Resource Provisioning Policies to Increase IaaS Provider's Profit in a Federated Cloud Environment

Adel Nadjaran Toosi*, Rodrigo N. Calheiros*, Ruppa K. Thulasiram†, and Rajkumar Buyya*

*Cloud Computing and Distributed Systems (CLOUDS) Laboratory
Department of Computer Science and Software Engineering, The University of Melbourne, Australia
Email: adeln@csse.unimelb.com.au, {rnc, rbuyya}@unimelb.edu.au

†Department of Computer Science, University of Manitoba, Canada
Email: tulsi@cs.umanitoba.ca

***Abstract*—Cloud Federation is a recent paradigm that helps Infrastructure as a Service (IaaS) providers to overcome resource limitation during spikes in demand for Virtual Machines (VMs) by outsourcing requests to other federation members. IaaS providers also have the option of terminating spot VMs, i.e, cheaper VMs that can be canceled to free resources for more profitable VM requests. By both approaches, providers can expect to reject less profitable requests. For IaaS providers, pricing and profit are two important factors, in addition to maintaining a high Quality of Service (QoS) and utilization of their resources to remain in the business. For this, a clear understanding of the usage pattern, types of requests, and infrastructure costs are necessary while making decisions to terminate spot VMs, outsourcing or contributing to the federation. In this paper, we propose policies that help in the decision-making process to increase resources utilization and profit. Simulation results indicate that the proposed policies enhance the profit, utilization, and QoS (smaller number of rejected VM requests) in a Cloud federation environment.**

I. Introduction

In recent years, Cloud Computing [1]–[3] has become a consolidated paradigm for delivery of services through on-demand provisioning of virtualized resources. By the emergence of this paradigm, along with support of companies like Amazon, Microsoft, and IBM, the long envisioned dream of computing as a utility finally has come true. Now customers are able to use resources and services in a pay-as-you-go manner from anywhere and at anytime. Among the different methods to deliver Cloud services, Infrastructure as a Service (IaaS) allows Cloud provider to sell resources in the form of Virtual Machines (VMs) to customers.

One of the key motivations for IaaS providers is the possibility of making profit by leveraging their available data center resources to serve potentially thousands of users. Therefore, Cloud providers aspire to accept as many new requests as possible with the main objective of maximizing profit; nevertheless, they must guarantee Quality of Service (QoS) based on the agreed Service Level Agreement (SLA) with customers. Achieving this goal requires efficient resource management strategies.

Most resource management strategies applied by providers hinder their market potential by limiting the amount of resources allocated to requests, so QoS is met in a conservative way. Alternatively, providers may relax QoS related to resource performance, so that physical servers can be over-subscribed and more requests can be served simultaneously.

To be able to offer QoS guarantees without limiting the number of accepted requests, providers must be able to dynamically increase the available resources to serve requests. One possible source for additional resources is idling resources from other providers. In order to enable such scenario, coordination between providers has to be achieved, possibly through establishment of a Cloud federation [4]–[6].

A Cloud federation allows providers to trade their resources through federation regulations. In this paradigm, providers aim to overcome resource limitation in their local infrastructure, which may result in rejection of customer requests, by outsourcing requests to other members of the federation. Moreover, Cloud federation allows underutilized providers to lease part of their resources to other members of the federation, usually at cheaper prices, in order to avoid wasting their non-storable compute resources. Both cases lead to enhancement in profit and elasticity for providers, if this opportunity is properly used. By this we mean that providers should make an intelligent decision about utilization of the federation (either as a contributor or as a consumer of resources) depending on different conditions that they might face.

A challenging condition for providers occurs when they dedicate part of their capacity in the form of spot VMs. Spot VMs are VMs that can be terminated by providers whenever the current value for running such VMs (defined by the provider) exceeds the value that the client is willing to pay for using such resources, as in the case of Amazon EC2 spot instances [7], [8]. This type of VMs can be provided to users at a lower cost than on-demand VMs, usually in the spot market, which works based on supply and demand. Existence of spot VMs certainly benefits IaaS Cloud providers, because spot VMs help them in making profit by increasing the utilization of the data center while waiting for incoming on-demand requests. When a federated Cloud provider receives an on-demand request for VMs and there are no idle resources within the data center, it has to decide between either increasing the spot price and terminating spot VMs, or outsourcing the request to another federation member.

Decision on outsourcing requests or renting part of idling resources to other providers is a complex problem that has been surveyed by several studies [9], [10]. To the best of our knowledge, the work in this paper is the first attempt to incorporate the outsourcing problem with option of terminating spot VMs within a data center. Our main objective is to maximize a provider's profit, by accommodating as many on-demand requests as possible. Our main contribution is proposing policies that help making decisions when providers have different choices regarding incoming requests: rejecting, outsourcing, or terminating spot leases to free resources for more profitable requests[1].

The remainder of this paper is organized as follows. In section II, we define the system model, including Cloud federation as well as customers and providers interaction. Section III describes the proposed policies and formalizes the decision equations. Evaluation and experimental environment are presented in section IV. Related work is reviewed in section V. Finally, we conclude our work and present future works in section VI.

