A Framework for Truthful Online Auctions in Cloud Computing with Heterogeneous User Demands

Hong Zhang, Hongbo Jiang, *Senior Member, IEEE*, Bo Li, *Fellow, IEEE*, Fangming Liu, *Member, IEEE*, Athanasios V. Vasilakos, and Jiangchuan Liu, *Senior Member, IEEE*

Abstract—Auction-style pricing policies can effectively reflect the underlying trends in demand and supply for the cloud resources, and thereby attracted a research interest recently. In particular, a desirable cloud auction design should be (1) online to timely reflect the fluctuation of supply-demand relations, (2) expressive to support the heterogeneous user demands, and (3) truthful to discourage users from cheating behaviors. Meeting these requirements simultaneously is non-trivial, and most existing auction mechanism designs do not directly apply. To meet these goals, this paper conducts the first work on a framework for truthful online cloud auctions where users with heterogeneous demands could come and leave on the fly. Concretely speaking, we first design a novel bidding language, wherein users' heterogeneous requirement on their desired allocation time, application type, and even how they value among different possible allocations can be flexibly and concisely expressed. Besides, building on top of our bidding language we propose COCA, an incentive-Compatible (truthful) Online Cloud Auction mechanism. To ensure truthfulness with heterogenous and online user demand, the design of COCA is driven by a monotonic payment rule and a utility-maximizing allocation rule. Moreover, our theoretical analysis shows that the worst-case performance of COCA can be well-bounded, and our further discussion shows that COCA performs well when some other important factors in online auction design are taken into consideration. Finally, in simulations the performance of COCA is seen to be comparable to the well-known off-line Vickrey-Clarke-Groves (VCG) mechanism [19].

Index Terms—Online auction mechanism, cloud resource allocation, truthfulness, heterogeneous user demands

1 Introduction

Cloud computing is meant to offer on-demand network access to configurable computing resources, and promises to deliver to cloud users fast and flexible provisioning of resources with the freedom from long-term investments [8]. Such a paradigm has motivated a wide interest in dynamic and market-based resource allocation mechanisms in order to dynamically reflect the equilibrium market price, and provide satisfactory resource allocation for both cloud consumers and providers [9].

As a quick and efficient approach to selling goods at market value, auction-style pricing polices have been widely applied, reflecting the underlying trends in demand and supply for the computing resources. Indeed, an auction-style pricing policy, so called Spot Instance [1], has been adopted by Amazon to dynamically allocate cloud resources among potential users. Such a design has attracted significant attentions from the research community, and prompted a number of studies [3], [23], [24], [31] on auction-style cloud pricing mechanism design. More specifically, by means of forecasting the demand of users, [31] tries to maximize the revenue for the cloud resource provider in cloud spot market via linear programming, and [23] proposes a suite of computationally efficient and truthful auction-style pricing mechanisms, so that users can fairly compete for resources and cloud providers can increase their overall revenue. Abhishek et al. [3] investigate truthful auction policies under Bayes-Nash Equilibrium [19] in a spot market model, and [24] focuses on a optimal segmentation of cloud resources between pay-as-you-go market and the spot market. More recently, Zhang et al. [30] design a truthful single-round auction using LP decomposition. Shi et al. [22] propose the first online combinatorial auction for the VM market which is proved to be truthful and computationally efficient. And in [21], another combinatorial auction framework is introduced which provides guarantees in both the provider's revenue and social welfare.[1] Although these studies have made significant progress towards a full-fledged market-driven cloud service, they

- *H. Zhang and H. Jiang are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, HuBei, China. E-mail: {hongzhangblaze, hongbojiang2004}@gmail.com.*
- *B. Li is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail: bli@cs.ust.hk.*
- *F. Liu is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China. E-mail: lfxad@cse.ust.hk.*
- *A.V. Vasilakos is with the Department of Computer Science, Kuwait University. E-mail: vasilako@ath.forthnet.gr.*
- *J. Liu is with the the School of Computing Science, Simon Fraser University, 8888 University Drv, Burnaby, BC V5A 1S6, Canada. E-mail: jcliu@cs.sfu.ca.*

Manuscript received 1 July 2014; revised 11 Feb. 2015; accepted 30 Mar. 2015. Date of publication 19 May 2015; date of current version 10 Feb. 2016. Recommended for acceptance by R. G. Melhem. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TC.2015.2435784

1. Detailed definition of social welfare please refer to Section 2.2.

0018-9340 © 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

fail to simultaneously meet the following requirements of a desirable cloud auction.

1.1 Design Requirements of Cloud Auction

Typical methods such as [1] and [31] apply a pricing policy that changes periodically, to simplify the cloud provider's operations. On the downside, cloud users often suffer from this simplicity. For example, a cloud user with an uninterruptible job which lasts for more than one period will face the threat of being outbid and losing its cloud usage in any of these periods. Plus, as the price only changes periodically (once per hour; or less frequently in many cases [1], [31]), the fluctuation of supply-demand relations, which is always drastic due to the inherent dynamics and burst nature of user demands, cannot be timely and efficiently reflected.

Cloud users often have a variety of application and valuation types (i.e., with heterogeneous demands). For instance, users who have analytic or batch jobs to run are *job-oriented*: they mostly concern about whether their jobs can be finished in time [25], [28]. And some other users are *resource-aggressive*, e.g., for an SaaS provider who purchases cloud resources to provision for the peak demand, attaining enough cloud resources in a specific time interval (rush hours) is of its primary consideration [8]. Existing studies [8], [23] only consider cloud users with a single valuation type for simplicity. Moreover, users' valuations could be multi-minded: a bidder may have a valuation of \$10 in total for five VMs, while having a valuation of \$8 in total if it gets three of them. Similarly, a bidder may have a valuation of \$10 if its job is finished in 3 hours, while having a valuation of \$8 if finished in 5 hours. Current designs [1], [21], [22], [23], [24], [30], [31] cannot reflect such complicated form of user demands.

Last but not the least, the cloud market could be vulnerable to selfish user behaviors: cloud users may manipulate auction outcomes and gain unfair advantages via untruthfully revealing their preference on cloud resources. These strategic (or so-called cheating) behaviors will hinder other qualified users, significantly degrade auction efficiency, and greatly discourage users from participation. Truthful design [23] (which ensures that a tenant will maximize its benefit by bidding truthfully) has been proposed under one-time or periodic auction settings, which is unable to serve cloud users come on-the-fly. And analysis under Bayes-Nash Equilibrium in [3] is not so practical as users are assumed to have only two different valuations.

1.2 Overview of Our Proposed Framework

This paper conducts the first work on a framework, as shown in Fig. 1, for truthful online cloud auctions where users with heterogeneous demands could come and leave on the fly. First, cloud auctions are all carried out in an *online* manner, i.e., bidders can request cloud resources whenever they need, and their requests are processed by the cloud provider instantaneously. Such flexibility, in accordance with the "pay as you go" cloud paradigm, makes online auctions particularly attractive in practice [11]. Plus, a *bidding language* is implemented in the client side to translate user-specific demands into requests, by which users' heterogeneous demands can be restricted to

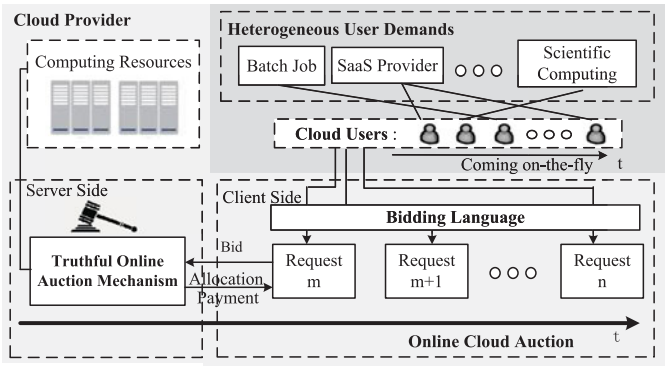


Fig. 1. Infrastructure of the framework for truthful online cloud auctions with heterogeneous user demands.

regulated and consistent forms while the details of the requirements can still be revealed. Finally, each request is then submitted to the server side through web service interfaces, and a *truthful* (also called incentive-compatible) online auction mechanism is implemented so that cloud users can be rationally motivated to reveal their truthful valuations in their requests.

To that end, our first contribution is a novel bidding language for our online cloud auction: We categorize bidders into three typical valuation types, and each of them can be specified by a valuation function, where users' valuations change with respect to the allocation they get. After that, each type of the valuation function can then be mapped into a corresponding request form which is more concise and regulated. Compared with many previous approaches which are more rigid in terms of the uniformity of request formats [1], [21], [22],[23], [24], [30], [31], our bidding language can flexibility and concisely reflect user requirement on allocation time/delay, application type, and even how they value among different possible allocations. We refer to such flexibility provided by our bidding language in defining user demand as "*expressiveness*".

On the downside, such expressiveness provided by our bidding language greatly complicates the problem of ensuring truthfulness, and existing auction mechanism designs [14], [15], [17], [26] cannot be directly applied. Therefore we present COCA, a truthful (incentive-compatible) online cloud auction mechanism building on top of our proposed bidding language. COCA is composed of two main building blocks: (1) a payment-function-based payment rule which is uniquely determined by the allocation result and the request submission time, and (2) an allocation rule that tries to maximize bidders' utility, which are proved to be the necessary and sufficient conditions for ensuring truthfulness. Based on these two rules, COCA ensures truthfulness by introducing a nondecreasing auxiliary pricing function in terms of the current supply-demand relations. After truthfulness is ensured, extensive theoretical analysis shows that the worst-case performance of COCA can be well-bounded, and further discussions show that COCA performs well in terms of other desired properties. Finally, in simulations the performance of COCA is seen to be comparable to the well-known off-line VCG mechanism.

The remainder of the paper proceeds as follows: Section 2 introduces our auction model and biding language. Section 3 presents COCA mechanism for online

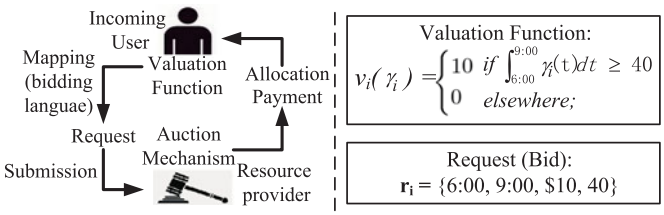


Fig. 2. An illustrative example of the online cloud auction.

