A Two-Sided Market Mechanism for Trading Big Data Computing Commodities

Lena Mashayekhy, Mahyar Movahed Nejad, Daniel Grosu
Department of Computer Science
Wayne State University
Detroit, MI 48202, USA
{mlena, mahyar, dgrosu}@wayne.edu

***Abstract*—The *big data* trend is generating compute-intensive and data-intensive applications requiring unique services that are different from conventional computing services. Therefore, there is a need to fundamentally address such requirements by developing market mechanisms for managing, trading, and pricing big data computing services. The cloud computing platforms have a great potential to meet the economic requirements of market mechanisms for big data applications due to their technological advances, cost benefit ratios, and easy to use services. We design a two-sided mechanism for trading computing resources for big data applications. Our proposed mechanism is universally strategy-proof, providing incentives for both cloud providers and users to voluntarily reveal their true private information. We perform extensive experiments to evaluate our proposed mechanism.**

***Keywords*-cloud computing; big data; two-sided auction.**

I. Introduction

The big data trend is generating massive data sets, and its processing technologies are becoming more available to users due to advances in IT infrastructures such as virtualization of resources and access to fast broadband networks. However, big data processing requires unique services that are different from conventional computing services. Therefore, there is a need to fundamentally address such requirements by developing economics and market mechanisms for managing, trading, and pricing big data computing services. Cloud-based technologies have a great potential to meet the requirements of new compute-intensive and data-intensive applications. Therefore, cloud providers can gain a part of the big data market share by facilitating and supporting big data processing as a pay-as-you-go model. However, gaining such market share comes at the cost of architectural changes to the current cloud frameworks along with designing economic mechanisms for extending cloud services into big data pay-as-you-go services.

Currently, cloud providers offer Infrastructure-as-a-Service (IaaS) to users in the form of virtual machines with limited computing resources. However, such services are not specifically designed for the big data applications. MapReduce along with its popular open source implementation, Hadoop, is the platform of choice for deploying and executing the majority of big data applications. Hadoop typically requires a cluster to be run on, especially when it comes to big data analytics procedures, where the amount of required computing resources can dramatically increase during the execution. Therefore, users require entire clusters for their big data applications. Such demand necessitates the design and deployment of markets for big data services in which entire clusters are the tradable goods. While there exists many studies that make MapReduce processing more efficient from different perspectives (e.g., [1], [2]), there is no study on trading big data computing commodities such as clusters.

When cloud providers offer big data services individually, users have to compare different cloud services with different service descriptions along with different performance characteristics. Such burden to users is further increased once the users want to switch between different cloud providers which may require long negotiation processes. Therefore, there is a need for trading platforms for big data computing commodities that facilitate their trading and usage. With such platforms, cloud providers and users are able to meet at a marketplace to trade compatible and comparable computing resources. Such markets should be able to accept orders from cloud providers and users and determine the price for the clusters, execute the financial closing of trades, and deliver the big data services. For example, a new initiative by Deutsche Börse Cloud Exchange, will launch in 2014 a vendor-neutral marketplace for cloud resources. This market will be a platform for offering, buying and deploying IaaS. The reader is referred to [3] for more details of its structure.

Auctions have been proven to be effective market-based mechanisms for cloud services. Main stream cloud provider powerhouses such as Amazon have been offering cloud services in a one-sided auction market for several years. Market-driven cloud auctions offer an efficient exchange environment. If designed well (e.g., considering incentives of both sides), a cloud auction creates a market in which it attracts both cloud providers and users. Such auctions give equal opportunity to the participants, and select those with the highest values as winners. In addition, they have to satisfy some economics properties such as *strategy-proofness* (i.e., give incentives to the participants to reveal their true valuations for the requested resources).

In this paper, we design a strategy-proof two-sided auction mechanism that gives incentives to both users and cloud providers to voluntarily reveal their true private information.

Each participant is self-interested, and tries to maximize its utility. To promote the transactions and attract both users and cloud providers, we design a mechanism that maximizes the social welfare, i.e., the summation of the broker's payoff and each participant's utility. Our proposed mechanism offers benefits to both cloud users and cloud providers by deploying the big data services of an exchange to facilitate trading and usage of cloud resources. Our proposed mechanism enables two-sided markets providing additional sales channels to cloud providers and enabling them to reach new user groups with low customer acquisition cost leading to higher utilization, more efficiency in sales and production, and reducing the overall cost in offering services.

