Mechanism Design for Stochastic Virtual Resource Allocation in Non-Cooperative Cloud Systems

Zhen Kong, Cheng-Zhong Xu
Dept. of Electrical and Computer Engineering
Wayne State University, Detroit, MI 48202,USA
{zkong, czxu}@wayne.edu

Minyi Guo
Dept. of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
guo-my@cs.sjtu.edu.cn

***Abstract*—Currently, virtualization technology has been widely adopted by cloud service providers to provide flexible and cost-effective resource sharing among users. On cloud platforms, computing resources are allocated on-demand dynamically and the application hosted on a virtual machine(VM) usually has the illusion of complete control of resources. Thus, a selfish VM may strategically compete for resource with other VMs to maximize its own benefit while at the cost of overall system performance. This problem poses new challenges to cloud providers, who must thwart non-cooperative behavior as well as allocating resource among selfish VMs efficiently. In this paper, we propose to utilize mechanism design to allocate resource among selfish VMs in a non-cooperative cloud environment. Because the accurate relationship between VM's valuation function and allocated resource may not be available in practice and the valuation function parameters may not noisefree, we also propose to apply stochastic approximation methods to get stochastic solution for allocation and payment outcomes. We show through theoretical analysis and simulations that the proposed stochastic mechanism is efficient and incentive compatible. That is, the incorporation of mechanism design for virtualized resource allocation is able to enforce cooperation and achieve efficient resource utilization among selfish VMs in non-cooperative cloud systems.**

***Index Terms*—Cloud computing, virtual machine, resource allocation, stochastic approximation, Vickrey-Clarke-Groves (VCG) mechanism design, game theory.**

I. Introduction

In recent years, cloud computing has been fast developing from a long-held dream into a commercially viable technology for users in search of a cost-effective storage and server solution [2]. Along with this new computing paradigm, computing resources are provided as services over the Internet and users can enjoy services from a shared pool of scalable computing resources in a pay-as-you-go manner with relatively low price. Currently, most cloud service providers use virtualization technology to provide flexible and cost-effective resource sharing, such as Amazon EC2 cloud computing infrastructure using Xen virtualization [3]. Generally speaking, virtualization technology allows a large number of virtual machines (VMs), each running a traditional operating system (OS) and hosting one or more services, to be consolidated on limited physical hardware as well as share physical processors and I/O interfaces with other VMs on the same physical machine. Furthermore, VM monitor (VMM), located between underlying hardware and guest OSes, acts as resource manager to control the resource provision and reconfiguration for the VMs. Because virtualization technology plays a key role in the realization of cloud computing, lots of effort have been exerted on the researches of system virtualization, such as resource allocation [5] and autoconfiguration [19].

Most of the past resource allocation works in literatures, such as [5], [17], [19], discussed in *cooperative* environments, where VMs are cooperative with each others and honestly report their real resource requests to VMM. Based on predefined criterions, VMM allocates resource according to VMs' requests, or dynamically adjusts resource configuration according to the observation of workload or VM environment changes. VMs themselves just passively comply with the algorithm and accepts the resource control results from VMM. Whereas in some practical scenarios, different VMs may host different applications belonging to different users. They may exhibit non-cooperative behaviors due to self-interests and try to gain their own advantages without regard to the overall system performance. By non-cooperative, we mean VMs care essentially about their own benefits, without any consideration about others. For example, since the application hosted on VM has the illusion that it has the complete control of physical resource, it can actively request resource and even request more than enough resource to maximize its own benefit. Consequently, the coexistence of several selfish VMs on a physical server will introduce the competition for resource among them. Although virtualization technology aims at performance isolation among different VMs, it has been shown that any bad behavior of one VM can still adversely affect the performance of other VMs by depriving the hypervisor and driver domain resources [19]. In [10], it is also demonstrated that in an ARM based Xen environment, unauthorized software downloaded to a VM from the Internet has the potential to be malicious or selfish and consume more than enough resource at the expense of others. Therefore, selfish misbehavior of VMs has become a truly critical problem for VM resource allocation in cloud environments, and needs to be scrutinized carefully.

Here, we propose a mechanism design approach for resource allocation that explicitly considers individual VMs' rationality so as to thwart selfish behavior and enforce truth-revealing for selfish VMs in this non-cooperative environment. Mechanism design [13] is the study of designing rules of a game or system to achieve a specific system-wide outcome, even though each player may be selfish. Because the considered VM system is a

centralized environment, mechanism design is a suitable tool by leveraging the existing messaging and control mechanism and resolving the competition among selfish VMs. By mechanism design, each VM reveals to the mechanism its private information (referred as "type"), such as the parameters defining a valuation function quantifying its preference on a specific resource allocation outcome. After collecting all VMs' types, the mechanism calculates the allocation outcome and the payment results paid by VMs accordingly. Incentive compatibility and social optimum are the most desirable criteria that mechanism designers tend to achieve. A mechanism is said to be incentive compatible if all of the players fare best when they truthfully reveal any private information. While social optimum means the mechanism can be used to maximize the aggregate utilities of all the players in the system. Vickrey-Clarke-Groves (VCG) [13] is a well-know mechanism satisfying these two goals simultaneously. We will focus our design on VCG in this cloud computing environment.