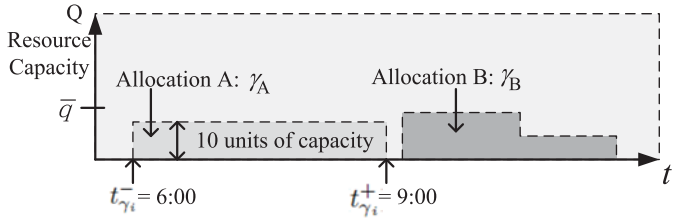


Fig. 3. The cloud resource and the presentation of allocation γ_i.

auctions. Section 4 conducts the competitive analysis of COCA. Section 5 discuss about COCA's performance in terms of some other important factors. Preliminary simulation results are illustrated in Section 6. Finally Section 7 concludes the paper.

2 Auction Model and Bidding Language

2.1 Online Auction Model for Cloud Resources

The auction procedure is shown in the left part of Fig. 2, cloud user (bidder) i with a specific valuation v_i (defined below) for the cloud resource arrives at an arbitrary time, maps (translates) its valuation into a request (bid) $\mathbf{r_i}$, and then submits $\mathbf{r_i}$ to the resource provider. After receiving the request, the resource provider is committed to determine the allocation γ_i and the payment pay_i immediately according to the adopted auction mechanism.

Resource—As shown in Fig. 3, we consider one cloud resource provider (e.g., a single data-center) who has a large number of computational resources [1], [8] with a fixed capacity Q in an infinite time interval $[0, \infty]$ [21], [22], [30].

Allocation Payment—An allocation γ_i presents how the resources are allocated to a cloud user i. A typical allocation γ_A, as shown in Fig. 3, presents the usage of cloud resource from 6:00 to 9:00 with a fixed 10 units of cloud capacity. As such, if we denote the start time (end time) of an allocation γ_i as $t^-_{\gamma_i}$ ($t^+_{\gamma_i}$) (as is shown in Fig. 3 for allocation A), an allocation γ_i can be regarded as a function over t, where $\gamma_i(t)$ ($t \in [t^-_{\gamma_i}, t^+_{\gamma_i}]$) is the instantaneous quantity of resources allocated to the user at time t. Additionally, like γ_B in Fig. 3, the capacity allocated is not necessarily time-invariant, in this paper we assume that $\gamma_i(t)$ can be varying within the range $[0, \overline{q}]$, and we denote all possible allocations to some user i as a set Γ_i. Moreover, we use $\gamma_i^* \in \Gamma_i$ to denote the *allocation decision*: the allocation finally determined for bidder i by the adopted auction mechanism, and we use $pay_i \in \mathbb{R}$ to represent the amount of money user i is required to pay.

Valuation—The valuation of bidder i is a function $v_i : \Gamma_i \to \mathbb{R}$, representing the benefit bidder i obtains from receiving a certain allocation γ_i of the cloud resource. Note that the valuation is known only to the bidder himself. Consider a cloud user with a job of size 40 (it takes 40 units of resource capacity running for one time unit to finish the job), who has a valuation of $10 if the job is carried out within [6:00,9:00]. The corresponding valuation function is presented in the right part of Fig. 2.

Utility—The utility $u_i(\gamma_i)$ refers to the "net profit" bidder i gets from an allocation γ_i [19], that is, $u_i(\gamma_i) = v_i(\gamma_i) - pay_i$. As bidders are assumed to be selfish, they may untruthfully reflect their bidding parameters in their requests in order to maximize their utility.

Social welfare—As a commonly used criterion to evaluate the performance (outcome) of an auction mechanism, social welfare refers to the sum of all the valuations of the allocated resources. Specifically, for any request sequence τ, the social welfare is defined as $E(\tau) = \sum_{i \in \tau} v_i(\gamma_i^*)$.

Request—To apply for cloud resources, a cloud user i submits a request $\mathbf{r_i}$ to the provider representing his valuation for the resource. A request is always represented as a set of bidding parameters. Recall the example shown in the right part of Fig. 2, obviously, in this case a concise request form of $\mathbf{r_i} = \{6{:}00, 9{:}00, \$10, 40\}$ is enough to reflect the entire valuation function. Here we call such mapping from valuations to requests as *bidding language*. Besides, we denote the *submission time* of request i (the time i arrives at the market) as t_{sub_i}. Note that we allow users to *reserve* resources, that is, in the above example user i can submit its request at anytime before 6:00.

2.2 Bidding Language for Heterogeneous User Demands

In this subsection, we turn to the mapping from valuations to requests. In practice, the user valuations are heterogeneous and often have complicated forms in the cloud market. This leads to a dilemma: *how can we present such heterogeneous valuation in requests with a concise and regulated form?* In response, we put forward in this paper a bidding language, by which the representation of requests captures as many features of the heterogeneous demands as possible, while keeping itself concise and consistent.

One challenge here is that a single request type cannot reveal the diversity of users' valuation types. As such, after investigating some different user requirements in cloud computing [5], [8], [9], [23], we categorize bidders into three typical valuation types, each with a corresponding valuation function. After that, each of the valuation functions is mapped into a corresponding concise request form respectively. As a result, each bidder can translate its specific valuation into a concise request according to its own type.

Valuation TYPE I: Job-oriented users.Valuation TYPE I: Job-oriented users. Analytic and batch jobs account for a large population in the cloud market [5], [8]. Generally, a job-oriented bidder i has a job with size $size_i$, and the job should be carried out within a time period $[a_i, d_i]$ (a_i denotes the earliest available time, and d_i denotes the deadline). It's worth mentioning that as we allow users to reserve resources, a_i can be greater or equal to t_{sub_i} in all three valuation types. Typically, bidder i has a valuation b_total_i if the job is finished before the deadline d_i, otherwise the longer the delay $delay_i$ is, the less the valuation will be [28]. To model this, we assume each bidder has a specific penalty rate pen_rate_i representing its valuation loss per unit delay time.

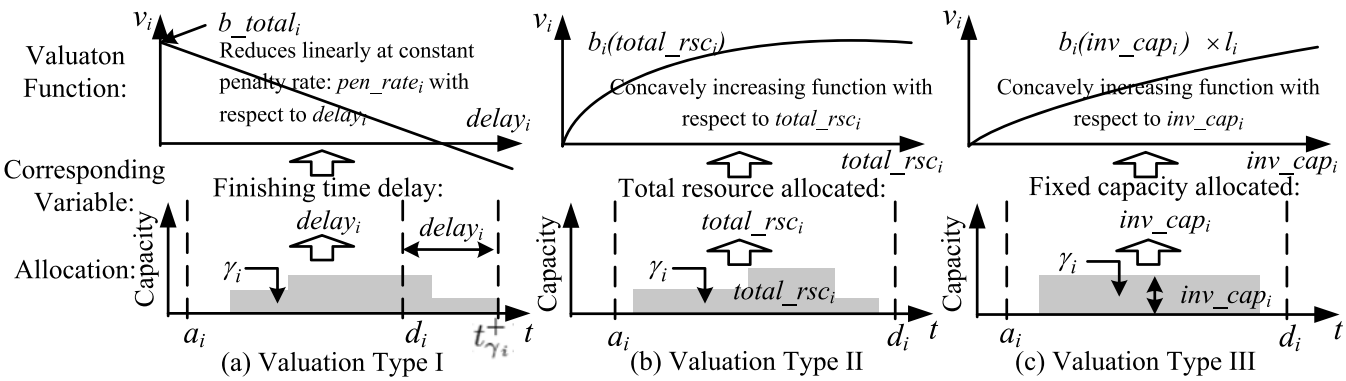


Fig. 4. The valuation function of three types of users.

As shown in Fig. 4a, we can specify the valuation function $v_i(\gamma_i)$ of job-oriented users as:

$$v_i(\gamma_i) = \begin{cases} b\_total_i - delay_i \cdot pen\_rate_i & \text{if } \int_{a_i}^{d_i+delay_i} \gamma_i(t)dt \geq size_i \\ 0 & elsewhere, \end{cases} \tag{1}$$

where $delay_i = max(t^+_{\gamma_i} - d_i, 0)$ uniquely corresponds to a given allocation γ_i. Recall the example shown in the right part of Fig. 2, obviously it belongs to such valuation type with pen_rate_i equals ∞. Accordingly, a request with the form: $\mathbf{r_i} = \{a_i, d_i, pen\_rate_i, b\_total_i, size_i\}$ is capable of reflecting the TYPE I valuation function.

Valuation TYPE II: Resource-aggressive users. For another typical kind of bidders, the biggest concern is to get sufficient number of resources in a specific time period. Such requirement is widely considered in traditional auction market settings, and it is also quite common in the cloud market. As an example, for an SaaS Provider who wants to provision for its peak-load during the rush hours $[a_i, d_i]$ [5], the more resources it acquires within such period, the more benefit it may get. As is shown in Fig. 4b, such a valuation function v_i of TYPE II can be specified as a nondecreasing concave function $b_i(\cdot)$, with respect to the total quantity of resource $total\_rsc_i$ allocated to bidder i within the preferred time period $[a_i, d_i]$. So we have: $v_i(\gamma_i) = b_i(total\_rsc_i)$, where $total\_rsc_i = \int_{a_i}^{d_i} \gamma_i(t)dt$. Accordingly, the request i of TYPE II can be organized as: $\mathbf{r}_i = \{a_i, d_i, b_i(total\_rsc_i)\}$.

Valuation TYPE III: Resource-aggressive users with time-invariant capacity requirements. Some users may be in need of time-invariant computing power. So for the third type, we consider users that require an invariable capacity of computing power (as the resource model in [4], [8]). Typically, such a cloud user i may request cloud resources of invariable capacity for a time length l_i, within a preferredte time duration $[a_i, d_i]$ ($l_i \leq d_i - a_i$). And the valuation can be presented by a concavely increasing function $b_i(\cdot)$ with respect to the invariant capacity inv_cap_i allocated to him. For example, a user may have a valuation of \$10 in total if $inv\_cap_i = 5$ (five VMs allocated to him), or a valuation of \$8 if $inv\_cap_i = 3$. Accordingly, we have $v_i(\gamma_i) = b_i(inv\_cap_i) \cdot l_i$, where $inv\_cap_i = min_{t\in[t^-_{\gamma_i}, t^+_{\gamma_i}]}(\gamma_i(t))$. Such a request can be organized as: $\mathbf{r}_i = \{a_i, d_i, l_i, b_i(inv\_cap_i)\}$.