Our Contribution. We propose a novel two-sided mechanism for trading big data commodities (called 2-SAMBA) considering a market with several cloud providers and cloud users. 2-SAMBA consists of winner and price determination phases. We represent the users' requests over time as a conflict graph that is used by 2-SAMBA to determine the winning cloud providers and users. 2-SAMBA determines the price that each user has to pay and the revenue that each cloud provider receives. Our mechanism promotes healthy competition by giving incentives to cloud providers and users to reveal their actual valuations. In addition, we model the optimal winning determination phase as an integer program. We provide a comprehensive assessment through extensive performance analysis experiments and compare the solutions obtained by 2-SAMBA with the optimal solutions.

Related Work. Designing one-sided auctions for trading cloud resources has attracted a great deal of attention. Wang et al. [4] proposed offline one-sided auction-based mechanisms for cloud resource pricing. Zhang et al. [5] proposed an online one-sided auction for resource allocation in clouds in the presence of only one type of resources. In our previous studies [6], [7], [8], we proposed truthful offline and online one-sided auction-based mechanisms for allocation and pricing of VMs with heterogeneous resources in clouds such that their profit is maximized and the resources are utilized efficiently. None of the above mentioned studies considered the design of two-sided mechanisms.

Several researchers have studied resource management in cloud federations to facilitate big data processing. Van den Bossche *et al.* [9] proposed a binary integer program formulation for public and private cloud federations in order to minimize the cost of outsourcing. Mashayekhy et al. [10] addressed the problem of federation formation in clouds and designed a coalitional game-based mechanism that enables the cloud providers to dynamically form a cloud federation maximizing their profit. They also developed cloud federation frameworks considering data protection [11]. Ma et al. [12] proposed a new instruction set architecture to unify myriads of compute nodes to form a big virtual machine, which can scale up to support large clusters to support big data computing.

Recently, designing market exchange frameworks to facilitate the trading of cloud resources has received an increasing interest from the cloud computing research community. Garg et al. [13] identified the various technical and market requirements and challenges in designing such an exchange market. Watzl [3] proposed a framework for exchange-based trading of cloud computing commodities. A version of his proposed framework is currently being implemented at Deutsche Börse Cloud Exchange AG.

Several studies focused on designing double-auction mechanisms for large-scale distributed systems. Tan and Gurd [14] proposed a double auction for grid resource allocation. Fujiwara et al. [15] proposed a combinatorial double auction for cloud resource allocation. Nallur and Bahsoon [16] proposed market-based heuristic algorithms using a continuous double-auction to allow applications to decide which services to choose. They considered an application as a multi-agent system and the cloud as a marketplace where many such applications self-adapt. However, they did not model complex seller-side behavior. None of the above mentioned studies can be adapted for trading big data commodities along with guaranteeing strategy-proofness.

The seminal paper by McAfee [17] introduced a strategy-proof double auction mechanism, where buyers and sellers exchange single units. However, McAfee's double auction cannot be directly used in our problem since our setting requires multiple bids/asks over time. In addition, our proposed mechanism is a randomized strategy-proof mechanism that uses conflict graphs to find partitions of users, and their bids.

Organization. The rest of the paper is organized as follows. In Section II, we describe the system model, and we introduce the problem of trading big data computing commodities. In Section III, we present our proposed mechanism. In Section IV, we evaluate the proposed mechanism by extensive experiments. In Section V, we summarize our results and present possible directions for future research.

II. System Model

In this section, we describe the system model consisting of a set of big data jobs, a set of cloud providers, and a broker as a mediator. Each job is owned by a user. Each user knows the characteristics of her own job and reports the job's specification to the broker. The broker is a trusted third party responsible for receiving requests, determining the winners and prices, billing the users, receiving the payment from users, and paying the participating cloud providers. There exists a set of cloud providers $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ available to provide big data computing resources to cloud users. The allocation occurs over a reservation system on a set of different time slots, denoted by T. Each cloud

provider $C_j \in \mathcal{C}$ offers one cluster to users in each time slot, and reports a minimum cost for it. In addition, the preferences of the cloud providers are represented as a vector $\mathbf{a} = (a_1, \ldots, a_m)$, where each element a_j denotes the minimum cost of cloud provider $C_j \in \mathcal{C}$. Cloud provider C_j's cost per time slot is $\frac{a_j}{|T|}$. Each user $i \in \mathcal{U}$, where $\mathcal{U}$ is the set of users, requests to use a cluster for one time slot. She specifies her preference b_i^t as the maximum price she is willing to pay for using a cluster at time slot t.