Player's valuation on allocated resource is one of the prerequisites for mechanism design. To the best of our knowledge, most applications using mechanism design, such as spectrum auction [9] and parallel job scheduling [4], assume that the players can get the exact and accurate expression of their valuation functions. However, this assumption may not hold in practice, especially in virtualization environments, where it is non-trivial to convert performance metrics to resource allocations on the virtualized node because of complex system infrastructures and resource dependencies [17]. Therefore, VM's accurate valuation function or model is not available practically. System designer can only utilize some indirect experimental approaches, such as profiling [22] and feedback control [17], to get the "noisy" measurements. This requires to obtain the mechanism design results based on such noisy inputs instead of the accurate inputs used in traditional situations, which in turn introduces additional challenges to virtual resource allocation in non-cooperative environments: how to develop mechanism to combat uncertainty caused by noisy measurements? and how to ensure it be efficient and incentive compatible? To answer these questions, we resort to stochastic approximation (SA) [14], [20] to get stochastic allocation and payment solutions based on noisy valuation functions and analyze the performances. Specifically, we proposed SA approaches to calculate resource allocation and payment results for VCG mechanism in this stochastic environment, and showed its efficiency and incentive compatibility through theoretic analysis and simulations, i.e., the selfish VMs can be enforced to report their types truthfully and the virtual resource can be allocated in a socially optimal manner.

The rest of this paper is organized as follows. System model is presented in Section II. Then in Section III, we discuss the VCG mechanism in deterministic case, where each VM have exact measurement of valuation function on obtained resource. While the mechanism design based on "noisy" estimates of valuation functions is proposed and analyzed in Section IV. Section V gives the simulation results. Section VI presents the related work. We conclude in Section VII.

II. System model

The proposed mechanism design framework is depicted in Fig. 1, which is based on Xen architecture. Xen is a high performance resource-managed VMM, which consists of two components: a hypervisor and a driver domain. The hypervisor provides the guest OS, also called a guest domain in Xen, the illusion of occupying the actual hardware devices. The hypervisor performs functions such as CPU scheduling, memory mapping and I/O handling for guest domains. The driver domain (Dom0) is a privileged VM which manages other guest VMs and executes resource allocation policies. Dom0 hosts unmodified device drivers for VMs; it also has direct access to actual hardware. Xen provides a control interface in the driver domain to manage the available resources, such as CPU times, virtual CPUs and physical memory, to each VM.

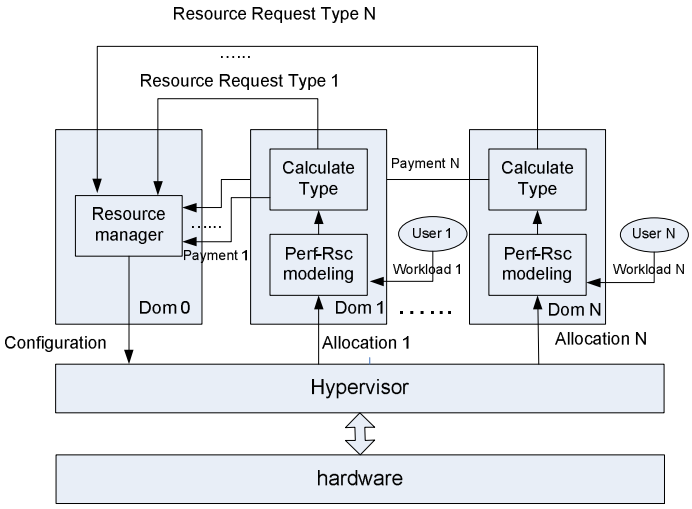


Fig. 1: Resource allocation framework in non-cooperative clouds.

The notations used throughout this paper are listed in Tab. I. We assume there are M VMs competing for computing resource from the Dom0. Each VM i requires r_i resources, such as CPU time, VCPU or memory, for its successful execution. The total available resource is Ω. To implement mechanism design, each VM i first estimates its valuation function according to current workload and SLA requirements through its performance-resource modeling functionality, then calculate its "type" θ_i and send it to Dom0. Here, type θ_i is the private information of VM i, which can be regarded as the parameters deciding the valuation function derived from the potentially allocated resources r_i under current workload and system conditions. For example, if VM i's valuation is linear as $V_i(r_i) = a_i * r_i + b_i$, its type would be $\theta_i = \{a_i, b_i\}$. Thus, we can define valuation function as $V_i(\theta_i, r_i)$. The true type profiles from all VMs is $\theta = \{\theta_1, \cdots, \theta_M\}$, with $\theta \subset \Theta, \Theta = \Theta_1 \times \cdots \times \Theta_M$. The reported type $\hat{\theta_i}$ may be different with the true type θ_i, and the announced type profile is $\hat{\theta} = \{\hat{\theta_1}, \cdots, \hat{\theta_M}\}$.

Upon receiving the reported type profile, the Dom0 will

determine the allocation and payment outcome for the participating VMs. The allocation outcome of the mechanism is defined as $\mathcal{R}(\theta, \Omega) = [r_1, \cdots, r_M]$, where $\mathcal{R} : \Theta \times \mathbb{R}_+ \rightarrow \mathbb{R}_+^M$ is an allocation function mapping the reported type profile and current available resource to the resource allocation. Meanwhile, the payment for allocated resources, $\tau(\theta, \Omega) = [\tau_1, \cdots, \tau_M]$ is also computed, where $\tau : \Theta \times \mathbb{R}_+ \rightarrow \mathbb{R}_-^M$ is a payment function that maps the reported type profile and current available resource to the payment that each VM needs to pay to the VMM. Therefore, the utility U_i that a VM i gains is $U_i(\theta_i, r_i) = V_i(\theta_i, r_i) - \tau(\theta_i, r_i)$.

As introduced in Section I, we propose to consider mechanism design in deterministic and stochastic situations, respectively. In the following, we describe our designs and analysis based on the above auction framework.

TABLE I: Notation.