II. System Model

In this section, firstly, we describe the interaction between customers and providers including different types of services. Afterward, the scenario, assumptions, and requirements for Cloud federation are discussed.

A. Interaction between customers and providers

Cloud computing providers, specifically IaaS providers, offer different types of VMs with different QoS and pricing models that help them support different types of applications and fulfill customers' requirements. This variety of QoS and pricing models also gives them more flexibility in resource management and to increase utilization. For example, Amazon Elastic Compute Cloud (EC2) [7], [8] offers three different pricing models, *on-demand*, *reserved* and *spot*.

In this work, we consider the scenario where providers support two different levels of QoS and pricing models which are mostly based on Amazon pricing models. By providers, we mean the set of autonomous IaaS Cloud providers who own a data center and serve a number of customers. Customers are either private users or other SaaS and PaaS providers, who submit requests to IaaS providers for instantiating VMs in either *on-demand* or *spot* types, based on their requirements.

On-demand VMs allow customers to pay for compute capacity by its hourly usage without a long-term commitment. Customers request for VMs, which are provisioned to them if the provider possesses enough resources, otherwise the request is rejected. After instantiation of VMs, customers can retain machines as long as they need them.

Spot VMs allow customers to reduce the cost of using VMs by accepting the risk of being canceled in favor of customers willing to pay more for the same resources. In this model, customers submit a spot VM request, including the number of spot VMs they want to instantiate, and the maximum price they are willing to pay per VM/hour, that is called bid. If the bid exceeds the current price, the request is served and VMs are instantiated. Otherwise, no VM is launched and the request remains in a pending state in the queue until the spot price goes below the bid price. VMs will run until either the customer decides to terminate them or the price goes above the bid.

The provider charges the customer based on the current price, which is calculated based on the minimum bid of running VMs in the system [11]. There is a correlation between resource availability and current price. In case of resource shortage, the provider terminates VMs of low bid and replaces them with higher bid VM requests or on-demand requests. Consequently, bidding at higher price decreases the likelihood of VM termination by the provider.

Here, we consider two types of spot VM requests: *one-time* and *persistent*. One-time are spot requests that are not restarted after termination by providers, whereas persistent are spot requests that are kept in the data center to be re-executed until completed. Providers automatically instantiate new VMs for the persistent request each time the current spot price goes below the bidding price of the persistent request.

The difference between on-demand and spot lies only in the guarantee about resources availability. Other QoS characteristics of VMs (such as memory and CPU power) are the same for both models and they are enforced by providers. Moreover, we assume that the user request has to be entirely served in the same data center.

In this study, we assume that infrastructure providers commit the actual amount of resources required by VMs, regardless of the actual users' usage pattern. This means that the resource manager does not apply methods of consolidation to increase the capacity of the data center [12], [13]. For instance, if two VMs requiring 1 unit of processing (e.g. EC2 compute unit[2]) and they are running on the same physical node with two units of processing, the resource manager will not initialize another VM on that node, even if VMs are not using the total computing power allocated to them.

Considering that under-provisioning of resources may lead to violations in SLAs with customers, providers have to use other methods to serve new on-demand requests and thus increase their profit. One alternative is increasing the spot VM price, which leads to cancellation of spot VMs and more room for on-demand requests. Another alternative is acquiring resources from other Cloud providers that can be used to serve new on-demand requests. To make this scenario possible, it is important that providers engage in a federation so that they sell idle resources to other federation members in a lower price than the customer's price. In exchange, they are also able to buy resources from other members when the demand increases for their resources. Interaction between federation members is detailed next.

[1] In this paper, terms VM and resource are used interchangeably.

[2] 1 EC2 Compute unit (ECU) is equivalent to a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor.

B. Cloud Federation

The Cloud federation scenario used in this paper is presented in Figure 1, which is mostly inspired by the InterCloud project [5]. Each provider is autonomous, and has its own customers. Federation can help providers to absorb overloads due to spikes in demand. At the center of this model, the *Cloud Exchange* service plays the role of information service directory. With the aim of finding available resources from the members of federation, providers send an inquiry to the Cloud Exchange Service in case of shortage of local resources. The Cloud Exchange is responsible for generating a list of providers with corresponding service prices that can handle the current request. Therefore, the resource availability and price list is used by providers to find suitable providers where requests can be redirected to.

Decision on allocating additional resources from a federated Cloud provider is performed by a component called *Cloud Coordinator*. The amount of idling capacity each provider shares with other members and the way providers price their resources is also decided by the Cloud Coordinator. These decisions significantly affect the profit of providers, and thus they are of paramount importance for the successful adoption of the federation paradigm by Cloud providers. Moreover, agreements between federation members are necessary in order to make the federation profitable to all its members. We call these agreements *Federation Level Agreement* (FLA).