Assumptions. First, aiming at a compelling user experience, preemption [13] is not allowed. Second, we do not assume any specific distribution of bidding parameters in the request $\mathbf{r_i}$—we only apply a very general assumption that the *unit valuation* (the valuation for one unit resource per unit time) is within a known interval $[\underline{p}, \overline{p}]$, and the job length of Type III bidders is within the interval $[\underline{l}, \overline{l}]$ in order to bound the social welfare.[2]

A simple tenant-provider interface design. In cloud market some users may not have a very good knowledge of their valuation functions, and some may have difficulty specifying their valuation functions. To make our bidding language one more step towards a practical and implementable design, we propose a simple interface to help users estimate their valuation function easily. Fig. 5 introduces the interface with a Type II user taken as an example.

Client Side: First, instead of reporting the valuation function $b_i(total\_rsc_i)$, a user can submit several critical parameter-value pairs, each representing a specific $total\_rsc_i$ and the corresponding valuation (a point on the valuation curve).[3] Then after providing a coarse-grained localization of the valuation function in step 1, the user should also specify its regression type (e.g., linear, piecewise linear, quadratic), in order to further detail the shape of the function.

Server side. With the given user type, critical parameter-value pairs and regression type, the provider is able to generate an initial valuation function using regression analysis. And finally, for some regression types, further modification is implemented to maintain the concavity and monotonicity of the function.

With the above interface, a user who knows its demands for crystal clear can get a precise valuation function estimation by providing many critical parameter-value pairs (or by submitting the valuation function directly), and for users without a sound knowledge on their valuation, the submission of two or three parameter-value pairs will also provide a coarse estimation. In addition, we note that such interface can also be extended to enable more complicated regression types, while a more sophisticated user valuation estimation design is beyond the scope of this paper.

2. Besides, it is also worth mentioning that the concavity assumption on $b_i(\cdot)$ for Type II and III users is only used later in the worst-case performance analysis (Section 4), and does not affect our analysis on ensuring truthfulness.

3. Pair (0,0) is added by default as we assume users always have 0 valuation if they get nothing.

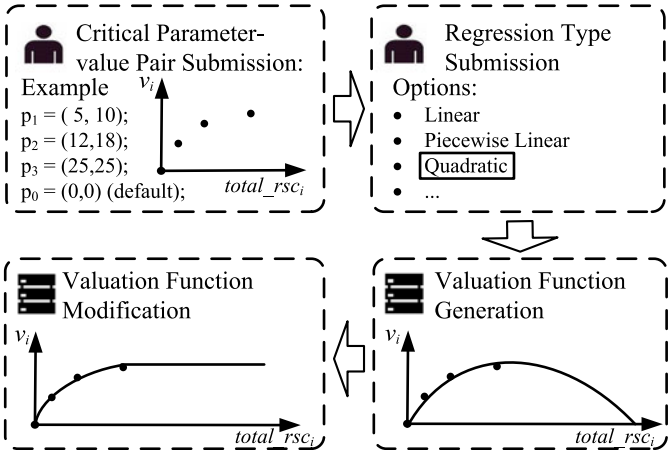


Fig. 5. An illustrative example of the tenant-provider interface for Type II users.

2.3 Performance Metrics for the Auction Mechanism

Truthfulness (also called incentive-compatibility) is one of the most critical property of auction mechanism [19]. As mentioned in Section 1, an auction could be vulnerable to market manipulation without truthfulness-guaranteed.

Definition 1. *(Truthfulness) An auction mechanism $\mathcal{A}$ is said to be truthful if, for any bidder i, regardless of the behaviors of other bidders, declaring a bid that truthfully reveals its valuation can maximize its utility.*

Existing pricing designs for cloud market [23], [24], [31] typically target at achieving the "optimal" allocation performance on revenue or social welfare. However, the optimal solution generally needs further information of the variation of user demands, which is very hard to estimate. To this end, in this paper we focus on an alternative problem: without assumptions on any specific distribution information on bidders' arrival or valuation, *how can we achieve a good worst-case performance on social welfare?* To evaluate the worst-case performance of an auction mechanism by its social welfare, a commonly adopted way is competitive analysis [14], [17]—to compare the allocation performance against the optimal offline solution—VCG mechanism.

Definition 2. *(Competitive ratio on social welfare) An auction mechanism $\mathcal{A}$ is ζ-competitive with respect to the social welfare if for every bidding sequence τ, $E_{\mathcal{A}}(\tau) \geq E_{VCG}(\tau)/\zeta$. Accordingly, ζ is the competitive ratio of $\mathcal{A}$.*

To that end, next in Section 3 we propose the online auction mechanism, called COCA, along with the proof for the truthfulness, followed by Section 4 where we conduct extensive competitive analysis on COCA.

3 Mechanism Design: Ensuring Truthfulness

In this section, first we present the challenges on ensuring truthfulness under our proposed online cloud auction model, which is followed by an in-depth analysis of design methodology for ensuring truthfulness. Then based on such design methodology, a truthful online cloud auction mechanism COCA is proposed building on top of our proposed bidding language in the previous section. Finally, extensions of COCA is discussed at the end of this section.

3.1 Design Challenges on Ensuing Truthfulness

Truthful online mechanism design has been extensively studied lately. Lavi and Nisan [17] first proposed the paradigm of truthful online auction and performed competitive analysis on both revenue and efficiency(social welfare). After that, many online mechanisms have been proposed under a great variety of scenarios, e.g., goods with limited supply [15], digital goods with infinity supply [7], and gradually expiring goods [18], etc. Accordingly, a question raises intuitively: *Can we simply adopt some existing auction mechanisms to achieve good allocation performance?*

Compared with most existing online auction settings, the expressiveness of our auction model are two-fold: First, to be more expressive, we consider more than one typical valuation types, each with a request form with more than one bidding parameter. Second, users' valuations are modeled as functions in terms of the allocation results they obtain. Undoubtedly, such an auction setting is more in accordance with the cloud market, however on the downside it greatly complicates the problem of ensuring truthfulness.

Unlike in single parameter settings where truthfulness can be characterized by a *monotonic allocation rule* and a *critical payment rule* [19], ensuring truthfulness is generally much more difficult in our auction setting with such expressiveness. As for our cloud auction model, cloud users might get extra benefit by cheating on any of the bidding parameters or even their valuation types. Specially, cloud resource is a kind of reusable goods [14]: instead of *obtaining* the resource forever, the cloud users are actually *renting*the usage of the resource for a certain period of time. As a consequence, bidders' valuations depend not only on the amount of resource allocated to them, but also on the time period where (and for how long) the resources are allocated to them, which makes online cloud auctions vulnerable to various types of "time cheating" [11], where users can strategically submit a false arrival and departure time to get a better allocation.

One related online auction mechanism design for reusable goods is presented in [14], and later adopted in online spectrum auction in [11]. However, such design is no longer available under our cloud auction model: First, considering the heterogeneous and complicated request form in our cloud auction model, the proposed algorithm cannot be directly used for Type I and Type II users. Second, remind that in our cloud auction model, users are allowed to *reserve* resources, such auction setting makes the assumption of *no early arrival and no late departure* [14], [20] no longer valid in our cloud auction model. Accordingly, if we adopt the mechanism in [11], [14] to Type III users, users can get extra benefit by strategically reporting two or more of their bidding parameters(e.g., an earlier earliest available time a_i together with a longer job length l_i, or a later deadline d_i together with a higher biding price function $b(\cdot)$) together). In addition, other truthful designs [26], [27] with specific user distribution for spectrum allocation cannot be applied since in our model no specific distribution information is assumed. Therefore, we believe an in-depth analysis on truthful mechanism design under our proposed model is worth the effort.

3.2 Design Methodology on Ensuring Truthfulness

3.2.1 How to Determine the Payment

Under our proposed auction model, the payment of a bidder i can be generally considered as a function: $pay_i = p_i(\gamma_i, t_{sub_i}, \mathbf{r_i})$.[4] However it may not be the simplest form of the payment function as the parameters are not totally independent. Indeed, the following lemma shows that the payment function can be further simplified as $pay_i = p_i(\gamma_i, t_{sub_i})$.

Lemma 1. *For any truthful auction algorithm $\mathcal{A}$, given γ_i and t_{sub_i}, the payment should be uniquely determined for any bidder i regardless of its request $\mathbf{r_i}$.*

Proof. We prove it by contradiction. Given some γ_i and t_{sub_i}, assume that there are two different requests $\mathbf{r_i}$ and $\mathbf{r'_i}$ with $p_i(\gamma_i, t_{sub_i}, \mathbf{r_i}) > p_i(\gamma_i, t_{sub_i}, \mathbf{r'_i})$. In this case, a bidder with true request $\mathbf{r_i}$ will increase its utility by declaring $\mathbf{r'_i}$. Therefore the auction is no longer truthful, completing the contradiction. □

Next we discuss how γ_i and t_{sub_i} are correlated to the payment function $p_i(\gamma_i, t_{sub_i})$.

Definition 3. *(Monotonic with allocation) We say $\gamma_i \succeq \gamma'_i$ if $\forall t, \gamma_i(t) \geq \gamma'_i(t)$. A payment function p_i is monotonic with allocation if for any t_{sub_i} and any allocation $\gamma_i \succeq \gamma'_i$, we have $p_i(\gamma_i, t_{sub_i}) \geq p_i(\gamma'_i, t_{sub_i})$.*

Definition 4. *(Monotonic with submission time) A payment function is monotonic with submission time if for any allocation γ_i and any submission time $t_{sub_i} \leq t'_{sub_i}$, we have $p_i(\gamma_i, t'_{sub_i}) \geq p_i(\gamma_i, t_{sub_i})$.*

Given the above definitions, we are now ready to present the following theorem about how to decide the payment in order to ensure truthfulness.

Theorem 1. *For any truthful online auction mechanism $\mathcal{A}$, the payment for any bidder i can be determined by a payment function $p_i(\gamma_i, t_{sub_i})$, which should be monotonic with submission time and monotonic with allocation.*

Proof. We prove it by contradiction. First, assume that p_i is not monotonic with allocation, that is, there exists a bidder i with two possible allocation decisions $\gamma'_i \succeq \gamma_i$, such that $p_i(\gamma_i, t_{sub_i}) \geq p_i(\gamma'_i, t_{sub_i})$. Denote $\mathbf{r'_i}$ and $\mathbf{r_i}$ as the requests lead to these two allocations respectively, then bidder i with truthful request $\mathbf{r_i}$ will increase its utility by declaring request $\mathbf{r'_i}$. That's a contradiction with the fact that $\mathcal{A}$ is truthful.