Both users and cloud providers report their preferences (bids and asks, respectively) to the broker, which is responsible for executing the two-sided mechanism in order to determine the allocation and pricing. Users and cloud providers submit their preferences as sealed bids to the broker in advance. This information is private to all users and cloud providers.

Given the above setting the problem of trading big data computing commodities (TBDCC) is to determine the allocation of clusters to users and the price of the clusters based on the submitted bids and asks. A mechanism for solving the TBDCC problem consists of two phases: winner determination and price determination. In the winner determination phase, the mechanism determines the assignment of clusters to users over time. If user i receives a cluster from cloud provider C_j at time t, $x_{ij}^t = 1$; otherwise, $x_{ij}^t = 0$. If the resources of the cloud provider C_j are allocated to a user, then $y_j = 1$. In the price determination phase, the mechanism determines the amount π_i^u that each user i must pay to the broker, and the amount π_j^c that each cloud provider C_j receives from the broker. Users have quasi-linear utility, that is, if user i is allocated, her utility u_i^u is the difference between her valuation and the amount of money transferred (i.e., $u_i^u = b_i^t - \pi_i^u$), and zero (i.e., $u_i^u = 0$), otherwise. If cloud provider C_j allocates a cluster to users, C_j has a utility of $u_j^c = \pi_j^c - a_j$; and zero (i.e., $u_j^c = 0$), otherwise. The broker's monetary payoff is calculated as follows:

$$\sum_{i \in \mathcal{U}} \pi_i^u - \sum_{j:C_j \in \mathcal{C}} \pi_j^c \tag{1}$$

which is the total payment received from the users minus the revenues of the cloud providers. If the broker's monetary payoff is non-negative, the auction is *ex-post budget balanced*. This property gives incentives to the broker to set up the auction.

Each participant is self-interested, and tries to maximize its utility. Our goal is to design a strategy-proof mechanism that solves the TBDCC problem and discourages users and cloud providers from gaming the system by untruthful reporting. To promote the transactions and attract both users and cloud providers, we design a mechanism that maximizes the social welfare, i.e., the summation of each participant's utility and the broker's payoff.

If all users and cloud providers bid truthfully, the optimal winner determination phase can be modeled by the following integer program (TBDCC-IP):

$$\text{Maximize} \sum_{t \in T} \sum_{i \in \mathcal{U}} \sum_{j:C_j \in \mathcal{C}} b_i^t x_{ij}^t - \sum_{j:C_j \in \mathcal{C}} a_j y_j \tag{2}$$

Subject to:

$$\sum_{i \in \mathcal{U}} x_{ij}^t \leq 1, \forall j : C_j \in \mathcal{C}, \forall t \in T \tag{3}$$

$$\sum_{j:C_j \in \mathcal{C}} \sum_{t \in T} x_{ij}^t \leq 1, \forall i \in \mathcal{U} \tag{4}$$

$$y_j \geq x_{ij}^t, \forall i \in \mathcal{U}, \forall t \in \mathcal{T}, \forall j : C_j \in \mathcal{C} \tag{5}$$

$$x_{ij}^t = \{0,1\}, \forall i \in \mathcal{U}, \forall j : C_j \in \mathcal{C}, \forall t \in \mathcal{T} \tag{6}$$

$$y_j = \{0,1\}, \forall j : C_j \in \mathcal{C} \tag{7}$$

where x_{ij}^t and y_j are binary decision variables defined as follows:

$$x_{ij}^t = \begin{cases} 1 & \text{if user } i\text{'s request is assigned to } C_j \text{ at } t, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

$$y_j = \begin{cases} 1 & \text{if } C_j\text{'s resources has been allocated,} \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

The objective function is to maximize the social welfare. Constraints (3) guarantee that each cloud provider at each time slot serves at most one user. Constraints (4) ensure that the request of each user is fulfilled at most once. Constraints (5) guarantee that if a user is assigned to a cloud provider, the cost of that cloud provider is considered in the objective. Constraints (6) and (7) represent the integrality requirements for the decision variables.

