A False-name-proof Combinatorial Auction for Resource Allocation in Cloud Computing

Genzhou Zhang\textsuperscript{1, a}, Yongzhu Zhao\textsuperscript{1, b}, Yuntao Guo\textsuperscript{2, c}, Tao Fang\textsuperscript{1, d}, Xiaolong Ren\textsuperscript{1, e}, Xiaoming Wang\textsuperscript{2, f} and Lin Wang\textsuperscript{3, g}

\textsuperscript{1}State Grid Shaanxi Electric Power Company, China
\textsuperscript{2}State Grid Information&Telecommunication Company of Shaanxi Electric Power Corporation, Xi’an, China
\textsuperscript{3}Nari Group Corporation/State Grid Electric Power Research Institute, Xi’an, China
\textsuperscript{a}zhanggenzhou@sn.sgcc.com.cn, \textsuperscript{b}zhaoyongzhu@sn.sgcc.com.cn, \textsuperscript{c}guoyuntao@xt.sn.sgcc.com.cn, \textsuperscript{d}fangtao@xt.sn.sgcc.com.cn, \textsuperscript{e}renxiaolong@sn.sgcc.com.cn, \textsuperscript{f}wangxiaoming@xt.sn.sgcc.com.cn, \textsuperscript{g}wanglin18@sgepri.sgcc.com.cn

Abstract. Cloud computing is an Internet-based computing and network model, which provides a market for sharing computing and storage resources. To motivate the cloud providers, the allocation of resources and their pricing becomes one of the challenges in this market. In this situation, designing auction-based mechanisms for cloud resource allocation have attracted a wide interest. Most of prior works focus on designing strategy-proof (a.k.a. truthful) cloud auction mechanisms to eliminate market manipulation. Strategy-proof auctions ensure that no bidders can improve their utilities by untruthful bidding. However, these strategy-proof auctions are fragile when false-name-bids are allowed. This is a new type of cheating where a bidder can gain profit by submitting bids using multiple fictitious names, and it is easy to form in cloud environment. Therefore, a new series of auction mechanisms are needed to guarantee the false-name-proofness. To tackle this issue, in this paper, we designed CACC, a combinatorial auction for resource allocation in Cloud Computing. To address the heterogeneity of cloud resource, the combinatorial auction model is adopted. We show CACC is both strategy-proof and false-name-proof. Moreover, CACC is computationally efficient. Through simulation experiments, we show that CACC is effective and efficient.

1. Introduction

Cloud computing is a new computing based business model where various resources such as CPU, Network, Storage, Memory etc. are offered as utility and are available on demand \cite{1} \cite{2}. The cloud service providers (e.g., Amazon \cite{3}, Google and Microsoft \cite{4}) can gain profit by using different pricing schemes \cite{2} when providing the resources. Cloud users want to use cloud services to execute their jobs or applications but by paying optimal price with desired QoS.

Auction is known as an efficient market-driven mechanism where resources can be efficiently distributed between sellers and buyers \cite{5}. As a result, auction-based mechanism for cloud resource
allocation has been well studied recently, e.g., [6], [11], [21], [25], [26]. When designing an auction mechanism, a critical property of strategy-proofness (a.k.a. truthfulness) is to be guaranteed. Strategy-proofness is essential to ensure auction fairness and resist market manipulation. In strategy-proof auctions, bidders cannot improve their utilities by untruthful bidding and thus bid their truth valuations. In this context, a number of strategy-proof auctions have been proposed, e.g., [6], [21], [25], [26].

However, these strategy-proof auctions suffer from a new type of cheating named as false-name bids where bidders can submit bids using multiple false names. This type of cheating is easy to form in various auctions running on Internet [13], [18], and it is hard to be detected. In this way, selfish bidders can manipulate the auction results by providing false-name bids, which will lead to untruthful bidding and unfair scarcity. Since cloud auctions are always run on Internet (e.g., Amazon [3], Google and Microsoft [4]), the same cheating is also easy to breed in cloud auctions. For instance, bidders can easily make multiple fictitious names just by registering multiple e-mail addresses. As a result, when designing the cloud auctions, we also need to resist false-name cheating while providing strategy-proofness. An effective way to solve this issue is to design a false-name-proof mechanism, where bidders are encouraged to bid truthfully using a single identifier. False-name-proof auctions ensure that profit gain via submitting false-name bids is impossible.