Second, we assume that p_i is not monotonic with submission time. That is, for some request $\mathbf{r_i}$ with submission time t_{sub_i}, $p_i(\gamma_i, t_{sub_i}) \geq p_i(\gamma_i, t'_{sub_i})$ holds for some $t'_{sub_i} \geq t_{sub_i}$. In this case, user i may increase its utility by declaring the same request $\mathbf{r_i}$ at such a later submission time t'_{sub_i}. That is also a contradiction with the fact that $\mathcal{A}$ is truthful. □

4. The function can be more formally written as $pay_i = p_i(\gamma_i, t_{sub_i}, \mathbf{r_i}, \Phi)$, where Φ denotes all the parameters which cannot be directly affected by the user strategy. Note that it is not necessary to explicitly show Φ in our analysis about truthfulness due to Φ's independence of user strategy.

Theorem 1 provides a payment rule which serves as a necessary condition to ensure truthfulness. Intuitively, a bidder may strategically delay its submission time t_{sub_i}, or try to manipulate the allocation decision by reporting an untruthful request $\mathbf{r'_i}$, so a later submission time, or a "better" allocation should lead to a higher payment.

3.2.2 How to Determine the Allocation

In response to the above payment function, we resort to a general allocation rule.

Proposition 1. *(Nisan et al. [19]) For any truthful auction mechanism $\mathcal{A}$, the auction mechanism optimizes for all bidders, i.e., the allocation decision maximizes the utility gain for each bidder i.*

Proposition 1 provides a generalized necessary condition to ensure truthfulness, and following this route, we present our guideline of how to determine the allocation under our auction model:

Theorem 2. *For any auction mechanism $\mathcal{A}$, denote all possible allocation results to some bidder i as a set Γ_{iA}, then for any bidder i the auction mechanism $\mathcal{A}$ will try to maximize its utility according to the constraint Γ_{iA} and its request $\mathbf{r_i}$. More formally we have*

$$\gamma_i^* = argmax_{\gamma_i \in \Gamma_{iA}}(\widetilde{v}_i(\gamma_i) - p_i(\gamma_i, t_{sub_i})),^5 \quad (2)$$

where $\widetilde{v}_i(\gamma_i)$ is the valuation function learned from request $\mathbf{r_i}$, and γ_i^* is the allocation finally determined for bidder i.

Proof. We prove this theorem by contradiction. Assume the allocation decision for bidder i does not maximize $(\widetilde{v}_i(\gamma) - pay_i)$, and we denote such an allocation as $\gamma'_i \in \{\Gamma\}$. Let $\gamma_i \in \Gamma_{iA}$ be the allocation that maximizes $(\widetilde{v}_i(\gamma) - pay_i)$, and let $\mathbf{r'_i}(\gamma)$ be some other bidding price function, by submitting which bidder i can get the allocation γ_i. So, if the resource provider makes an allocation $\gamma'_i \neq \gamma_i$, bidder i will increase its utility by declaring a request $\mathbf{r'_i}(\gamma)$. It then contradicts with the assumption that the auction is truthful. So Eq. (2) must be satisfied to ensure truthfulness. □

Theorem 2 provides an *allocation rule* which serves as another necessary condition to ensure truthfulness in our cloud auction model. To help better understand the theorem, it is worth mentioning that Γ_{iA} may not be equivalent to Γ_i, which is the general set of all possible allocations. On one hand, the auction mechanism $\mathcal{A}$ may restrict the possible allocation results. As an extreme example, an auction mechanism which refuses all the user requests ($\Gamma_{iA} = \emptyset$) can be truthful as it satisfies Eq. (2). On the other hand, in an online auction the possible allocation results at some time can also be restricted by the previous allocation decisions. More specifically, for some bidder i, Γ_{iA} may be restricted for the reason that all the resources in the interested time period have already been allocated to bidders with submission time earlier than t_{sub_i}.

5. Here we adopt a general assumption that $p_i(\emptyset, t_{sub_i}) = 0$ (a user always pays 0 if it gets nothing), thus users will not receive negative utilities.

3.2.3 Necessary and Sufficient Conditions for Ensuring Truthfulness

The previous subsection has provided a payment rule (in Theorem 1) and an allocation rule (in Theorem 2), both of which serve as necessary conditions to ensure truthfulness in our cloud auction model. More interestingly, in this subsection our further analysis shows that these two rules together also serve as the sufficient conditions for ensuring truthfulness.

Let $\mathbf{r'}_i$ be any untruthful request derived from the truthful request $\mathbf{r}_i$ by making arbitrary changes on the reported valuation function $\widetilde{v}_i(\gamma_i)$ (changing any of its bidding parameters or its request type). In this case we have the following theorem:

Theorem 3. *An online auction mechanism $\mathcal{A}$ is truthful if and only if for any bidder i:*

1) *the payment can be determined by a payment function $p_i(\gamma_i, t_{sub_i})$, which is monotonic with submission time and monotonic with allocation;*
2) *the auction mechanism $\mathcal{A}$ will try to maximize i's utility according to the constraint Γ_{iA} and i's request $\mathbf{r_i}$, that is, $\gamma_i^* = argmax_{\gamma_i \in \Gamma_{iA}}(\widetilde{v}_i(\gamma_i) - p_i(\gamma_i, t_{sub_i}))$.*

Proof. *1) Proof of the "only if" part:* please refer to the proofs of Theorem 1 and Theorem 2.

2) Proof of the "if" part: Let $\mathbf{r''}_i$ be any untruthful request derived from the truthful request $\mathbf{r}_i$ by making arbitrary changes on the reported valuation function $\widetilde{v}_i(\gamma_i)$(changing any of its bidding parameters or its request type) and the submission time t_{sub_i} (changed to t''_{sub_i}). Denote $u_i(\mathbf{r}_i)$ and $u_i(\mathbf{r''}_i)$ as the utility bidder i gets by submitting request $\mathbf{r}_i$ and $\mathbf{r''}_i$ respectively.

First, it is derived from Theorem 2 that γ_i^* maximizes $u_i(\gamma_i)$. Therefore, if we denote the allocation decision for reporting $\mathbf{r}_i$ and $\mathbf{r}''_i$ as γ_i^*, and γ''_i respectively, the following inequation holds:

$$u_i(\mathbf{r}_i) = v_i(\gamma_i^*) - p_i(\gamma_i^*, t_{sub_i}) \geq v_i(\gamma''_i) - p_i(\gamma''_i, t_{sub_i}). \quad (3)$$

Second, note that the true submission time is defined as the time that a user arrives at the market, which is also the first time it is aware of its demand. Thus, it's not possible for any user to report a submission time earlier than its true submission time [14]. Since $p_i(\gamma_i, t_{sub_i})$ is monotonic with submission time, we have $p_i(\gamma_i, t_{sub_i}) \leq p_i(\gamma_i, t''_{sub_i})$ for any γ_i and $t''_{sub_i} \geq t_{sub_i}$. Hence the following inequations holds:

$$v_i(\gamma''_i) - p_i(\gamma''_i, t_{sub_i}) \geq v_i(\gamma''_i) - p_i(\gamma''_i, t''_{sub_i}) = u_i(\mathbf{r}''_i). \quad (4)$$

Eq. (3) and Eq. (4) imply that $u_i(\mathbf{r}_i) \geq u_i(\mathbf{r}''_i)$. This result demonstrate that reporting the true request $\mathbf{r_i}$ always results in a better utility than reporting any untruthful request $\mathbf{r}''_\mathbf{i}$. That is, the auction is truthful if the payment rule in Theorem 1 and the allocation rule in Theorem 2 are adopted, the theorem holds. □

3.3 COCA: A Truthful Online Cloud Auction Design

Motivated by the aforementioned design methodology, let's turn to the design of COCA in detail. We start with introducing how we construct the payment function p_i for every coming bidder i such that it is monotonic with allocation and submission time.

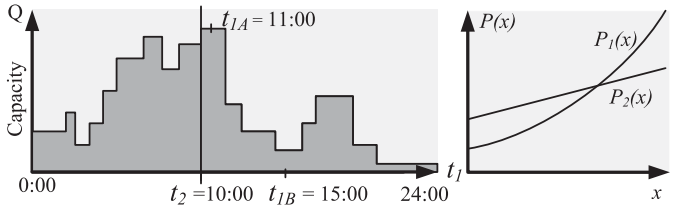


Fig. 6. The utilization rate $U(t_1, t_2)$ with $t_2 =$ 10:00 and the auxiliary pricing function $P(x)$.

3.3.1 Payment Function Construction

Intuitively, COCA's payment function is committed to reflecting the current equilibrium market price—the resource should be charged more in "hot" time periods (where there are a greater number of user demands). Similar to [17], COCA exploits an *auxiliary pricing function* $P(x)$ with respect to the current *utilization rate* U to help the resource provider decide the payment function.

Reserved resource utilization rate U. Note that COCA allows a bidder to reserve resources that are not available in the current time period. To that end, we define a variable—the *reserved resource utilization rate* U (we call it utilization rate for short henceforth). Formally, denote all the allocation decisions $\gamma_i^*, i = 1, 2 \ldots$ made by time t_2 as a set Γ_{t_2}, then we have $U(t_1, t_2) = \sum_{\gamma_i^* \in \Gamma_{t_2}} \gamma_i^*(t_1)/Q$. Obviously we have $U \in [0, 1]$. With such a definition, U can clearly reflect the status of how the cloud resources are allocated (reserved) *at* time t_1 according to the allocation decisions made *by* time t_2. As an example, Fig. 6 shows the utilization rate within one day ($t_1 \in$ [0:00,24:00]) by the time $t_2 =$ 10:00. A high utilization rate is observed at U(11:00,10:00), which indicates that the resources at time t_{1A} =11:00 have almost been sold out by 10:00. On the contrary, a low value at U(15:00,10:00) implies that there are still a lot of unallocated resources at time $t_{1B} =$ 15:00 by 10:00. Specifically, we denote $U(t_1, t_{sub_i})$ ($U(t_1, t^+_{sub_i})$) as the utilization rate at time t_1 before (after) the allocation of request $\mathbf{r_i}$ submitted at t_{sub_i}. Accordingly we have $U(t_1, t^+_{sub_i}) - U(t_1, t_{sub_i}) = \gamma_i(t_1)/Q$. In addition, it's obvious that $\forall t_1, t_2$, we have $t_2 \leq t'_2 \Rightarrow U(t_1, t_2) \leq U(t_1, t'_2)$.