The solution to TBDCC-IP will be used in our experiments as a benchmark for the winner determination phase of the proposed mechanism.

III. Two-Sided Mechanism for Trading Big Data Computing Commodities

Our goal is to design a strategy-proof two-sided mechanism for trading resources for big data applications. The mechanism, called 2-SAMBA (2-Sided Auction Mechanism for Bigdata Applications) is given in Algorithm 1 and consists of a winner determination phase and a price determination phase.

The mechanism first partitions the users into disjoint sets $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_P\}$ such that the users in each sub-partition do not overlap in terms of their requested time slots (lines 2-16). 2-SAMBA partitions the users in a way that conflicting users are not assigned to the same cloud provider. The mechanism creates a conflict graph $G^t(V^t, E^t)$ for each time slot t, where a vertex $v \in V^t$ represents a user, and an edge $(v, w) \in E^t$ represents a conflict between two users, u and w. A conflict between two users occurs when they submit bids to use a cluster for the same time slot t (lines 3-6). 2-SAMBA extracts a vertex from each conflict graph randomly, and assigns it to $\mathcal{P}_l$, where l tracks the number of current sub-partitions (lines 8-14). 2-SAMBA determines the sets $\mathcal{P}$, which represent the partitioning of the set of users. The users that are part of a set $\mathcal{P}_l$ do not conflict with each other, and can be assigned to the same cloud provider.

Then, for each sub-partition $\mathcal{P}_p \in \mathcal{P}$, 2-SAMBA finds a bid β_p, called the sub-partition bid, (lines 17-18) as follows:

$$\beta_p = |\mathcal{P}_p| \cdot \min_{i \in \mathcal{P}_p} b_i^t. \tag{10}$$

The sub-partition bid is determined by the number of users in the sub-partition and the minimum bid submitted by the users in that sub-partition. 2-SAMBA sorts the cloud providers and sub-partitions based on their asks/bids (lines 19-20). Then based on the orders, it finds the top k pairs of cloud provider and sub-partition for which the sub-partition bid is higher than the cloud provider's ask (lines 21-26). The users belonging to the first $k-1$ sub-partitions and the first $k-1$ cloud providers are selected as winners, and they are matched (line 27). For the winning cloud providers, 2-SAMBA sets $y_l = 1$, where l is the index of the cloud provider in the sorted list $\mathcal{C}^s$ (lines 28-29). For the winning users, 2-SAMBA sets $x_{il}^t = 1$, where t is the time slot requested by user i, and l is the index of the cloud provider and the sub-partition that user i belongs to in the sorted list $\mathcal{C}^s$ and $\mathcal{P}^s$, respectively (lines 28-32). This concludes the winner determination phase of the mechanism.

The winning cloud providers receive as payments the cost associated with the k-th cloud provider (lines 34-36). The remaining cloud providers do not receive any payment since they are not winners (lines 37-38). The users who belong to the winning sub-partitions are then charged a payment π_i^u as follows:

$$\pi_i^u = \frac{\beta_k}{|\mathcal{P}_p|} \tag{11}$$

where π_i^u is based on the k-th sub-partition bid (i.e., β_k) and the size of the winning sub-partition that user i belongs to (lines 39-41). The remaining users (i.e., non-winning users) are not charged (lines 42-44). This concludes the payment determination phase and 2-SAMBA returns the allocation and the payments (line 45).

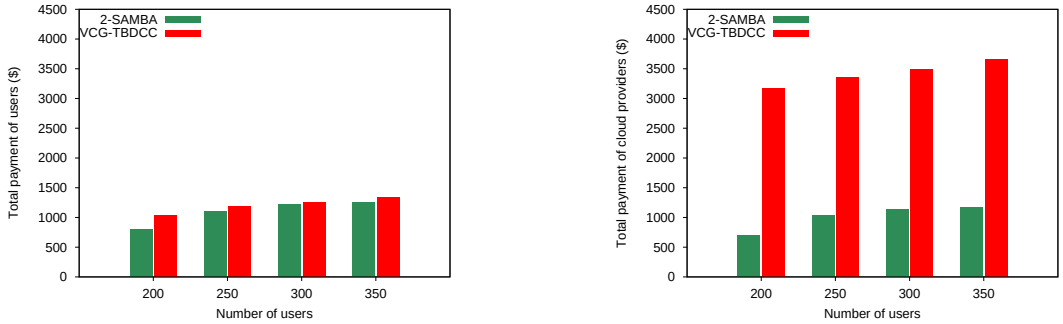
Since 2-SAMBA selects the users to form a partition randomly, it is a randomized mechanism. 2-SAMBA is *universally strategy-proof* [18] for both users and cloud providers, i.e., it is strategy-proof for any possible selection of users to form sub-partitions from the conflict graph. It is also an ex-post budget balanced mechanism. Due to space limitations we are not able to present the proofs of these properties.