In this study, FLA requires that each provider dynamically prices its contributed resources (VMs) based on the idling capacity of its data center. Therefore, the instant federation price of a resource per hour can be computed as

$$F = \frac{M_p - M_{idle}}{M_p} \cdot (F_{max} - F_{min}) + F_{min} \ , \qquad (1)$$

where F is the resource's federation price; M_p and M_{idle} are total capacity and idling capacity of the provider data center respectively, F_{max} is the on-demand VM price to customers and F_{min} is the minimum profitable price for the provider. The provider does not sell resources for prices smaller than F_{min}. We discuss later in Section IV about F_{min} that is anything larger than the price that compensates costs of keeping nodes of the data center up.

This pricing mechanism facilitates load balancing between federated providers, since it results in cheaper price for providers with larger amount of resources.

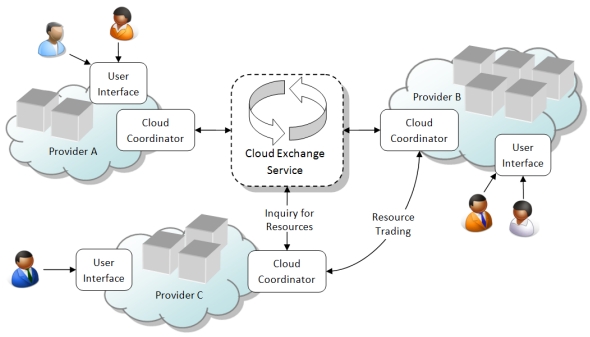


Fig. 1. Cloud Federation Architecture

A relevant issue about this mechanism is whether it reveals sensitive information about the provider, such as a provider's actual resource utilization (which might lead to inferences about a provider's revenue). Nevertheless, Equation 1 does not reveal such sensitive information, since providers are free to advertise a subset of their resources, and thus members cannot determine the overall utilization of other member's resources.

Considering the above scenario and assumptions, various policies are proposed in the next section to investigate how different decisions made by providers affect their profit and reputation with customers, when they provide the mentioned types of services.

III. Proposed Policies

A Cloud provider may receive a request for on-demand VMs from its customer. Due to unavailability of resources, this request may not be served locally without violating QoS for other running VMs. Under such circumstances, the provider might decide to cancel a sufficient number of spot VMs (lowest bid first) to be able to accommodate more profitable on-demand requests. Also providers may look up to the federation, which provides an opportunity for outsourcing local requests to other members. In this case, an on-demand request received by a Provider A is actually served with resources from a Provider B. Provider B charges Provider A at the federation cost as given in Equation 1, which is typically lower than the prices that both Provider A and Provider B would charge the on-demand customers.

We propose policies that help the provider to increase profit, resource utilization, and user satisfaction, when providers are federation members and benefit from outsourcing requests and also they are able to terminate spot VMs for serving on-demand requests. These policies only address the possibility of outsourcing on-demand requests. Outsourcing spot requests is not considered in the current work, since the proposed policies are not designed to handle highly fluctuating prices of spot VMs.

To this end, we describe the proposed policies, which differ in how they handle new on-demand requests when they cannot be served by available local resources.

1) Non-Federated Totally In-house (NFTI): In this policy, firstly providers consider termination of spot VMs with lower bids to accommodate a more profitable on-demand request. If this action does not release enough resources for the new on-demand request, it is rejected. This policy is considered as a base policy in order to allow verification of maximum profit a provider can make without the federation.

2) Federation-Aware Outsourcing Oriented (FAOO): In this policy, each fully utilized provider firstly checks the Cloud exchange service for available offers by other members. Then, it outsources the request to the provider that offers the cheapest price. If outsourcing is not possible, Spot VMs are terminated as a last resort to accommodate the new on-demand request. This policy is considered to show whether outsourcing is always a profitable decision when fully utilized providers

receive an on-demand request and spot termination is also possible.

3) Federation-Aware Profit Oriented (FAPO): This policy compares the profit of outsourcing with termination of spot VMs. The idea behind this algorithm is that, in one hand, termination of spot VMs result in profit loss, and on the other hand, replacing spot VMs by on-demand ones increase the profit. Moreover, termination of spot VMs may result in spot price increase. In this policy, decisions are made based on $P(t)$, the *instant profit* of the provider in time t , which is given by

$$P(t) = R(t) - C(t) \ , \tag{2}$$

where $R(t)$ and $C(t)$ are revenue and cost at time t, respectively. $R(t)$ can be obtained as follow:

$$R(t) = R_o(t) + R_s(t) + R_{fed}(t) + R_{out}(t) \ , \tag{3}$$

where $R_o(t)$, $R_s(t)$, $R_{fed}(t)$ and $R_{out}(t)$ are revenue of on-demand, spot, contributed to federation and outsourced resources. By contributed to federation resources, we mean those local resources used by other members of the federation to serve their customers. $R_o(t)$ and $R_{out}(t)$ are calculated based on the following equations:

$$R_o(t) = vm_o(t) \cdot F_o \ , \tag{4}$$

$$R_{out}(t) = vm_{out}(t) \cdot F_o \ , \tag{5}$$

where F_o is the on-demand resource price per resource per hour, which is a constant value for all providers and $vm_o(t)$ and $vm_{out}(t)$ are the number of on-demand VMs running locally and outsourced VMs, respectively. $R_s(t)$ is given by

$$R_s(t) = vm_s(t) \cdot F_s(t) \ , \tag{6}$$

where $vm_s(t)$ is the number of running spot VMs and $F_s(t)$ is the price of the spot VMs at time t.