As discovered in [14], prior strategy-proof auction designs like [6], [21], [25], [26] failed to be false-name-proof. [14] is the pioneer work where a false-name-proof auction is designed for resource allocation in cloud computing market. In [14], to make the pricing scheme being feasible, the authors introduce the concept of instance weight to deal with VM instance heterogeneity. In other words, any type of VM instances can be normalized to one type via instance weight finally. However, in practical systems, the heterogeneity of VM instances is hard to be normalized. In this paper, we use the combinatorial auction model to tackle this issue and then proposed CACC, a false-name-proof Combinatorial Auction for resource allocation in Cloud Computing. Intuitively, to ensure the false-name-proofness, the price of buying a bundle of VM instances must be less than or equal to the sum of prices of buying these instances separately using multiple identifiers. To achieve this, CACC adopts the price-oriented rationing-free (PORF) protocol. In other words, CACC firstly determines the prices for bidders, and then determines the allocation. Moreover, the charged price for each bidder is ensured to be independent of the bidder itself. We show CACC is false-name-proof and computationally efficient, and simulation experiments verified the performance of CACC. The main contributions of this paper can be summarized as follows.

- We study the problem of resource allocation in cloud using a combinatorial auction model, and we then design CACC, a Combinatorial Auction for resource allocation in Cloud Computing.
- CACC not only achieves strategy-proofness but also resists the false-name bid cheating. Moreover, CACC is computationally efficient.
- Through simulation experiments, we show CACC is effective and efficient.

The remainder of the paper proceeds as follows: Section II introduces the preliminaries on cloud auctions. CACC is proposed in Section III. Experimental results are given in Section IV. Section V concludes this paper.

2. Preliminaries

In this section, we give out the system model and the problem formulation. Following that, we introduce the objectives to design efficient, economic-robust cloud auction mechanisms.

2.1. System Model

For easy description, we denote \( [0, 1, \ldots, n] \) by \( |n| \) and \( [1, 2, \ldots, n] \) by \( |n|+ \) for a nonnegative integer \( n \).

We consider a typical scenario where one cloud provider auction a set of VM instances to multiple users. Let \( M \) be the number of types of different VM configurations (types), denoted as \( \text{VM}_1, \text{VM}_2, \ldots, \text{VM}_M \). Each type corresponds to a certain number and speed of CPUs, memories, etc., and each type \( \text{VM}_m \) has \( k_m \) copies for allocation.
There are \( N \) cloud users (the bidders) participating the auction, and let \([n]^+\) be the set of bidders. Each bidder \( i \) requests a bundle of VM instances denoted by a vector \( d_i = (d_{i1}, d_{i2}, \ldots , d_{im}) \), where \( d_{im} \) represents the required number of VMs. For bidder \( i \), let \( v_i \) be the valuation of bidder \( i \) for the required bundle of instances. Bidders are always selfish, and thus when bidder \( i \) submits bid \( b_i \) for bundle \( d_i \), denoted by tuple \( (b_i, d_i) \), \( b_i \) does not have to be equal to the true valuation \( v_i \).

After the allocation of VM instances, the price \( p_i(b_i, b_{-i}) \) is charged from each bidder \( i \), and we define \( p_i(b_i, b_{-i}) = 0 \) if bidder \( i \) loses. Here \( b_{-i} \) denotes the vector including all the bids from \( b_1 \) to \( b_N \) except \( b_i \). We simplify the notations \( v_i(d_i), p_i(b_i, b_{-i}) \) and \( u_i(b_i, b_{-i}) \) as \( v_i, p_i \) and \( u_i \), respectively, if no confusion is incurred.

### 2.2. Problem Formulation

In auction design, we study the case that the bidders are single-minded [24], i.e., they can only accept allocations of either the whole requested bundle (or any superset) or nothing. Note that this is the assumption for most existing work on cloud auctions (e.g., [25], [26]).

Since the aim of this work is to benefit both users and providers, the goal of the resource allocation problem is to maximize social welfare, which is defined as the sum of bids submitted by all winners. Social welfare is a strong indicator of how efficiently the buyers make use of the sold VM instances. We use the following binary variable \( x_i \) to formally describe the allocation problem. If \( x_i = 1 \), which indicates bidder \( i \) is a winner. Otherwise, \( x_i = 0 \).