Notation	Description
θ_i	true type of VM i
θ	type profile from all VMs
Θ_i	the set of possible types of VM i
Θ	the set of type profiles for all VMs
$\hat{\theta_i}$	reported type of VM i
$\hat{\theta}$	reported type profile for all VMs
r_i	resource allocated to VM i
$\mathbf{r}$	resource allocation set $\{r_i\}$ for all VMs
$\mathbf{r}^*$	optimal resource allocation
Ω	current available resource
$r(\theta, \Omega)$	resource allocation outcome
τ_i	the payment of VM i
$\tau(\theta, \Omega)$	the set of payment of all VMs
U_i	utility of VM i
$V_i(\theta_i, r_i)$	VM i's valuation function on obtained r_i under his type θ_i
$r_{i,n}$	the n^{th} iteration of the estimate of r_i
$\{r_{i,n}\}$	the sequence of estimate of r_i
$\{\mathbf{r}_n\}$	the sequence of estimate of $\mathbf{r}$
$Y_{i,n}(\theta_{i,n}, r_{i,n})$	the noisy measurement of valuation function by VM i at the n^{th} iteration
$\xi_{i,n}$	the measurement error for VM i's valuation function at the n^{th} iteration
$f(\mathbf{r})$	the objective function for maximization
$\phi(\mathbf{r})$	the constraint function

III. Deterministic Analysis

In the study of resource allocation in distributed computing systems, it is of prime importance to quantitatively describe the relationship between resource and performance. Lots of work characterize it in a deterministic manner. That is, the resource demand of each job is deterministic without uncertainty. There is no resource contention between jobs and the execution time of the jobs are highly predictable. Because deterministic performance model has been widely and efficiently used for load balance and resource allocation problems, it is a nature idea to utilize deterministic model in cloud systems. Specifically, we assume each VM i can derive its exact valuation function $V_i(r_i)$, and V_i is quasi-concave on resource r_i. To be noticed, this valuation is not known by Dom0 unless being informed. Thus, a selfish VM can notify the Dom0 a bogus valuation instead of its true value to obtain unfair benefit at the cost of others. As discussed in Section I, we use VCG mechanism to enforce cooperation among selfish VMs.

Specifically, VCG mechanism allocates resource to maximize the declared social welfare in the following way.

$$\mathbf{r}^* = \{r_1^*, r_2^*, \cdots, r_M^*\} = \arg\max_{\{r_i\}} \sum_i V_i(\hat{\theta_i}, r_i) \qquad (1)$$

subject to:

$$(I) \quad \sum_i r_i = \Omega; \qquad (II) \quad 0 \leq r_i \leq \Omega,$$

where the objective function of (1) means that the resource is allocated to maximize the aggregate valuation of all VMs, Constraint (I) defines the total resource budget, and Constraint (II) defines resource r_i as a real parameter between 0 and Ω. Actually, if r_i represents VCPU, it should be an integer. Here, we relax it to a positive real number to formulate the optimization as a convex problem. This is a practical relaxation because VM i can be regarded as obtaining $\frac{r_i}{\Omega} * 100\%$ VCPU resource if its allocation is r_i.

While based on VCG mechanism, the payment paid by each VM should be:

$$\tau_i(\hat{\theta_j}, \Omega) = \max_{\mathbf{r}_{-i}} \sum_{j \neq i} V_j(\hat{\theta_j}, r_j) - \sum_{j \neq i} V_j(\hat{\theta_j}, r_j^*), \qquad (2)$$

where $\mathbf{r}_{-i}$ represents the resource allocation results when VM i is absent in the system.

The first term of (2) is the maximum aggregated utility that all other VMs can derive if VM i does not participate in the resource competition. The second term is the sum of aggregated valuations of the other VMs except VM i under optimal resource allocation in the presence of VM i. It is clear that the first term is not less than the second term since the second one is the maximum summation of utilities for all the VMs except VM i. Thus, the payment paid by each VM corresponds to the loss of declared welfare it imposes to the others through its presence.

Theorem 1. *With VCG mechanism, selfish VM can be enforced to report utility function truthfully.*

Proof: When reporting $\hat{\theta}$, the utility of VM i is

$$\begin{aligned} U_i(\hat{\theta_i}, r_i) &= V_i(\theta_i, r_i^*) - \tau(\hat{\theta_i}, r_i) \\ &= V_i(\theta_i, r_i^*) - [\max_{\mathbf{r}_{-i}} \sum_{j \neq i} V_j(\hat{\theta_j}, r_j) - \sum_{j \neq i} V_j(\hat{\theta_j}, r_j^*)] \\ &= [V_i(\theta_i, r_i^*) + \sum_{j \neq i} V_j(\hat{\theta_j}, r_j^*)] - \max_{\mathbf{r}_{-i}} \sum_{j \neq i} V_j(\hat{\theta_j}, r_j). \end{aligned} \qquad (3)$$

The last item in (3) is independent of the strategic behavior

of VM i, thus, his optimal strategy is to solve the following optimization problem

$$max(V_i(\theta_i, r_i^*) + \sum_{j \neq i} V_j(\hat{\theta_j}, r_j^*)). \quad (4)$$

We rewrite the social welfare maximization function in (1) as follows:

$$\max_{\{\hat{\theta_i}\}} \sum_i V_i(\hat{\theta_i}, r_i) = \max_{\{\hat{\theta_i}\}} (V_i(\hat{\theta_i}, r_i) + \sum_{j \neq i} V_j(\hat{\theta_j}, r_j)). \quad (5)$$

Comparing (4) and (5), a dominant strategy for VM i is to declare its true type θ_j as as to maximize its own utility, regardless the other VM's strategies. ■

We take a simple example with 2 VMs to illustrate the payment mechanism in Tab. II. Suppose the total available resource, e.g., VCPU, is 10. VM-1 and VM-2's true valuation functions are $r_i + 5$ and $2r_i + 3$, respectively. When both VMs report truthfully, resource allocation $\mathbf{r}$ is $\{0, 10\}$. In this situation, VM-1's valuation is 5, its payment is 0, and the resulting utility is 5. On the other hand, if VM-1 reports a bogus type $2r_i + 5$, then it will get 10 VCPUs and a valuation of 15 corresponding. However, in this case, when VM-1 cheats, VM-2 gets a valuation of 3; if VM-1 is removed, VM-2 will get whole resource and its valuation is 23, then, according to (2), VM-1's payment will be $23 - 3 = 20$. Thus, although the valuation is increased to 15, its utility drops to $15 - 20 = -5$. Therefore, VM-1 is punished by VCG mechanism for its cheating.