$R_{fed}(t)$ is calculated based on the summation of all VMs contributed to federation according to the following equation:

$$R_{fed}(t) = \sum_{i=1}^{vm_{fed}(t)} F_{fed_i} \ , \tag{7}$$

where $vm_{fed}(t)$ is the number of VMs contributed to federation and F_{fed_i} is the associated price for each VM contributed to federation.

In order to determine $C(t)$ in Equation 2, both operational cost and cost of outsourcing are considered. Therefore, $C(t)$ is given by

$$C(t) = C_p(t) + C_{out}(t) \ , \tag{8}$$

where $C_P(t)$ is the operational cost, which includes cost of acquiring and operating the data center nodes (i.e, hardware and software acquisition, staff salary, power consumption, cooling costs, physical space, amortization of facilities, etc). $C_{out}(t)$ is the cost of outsourced VMs that a provider pays to federation members hosting its requests. Without loss of generality, we can assume that a constant value for $C_p(t)$ represents various combination of the costs and effect of change of constituting parameters. Further deeper analysis of the cost in the constituting parameters on $C_p(t)$ is left as a future analytical work. Here, a constant value for $C_p(t)$ is assumed, whereas $C_{out}(t)$ depends on the renting price of the outsourced VMs. $C_{out}(t)$ is given by

$$C_{out}(t) = \sum_{i=1}^{vm_{out}(t)} F_{out_i} \ , \tag{9}$$

where F_{out_i} is the amount paid for each outsourced VM vm_i.

By putting all the above equations together, $P(t)$ is calculated as follows:

$$\begin{aligned} P(t) =& vm_o(t) \cdot F_o + vm_s(t) \cdot F_s(t) + \sum_{i=1}^{vm_{fed}(t)} F_{fed_i} \\ &+ vm_{out}(t) \cdot F_o - C_p(t) - \sum_{i=1}^{vm_{out}(t)} F_{out_i}. \end{aligned} \tag{10}$$

Considering Equation 10, *FAPO* policy has two choices in order to improve provider's profit. When a request for n on-demand VMs arrives at time t, and local infrastructure can only accommodate m VMs ($m < n$), it can either decide to terminate the $n - m$ lowest value bids spot VMs or outsource the new request. To choose the best approach, *FAPO* policy estimates the instant profit in future time t' for both approaches ($t' = t + \varepsilon,\ \varepsilon \to 0$).

The spot price is always set to the value of the lowest bid being served. Therefore, termination of the $n - m$ lowest bid spot VMs may increse spot price from $F_s(t)$ to $F_s(t')$ ($F_s(t) \geq F_s(t')$). But, according to the proposed model, this increment only affects those spot requests whose accounting period expires on or after t', because the price of a spot VM is set at the beginning of each accounting period, which is one hour. Assuming k as the number of spot VMs whose accounting period start on t', we replace $k \cdot F_s(t)$ with $k \cdot F_s(t')$ in Equation 10 to obtain estimated profit of terminating spots after accommodating n on-demands VMs, $P_1(t')$ as

$$\begin{aligned} P_1(t') =& (vm_o(t) + n) \cdot F_o + vm_{out}(t) \cdot F_o - C_p(t) \\ &+ (vm_s(t) - (n - m) - k) \cdot F_s(t) + k \cdot F_s(t') \\ &+ \sum_{i=1}^{vm_{fed}(t)} F_{fed_i} - \sum_{i=1}^{vm_{out}(t)} F_{out_i}. \end{aligned} \tag{11}$$

Moreover, in the case of outsourcing, estimated profit at time t', $P_2(t')$, is defined as

$$\begin{aligned} P_2(t') =& vm_o(t) \cdot F_o + vm_{out}(t) \cdot F_o \\ &+ n \cdot F_o - C_p(t) + vm_s(t) \cdot F_s(t) \\ &+ \sum_{i=1}^{vm_{fed}(t)} F_{fed_i} - \sum_{i=1}^{vm_{out}(t)} F_{out_i} - n \cdot F_{offer} \ , \end{aligned} \tag{12}$$

where F_{offer} is the lowest offered price in the federation. The policy compares $P_1(t')$ and $P_2(t')$ to make its decision. Therefore, similar terms can be eliminated from both equations and they are calculated as follows:

$$P_1(t') - P_2(t') = k \cdot F_s(t') - (n - m + k) \cdot F_s(t) + n \cdot F_{offer}. \quad (13)$$

Consequently, the policy decides to outsource requests when $P_1(t') - P_2(t') < 0$; if $P_1(t') - P_2(t') \geq 0$ termination of spot VMs is more profitable and it will be recommended by the policy.