Auxiliary pricing function $P(x)$: To help the resource provider determine the payment function, in COCA we introduce an auxiliary pricing function $P(x)$ which is predetermined by the resource provider before the auction process. $P(x)$ explicitly presents the "marginal price" with respect to the utilization rate. That is, for any piece of allocation γ_i with

$$\begin{cases} t^+_{\gamma_i} - t^-_{\gamma_i} = \Delta l \to 0 \\ \gamma_i(t) = \Delta q \to 0 \qquad \forall t \in [t^-_{\gamma_i}, t^+_{\gamma_i}]. \end{cases} \quad (5)$$

The payment for such γ_i is calculated as:

$$p_i(\gamma_i, t_{sub_i}) = P(U(t^-_{\gamma_i}, t_{sub_i})) \cdot \Delta l \cdot \Delta q, \quad (6)$$

where $U(t^-_{\gamma_i}, t_{sub_i})$ presents the utilization rate reserved at $t^-_{\gamma_i}$ by the time user i submits its request.

Algorithm 1. COCA Mechanism Design

Input:
1) The request sequence τ: $\{\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, ...\mathbf{r}_\infty\}$, such that $t_{sub_1} < t_{sub_2} < ... < t_{sub_\infty}$;
2) A nondecreasing auxiliary pricing function $P(x)$

Output:
1) The allocation decision γ_i^* for each request $\mathbf{r}_i$;
2) Payment decision pay_i for each request $\mathbf{r}_i$

1: **Initialization of the utilization rate:**
2: $\forall\ t \in [0, \infty], U(t, t_{current}) = 0$ % $t_{current}$ refers to the current time.
3: **for** $i = 1\ to \infty$ **do**
4: **Constructing payment function:**
5: $p_i(\gamma_i, t_{sub_i}) = \int_{t_{\gamma_i}^-}^{t_{\gamma_i}^+} \int_{U(t,t_{sub_i})}^{U(t,t_{sub_i})+\gamma_i(t)/Q} P(x) \cdot Q dx dt$
6: **Allocation rule:**
7: Check the type $type_i$ of $\mathbf{r}_i$;
8: **switch** $type_i$
9: **case** 1 % Allocation determination for Type I bidder:

$$\gamma_i^* = argmax_{\{\gamma_i\}}(b\_total_i - pen\_rate_i \cdot delay_i - p_i(\gamma_i, t_{sub_i}))$$
$$\text{s.t. } \forall t, U(t, t_{sub_i}) + \gamma_i(t)/Q \leq 1$$
$$\int_{a_i}^{d_i + delay_i} \gamma_i(t) \geq size_i$$

where $delay_i = max(t_{\gamma_i}^+ - d_i, 0)$
% Find the allocation that maximizes i's utility if the job is accepted.
10: **if** $b_{total_i} - pen\_rate_i \cdot delay_i^* - p_i(\gamma_i^*, t_{sub_i}) < 0$ **then**
11: set $\gamma_i^* = \emptyset$
% If the maximum utility of accepting the job is negative, then reject the job.
12: **end if**
13: **end case**
14: **case** 2 % Allocation determination for Type II bidder:

$$\gamma_i^* = argmax_{\{\gamma_i\}}(b_i(total\_rsc_i) - p_i(\gamma_i, t_{sub_i}))$$
$$\text{s.t. } \forall t, U(t, t_{sub_i}) + \gamma_i(t)/Q \leq 1$$

where $total\_rsc_i = \int_{a_i}^{d_i} \gamma_i(t) dt$
% Find the allocation that maximizes i's utility.
15: **end case**
16: **case** 3 % Allocation determination for Type III bidder:

$$\gamma_i^* = argmax_{\{\gamma_i\}}(b_i(inv\_cap_i) \cdot l_i - p_i(\gamma_i, t_{sub_i}))$$
$$\text{s.t. } \forall t, U(t, t_{sub_i}) + \gamma_i(t)/Q \leq 1$$
$$\forall t_1, t_2 \in [t_{\gamma_i}^-, t_{\gamma_i}^+], inv\_cap_i = \gamma_i(t_1) = \gamma_i(t_2)$$

% Find the allocation that maximizes i's utility.
17: **end case**
18: **end switch**
19: **Payment rule :**
20: $pay_i = p_i(\gamma_i^*, t_{sub_i})$
21: **Updating the utilization rate:**
22: $\forall t \in [t_{\gamma_i}^{*-}, t_{\gamma_i}^{*+}], U(t, t_{current}) = U(t, t_{current}) + \gamma_i^*(t)/Q$
23: **end for**

In such a way, the total payment for any γ_i can be calculated by dividing it into such small pieces and summing them up, thus our payment function has the following form:

$$p_i(\gamma_i, t_{sub_i}) = \int_{t_{\gamma_i}^-}^{t_{\gamma_i}^+} \int_{U(t,t_{sub_i})}^{U(t,t_{sub_i})+\gamma_i(t)/Q} P(x) \cdot Q dx dt. \quad (7)$$

It is noted that given a nondecreasing $P(x)$, the payment function satisfies all the necessary conditions given in Theorem 1 to ensure truthfulness. Moreover, we note that the outcome of COCA (e.g. revenue, social welfare, etc.) directly depends on the choice of $P(x)$, and a provider-specific $P(x)$ can be implemented in order to realize different performance goals. Later in Section 4 we will show that COCA's allocation performance with respect to social welfare greatly depends on the choice of such $P(x)$, and a good competitive ratio can be achieved by carefully designing $P(x)$.

3.3.2 Mechanism Description

We now describe our design of COCA, shown in Algorithm 1, step by step:

- *Step 1: Lines 4-5 (Constructing payment function).* For any bidder i, the payment function is given as $p_i(\gamma_i, t_{sub_i}) = \int_{t_{\gamma_i}^-}^{t_{\gamma_i}^+} \int_{U(t,t_{sub_i})}^{U(t,t_{sub_i})+\gamma_i(t)/Q} P(x) \cdot Q dx dt$, where $P(x)$ can be any nondecreasing function.
- *Step 2: Lines 6-18 (Allocation rule).* The allocation tries to maximize the utility gain for each bidder i according to its request $\mathbf{r_i}$ and the current utilization rate, that is

$$\gamma_i^* = argmax_{\gamma_i}(\widetilde{v}_i(\gamma_i) - p_i(\gamma_i, t_{sub_i}))$$
$$\text{s.t.} \forall t, U(t, t_{sub_i}) + \gamma_i(t)/Q \leq 1, \quad (8)$$

  where $\widetilde{v}_i(\gamma_i)$ is the valuation function learned from the request $\mathbf{r_i}$.
- *Step 3: Lines 19-20 (Payment rule).* Determine the payment according to the payment function p_i and the allocation decision γ_i^*: $pay_i = p_i(\gamma_i^*, t_{sub_i})$.
- *Step 4: Lines 21-22 (Updating the utilization rate).* Update the utilization rate U according to the allocation decision γ_i^*.

Following the above steps, we present our detailed design of COCA in Algorithm 1. It is shown that as bidders are categorized to the three valuation types introduced in Section 2.2, corresponding allocation rules can be applied accordingly. As a consequence, it is not surprising that truthfulness can be ensured by such mechanism design:

Theorem 4. *COCA is a truthful auction mechanism under our online cloud auction model.*

Proof. According to the mechanism description, the payment rule and allocation rule of COCA satisfies the rules proposed in Theorem 3, then in line with Theorem 3, COCA is truthful under our online cloud auction model.□

Meanwhile, such a design spontaneously balances the workload. Since according to the allocation rule, users will be more likely to be allocated in time durations where the current utilization rate is lower, as the payment will be lower according to $p_i(\gamma_i, t_{sub_i})$ (with a nondecreasing $P(x)$).

Remark: One may argue that COCA offers less flexibility as the allocation and payment is based on the auxiliary

pricing function $P(x)$. More accurately, we note that the allocation and payment of COCA mechanism for any bidder i is determined by the following three factors: (1) the request $\mathbf{r_i}$; (2) the requests which have been accepted before bid i; (3) the auxiliary pricing function $P(x)$. Actually, for online cloud auction design, it is always the case that: To ensure truthfulness under a more complex model and a more general assumption, the designed auction mechanism has to suffer from less flexibility. This is also the reason why we introduce the auxiliary pricing function $P(x)$ to help determine the payment. Essentially speaking, without further restrictions or assumptions, there exists a general trade off between the generality of auction model and the flexibility the auction algorithm can enjoy.

Algorithm 2. COCA$^+$ Mechanism Design

1: ...
2: **Allocation rule:**
3: **for** any incoming request $\mathbf{r_i}$ **do**
4: **if** $\mathbf{r_i}$ belongs to one of the 3 valuation types **then**
5: **goto** Algorithm 1
6: **else**
7: Determine the allocation by the general allocation rule:

$$\gamma_i^* = argmax_{\gamma_i}(\widetilde{v}_i(\gamma_i) - p_i(\gamma_i, t_{sub_i}))$$
$$\text{s.t. } \forall t, U(t, t_{sub_i}) + \gamma_i(t)/Q \leq 1$$

8: **end if**
9: **end for**
10: ...

Extension to general valuation types. While we claim that the three valuation types in our model are typical, they are by no means exclusive in our auction mechanism. Instead, COCA can be extended (denoted as COCA$^+$) to deal with any other type of heterogeneous user demands, and the extension only requires minor modification in the allocation rule. As is shown in Algorithm 2, if a user does not belong to any of the three given valuation types, he can submit its request in a general form $\mathbf{r_i} = \widetilde{v}_i(\gamma_i)$, reflecting its valuation with respect to the allocation γ_i. Consequently the allocation will be determined according to the general allocation rule shown in Step 2. By doing so, COCA$^+$ remains truthful for any unknown type of user demands.[6]

Extension to multiple resource types. COCA can be further extended (denoted as COCA^{++}) to fit a more general resource model with multiple kinds of cloud resource types, such as computing resources, bandwidth, storage capacity, etc. Denote the generalized resource model as an n-dimensional vector $\mathbf{Q} = \{Q_1, Q_2, \ldots Q_n\}$, with each component corresponds to the resource capacity of one resource type. As such, the allocation to some bidder i can be considered as a corresponding n-dimensional vector $\boldsymbol{\gamma_i} = \{\gamma_{1i}, \gamma_{2i}, \ldots \gamma_{ni}\}$, and similarly the utilization rate can be considered as $\mathbf{U} = \{U_1, U_2, \ldots U_n\}$. In this case, users can submit its request in a general form $\mathbf{r_i} = \widetilde{v}_i(\boldsymbol{\gamma_i})$, reflecting its valuation with respect to the n-dimensional allocation $\boldsymbol{\gamma_i}$. If we do not consider the correlation of price among different resource types, a set of functions: $\mathbf{P} = \{P_1(x_1), P_2(x_2), {}_n(x_n)\}$ can be applied to ensure truthfulness if $d(P_k(x_k))/dx_k \geq 0$ holds $\forall k = 1$ to n. The detailed design is shown in Algorithm 3.