The VM instance allocation to achieve optimal social welfare can thus be formulated as an Integer Programming (IP) problem, denoted as \( IP([N]^+) \):

\[
W = \max \sum_{i=1}^{N} b_i x_i \quad \text{s.t.} \quad \sum_{i=1}^{N} d_{im} x_i \leq k_m, \quad \forall m \in [M]^+ \\
x_i \in \{0, 1\}, \quad \forall i \in [N]^+ 
\]

In the above formulation, the social welfare, i.e., Eq. (1) is the objective to be maximized. Constraint Eq. (2) specifies that the sum of allocated VM instances is no more than the total number of provided VM instances. Eq. (3) depicts that the decision variable \( x_i \), where \( x_i \) equals 1 if user \( i \) wins and 0 otherwise.

### 2.3. Design Targets

When designing the auction mechanism, some critical economic properties are to be satisfied. In this paper, we aim to design the auction mechanism satisfying the following three properties.

1. **Strategy-proofness** [16] [17] An auction mechanism is strategy-proof (or truthful) if for any bidder \( i \), \( u_i(v_i, b_{-i}) \geq u_i(b_i, b_{-i}) \) holds under any given \( b_{-i} \).

2. **False-name-proofness** [18] [19] An auction mechanism is false-name-proof if for any bidder \( i \) participating the auction using \( k \) false names \( i_1, \ldots , i_k \), the following equation holds under any given \( b_{-i} \):

\[
u_i(v_i, b_{-i}) \geq \sum_{j=1}^{k} u_{i_j}(b_{i_j}, b_{-i} \cup \{j\})
\]

Where \( I_{-j} = \{b_{i_l} : l \in [k]^+, l \neq j\} \). Strategy-proofness (a.k.a. truthfulness) nullifies improved utility from cheating on bid valuation, while false-name-proofness from cheating on both false-name bids and bid valuation. It is worth mentioning that, false-name-proofness generalizes the concept of strategy-
proofness. In other words, false-name-proofness is a sufficient but in general not a necessary condition of strategy-proofness.

(3) Computational Efficiency. Conventionally, the auction mechanism is required to be run in a polynomial time complexity.

In practical systems, computational efficiency is also an important property to be satisfied. However, designing the above combinatorial auction mechanism with the goal of maximizing social welfare (Eq. 2) is a NP-hard problem [23]. Since the economic properties are critical to implement the auction, we focus on designing the combinatorial auction mechanism satisfying the economic properties primarily while maximizing efficiency approximately. As a conventional approach, this has also been applied in other auction designs [21], [22].

3. Our Auction Design
In this section, we give out the main algorithms of our auction design named as CACC.

3.1. Main Algorithms
CACC consists of a pricing algorithm to compute the payment for each bidder and an allocation method to determine winners. CACC conduct a price-oriented rationing-free (PORF) strategy which is critical to ensure the false-name-proofness. In other words, CACC first decides the price for each bidder, and then determines the winners according to the finely computed prices. This is different from a traditional auction mechanism.

(1) Computing Prices. To ensure the computational efficiency of CACC, we need approximately solve the problem of Eq. (2). We remove the integrality constraint and get the following linear program relaxation, denoted as $LP([N]^+)$:

$$W = \max \sum_{i=1}^{N} p_i x_i \text{ s.t.}$$

$$\sum_{i=1}^{N} d_{m,i} x_i \leq k_m, \quad \forall m \in [M]^+,$$

$$x_i \in [0, 1], \quad \forall i \in [N]^+. \tag{7}$$

After getting the LP solution, we ignore the bidders whose LP solutions are less than 1. These bidders will lose in the allocation procedure, and thus cannot be allocated. These bidders are included in the set $D$. And we compute the prices for those with LP solution equaling to 1. For each bidder $i$ with $x_i = 1$, we resolve the linear program relaxation problem again with the bidders except bidder $i$, i.e., $LP([N]^+ \setminus i)$. Then we check the bidders in $C$ one by one, if the bidder $j \in D$ and $x_j = 1$ in solution of $LP([N]^+ \setminus i)$. Then $j$ is included in set $C$. This set is called the conflicting set for bidder $i$. The price for bidder $i$ is the highest bid of these bidders in $C$, i.e., $p_i = \max_{j \in C} b_j$. The detailed algorithm is described in Algorithm 1.