It is worth noting that none of the policies takes into account the duration of the request, because the provider does not have information about how long current VMs will remain in the system. Strategies that consider prediction of future resource availability to drive decisions will be explored in future works.

IV. Evaluation

This section presents an evaluation of the policies presented in the previous section. First, we describe simulation settings and performance metrics, and then experimental results are presented and discussed.

A. Experimental Settings

The experiments presented in this section were developed using CloudSim [14] discrete-event Cloud simulator.

The simulated Cloud scenario is composed of a federation containing IaaS Cloud providers. The number of providers is one of the simulation parameters, and we evaluated the effect of the policies considering different number of federation members.

For the sake of simplicity, we can assume only one type of VM is offered by providers. The VM configuration is inspired by Amazon EC2 small instances (1 CPU core, 1.7 GB RAM, 1 EC2 Compute Unit, and 160 GB of local storage). Adding different types to the model can be considered as an extension of the current work.

Providers follow the pricing model of Amazon Web Services (AWS) [7], [8]. That is, all providers charge their customers based on hourly usage, at the cost of $0.085 per hour per on-demand VM. In the case of spot VMs, the provider charges customers based on the spot price, which fluctuates periodically according to the minimum bid in the system and resource availability. The price for each spot VM is set at the beginning of each VM-hour for the entire hour.

On the customer side, for the purpose of the bidding algorithm of spot requests, we assume that customers do not bid higher than the on-demand price, because higher bids behave like on-demand requests (they are never canceled) but generate even higher profit for the providers. Our bidding algorithm generates a uniformly-distributed random value between the minimum of bid $0.020 and maximum of $0.085. The minimum price is set in such a way that the value offered by customers is still enough to cover operational costs of serving the request, even though it can result in no profit for the provider.We also used $0.020 and $0.085 for F_{min} and F_{max} respectively in Equation 1 for pricing resources contributed by each provider in the federation.

Each simulated data center contains 128 servers, and each one supports 8 VMs. So, each provider is able to concurrently host 1024 VMs. We assumed that operational costs are constant and the same for all the providers, so they are not considered in the experiment.

For the sake of accuracy, each experiment is carried out 20 times by using different workloads and the average of the results is reported. We explain the workload setup details in the following subsection.

B. Workload setup

Due to lack of publicly available workload models and real traces of IaaS Clouds, we apply a nine day long workload generated by the Lublin workload model [15]. To adapt this model to our scenario, we consider the number of nodes in each Lublin request as the number of VMs of the request, and this number is limited to 32 simultaneous VMs for each request. The first 12 hours of simulation and the last 36 hours are warm-up and cool-down periods respectively, and they are discarded from results. Therefore, a one week long period of simulation is considered.

The only parameters of the Lublin workload model changed for these experiments are job runtime parameters. We changed the first parameter of the Gamma distribution for runtime in Lublin model from the default value of 4.2 to 7.2 in order to generate longer VM requests. The user workload submitted to each provider is generated based on the above configuration. Providers are equidistantly distributed among time zones. Because the generated load has a daily cycle with peak hours, providers' loads vary. For example, while provider A is at its peak period, provider B will be at its off-peak time.

We intend to study the behavior of different policies in different situations. For this purpose, effects of four input parameters on the providers are investigated.

The first input parameter is the system load. The difference in the proposed policies lies on the action taken at the time at which a provider's resources are fully utilized and requires additional resources. In this direction, the arrival rate of requests has been selected as the most suitable parameter to adjust the load of a provider. With the intention of changing providers load, arrival rate of requests has been changed by varying the $aarr$ parameter of the model between 8.2 and 6.4. $aarr$ is the shape parameter of the Gamma distribution.

Another parameter that impacts the policies' performance is the ratio of spot requests to total requests (on-demand plus spot requests). We named this parameter β ($0 \leqslant \beta \leqslant 100\%$), and this is defined as follows. After generating the 9 days long workload, we randomly select some of the generated requests as spot requests so β percent of the requests are spot requests.

The third parameter evaluated, α, contains the rate of spot requests that are persistent. It impacts the provider's profit because it determines the amount of requests that are going to be kept in the provider to be served when resources become available.

Finally, *Number of providers* is another parameter that is considered. This parameter is important because it increases the chance of members finding other members with lower load to select as the target of outsourced requests.