TABLE II: Illustration example of VCG mechanism.

	(1) No VMs lying		(2)VM-1 lying	
Reported type	$r_i + 5$	$2r_i + 3$	$2r_i + 5$	$2r_i + 3$
Allocation	0	10	10	0
Valuation	5	23	15	3
Payment	0	10	20	0
Utility	5	13	-5	3

Thus, from Theorem 1, we can see that no VM has incentive to lie its type to the Dom0 by VCG mechanism. Then, after collecting true resource requests from all VMs, the Dom0 can allocate resource efficiently according to (1) to maximize the social welfare in this deterministic situation.

IV. Stochastic Analysis

The above VCG mechanism works well in deterministic situation where the players can get accurate valuation functions. However, this deterministic assumption doesn't hold in practice, especially for virtualized cloud environments. First, it's non-trivial to convert individual application performance requirements to resource allocations on virtualized node because of complex system infrastructures and resource dependencies [17]. Second, current virtualization technique does not guarantee perfect performance isolation between VMs [18], and the resultant performance interference between consolidated VMs complicates the relationship between resource allocations and QoS performance [16].

Therefore, VM's accurate valuation function $V_i(\theta_i, r_i)$ is not available in virtualized cloud systems. System designers can only utilize some indirect experimental approaches, such as profiling, to get the "noisy" measurements Y_i at any desired value of r_i. Consequently, the deterministic solution of resource allocation cannot be obtained in this noisy environment, and we need to obtain the mechanism design results based on such noisy inputs instead of the accurate inputs used in traditional situations. This change in turn introduces additional challenges to virtual resource allocation in non-cooperative cloud environments, i.e., how to develop mechanism to combat uncertainty caused by noisy measurements? and how to ensure it be efficient and incentive compatible? To answer these questions, we resort to stochastic approximation (SA) [14], [20] to get stochastic allocation and payment solutions based on noisy valuation functions.

Specifically, to get a stochastic solution from SA approaches, we first assume that the noise $\{\xi_n\}$ is a sequence of mutually independent random variables, and then define the "noisy" measurement of the value of $V_{i,n}(\theta_{i,n}, r_{i,n})$ as a random vector $Y_{i,n}$:

$$Y_{i,n}(\theta_{i,n}, r_{i,n}) = V_{i,n}(\theta_{i,n}, r_{i,n}) + \xi_{i,n}. \quad (6)$$

Now, let re-express the original constrained optimization problem in (1) as

$$Maximize\, f(\mathbf{r}) = f(r_1, ..., r_M) = \sum_{i=1}^{i=M} Y_i(\theta_i, r_i) \quad (7)$$

subject to:

$$(I) \quad \phi(\mathbf{r}) = \sum_i r_i - \Omega \ = 0, \qquad 0 \leq r_i \leq \Omega.$$

Here, the constraint function $\phi(\mathbf{r})$ is assumed known a priori, but $f(\mathbf{r})$ is unknown. In the following, we utilize stochastic approximation methods to find recursive and stochastic solution of $\{r_n\}$ from the noisy observation $Y_{i,n}$ instead of $V_{i,n}$.

A. Allocation Analysis

SA methods are a family of iterative stochastic optimization algorithms that attempt to solve zero-finding or minimization functions which cannot be computed directly, but only estimated via noisy observations. Specifically, after converting (7) to a minimization problem by applying a negative sign on it, we implement the Kiefer-Wolfowitz (K-W)algorithm [12] to estimate its optimal solution subject to the resource constraint. $\mathbf{r}_{n+1}$ at iteration $n + 1$ can be estimated by

$$\mathbf{r}_{n+1} = \Pi_\phi(\mathbf{r}_n - a_n \cdot \hat{\nabla}_{\mathbf{r}}(-f(\mathbf{r}_n))), \quad (8)$$

where $\hat{\nabla}_{\mathbf{r}}(-f(\mathbf{r}_n))$ is an estimate of the gradient of $-f(\mathbf{r}_n)$ with respect to $\mathbf{r}$, $\{a_n\}$ is a gain sequence of positive real numbers with $\sum_n(a_n) = \infty$ and $\sum_n(a_n^2) < \infty$, and Π_ϕ

is a projection onto constraint set ϕ. According to K-W algorithm, the gradient can be estimated by a finite difference. Specifically, by let $c_n \to 0$ be a finite difference interval and e_n be the standard unit vector in the n^{th} coordinate direction, we can get $\hat{\nabla}_{\mathbf{r}}(f(\mathbf{r}_n))$ as

$$\hat{\nabla}_{\mathbf{r}}(f(\mathbf{r}_n)) = \frac{[f(r+e_1c_n),...,f(r+e_mc_n)]^T}{2c_n} - \frac{[f(r-e_1c_n),...,f(r-e_mc_n)]^T}{2c_n}. \tag{9}$$

The basic idea of the K-W algorithm is to tune the resource allocation estimated each iteration by a small step in the positive and negative directions from its current value, and collect measurements of the next observation variable such that (8) will converge toward the optimal value of (7). The convergence is proven in the following.