(2) Determine Winners. Based on the computed price of each bidder, Algorithm 2 sequentially checks each bidder in $[N]^+$. If the computed price is positive and the utility of bidder $i$ is positive, function $Assign(i, d_i)$ assigns the bundle $d_i$ to bidder $i$.S. Otherwise, bidder $i$ loses with no charge, and it utility is set to 0. This is because we need to ensure that no bidders will be charged more than its bid valuation, to incentivize bidders to participate the auction.
Algorithm 1: CACC-Pricing Algorithm.

Input:
The bidders, $[N]^+$;
The bid vector, $b$;
The demand vector, $d$;
The resource capacity, $K$;

Output:
The prices for bidders;
1. Solve $LP([N]^+)$;
2. IF $x_i < 1$
3. \[ D \leftarrow i; p_i = -1; \]
4. IF $x_i == 1$
5. Solve $LP([N]^+ \setminus i)$;
6. for each $j \in D$
7. {IF $x_j == 1$
\[ C \leftarrow j; \]}
8. $p_i = \max_{j \in \{b:j\}}$;

Algorithm 2: CACC-Allocation Algorithm.

Input:
The bidders, $[N]^+$;
The bid vector, $b$;
The demand vector, $d$;
The resource capacity, $K$;
The computed prices, $P$;

Output:
Determine winners and allocation;
1. For each bidder $i \in [N]^+$
2. IF $(p_i > 0) \& \& (v_i - p_i \geq 0)$
3. Assign($i$, $d_i$);
4. $u_i = v_i - p_i$
5. ELSE
6. $u_i = 0$;

3.2. Auction Properties

Now we prove CACC is false-name proof. We first characterize the auction by establishing some lemmas. According to the theorem proved in [20], we know that a price-oriented rationing-free (PORF) mechanism with the weakly-anonymous pricing rule (WAP) satisfies the no super-additive price increase (NSA) condition, then it is false-name proof.

Lemma 1: CACC is a price-oriented rationing-free (PORF) mechanism.

Proof: According to the definition of PORF in [20], we know CACC firstly determines the prices for each bidder $i$ and then determines the allocation for winners. In addition, the price for bidder $i$ is independent of $i$’s declared bid since bidder $i$ is excluded when computing its price. As a result, CACC is price-oriented. Secondly, since each bidder $i$ is single-minded, the allocated bundle $d_i$ is the optimal allocation which maximizes $i$’s utility. In summary, CACC is a PORF mechanism.

Lemma 2: CACC satisfies weakly-anonymous pricing (WAP) rule.

Proof: We know the price of bidder $i$ is independent of bidder $i$’s bid valuation, but is dependent of the bids of other bidders. Therefore, the price of bidder $i$ can be described as a function of bids of other bidders, i.e., the price can be described as $p(d_i, \Theta_X)$, where $X$ is the set of bidders except $i$, and $\Theta_X$ is the
set of bids of bidders in $X$. Then according to the definition of WAP in [20], we know CACC satisfies WAP rule.

**Lemma 3:** CACC satisfies no super-additive price increase (NSA) condition.

**Proof:** According to the definition of NSA in [20], we need to prove that: in pricing scheme, for any subset of bidders $S \subseteq [N]^*$ and $X = [N]^* \setminus S$, for $i \in S$, let us denote $B_i$ as a bundle that maximizes $i$'s utility, then $\sum_{i \in S} p(B_i \cup \{b_i\} \cup \Theta_X) \geq p(\cup_{i \in S} B_i \cup \Theta_X)$.

Now if bidder $i$ requesting bundle $B \cup B'$, then by the pricing scheme, bidder $j$ with request $B$ and bidder $j'$ with $B'$ will conflict with bidder $i$. Therefore, we get $p(B \cup B', \Theta_X) = \max(p(B, \Theta_X), p(B', \Theta_X))$. Therefore, we get:

$$p(B \cup B', \Theta_X) \leq p(B, \Theta_X) + p(B', \Theta_X)$$

Therefore, the following formula holds.

$$p(\cup_{i \in S} B_i, \Theta_X) = \max_{i \in S} p(B_i, \Theta_X) \leq \sum_{i \in S} p(B_i, \Theta_X)$$

Furthermore, in the pricing scheme, prices increase monotonically by adding bidders, i.e., for all $X \in X'$, since more bidders will cause more conflicting bidders, we have $p(B, \Theta_X) \leq p(B, \Theta_X')$. Therefore, for each $i$, $p(B_i \cup \{b_i\} \cup \Theta_X) \geq p(B_i, \Theta_X)$, as a result, the NSA condition holds.