Algorithm 3. COCA^{++} Mechanism Design

1: ...
2: **for** any incoming request $\mathbf{r_i}$ **do**
3: **Constructing payment function:**
4:

$$p_i(\boldsymbol{\gamma_i}, t_{sub_i}) = \int_{t^-_{\boldsymbol{\gamma_i}}}^{t^+_{\boldsymbol{\gamma_i}}} \sum_{k=1}^{n} \int_{U_n(t,t_{sub_i})}^{U_n(t,t_{sub_i})+\gamma_{1i}(t)} P_n(x) \cdot Q_n dx dt \quad (9)$$

5: **Allocation rule:**
6: Determine the allocation by the general allocation rule:

$$\boldsymbol{\gamma_i}^* = argmax_{\boldsymbol{\gamma_i}}(\widetilde{v}_i(\boldsymbol{\gamma_i}) - p_i(\boldsymbol{\gamma_i}, t_{sub_i}))$$
$$\text{s.t. } \forall t, n, U_n(t, t_{sub_i}) + \gamma_{ni}(t)/Q_n \leq 1$$

7: **Payment rule:**
8: $pay_i = p_i(\boldsymbol{\gamma_i}^*, t_{sub_i})$
9: **Updating the utilization rate:**
10: $\forall t \in [t^{*-}_{\boldsymbol{\gamma_i}}, t^{*+}_{\boldsymbol{\gamma_i}}], \mathbf{U}(t, t_{current}) = \mathbf{U}(t, t_{current}) + \boldsymbol{\gamma_i}^*(t)$
11: **end for**

4 Mechanism Design: Achieving a Nontrivial Competitive Ratio on Social Welfare

Based on the above mechanism design of COCA, truthfulness can be ensured by the implementation of a nondecreasing auxiliary pricing function $P(x)$. Now we have the following two problems still unsolved—*(i). how to determine the auxiliary pricing function $P(x)$? (ii). how can COCA achieve a nontrivial competitive ratio on social welfare (defined in Section 2.2)?* In this section we show that the answers to these two questions are closely correlated—the competitive ratio highly depends on the auxiliary pricing function $P(x)$. Moreover, we show that the competitive ratio of COCA can be well-bounded by appropriately constructing the auxiliary pricing function $P(x)$.

4.1 Competitive Analysis for a Single Bidder Type

In this subsection, we consider the scenario where the cloud users in a request sequence τ only belong to a *single request type*. In the following analysis, we will show how the competitive ratio of COCA on social welfare is determined by the choice of the auxiliary pricing function $P(x)$.

4.1.1 Competitive Analysis for Type II Bidders

Recall that we apply a general assumption that the *unit valuation* (the valuation for one unit resource per unit time) is within a known interval $[\underline{p}, \overline{p}]$. Under such assumption, denote the social welfare achieved when VCG (COCA) is applied as E_{VCG} (E_{COCA}), we have the following theorem for Type II bidders:

Theorem 5. *For any request sequence τ consisting of Type II bidders, we have $E_{COCA}(\tau) \geq E_{VCG}(\tau)/(1 + g_2)$, where $g_2 = max_{\{x \in [P^{-1}(\underline{p}),1]\}}(P(x)/\int_0^x P(u)du)$.*[7]

6. This can be proved in the same way as we did in Theorem 3.

7. As P is not necessarily strictly increasing, here we simply denote $P^{-1}(x)$ as the maximal value y which satisfies $x = P(y)$.

Theorem 5 indicates that any sequence composed of Type II bidders has a competitive ratio of $1 + g_2$, where g_2 is directly determined by the auxiliary pricing function $P(x)$. Thus, we can minimize the competitive ratio by solving the following optimization problem:

$$\min_{\{P(x)\}} g_2 = max_{\{x \in [P^{-1}(\underline{p}),1]\}} P(x) / \int_0^x P(u)du$$
$$\text{s.t.} P^{-1}(\overline{p}) = P^{-1}(\underline{p}) + \int_{\underline{p}}^{\overline{p}} (P^{-1}(x))' dx \leq 1,$$

where the inequality constraint implicitly insures that the quantity of resource allocated at any time should be less than Q (i.e.,utilization rate less than one). To solve this, we use a similar technique in [12], [17], figuring out an auxiliary pricing function $P_1(x)$ such that the competitive ratio can be minimized.

Corollary 1. *For any request sequence τ consisting of Type II bidders, with auxiliary pricing function*

$$P_1(x) = \begin{cases} \overline{p}/e^{(1-x)\cdot r} & 1/r \leq x \leq 1, \\ \underline{p} & 0 \leq x < 1/r. \end{cases} \quad (10)$$

COCA is $(1 + r)$-competitive where $r = 1 + ln(\overline{p}/\underline{p})$.

4.1.2 Competitive Analysis for Type I Bidders

Achieving a good competitive ratio in the *general case* with no further constrains or assumptions is more difficult for Type I bidders. The reason is that a user of high unit valuation together with a very high penalty rate may be directly rejected if it is blocked by previous allocations. Instead, here we consider a common but less general case in which the resources haven't been fully utilized: we call it the *underload* case if when COCA is applied, for all t we have $U(t, \infty) \leq 1 - \overline{q}/Q$.[8]

Note that since the resource provider always has a very large resource capacity Q, $1 - \overline{q}/Q$ will be very close to 1, accordingly the underload case defined above is very likely to happen if there are not so many bidders with high unit valuation asking for a same time period. The following theorem shows that for Type I bidders, we can achieve a competitive ratio comparable to that of Type II bidders in such underload case.

Theorem 6. *For any request sequence τ consisting of Type I bidders, we have $E_{COCA}(\tau) \geq E_{VCG}(\tau)/(1 + g_1)$ in the underload case, where $g_1 = max_{\{x \in [P^{-1}(\underline{p}),1]\}}(P(min(1, x + \overline{q}/Q))/\int_0^x P(u)du)$.*

Observe that g_1 is quite similar to g_2. Then according to Theorem 6, in the following corollary we show that auxiliary function $P_1(x)$ can also be applied to achieve a good competitive ratio for Type I bidders in the underload case.

Corollary 2. *For any request sequence τ consisting of Type I bidders, with auxiliary pricing function $P_1(x)$, COCA is $(1 + e \cdot r)$-competitive in the underload case if $\overline{q} \leq Q/r$, where $r = 1 + ln(\overline{p}/\underline{p})$.*

8. Note that in Section 2.2 we make the assumption that $\gamma_i(t)$ is within the range $[0, \overline{q}]$.

4.1.3 Competitive Analysis for Type III Bidders

In the following analysis for Type III bidders, we show the relation between the competitive ratio and the choice of $P(x)$ in both the general case and the underload case.

Theorem 7. *For any request sequence τ consisting of Type III bidders, $E_{COCA}(\tau) \geq E_{VCG}(\tau)/(1 + g_1)$ holds in the underload case. And for the general case, we have $E_{COCA}(\tau) \geq E_{VCG}(\tau)/(1 + g_1 + g_3)$, where $g_3 = \overline{l} \cdot \overline{p}/\underline{l} \cdot \int_0^{(1-\overline{q}/Q)/2} P(u)du$.*

Theorem 7 indicates that the competitive ratio has a more complex form in the general case, where both g_1 and g_3 have to be considered in order to achieve a good competitive ratio. The following corollary shows that we can obtain a non-trivial competitive ratio for both the underload case and the general case, by using auxiliary function $P_1(x)$ and $P_3(x)$ respectively.

Corollary 3. *For any request sequence τ consisting of bidders of Request Type III, with auxiliary pricing function $P_1(x)$, COCA is $(1 + e \cdot r_1)$-competitive in the underload case if $\overline{q} \leq Q/r_1$, where $r_1 = 1 + ln(\overline{p}/\underline{p})$. And for the general case, with auxiliary pricing function*

$$P_3(x) = \begin{cases} (\overline{p} \cdot \overline{l}/\underline{l})/e^{(0.5\cdot(1+\overline{q}/Q)-x)\cdot r_3} & 1/r_3 \leq x \leq 1 \\ \underline{p} & 0 \leq x < 1/r_3. \end{cases} \quad (11)$$

COCA is $(1 + 2e \cdot r_3)$-competitive if $\overline{q} \leq Q/r_3$, where $r_3 = (1 + ln(\overline{p} \cdot \overline{l}/\underline{p} \cdot \underline{l}))/(0.5 \cdot (1 + \overline{q}/Q))$.

4.2 Competitive Analysis for the Mixture Arrival Case

In Section 4.1 we assume users in a request sequence to be of a *single type*. Since we have three request types, *what competitive ratio can we achieve if bidders of different types come in a mixed manner?* To answer this question, we conduct competitive analysis for the mixture arrival case as follows.

Theorem 8. *For any request sequence τ consisting of bidders of Type I, II, and III, $E_{COCA}(\tau) \geq E_{VCG}(\tau)/(3 + 2 \cdot g_1 + g_2)$ holds in the underload case.*

Theorem 8 tells us how the competitive ratio in the mixture case is determined by the competitive ratio under the scenario where only a single bidder type is considered. Finally, according to Theorem 5 to Theorem 8, a non-trivial bound on competitive ratio is given in the following proposition.

Corollary 4. *For any request sequence τ consisting of bidders of Request Type I, II and III, with auxiliary pricing function $P_1(x)$, COCA is $O(log(\overline{p}/\underline{p}))$-competitive in the underload case as long as $\overline{q} \leq Q/ln(\overline{p}/\underline{p})$.*

Tightness of the achieved competitive ratio. It is worth mentioning that all the three request types can be reduced to a simple case in [17],[9] where the competitive ratio on social welfare is proven to be bounded by $O(log(\overline{p}/\underline{p}))$ for any truthful online auction. This result indicates the tightness of

9. More specifically, Type II and III can be reduced into the "one divisible good" case, and Type I can be reduced into the "k indivisible goods" case.

the achieved competitive ratio in an asymptotic sense, and it also applies to the mixed arrival case.