Theorem 2. *When all VM report truthfully, under any initial condition $\{r_{i,0}\}$, the sequence $\{r_n\}$ generated by (8) converges weakly to a local optimizer $\boldsymbol{r}^*$.*

Proof:

We first interpolate the iterates $\{r_n\}$ into a continuous time interpolation process r_n^ϵ with interpolation intervals ϵ_n, and define r_n^ϵ as follows:

$$\overline{r}_i^{\epsilon_i}(t) = \begin{cases} r_{i,0} & for \quad t<0 \\ r_{i,n} & for \quad n\epsilon_i - \epsilon_i \le t \le n\epsilon_i,\ n=1,2,\cdots \end{cases} \tag{10}$$

Then we introduce the augmented Lagrangian to express the objective function (7) as

$$W(\mathbf{r},\lambda) = -f(\mathbf{r}) + \lambda^T \cdot \phi(\mathbf{r}) + (\frac{k}{2})|\phi(\mathbf{r})|^2, \tag{11}$$

where k be a positive real number, and the superscript T represents transpose.

Because ξ_n is mutually independent random variables and $W(r,\lambda)$ is integrable, as shown in [14], when $\epsilon_n \to 0$, $\overline{r}_i^{\epsilon_i}(t)$ converges weakly in trajectory to the continuous-time function $\overline{r}(t)$. Here, $\overline{r}(t)$ is the solution of the ordinary differential equation (ODE)

$$\frac{d\overline{r}}{dt} = -W(\overline{r},\lambda). \tag{12}$$

Because $f(\mathbf{r})$ is bounded and quasi-concave with respect to $\mathbf{r}$, the ODE in (12) is asymptotically Liapunov stable. Consequently, as $\epsilon_n \to 0$, the sequence $\{r_n\}$ generated by (8) converges weakly to a local optimizer $\mathbf{r}^*$. ■

From this theorem, the introduced SA approaches in (8) can lead $\{r_n\}$ to a stable value statistically, which is also the maximizer of the social welfare in deterministic situation. Therefore, even with noisy type reported from individual VMs, Dom0 can still calculate the optimal resource allocation results iteratively and stochastically.

B. Payment Analysis

The prerequisite for (8) converging to an optimizer is that all VMs report truthfully. But how to enforce truthful-telling, i.e., incentive compatibility, in this stochastic environment has not been investigated yet. Now, our goal is to find payment method such that truth-telling could be also achieved even under noisy estimates. Based on the observation value Y, we modified the VCG payment function in (2) for VM i at step n in the following:

$$\tau_{i,n}(\hat{\theta}_{i,n},\Omega) = \max_{r_{-i,n}} \sum_{j\neq i} Y_{j,n}(\hat{\theta}_{j,n},\tilde{r}_{j,n}) - \sum_{j\neq i} Y_{j,n}(\hat{\theta}_{j,n},r_{j,n}), \tag{13}$$

where $\tilde{r}_{j,n}$ is the recursive solution when VM i is absent. Similar to (2), it represents the recursive inconvenience caused to other VMs by VM i at step n. We then show its incentive compatible feature in the following theorem.

Theorem 3. *In this stochastic situation, truthful revealing is a dominant strategy for VM if payment approach in (13) and stochastic resource allocation $\{r_n\}$ in (8) are applied.*

Proof: Similar to the proof of Theorem 1, the utility of VM i at iteration n is

$$\begin{aligned} U_{i,n}(\hat{\theta_{i,n}},r_{i,n}) &= V_{i,n}(\theta_{i,n},r_{i,n}) - \tau(\hat{\theta_{i,n}},r_{i,n}) \\ &= [V_{i,n}(\theta_{i,n},r_{i,n}) + \sum_{j\neq i} Y_{j,n}(\hat{\theta_{j,n}},r_{j,n})] \\ &\quad - \max_{r_{-i,n}} \sum_{j\neq i} Y_{j,n}(\hat{\theta_{j,n}},\tilde{r}_{j,n}). \end{aligned} \tag{14}$$

Because $r_{i,n}$ will converge to $\mathbf{r}^*$, which is the maximizer for the aggregate valuations, i.e.,

$$\begin{aligned} r_i^* &= \lim_{n\to\infty}\{r_{i,n}\} \\ &= \lim_{n\to\infty}(max(V_i(\hat{\theta_{i,n}},r_{i,n}) + \sum_{j\neq i} V_{j,n}(\hat{\theta_{j,n}},r_{j,n})))). \end{aligned} \tag{15}$$

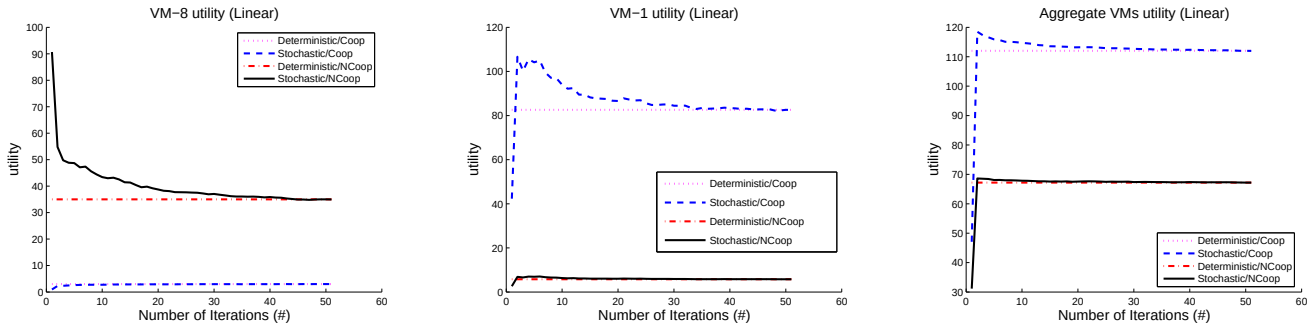
Therefore, when using (8) and (13), the true value $\theta_{i,n}$ can maximize $[V_{i,n}(\theta_{i,n},r_{i,n}) + \sum_{j\neq i} V_{j,n}(\hat{\theta_{j,n}},r_{j,n})]$ recursively, and so on its utility $U_{i,n}$. Thus, truthful revealing is a dominant strategy for VM i. ■

In another words, the modified VCG payment in (13) is incentive compatible in this stochastic situation. When Dom0 charges VMs according to this payment function, the selfish VMs will not deviate from truthful-telling. Consequently, the stochastic allocation method in (8) will take effect and lead the system to social optimal stochastically and recursively.