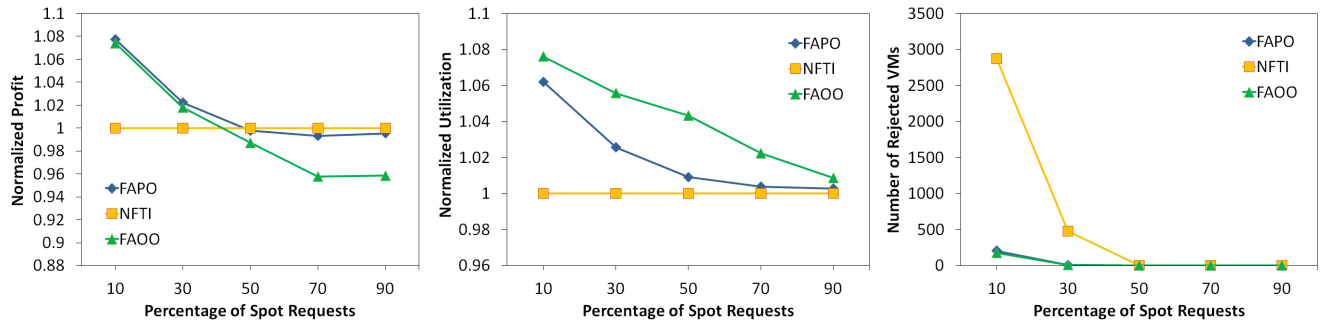


Fig. 2. Impact of percentage of spot requests on (a) Profit (b) Utilization, and (c) Number of rejected on-demand VMs for a provider with different policies.

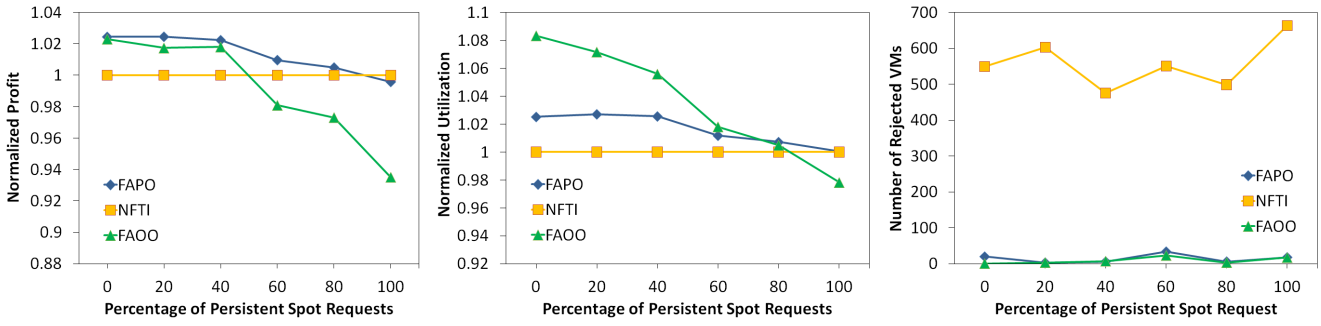


Fig. 3. Impact of percentage of persistent spot requests on (a) Profit (b) Utilization, and (c) Number of rejected on-demand VMs, for a provider with policies.

C. Performance Metrics

We applied the following metrics to analyze the impact of the proposed policies in the providers:

1) *Profit.* This metric is calculated for each provider and it is defined as the achieved revenue during a time period minus the cost incurred in the same time period. In our results, we ignore operational costs. Therefore,

$$Profit(\Delta t) = Revenue(\Delta t) - Cost_{out}(\Delta t) \ , \quad (14)$$

where $Revenue(\Delta t)$ is the revenue obtained during Δt including on-demand, spot, contributed to the federation, and outsourced requests, whereas $Cost_{out}(\Delta t)$ is the cost of the outsourcing VMs at the same period.

2) *Utilization.* This metric is defined as the ratio between the number of hours of VMs used by requests (both local and contributed to the federation) and the maximum number of hours of VMs in a time period.

$$Utilization(\Delta t) = \frac{\sum_{i=1}^{vm} runtime(vm_i)}{vm_{max} \cdot \Delta t} \ , \quad (15)$$

where vm is the total number of VMs including on-demand, spot and contributed to federation VMs and vm_{max} is the maximum number of VMs that a provider can run simultaneously in its data center. $runtime(vm_i)$ shows the corresponding runtime for each VM.

3) *Number of rejected on-demand VMs.* This metric shows the number of on-demand VMs rejected. It considers only on-demand requests, because providers never reject spot VM requests which are kept in waiting queues until the bid price is reached. It does not show the number of rejected requests because each request may contain demand for more than one VM. We select this metric instead of rejected number of requests because it better shows potential revenue losses.

D. Results

Results presented for profit and utilization are the normalized values for each metric using the result obtained for the *NFTI* policy as the base value. Since the *NFTI* policy reflects the situation where providers do not explore capacities of the federation, the use of normalized values allows us to quantify the benefits of federation-aware policies on each provider.

1) Impact of percentage of spot requests: The first experiment aimed at evaluating how changes in the ratio between spot and on-demand requests affect performance metrics.

This experiment's simulation scenario consists of 5 providers, each one with a workload whose $aarr$ value is 6.7 and 40% of the spot requests as persistent requests. The workload for each provider was generated as described in Section IV-B, but the percentage of spot VM requests (β) varied between 10% and 90% of the total amount of requests submitted to each provider. Results for this scenario are presented in Figure 2.