Algorithm 1 2-SAMBA Mechanism

1: {*Phase I: Winner determination*}
2: $\mathcal{U} \leftarrow$ set of users
3: **for all** $t = 1, \ldots, T$ **do**
4: V^t: users $\in \mathcal{U}$ bidding to use a cluster at time t
5: E^t: pairs of users $\in V^t$
6: Build a conflict graph $G^t(V^t, E^t)$
7: $l = 1$
8: **while** there is a vertex in any G^t **do**
9: $\mathcal{P}_l = \emptyset$
10: **for all** $t = 1, \ldots, T$ **do**
11: **if** $V^t \neq \emptyset$ **then**
12: $v \leftarrow$ randomly extract a vertex from G^t
13: $\mathcal{P}_l \leftarrow \mathcal{P}_l \cup \{v\}$
14: $l++$
15: $P = l - 1$
16: $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_P\}$
17: **for all** $p = 1, \ldots, P$ **do**
18: $\beta_p = |\mathcal{P}_p| \cdot \min_{i \in \mathcal{P}_p} b_i^t$
19: $\mathcal{C}^s \leftarrow$ Sort all $C_j \in \mathcal{C}$ in non-decreasing order of a_j
20: $\mathcal{P}^s \leftarrow$ Sort all $\mathcal{P}_p \in \mathcal{P}$ in non-increasing order of β_p
21: $k = 0$
22: **for all** $l = 1, \ldots, \min\{P, m\}$ **do**
23: $\alpha = a_l$, where C_l is the l-th cloud provider in $\mathcal{C}^s$
24: $\beta = \beta_l$, where $\mathcal{P}_l$ is the l-th partition of the users in $\mathcal{P}^s$
25: **if** $\beta \geq \alpha$ **then**
26: $k++$
27: Match the first k-1 pairs of sub-partitions of users and cloud providers
28: **for all** $l = 1, \ldots, k-1$ **do**
29: $y_l = 1$
30: **for all** $i \in \mathcal{P}_l$ **do**
31: **if** user i bids for time slot t **then**
32: $x_{il}^t = 1$
33: {*Phase II: Price determination*}
34: **for all** $j : C_j \in \mathcal{C}^s$ **do**
35: **if** $y_j = 1$ **then**
36: $\pi_j^c = a_k$
37: **else**
38: $\pi_j^c = 0$
39: **for all** $p = 1, \ldots, k-1$ **do**
40: **for all** $i \in \mathcal{P}_p$ **do**
41: $\pi_i^u = \frac{\beta_k}{|\mathcal{P}_p|}$
42: **for all** $p = k, \ldots, P$ **do**
43: **for all** $i \in \mathcal{P}_p$ **do**
44: $\pi_i^u = 0$
45: **return** x, y, π

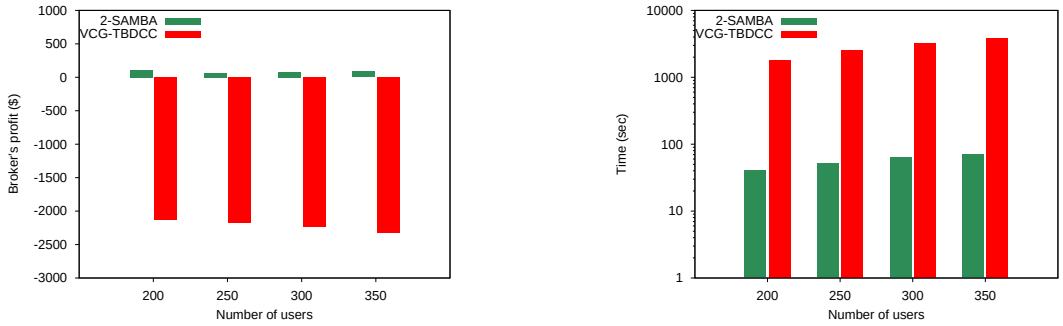
IV. Experimental Results

We perform extensive experiments in order to investigate the performance of the proposed mechanism 2-SAMBA. We compare the performance of 2-SAMBA with another strategy-proof mechanism called VCG-TBDCC. The VCG-



(a) (b)

Figure 1: 2-SAMBA vs. VCG-TBDCC: (a) Payment paid by users; (b) Payment received by cloud providers.