V. Numerical analysis

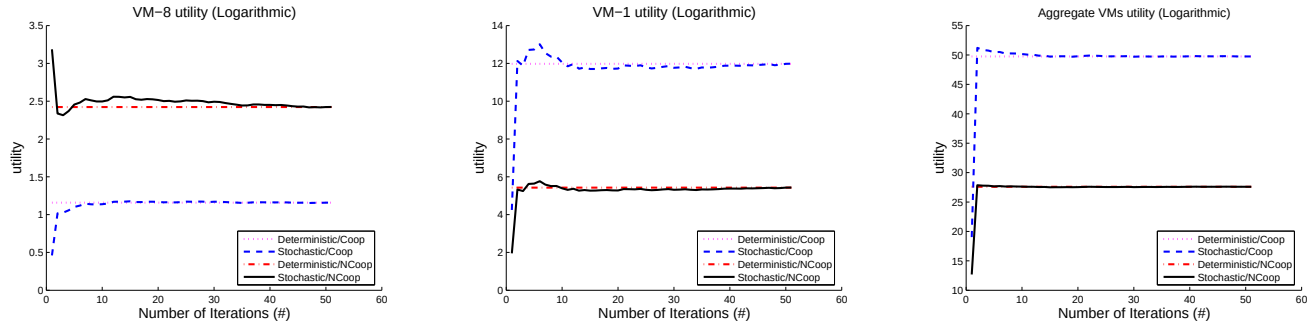
A. Impact of selfish behavior

We conduct simulations to show the performance of above proposed SA approaches. The overall simulation plan is that we first investigate the impact of selfish behavior on system



(a) VM-8's utility (b) VM-1's utility (c) Aggregate utility

Fig. 2: Utility performance under linear valuation function.



(a) VM-8's utility (b) VM-1's utility (c) Aggregate utility

Fig. 3: Utility performance under logarithmic valuation function.

performance under different kinds of valuation functions, then we will study the effectiveness of VCG mechanism in this non-cooperative environment. The simulated virtual machine environment consists of 1 Dom0 and 8 guest domains. We also consider there are 30 virtual CPUs available for allocation, i.e., $M = 8$ and $\Omega = 30$. Among them, VM-8 is assumed to be selfish and may report bogus type to maximize its own benefit in non-cooperative situations. We also assume system conditions change slowly; thus, the stochastic allocation results can converge to be stable before the valuation functions changes. Under this assumption and without loss of generality, we set the valuation function of each VM unchanged during the whole simulation period to show the utility performance and see if the recursive solution in stochastic case can converge to the optimum point in deterministic situation. In the simulation, each test case has been repeated for 100 trials and then we plot the figures with average value.

We first assume VM's valuation function is linear, i.e., $V_i(r_i) = a_i \cdot r_i + b_i$, $i \in \{1 \dots M\}$, where $a_i = 2.4 - 0.2 \cdot (i-1)$ and $b_i = 3.4 - 0.2 \cdot (i-1)$. And the random measure noisy ξ_i is uniformly distributed between 0 and 1. The assumption to use linear model is that although the performance model is usually non-linear and workload-dependent, the system behavior in the neighborhood of an operating point can be approximated locally by a linear model [17], which is also verified by experiment results in [7]. In a cooperative environment, all M VMs report their true types to Dom0. Whereas in non-cooperative situations, VM-8 deliberately tries to misbehave by reporting bogus type $\hat{\theta}_8$ representing $\hat{V}_8(r_8) = 3.6 \cdot r_i + 5.1$ rather than its true valuation function $V_8(r_8) = 1 \cdot r_i + 2$ to DomU. This is a reasonable setting to simulate selfish behavior because the selfish VM may get unfair resource share and increase its utility with this bogus valuation. While as shown in Figure 2(a), in non-cooperative case, VM-8's utility increases from 3 to 35 by behaving selfishly, or an increase of 11.67 times that in cooperative environments. Thus, the selfish VM really has the intention to report bogus information to improve its own benefit. In addition, the utility calculated in stochastic environment can converge to the stable value achieved in deterministic case eventually, which verifies the feasibility of applying stochastic approximation methods in this noisy environment. However, as a result of VM-8 using this selfish strategy, VM-1's utility shown in Figure 2(b) drops dramatically from 82.6 to 5.8, or a decrease of 92.98%. While from Figure 2(c), the system aggregate utility is also decreased 40% from 112 to 67.2. Therefore, the existence of selfish behavior in non-cooperative environments could significantly degrade the resource allocation performances.

Because the linear valuation function cannot accurately model the "diminishing return" effect, i.e, the utility of a resource saturates as the quantity of resource received is larger than that is necessary to achieve a certain level of

TABLE III: Comparisons of valuations, payments and utilities under linear valuation in the two cases.

VM	(1) No VMs lying			(2) Only VM-8 lying			∇Utility	∇Payment
	Valuation	Payment	Utility	Valuation	Payment	Utility		
1	82.6	72.6	10.0	5.8	1.0	4.8	-5.2	-71.6
8	3.0	2.4	0.6	35.0	79.2	-44.2	-44.8	+76.8

TABLE IV: Comparisons of valuations, payments and utilities under logarithmic valuation in the two cases.