Figure 2(a) shows that, for smaller amount of spot VM requests, exploiting the potential of federation for outsourcing or contributing to the federation helps providers to enhance profit. After the point that 50% of requests are spot requests, providers are not able to increase profit since spot VM price is the most effective factor on the profit. Moreover, when a significant number of running VMs in the data center belongs

to spot requests, providers are able to absorb spikes in demand just by terminating spot VMs.

Besides the fact that the presence of more spot VMs in data center decreases the potential of the federation for making profit, the *FAPO* policy has better performance comparing to the *FAOO* policy with higher profit and less utilization.

As shown in Figure 2(b), the utilization achieved with the *FAOO* and *FAPO* policies are always higher than with *NFTI*, because providers dedicate part of their capacity to the federation. However, *FAPO* witnesses smaller utilization, although it generates more profit than *FAOO*. This is due to the fact that *FAOO* contributes to the federation more than *FAPO*, and also terminates less spot VMs than *FAPO*, which results in lower spot price.

As we expected, by increasing the amount of spot requests, it is less likely that on-demand requests while rejected, because the chance of finding a spot VM to be terminated increases. Moreover, Figure 2(b) shows that less rejections occur for those policies that benefit from federation, especially when the percentage of spot VMs is low.

2) Impact of percentage of persistent spot requests: This second experiment demonstrates the effects of changing the percentage of persistent spot VMs requests on providers. This experiment's simulation scenario consists of 5 data centers, $aarr$ value of 6.7 and 30% of spot requests among the total number of requests for each provider. The workload for each provider was generated as described in Section IV-B, but the percentage of persistent spot VMs (α) varied between 0% and 100% of the total amount of spot requests submitted to each provider. Results for this scenario are presented in Figure 3.

More persistent spot VM requests results in less usage discontinuation, since even after termination of spot VMs the system retains the requests themselves, which can be served in a later stage. Percentage of persistent spot VMs is significant, since termination of spots VMs that are persistent does not cause load and revenue loss, but increases the current spot price, and consequently provider profit.

Therefore, according to Figure 3(a), profit making of the *FAOO* policy drastically decreases after a point where 50% of spot market is used by persistent spot VMs, because this policy causes less spot VMs termination than other policies. The *FAPO* policy shows better performance in comparison to other policies regarding profit, as it benefits from outsourcing in lower percentage of persistent spot VMs, and termination of spot VMs in higher percentage of persistent spot VMs.

It is expected that a smooth increase in utilization will occur when there are more persistent spot VMs, however, it is not observable in Figure 3(b). This is because spot termination does not result in losing part of VM requests. Thus, when there is a higher percentage of persistent spot VMs, policies converge to a specific utilization point because the total load remains constant in higher persistency of spot VMs.

Finally, the percentage of persistent spot requests does not have a significant effect on the number of rejected on-demand VMs. However, due to the lack of outsourcing choice, a higher number of rejected requests is seen when *NFTI* is applied.

3) Impact of the load: The third experiment evaluates the effect of load variation. This experiment's simulation scenario consists of 5 data centers, α of 40% and β of 30%. The workload for each provider was generated as described in Section IV-B, but the $aarr$ parameter of the Lublin workload varied between 8.2 and 6.4 to vary the number of generated requests for all the providers.

Figure 4 shows the impact of varying the $aarr$ parameter, which results in a different number of requests, on the proposed policies. Since our policies are triggered when the provider is fully utilized, load is the most influential parameter in our experiments. By increasing the number of total requests, and consequently provider's load, the provider frequently experiences a situation where it has to decide between outsourcing and terminating spot VMs. According to Figures 4(a) and 4(b) *FAPO* and *FAOO*, which support outsourcing, have higher profit and utilization by increasing load. However, the *FAPO* policy outperforms the *NFTI* policy by having a higher profit with smaller utilization. The difference between *FAPO* and *FAOO* is more observable at higher loads.

By decreasing the number of requests, the number of on-demand VM rejection also decreases, because providers are subject to a lower load. The *NFTI* policy is more sensitive to this effect, because it does not support the outsourcing option.

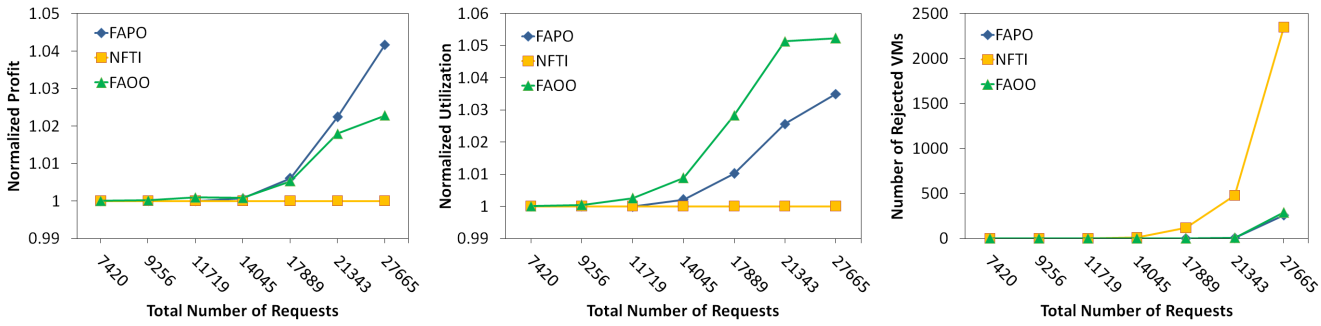
4) Impact of number of providers in the federation: This experiment evaluates the impact of number of participants in the federation on the results delivered by each policy. In this experiment, α is 40%, β is 30%, and $aarr$ is 6.7. The experiment was repeated with 3, 5, and 7 providers. The results are presented in Figure 5.