Based on the above lemmas, we obtain the main theorem.

**Theorem 1:** CACC is false-name-proof.

**Proof:** According to the Theorem 2 in [20], CACC is false-name-proof directly by the above three lemmas: lemma (1), lemma (2) and lemma (3).

**Computational Complexity:** Now we analyse the computational complexity of CACC. In pricing scheme of CACC, it takes at most $N + 1$ times of the time consumed by the LP process with bounded $N \times K$ parameters [15]. Note that we use Karmarkar's Algorithm to obtain the LP solution in this paper. Karmarkar's Algorithm belongs to the interior point method, which generates a sequence of points inside the feasible region and finally approaches the optimal vertex [15] and its computational complexity is bounded by $O(N^4K^4)$. For the allocation algorithm, it takes only $O(N)$ to check each bidder $i$, if its utility is greater than 0, then its demand is satisfied. Therefore, the overall computational complexity is bounded by $N + 1$ times of the time consumed by the LP process.

4. **Performance Evolutions**

In this section, we conduct comprehensive simulations to evaluate the performance of the proposed auction mechanism.

4.1. **Simulation Setup**

We implement the mechanism in Windows with Intel Core i5-2520 CPU 2.5GHz using Matlab 2011b. The number of types of VM instance is set as $M = 4$. In other words, the cloud provider offers four types of VM instances: small, medium, large, and huge. For simplicity, the number of each type of VM instances to be auctioned is set to be equal. Each bidder's request for each type of VMs is set as a random integer number over the range $[0, d_{\text{max}}]$. The bids of users are generated randomly from the range $[0, b_{\text{max}}]$. We set $b_{\text{max}} = 50$ and $d_{\text{max}} = 10$ and $K = 300$ initially. To overcome the impact of randomness, the results are all averaged over 50 times of running.

We evaluate the performance of TDAP from different aspects, including false-name-proofness and system efficiency.
4.2. False-name-proofness

We first verify the truthfulness of CACC, by showing that no bidders can improve their utilities by bidding untruthfully. Therefore, we randomly choose one losing bidder and one winning bidder, and then examine how their utilities change with his bid. The results in Fig.1.a show that a bidder will obtain positive utility when bidding higher than its truth valuation, but cannot improve its utility by bidding untruthfully. Another losing buyer will get negative utility when bidding higher than its true valuation. In summary, no matter what other bids it takes, buyers cannot improve the utility. Similarly, we examine how their utilities changes when they bid via multiple names. The results in Fig.b still show that bidding via multiple names cannot improve the utility.

4.3. System Efficiency

We evaluate the system efficiency of CACC using the metric of social welfare. We compare CACC with the pioneer design FAITH [14]. The results are shown in Fig.2. From the results, we observe that: (1) when the number of bidders is small, they perform similarly. This is because the resource is abundant for both schemes and most of bidders will win. (2) When the number of bidders is great, CACC performs a little worse and the performance gap is limited, always less than 5%. Compared with FAITH, the biggest advantage of CACC is that no instance weight parameter is required. This parameter is hard to set in practical systems. CACC solved this issue and the sacrificed performance is limited.

Figure 1. The utility of a bidder in the auction

Figure 2. Compare CACC with FAITH in terms of social welfare.
5. Conclusion

In this paper, we study the problem of designing auction mechanism for resource allocation in cloud computing. We use a combinatorial auction to address the heterogeneity of VM instance. In this situation, designing a robust auction mechanism needs to solve NP-hard problem. Moreover, a new type of cheating named false-name bids needs to be addressed. Therefore, in this paper, we propose CACC, a false-name-proof Combinatorial Auction for resource allocation in Cloud Computing. CACC is shown to be false-name-proof and efficient.

Acknowledgments

This work was financially supported by the science and technology projects of headquarters of State Grid Corporation (Research and application of non-stop state Adaptive upgrade technology for information system).