VM	(1) No VMs lying			(2) Only VM-8 lying			∇Utility	∇Payment
	Valuation	Payment	Utility	Valuation	Payment	Utility		
1	11.9	2.5	9.4	11.3	1.8	9.5	+0.1	-0.7
8	1.2	0.4	0.8	2.5	3.4	-0.9	-1.7	+3.0

performance requirement, we then try to capture the preference of cloud service users by using the following logarithmic function, i.e., $V_i(r_i) = a_i \cdot log(r_i + b_i)$, $i \in \{1 \dots M\}$, where $a_i = 4.5 - 0.5 \cdot (i-1)$ and $b_i = 5.5 - 0.5 \cdot (i-1)$. VM-8 also deliberately reports bogus type $\hat{\theta}_8$ representing $\hat{V}_8(r_8) = 6.75 \cdot log(r_i + 8.25)$ rather than its true valuation function $V_8(r_8) = 1 \cdot log(r_i + 2)$ to DomU. The simulation results are plotted in Figure 3(a), Figure 3(b) and Figure 3(c). It also shows the recursive solution r_i obtained by SA algorithm can converges to the aggregate utility optimizer in deterministic situation. Moreover, we can see from the shown results that the selfish VM-8's utility increases by about 109.38%, while the honest VM-1's utility and system aggregate utility drop by 54.72% and 44.53%, respectively.

B. Effectiveness of stochastic mechanism design algorithm

We further verify the effectiveness of VCG algorithm, i.e., the selfish user will be penalized if it lies about its resource requirement. In this simulation, we still compare the valuation, utility as well as the payments paid by users in stable state under two scenarios: (1) no user is lying about its valuation, and (2) only VM-8 is lying about its valuation, but the others are telling the truth. Tab. III and Tab. IV show the efficiency and payment performance under linear and logarithmic valuation functions, respectively.

From Tab. III, we can see that when all VMs reveal the true valuations, the resources are allocated among users efficiently. However, when VM-8 exaggerates its valuation, although its valuation is improved from 3.0 to 35.0, its payment is also increased from 2.4 to 79.2, and the resulting utility becomes negative. by using the VCG mechanism, the lying of selfish VM is penalized through a significantly increased payment. We obtained the similar results under logarithmic valuation functions from Tab. IV, where a selfish VM is punished by an additional payment and achieve a negative utility at last. Thus, a rational VM will not lie about its valuation of resource because the increased payment cannot be compensated by the improved valuation. Consequently, selfish VMs will be enticed to report their valuations honestly. Therefore, by using stochastic resource allocation approach in (8) and payment function in (13), the virtual resource in a non-cooperative cloud system can be efficiently allocated in a socially optimal manner, and the selfish VM can also be enforced to report its type truthfully.

VI. Related work

Resource allocation problem has been an important topic in virtualized cloud environments. Many researches have applied control-theoretic approaches (e.g., [7], [17], [23]) to realize automated resource allocation and service level managements in cloud systems. For example, Padala *et al.* [17] have combined an online model estimator and a multi-input multi-output (MIMO) resource controller to dynamically allocate resource to applications running on shared virtualized environments. Gong and Xu [7] have proposed a feedback control-based coordination system providing guarantees on a service level agreement with respect to performance and a power budget in virtualized environments. Zhang *et al.* [23] have presented two control loops to realize on-demand resource optimization and allocation in cloud infrastructures. Although these approaches can adaptively allocate resource according to dynamic workload and environments, they don't take the non-cooperative situation into consideration.

Recently, game theory has been used extensively to solve various resource competition problems in distributed computering and network systems, such as job allocation, routing, wireless cashing, etc. For instance, Kwok *et al.* [15] presented a hierarchical Grid computing model by taking machine selfishness into consideration and proposed a non-cooperative game theory based intrasite job execution mechanism. They found that the optimal selfish strategy significantly outperforms the Nash selfish strategy. But they assumed truthful valuation revelation in the system. Ghosh *et al.* [6] proposed a fair pricing strategy in mobile grid computing system and designed a game-theoretic algorithm to maximize the revenue of the grid community. Grosu *et al.* [8] used noncooperative game theoretic principles for resource allocation and load balancing in distributed systems, where the problem is formulated by VCG mechanism and overall expected response time is optimized. However, both these work are based on accurate performance measurements, which is different with our model. Khan *et al.* [11] discussed the use of cooperative game theory in grid system and proposed Nash bargain solution to optimize energy consumption and response time. Its assumption

of cooperative environment is different with non-cooperative environment discussed in this paper.

Actually, the above work just discussed game-theoretic resource allocation in Grid computing systems. To the best of our knowledge, game-theoretic modeling and design for non-cooperative resource allocation in virtualized cloud systems is still a relatively unexplored research problem. Wei *et al.* [21] formulated the resource allocation in cloud as a task scheduling problem with QoS constraints and proposed a game-theoretic approximated solution. However, they supposed that the execution time of each subtask is known to the scheduler, which is unrealistic in virtualized cloud systems for performance interference and dynamic environment change make it impossible to get such information clearly. While in our paper, we utilized stochastic approximation approach to model and analyze QoS performance under various virtual resource allocations. An *et al.* [1] investigated the social welfare and regulated demand and supply in cloud computing systems based on evolutionary game theory. Although the proposed evolutionary dynamics can always converge to evolutionary stable strategies, the author did not study the impact of selfish behavior on cloud system performance, and they did not explicitly consider the practical characteristics and constraints of virtualization, either.

VII. Conclusions

In this paper, we investigate the problem of virtual resource allocation in non-cooperative cloud environments, where computing resource are provided dynamically in pay-as-you-go manners and virtual machines can selfishly request resource to maximize its own benefit. Furthermore, in our discussed system, the accurate relationship between VM's valuation function and allocated resource can not be obtained in practice and only "noisy" measurements are available, thus, classical VCG mechanism can not be utilized directly. Through stochastic approximation method, we then modify VCG and propose a new stochastic game-theoretic mechanism to realize efficient and incentive compatible resource allocation in this noisy environments. Though theoretical analysis and simulations, we show that when using our proposed stochastic resource allocation and payment approaches, the selfish VMs can be enforced to report their types truthfully and the virtual resource can be allocated efficiently. Thus, the cloud service providers may utilize this solution to share virtual resource among different users in non-cooperative cloud systems.