(a) (b)

Figure 2: 2-SAMBA vs. VCG-TBDCC: (a) Broker's revenue; (b) Execution time.

based double auction mechanism, VCG-TBDCC, consists of the winner determination obtained by solving TBDCC-IP and the price determination based on participants' marginal contribution to social welfare. An optimal winner determination algorithm with well-known VCG (Vickrey-Clarke-Groves) payments provides a strategy-proof mechanism [19]. We use the IBM ILOG CPLEX Optimization Studio Multiplatform Multilingual eAssembly to solve the TBDCC-IP optimally. The mechanisms are implemented in C++ and the experiments are conducted on AMD 2.4GHz Dual Proc Dual Core nodes with 16GB RAM which are part of the WSU Grid System. In this section, we describe the experimental setup and analyze the experimental results.

A. Experimental Setup

The users' requests are generated based on realistic data combining publicly available information provided by Amazon as follows. We generate bids/asks based on Amazon Spot market report on users bidding strategies [20]. Amazon regularly updates its spot price history based on the past 90 days of activity. Amazon reported that most users bid between the price of reserved instances and on-demand prices. By doing so, these users saved between 50% to 66% compared to the on demand prices. Cluster compute instances for Amazon EU (Ireland) region cost $3.2 per hour based on Amazon EC2 and Amazon Elastic MapReduce prices. We consider 6 hours as a time slot, and we generate the users' and cloud providers' bids/asks based on a uniform distribution with the average of 19.2. We consider 8 cloud providers with 28 time slots (a week), and 200 to 350 users.

B. Analysis of Results

Figs. 1a and 1b show the total payment of users paid to the broker and the total payment that cloud providers receive from the broker, respectively. In both figures, with the increase in the number of requests from users, the total payment increases. For each fixed number of users, the total payment by users determined by 2-SAMBA is greater than the total payment received by cloud providers. For example, for 2-SAMBA and the case with 200 users, the total payment of users is $797.552 and the total payment received by cloud providers is $688.296. This shows that 2-SAMBA is ex-post budget-balanced. However, these figures show that VCG-TBDCC is not ex-post budget-balanced, which is a key requirement for many trading environments. For example, for VCG-TBDCC and the case with 200 users, the total payment by users is $1036.52, while the total payment

received by cloud providers is $3164.61, resulting in a loss for the broker.

The broker's profit is shown in Fig. 2a, which is the total payment received from the users minus the revenues of the cloud providers. The results show that 2-SAMBA obtains a positive profit for the broker, while VCG-TBDCC obtains a negative profit for the broker in all cases. For example, for the case with 200 users, 2-SAMBA and VCG-TBDCC obtain $109.256 and -$2128.08 as broker's profit, respectively. The total profit of the broker obtained by 2-SAMBA is 7.65% of the total payment received from users. However, in the case of VCG-TBDCC, the broker incurs a loss, making it unsuitable for practical implementation.

Fig. 2b shows the execution time of the mechanisms. 2-SAMBA is very fast, being able to find the solutions in less than 75 seconds. This makes 2-SAMBA suitable for use in two-sided markets with high demand. The results show that the execution time of 2-SAMBA is about two orders of magnitude less than that of VCG-TBDCC.

From all the above results, we conclude that 2-SAMBA is a suitable 2-sided mechanism for trading big data computing commodities.

V. Conclusion

Cloud providers should expect a dramatic increase in the demand for computing resources for big data applications in the near future. Such demand necessitates the design of novel market mechanisms for users and cloud providers in order to facilitate the trading of computing resources for big data applications.

In this paper, we designed a novel two-sided auction mechanism, 2-SAMBA, which offers benefits to both, cloud users and cloud providers, by deploying the big data services of an exchange facilitating trading and usage of cloud resources. 2-SAMBA is universal strategy-proof, and thus, it prevents market manipulations by cloud providers and users. We performed extensive experiments to evaluate our proposed mechanism. The results showed that 2-SAMBA provides an effective way of pricing the big data commodities taking into account the incentives of the users, the cloud providers, and the broker.