5 Discussions on Other Desired Properties

In this section we discuss about some other important factors concerned in online auction design. And we show that COCA performs well when these factors are taken into consideration.

5.1 Computational Complexity

Note that COCA presented in Algorithm 1 only depends on current information, and it is without any backward iteration. Recall that in the above analysis we consider a continuous time period from 0 to ∞ for simplicity. However in practice, a very loose restriction on the current available time period $[0, T]$ (e.g., one day or one week from now), and a very small time granularity s (e.g., one second or one minute) are utilized. In this case COCA can be quite computationally efficient: Since $P(x)$ is nondecreasing, numeral solutionis of all the optimization problems in the allocation rule can be efficiently obtained. More intuitively, if we consider discrete resource with capacity $|Q|$, COCA achieves a computational complexity of $O(|Q||L|^2 log(|Q|))$ where $|L| = T/s$.

Here we prove this conclusion for Type III users, and the proof for Type I and Type II users can be done in a similar way: Remind that type III users require time-invariant allocation capacity with a fixed length l_i. As a consequence, the allocation γ_i for any type III users must be rectangular shaped (like the allocation A in Fig. 3), and thus can be characterized by $t^-_{\gamma_i}$(the starting time of allocation) and inv_cap_i (the bandwidth it receives). Then to find the exact allocation for any type III users, we first substitute γ_i by $t^-_{\gamma_i}$ and inv_cap_i in the payment function of COCA (step 1). Recall that we have $v_i(\gamma_i) = b_i(inv\_cap_i) \cdot l_i$, where $inv\_cap_i = \gamma_i(t)$ for all $t \in [t^-_{\gamma_i}, t^+_{\gamma_i}]$, accordingly we have

$$p_i(inv\_cap_i, t^-_{\gamma_i}, t_{sub_i}) = \int_{t^-_{\gamma_i}}^{t^-_{\gamma_i}+l_i} \int_{U(t,t_{sub_i})}^{U(t,t_{sub_i})+inv\_cap_i/Q} P(x) \cdot Q dx dt. \tag{12}$$

Then replace γ_i by $t^-_{\gamma_i}$ and inv_cap_i in Eq. (8) we get

$$\begin{aligned} &< inv\_cap_i^*, t^{-*}_{\gamma_i} >= argmax_{<inv\_cap_i, t^-_{\gamma_i}>}(b_i(inv\_cap_i) \cdot l_i - \\ &\int_{t^-_{\gamma_i}}^{t^-_{\gamma_i}+l_i} \int_{U(t,t_{sub_i})}^{U(t,t_{sub_i})+inv\_cap_i/Q} P(x) \cdot Q dx dt) \\ &\text{s.t. } \forall t \in [t^-_{\gamma_i}, t^-_{\gamma_i} + l_i], U(t, t_{sub_i}) + inv\_cap_i/Q \leq 1. \end{aligned} \tag{13}$$

Note that $b_i(inv\_cap_i)$ is concave and non-decreasing, and $P(x)$ is convex and non-deceasing. Thus for any fixed $t^-_{\gamma_i}$, it takes at most $O(log(|Q|))$ iterations to find the optimal $inv\_cap_i^*$ by bisection method [2] in Eq. (13), and each iteration takes $O(|Q||L|)$ operations. Repeating this procedure for all possible choices of $t^-_{\gamma_i}$ results in a total complexity of $O(|Q||L|^2 log(|Q|))$.

5.2 Retry, Re-Sell and Collusion

Truthful mechanisms always make certain assumptions on the requests submitted, e.g., it is widely assumed that each request comes from an independent agent who offers one and only one bid to the auction [6], and collusion is usually not considered. Whereas such assumptions are not realistic in the cloud resource allocation problem. Retry, re-sell and collusion are commonly concerned factors that may prevent theoretically truthful mechanisms from being applied in real markets. In the following context, We briefly discuss about these factors and then show the performance of COCA when these factors are taken into consideration.

Retry. If the payment for a certain allocation fluctuates over time, it might be beneficial for a bidder to submit an untruthful request $\mathbf{r_i}$ and try it for many times. However, remind that the auxiliary pricing function is monotonically increasing, so the payment for a certain allocation will never decrease as t_{sub_i} increases when COCA is applied. As such, according to the allocation rule, it can be derived that a repeated request with the same bidding parameters (except for a later t_{sub_i}) will never result in a better utility gain. Therefore, *COCA is resilient against retries.*

Re-sell. Re-sellers refer to bidders who buy in resources at a low price and sell them out when the price goes higher, As an ubiquitous market phenomenon, such re-selling behaviors are regulated, rather than prohibited when COCA is applied. More specifically, it is reasonable to consider a reseller's valuation on the resources as the estimated price at which the resources can later be sold out. Then according to Theorem 3, it is easy to reach to the conclusion that: *If COCA is applied, re-sellers will also maximize their utility by truthfully revealing their valuation on the resource.*Accordingly, the re-selling behaviors can also be well regulated.

Collusion. Collusion refers to a group of bidders who agree to cheat on their valuations to get some extra utility gain [16]. Lots of works have focused on this problem [10], whereas in truthful auction design, such a problem has always been ignored. Strictly speaking, COCA is not collusion-proof—after bidders exchanging their information, the overall utility gain may increase as they may negotiate and reach to a new bidding sequence where the overall utility gain can be optimized. However, note that COCA introduces a predetermined auxiliary pricing function $P(x)$, and it can be derived from Algorithm 1 that the payment for any certain allocation γ_i is uniquely determined if the arrival time t_{sub_i} is fixed. *Such design ensures that no collusion can decrease the total payment for a certain allocation.* Further, it can be clearly verified that such utility gain (due to information exchange among bidders) is essentially achieved by avoiding (minimizing) the conflict among users, rather than reducing the overall payment.

6 Simulation Results

In this section we propose simple simulations to evaluate COCA under illustrative bid distributions and arrival models. We focus on examining the allocation performance of COCA on social welfare compared with the off-line VCG mechanism. We haven't compared COCA with existing online auction mechanisms because no prior solutions have achieved the generalized truthfulness in our online cloud auction setting.

We consider a cloud resource provider of a fixed capacity $Q = 10^4$ (i.e., the provider is able to host up to 10^4 VMs simultaneously), and here a simple simulation

TABLE 1
Implementation Configuration

t_{sub_i}	a_i	d_i	$\overline{q}$	β
$[1, 500]$	$[t_{sub_i}, min(t_{sub_i}+100, 500)]$	$[a_i, 500]$	100	$\overline{p}/\underline{p}$

Type	l_i	$size_i$	$b_i(\cdot)$
I	-	$[10^3, 10^5]$	$b\_total_i = \rho_i \cdot size_i; \rho_i \in [1, \beta]$
II	-	-	$b_i(total\_rsc_i) = \rho_i \cdot total\_rsc_i$
III	$[5, 200]$	-	$b_i(inv\_cap_i) = \rho_i \cdot inv\_cap_i \cdot l_i$

model is used: the bidding parameters are assumed to be uniformly distributed (detailed settings refer to Table 1), and the (marginal) unit valuation is a fixed number $\rho \in [1, \beta]$, where $\beta = \overline{p}/\underline{p}$, which refers to the ratio between the highest and lowest unit valuation (mentioned in the assumptions in Section 2). Moreover, we assume a penalty rate of ∞ for Type I bidders.

In our simulation, we create dynamic arrival scenarios for all three valuation types separately, as well as the mixed arrival case where each type contributes to around one third of the population. Each scenario is simulated for 3,000 runs, and each run lasts for 500 time units. In particular, the number of requests generated under each scenario is uniformly distributed from 100 to 2,000, and the request arrival time also follows a uniform distribution. We apply both COCA and VCG in all the four scenarios and compare the achieved social welfare.

Fig. 7 shows the allocation performance of COCA on social welfare compared with the optimal solution (VCG mechanism) in both worst case and average case. As is discussed in Section 4, we use auxiliary pricing function $P_1(x)$ for Type I and Type II users. It can be observed in Figs. 7a and 7b that the worst performance in 3,000 runs is always better than the performance lower bound (1 over the competitive ratio) calculated in Section IV-C, and the average performance is always over 50 percent compared with the optimal allocation. In Fig. 7c, we show the allocation performance for Type III users when P_1 and P_3 are used respectively. It can be observed that P_1 outperforms P_3 in both average performance and worst performance. Briefly speaking, the reason is that there exist some extreme "bad" cases for Request Type III when we analyze the performance lower bound shown in Fig. 7c. To achieve a better lower bound, P_3 has to be constructed in such a way where the allocation performance in most cases becomes less satisfactory. However the possibility that the extreme case appears is too small, so it hardly happens in 3,000 runs under our simulation model. In Fig. 7d, we plot the performance of COCA for the mixture arrival case with auxiliary pricing function $P_1(x)$. It is observed that the performance is very similar to the single-arrival case shown above.

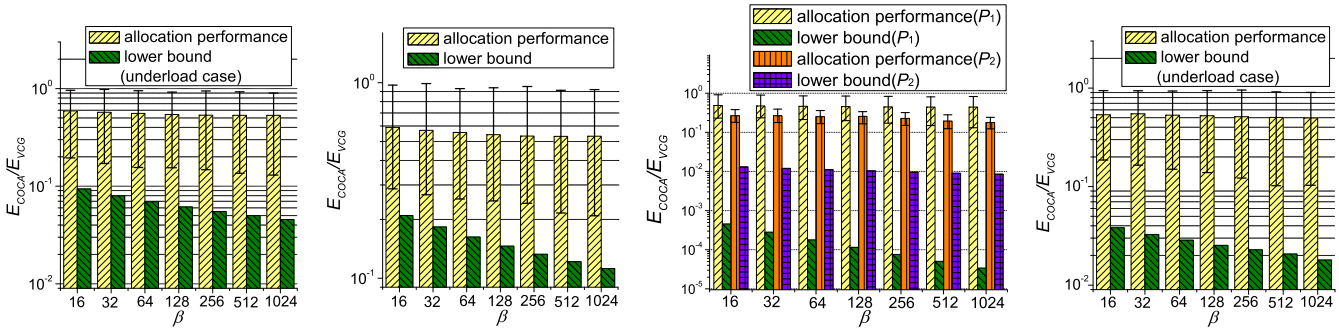
Overall, from the simulation results we can conclude that: *first*, it is clearly observed that the ratios of COCA over VCG in terms of social welfare are quite close to 1 in all cases, indicating that COCA is comparable to the off-line VCG mechanism (i.e., the optimal solution) under our simulation model; *second*, with $\beta = \overline{p}/\underline{p}$ increases exponentially, the worst performance of COCA (in 3,000 runs) decreases very slowly in all cases. Such results are in good agreement with the theoretical analysis in Section 4, that COCA achieves a competitive ratio of $O(log(\overline{p}/\underline{p}))$ rather than $O(\overline{p}/\underline{p})$.