In the next step, because the proposed stochastic payment method is very complex, we plan to design an approximation method to strike a balance between computational efficiency and performance accuracy. And it's also important to implement the proposed approaches in practical virtualization cloud system and test its performance with real workload.

Acknowledgements

This work was supported by U. S. NSF under grants CRI-0808232, CNS-0702488, CNS-0914330, CCF-1016966, and NSF of China under grant 61028005.

References

[1] B. An, A. V. Vasilakos, and V. Lesser, "Evolutionary stable resource pricing strategies." in *Proceedings of ACM SIGCOMM 2009 (poster)*, Barcelona, Spain, Aug. 2009.

[2] M. Armbrust *et al.*, "Above the Clouds: A Berkeley View of Cloud Computing." *Technical Report No.UCB/EECS-2009-28, UC Berkeley*, Feb. 2009.

[3] P. Barham *et al.*, "Xen and the art of virtualization." in *Proceedings of 19th ACM Symposium on Operating Systems Principles (SOSP 2003)*, New York, NY, USA, pp. 164-177, Oct. 2003.

[4] T. E. Carroll and D. Grosu, "An Incentive-Compatible Mechanism for Scheduling Non-Malleable Parallel Jobs with Individual Deadlines," in *Proceedings of the 37th International Conference on Parallel Processing (ICPP 2008)*, Portland, Oregon, USA, pp. 107-114, Sept. 2008.

[5] F. Chang, J. Ren, and R. Viswanathan, "Optimal Resource Allocation in Clouds," in *Proceedings of IEEE 3rd International Conference on Cloud Computing (Cloud 2010)*, Miami, FL, USA, pp.418-425, July 2010.

[6] P. Ghosh *et al.*, "A Game Theory based Pricing Strategy for Job Allocation in Mobile Grids", in *Proceedings of 18th International Parallel and Distributed Processing Symposium (IPDPS 2004)*, Santa Fe, NM, USA, Apr. 2004.

[7] J. Gong and C.-Z. Xu, "vPnP: Automated Coordination of Power and Performance in Virtualized Datacenters", in *IEEE International Workshop on Quality of Service (IWQoS'10)*, Beijing, China, June 2010.

[8] D. Grosu and A. T. Chronopoulos, "Noncooperative load balancing in distributed systems." *J. Parallel Distrib. Comput*, vol .65, no. 9, pp. 1022-1034, Sep. 2005.

[9] J. Jia *et al.*, "Revenue generation for truthful spectrum auction in dynamic spectrum access." in *Proceedings of ACM MobiHoc 2009*, New Orleans, LA, USA, May 2009.

[10] S-M. Lee *et al.*, "Fine-grained i/o access control of the mobile devices based on the xen architecture." in *Proceedings of the 15th annual international conference on Mobile computing and networking (MobiCom 2009)*, Beijing, China, pp. 273-284, Sept. 2009.

[11] S. U. Khan, and I. Ahmad, "A Cooperative Game Theoretical Technique for Joint Optimization of Energy Consumption and Response Time in Computational Grids", *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 3, pp. 346 - 360,March 2009.

[12] J. Kiefer and J. Wolfowitz, "Stochastic Estimation of the Maximum of a Regression Function", *Annals of Mathematical Statistics*, pp. 462-466 Sept. 1952.

[13] V. Krishna, *Auction Theory*, Academic Press, 2002.

[14] H. J. Kushner and G. G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*, Springer, 2003.

[15] Y. K. Kwok, K. Hwang, and S. Song, "Selfish Grids: Game Theoretic Modeling and NAS/PAS Benchmark Evaluation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 5, pp. 621-636, May 2007.

[16] R. Nathuji, A. Kansal and A. Ghaffarkhah, "Q-Clouds: Managing Performance Interference Effects for QoS-Aware Clouds", in *Proceedings of EuroSys'10*, Paris, France, April 2010.

[17] P. Padala *et al.*, "Automated Control of Multiple Virtualized Resources." in *Proceedings of EuroSys'09*, Nuremberg, Germany, Apr. 2009.

[18] X. Pu *et al.*, "Understanding Performance Interference of I/O Workload in Virtualized Cloud Environments," in *Proceedings of IEEE 3rd International Conference on Cloud Computing (Cloud 2010)*, Miami, FL, USA, pp.51-58, July 2010.

[19] J. Rao *et al.*, "VCONF: a reinforcement learning approach to virtual machines auto-configuration." in *Proceedings of 6th IEEE International Conference on Autonomic Computing (ICAC '09)*, Barcelona, Spain, June 2009.

[20] J. C. Spall, *Introduction to Stochastic Search and Optimization*, John Wiley & Sons, 2003.

[21] G. Wei, *et al.*, "A Game-theoretic Method of Fair Resource Allocation for Cloud Computing Services," *Journal of Supercomput*, vol. 54, no. 2, pp. 252-269, 2010.

[22] T. Wood *et al.*, "Profiling and Modeling Resource Usage of Virtualized Applications." in *Proceedings of ACM/IFIP/USENIX 9th International Middleware Conference (Middleware 2008)*, Leuven, Belgium, pp. 366-387, Dec. 2008.

[23] Y. Zhang *et al.*, "Integrating Resource Consumption and Allocation for Infrastructure Resources on-Demand," in *Proceedings of IEEE 3rd International Conference on Cloud Computing (Cloud 2010)*, Miami, FL, USA, pp.75-82, July 2010.