References

[1] R. Buyya, “Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility,” IEEE/ACM International Symposium on Cluster Computing & the Grid, 2009.
[2] I.A.T. Hashem, I. Yaqoob, N.B. Anuar, S. Mokhtar, A. Gani, and S.U. Khan, “The rise of big data on cloud computing: Review and open research issues. Journal of Information Systems,” vol.47 (1), pp.98-115, 2015.
[3] AWS 2016. Amazon EC2 SpotInstances. URL:https://aws.amazon.com/cn/ec2/spot/.
[4] Microsoft 2016. Mocrosoft Azure. URL: https://azure.microsoft.com/en-in/.
[5] V. Krishna, “Auction Theory,” Academic Press, March 2002.
[6] L. Zhang, Z. Li, and C. Wu, “Dynamic resource provisioning in cloud computing: A randomized auction approach,” In Proc. of INFOCOM, IEEE, 2014.
[7] J. Zhao, H. Li, C.Wu, Z. Li, Z. Zhang, and F. Lau, “Dynamic pricing and profit maximization for the cloud with geo-distributed data centers,” In Proc. of INFOCOM. IEEE, 2014.
[8] G. Baranwal and D.P. Vidyarthi, “A fair multi-attribute combinatorial double auction model for resource allocation in cloud computing.” Journal of Systems and Software, vol.108, pp.60-75, 2015.
[9] P. Samimi, Y. Teimouri, and M. Mukhtar, “A combinatorial double auction resource allocation model in cloud computing.” Information Sciences, vol.357, pp.201-216, 2016.
[10] D. Kumar, G. Baranwal, Z. Raza, and D.P. Vidyarthi, “A systematic study of double auction mechanisms in cloud computing.” Journal of Systems and Software, vol.125, pp.234-255, 2017.
[11] S.A. Tafsiri and S. Yousefi. Combinatorial double auction-based resource allocation mechanism in cloud computing market. Journal of Systems and Software, 137: 322–334, 2018.
[12] M. Yokoo, Y. Sakurai, and S. Matsubara, “The effect of false-name declarations in mechanism design: Towards collective decision making on the internet,” In Proc. of ICDCS. IEEE, 2010.
[13] M. Yokoo, Y. Sakurai, and Y. Matsubara, “The effect of false-name bids in combinatorial auctions: New fraud in internet auctions,” Games and Economic Behavior, vol. 46 (1), pp.174-188, 2004.
[14] Q. Wang, B. Ye, B. Tang, S. Guo, and S. Lu, “False-name-proof auctions for cloud resource allocation,” In Proc. of ICDCS. IEEE, 2015.
[15] H. Karloff, “Linear programming.” Birkhauser, 1991.
[16] A. Mas-Colell, M. D. Whinston, and J. R. Green, “Microeconomic Theory,” Oxford Press, USA, 1995.
[17] M. J. Osborne and A. Rubenstein, “A Course in Game Theory,” MIT Press, USA, 1994.
[18] T. Todo, A. Iwasaki, M. Yokoo, and S. Sakurai, “Characterizing false-name-proof allocation rules in combinatorial auctions,” In Proc. of AAMAS, 2009.
[19] M. Yokoo, Y. Sakurai, and Y. Matsubara, “The effect of false-name bids in combinatorial
auctions: New fraud in internet auctions,” Games and Economic Behavior, 46 (1): 174-188, 2004.

[20] M. Yokoo, “The characterization of strategy/false-name proof combinatorial auction protocols: Price-oriented, rationing-free protocol,” In Proc. of IJCAI, 2003.

[21] D. Kumar, G. Baranwal, Z. Raza, and D.P. Vidyarthi, “A truthful combinatorial double auction-based marketplace mechanism for cloud computing,” Journal of Systems and Software, vol.140, pp.91-108, 2018.

[22] Q. Wang, B. Ye, B. Tang, T. Xu, S. Guo, S. Lu, W. Zhuang, “Robust Large-Scale Spectrum Auctions against False-name Bids,” IEEE Transactions on Mobile Computing, vol.16 (6), pp.1730-1743, 2017.

[23] M. H. Rothkopf, A. Pekec, and R. M. Harstad, “Computationally manageable combinatorial auctions,” Management Science, vol.44 (8), pp.1131-1147, 1998.

[24] D. Lehmann, L.I. O’Callaghan, and Y. Shoham, “Truth revelation in approximately efficient combinatorial auctions,” Journal of the ACM, vol.49 (5), pp.577-602, 2002.

[25] Q. Wang, K. Ren, and X. Meng, “When cloud meets ebay: Towards effective pricing for cloud computing,” In Proc. of IEEE INFOCOM, 2012.

[26] S. Zaman and D. Grosu, “Combinatorial auction-based allocation of virtual machine instances in clouds,” Journal of Parallel and Distributed Computing, vol.73 (4), pp.495-508, 2012.