Acknowledgment

This research was supported in part by NSF grants DGE-0654014 and CNS-1116787.

References

[1] Y. Zheng, N. B. Shroff, and P. Sinha, "A new analytical technique for designing provably efficient mapreduce schedulers," in *Proc. of the IEEE INFOCOM*, 2013, pp. 1600–1608.

[2] L. Mashayekhy, M. Nejad, D. Grosu, Q. Zhang, and W. Shi, "Energy-aware scheduling of mapreduce jobs for big data applications," *IEEE Trans. on Parallel and Distributed Systems*, PrePrints, 2014. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2358556

[3] J. Watzl, "A framework for exchange-based trading of cloud computing commodities," Ph.D. dissertation, Ludwig Maximilian University of Munich, 2014.

[4] Q. Wang, K. Ren, and X. Meng, "When cloud meets ebay: Towards effective pricing for cloud computing," in *Proc. of the IEEE INFOCOM*, 2012, pp. 936–944.

[5] H. Zhang, B. Li, H. Jiang, F. Liu, A. V. Vasilakos, and J. Liu, "A framework for truthful online auctions in cloud computing with heterogeneous user demands," in *Proc. of IEEE INFOCOM*, 2013, pp. 1510–1518.

[6] M. M. Nejad, L. Mashayekhy, and D. Grosu, "Truthful greedy mechanisms for dynamic virtual machine provisioning and allocation in clouds," *IEEE Trans. on Parallel and Distributed Systems*, PrePrints, 2014. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2308224

[7] L. Mashayekhy, M. Nejad, and D. Grosu, "A PTAS mechanism for provisioning and allocation of heterogeneous cloud resources," *IEEE Trans. on Parallel and Distributed Systems*, PrePrints, 2014. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2355228

[8] L. Mashayekhy, M. M. Nejad, D. Grosu, and A. V. Vasilakos, "Incentive-compatible online mechanisms for resource provisioning and allocation in clouds," in *Proc. of the 7th IEEE Intl. Conf. on Cloud Computing*, 2014.

[9] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads," in *Proc. of the 3rd IEEE Intl. Conf. on Cloud Computing*, 2010, pp. 228–235.

[10] L. Mashayekhy, M. M. Nejad, and D. Grosu, "Cloud federations in the sky: Formation game and mechanism," *IEEE Trans. on Cloud Computing*, PrePrints, 2014. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TCC.2014.2338323

[11] ——, "A framework for data protection in cloud federations," in *Proc. 43rd Intl. Conf. on Parallel Processing*, 2014.

[12] Z. Ma, Z. Sheng, and L. Gu, "Dvm: A big virtual machine for cloud computing," *IEEE Trans. on Computers*, 2013.

[13] S. K. Garg, C. Vecchiola, and R. Buyya, "Mandi: a market exchange for trading utility and cloud computing services," *J. of Supercomputing*, vol. 64, no. 3, pp. 1153–1174, 2013.

[14] Z. Tan and J. R. Gurd, "Market-based grid resource allocation using a stable continuous double auction," in *Proc. 8th IEEE/ACM Intl Conf. on Grid Comp.*, 2007, pp. 283–290.

[15] I. Fujiwara, K. Aida, and I. Ono, "Applying double-sided combinational auctions to resource allocation in cloud computing," in *Proc. 10th IEEE/IPSJ Intl Symp. on Applications and the Internet*, 2010, pp. 7–14.

[16] V. Nallur and R. Bahsoon, "A decentralized self-adaptation mechanism for service-based applications in the cloud," *IEEE Trans. on Software Eng.*, vol. 39, no. 5, pp. 591–612, 2013.

[17] R. P. McAfee, "A dominant strategy double auction," *Journal of Economic Theory*, vol. 56, no. 2, pp. 434–450, 1992.

[18] S. Dobzinski and S. Dughmi, "On the power of randomization in algorithmic mechanism design," in *Proc. 50th IEEE Symp. on Foundations of Computer Science*, 2009, pp. 505–514.

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

[20] "Amazon EC2 Spot Instance Curriculum," http://aws.amazon.com/ec2/spot-tutorials/.