7 Conclusion

This paper conducts the first work on truthful online auction design in cloud computing where users with heterogeneous demands could come and leave on the fly. First, for cloud consumers with heterogeneous demands we propose a novel bidding language, by which user-specific demands can be revealed in a concise and regulated request form. Second we propose the first truthful online cloud auction mechanism, COCA, which is based on a payment rule and an allocation rule which serve as the necessary and sufficient conditions for ensuring truthfulness. We also implement competitive analysis on COCA in terms of social welfare, which shows that the worst-case performance of COCA can be well-bounded. Then our further discussion shows that COCA performs well when some other important factors in online auction design are taken into consideration. Finally, in simulations the performance of COCA is seen to be comparable to the well-known off-line VCG mechanism.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61202460, Grant 61271226, and Grant 61272410; by the National Natural Science Foundation of Hubei Province under Grant 2014CFA040; and by the Fundamental Research Funds for the Central Universities under Grant 2015QN073. Dr. Bo Li's work was supported in part by a grant from NSFC/RGC under the contract HKUST610/11, by grants from HKUST under the contract RPC11EG29, SRFI11EG17-C and SBI09/10.EG01-C, by a grant from Guangdong Bureau of Science



(a) Type I Bidders (b) Type II Bidders (c) Type III Bidders (d) The mixture arrival case

Fig. 7. Allocation performance of COCA compared with VCG mechanism (with fifth and 95th percentiles).

and Technology under the contract GDST11EG06, by a grant from China Cache Corp. under the contract CCNT12EG01. The corresponding author of this paper is Hongbo Jiang. An earlier version of this work appeared as [29].

References

[1] AmazonEC2SpotInstances [Online]. Available: http://aws.amazon.com/ec2/spot-instances/, 2013.
[2] Bisection method [Online]. Available: http://en.wikipedia.org/wiki/Bisection_method, 2010.
[3] V. Abhishek, I. Kash, and P. Key, "Fixed and market pricing for cloud services," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2012, pp. 157–162.
[4] E. Angel, E. Bampis, and F. Pascual,"Truthful algorithms for scheduling selfish tasks on parallel machines," *Theoretical Comput. Sci.*, vol. 369, no. 1, pp. 157–168, 2006.
[5] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "Above the clouds: A view of cloud computing," UC Berkeley, 2010.
[6] B. Awerbuch, Y. Azar, and A. Meyerson, "Reducing truth-telling online mechanisms to online optimization," in *Proc. 35th Annu. ACM Symp. Theory Comput.*, 2003, pp. 503–510.
[7] Z. Bar-Yossef, K. Hildrum, and F. Wu, "Incentive-compatible online auctions for digital goods," in *Proc. 13th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2002, pp. 964–970.
[8] W. Baumol, "On taxation and the control of externalities," *Am. Econ. Rev.*, vol. 62, no. 3, pp. 307–322, 1972.
[9] R. Buyya, C. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities," in *Proc. 10th IEEE Int. Conf. High Perform. Comput. Commun.*, 2008, pp. 5–13.
[10] Y. Che and J. Kim, "Optimal collusion-proof auctions," *J. Econ. Theory*, vol. 144, no. 2, pp. 565–603, 2009.
[11] L. Deek, X. Zhou, K. Almeroth, and H. Zheng, "To preempt or not: Tackling bid and time-based cheating in online spectrum auctions," in *Proc. IEEE INFOCOM*, 2011, pp. 2219–2227.
[12] R. El-Yaniv, A. Fiat, R. Karp, and G. Turpin, "Competitive analysis of financial games," in *Proc. 33rd Annu. Symp. Found. Comput. Sci.*, 1992, pp. 327–333.
[13] J. Goossens, P. Richard, and P. Richard, "Socially optimal pricing of cloud computing resources," in *Proc. Eur. Workshop Project Manag. Scheduling*, 2004, pp. 322–331.
[14] M. Hajiaghayi, "Online auctions with re-usable goods," in *Proc. 6th ACM Conf. Electron. Commerce*, 2005, pp. 165–174.
[15] M. Hajiaghayi, R. Kleinberg, and D. Parkes, "Adaptive limited-supply online auctions," in *Proc. 5th ACM Conf. Electron. Commerce*, 2004, pp. 71–80.
[16] P. Klemperer,"What really matters in auction design," *J. Econ. Perspectives*, vol. 16, no. 1, pp. 169–189, 2002.
[17] R. Lavi and N. Nisan,"Competitive analysis of incentive compatible on-line auctions," *Theoretical Comput. Sci.*, vol. 310, no. 1, pp. 159–180, 2004.
[18] R. Lavi and N. Nisan, "Online ascending auctions for gradually expiring items," in *Proc. 16th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2005, pp. 1146–1155.
[19] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, *Algorithmic Game Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2007.
[20] R. Porter, "Mechanism design for online real-time scheduling," in *Proc. 5th ACM Conf. Electron. Commerce*, 2004, pp. 61–70.
[21] W. Shi, C. Wu, and Z. Li, "RSMOA: A revenue and social welfare maximizing online auction for dynamic cloud resource provisioning," in *Proc. Int. Workshop Quality Service*, 2014, pp. 1–10.
[22] W. Shi, L. Zhang, C. Wu, Z. Li, and F. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," in *Proc. ACM Int. Conf. Meas. Model. Comput. Syst.*, 2014, pp. 71–83.
[23] Q. Wang, K. Ren, and X. Meng, "When cloud meets ebay: Towards effective pricing for cloud computing," in *Proc. IEEE INFOCOM*, 2012, pp. 936–944.
[24] W. Wang, B. Li, and B. Liang, "Towards optimal capacity segmentation with hybrid cloud pricing," in *Proc. IEEE 32nd Int. Conf. Distrib. Comput. Syst.*, 2012, pp. 425–434.
[25] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron, "Better never than late: Meeting deadlines in datacenter networks," *SIGCOMM-Comput. Commun. Rev.*, vol. 41, no. 4, p. 50, 2011.
[26] P. Xu and X. Li, "TOFU: Semi-truthful online frequency allocation mechanism for wireless networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 2, pp. 433–446, Apr. 2011.
[27] P. Xu, S. Wang, and X. Li, "SALSA: Strategyproof online spectrum admissions for wireless networks," *IEEE Trans. Comput.*, vol. 59, no. 12, pp. 1691–1702, Dec. 2010.
[28] C. Yeo and R. Buyya, "Service level agreement based allocation of cluster resources: Handling penalty to enhance utility," in *Proc. IEEE Int. Cluster Comput.*, 2005, pp. 1–10.
[29] H. Zhang, B. Li, H. Jiang, F. Liu, A. Vasilakos, and J. Liu, "A framework for truthful online auctions in cloud computing with heterogeneous user demands," in *Proc. IEEE Conf. Computer Commun.*, 2013, pp. 1510–1518.
[30] L. Zhang, Z. Li, and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach." in *Proc. IEEE INFOCOM*, 2014, pp. 433–441.
[31] Q. Zhang, E. Grses, R. Boutaba, and J. Xiao, "Dynamic resource allocation for spot markets in clouds," in *Proc. 11th USENIX Conf. Hot Topics Manag. Internet, Cloud, Enterprise Netw. Services*, 2011, p. 1.

Hong Zhang received the BS degree from the Huazhong University of Science and Technology, China, in 2010. He is currently working toward the MS degree in the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, China. His research interests include algorithms in wireless communication and cloud computing, especially game theoretic-based mechanism design.

Hongbo Jiang received the BS and MS degrees from Huazhong University of Science and Technology, China. He received the PhD degree from Case Western Reserve University in 2008. After that he joined the faculty of Huazhong University of Science and Technology, where he is currently a full professor and the department dean. His research interests include computer networking, especially algorithms and architectures for wireless networks. He is a senior member of the IEEE.

Bo Li received the BEng degree in the computer science from Tsinghua University, Beijing, and the PhD degree in the electrical and computer engineering from the University of Massachusetts at Amherst. He is a professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. He is a Cheung Kong chair professor in Shanghai Jiao Tong University, China, and the chief technical advisor in China Cache Corp. (NASDAQ:CCIH). He was previously with IBM Networking System Division, Research Triangle Park, and an adjunct researcher at Microsoft Research Asia. His recent interests include: large-scale content distribution in the Internet, Peer-to-Peer media streaming, the Internet topology, cloud computing, green computing and communications. He is a fellow of the IEEE.

Fangming Liu received the BEngr degree in 2005 from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, and the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology in 2011. He is currently an associate professor in the Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. From August 2009 to February 2010, he was a visiting student at the Computer Engineering Group, Department of Electrical and Computer Engineering, University of Toronto, Canada. His research interests are in the area of peer-to-peer networks, rich-media distribution, cloud computing and large-scale datacenter networking. He is a member of the IEEE and IEEE Communications Society. He is a member of the IEEE.

Athanasios V. Vasilakos received the BS degree in electrical and computer engineering from the University of Thrace, Xanthi, Greece, in 1983, the MS degree in computer engineering from the University of Massachusetts, Amherst, in 1986, and the PhD degree in computer engineering from the University of Patras, Patras, Greece, in 1988. He is currently a visiting professor with the National Technical University of Athens (NTUA), Athens, Greece. He has authored or coauthored more than 200 technical papers in international journals and conferences. He is the author/coauthor of five books and 20 book chapters in the areas of communications. He is the general chair of the Council of Computing and Communications of the European Alliances for Innovation.

Jiangchuan Liu received the BEng degree from Tsinghua University, China, in 1999, and the PhD degree from The Hong Kong University of Science and Technology in 2003. He co-received the Best Student Paper Award of IWQoS2008 and the Multimedia Communications Best Paper Award from the IEEE Communications Society. He is currently an associate professor at Simon Fraser University, British Columbia, Canada, and was an assistant professor at The Chinese University of Hong Kong from 2003 to 2004. His research interests include cloud computing, peer-to-peer systems, multimedia communications, and wireless networking. He is an associate editor of the *IEEE Transactions on Multimedia*, and an editor of the *IEEE Communications Surveys and Tutorials*. He is a senior member of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.