By increasing the number of providers, policies with an outsourcing option have a smaller number of rejected on-demand VMs, because it is more likely that a provider can be found who could serve the outsourced request. Increasing the number of providers does not have any impact in *NFTI*, as in this policy there is no interaction with federation members. For *FAOO* and *FAPO*, an increase in the number of providers in the federation results in lower profit, because of better matching in supply and demand. That is, the offer price for contributing to the federation falls and outsourcing becomes more profitable.

V. Related Work

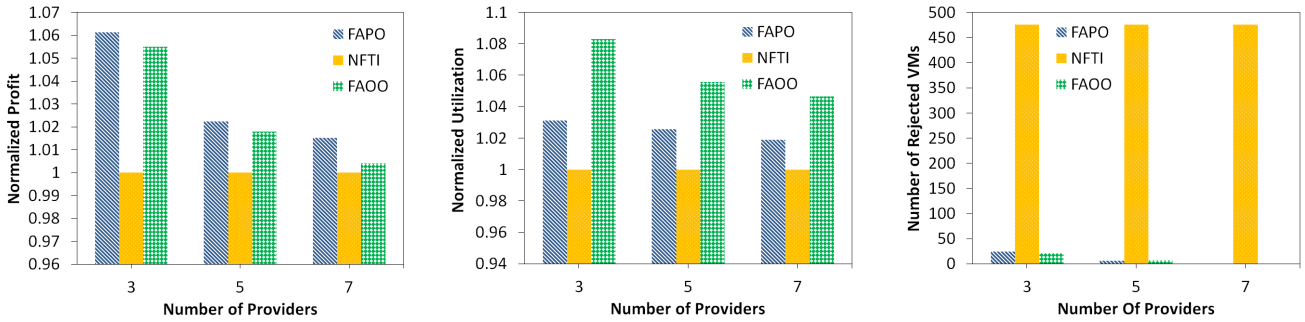
Despite several recently proposed platforms for Cloud federation [4], [5], [16], with different motivations and incentives for parties to join it, many fundamental problems and questions about federation remain unanswered. One of these problems is deciding when providers should outsource their local requests to other participants of the federation or how many and at what price they should provide resources to the federation. The outsourcing problem is not considered only in the context of federated Clouds; it was also investigated as a way of increasing capacity or scalability of applications in hybrid Clouds [17], distributed grid environment [18], and clusters [19].

Goiri et al. [9] present a profit-driven policy for decisions related to outsourcing or selling idling resources. On that



(a) (b) (c)

Fig. 4. Impact of load on (a) Profit (b) Utilization, and (c) Number of rejected on-demand VMs, for a provider with different policies.



(a) (b) (c)

Fig. 5. Impact of number of providers on (a) Profit (b) Utilization, and (c) Number of rejected on-demand VMs for a provider with different policies.

approach, providers have the option of shutting down unused nodes of the data center to save power. However, they did not take into account different types of VMs (e.g. on-demand and spot) and possible actions like terminating low priority leases.

A user satisfaction-oriented scheduling algorithm for service requests was proposed by Lee et al. [20]. Such an algorithm tries to maximize Cloud providers' profit by accepting as many service requests as possible, as long as QoS is kept at a certain level. In this regard, contracting with other service providers was taken into account as a method to avoid rejection of user requests. One of the main differences between this and our approach is that we specifically focus on federation of IaaS providers that serve requests for VMs. Moreover, we claim that federation, rather than being merely a technique for avoiding service rejection at provider level, can also be a source of profit for providers by allowing them to negotiate otherwise wasted resources at competitive prices.

The problem of how to value resources and how price may impact utilization is not a trivial one. Current public Cloud service providers like Amazon, GoGrid and RackSpace usually adopt fixed pricing strategies for the infrastructure services they provide. However, fixed pricing models are not suitable for federated environments as a policy to be applied between its participants, because it neither reflects current market price of resources due to dynamism in supply and demand nor generates any incentives for providers to join the federation. Dynamic pricing of resources, however, lies outside the scope of current work, and has been a subject of other studies [21]. Hence, in this work, a policy based on the provider utilization, is applied by federated providers to dynamically value resources.

The subject of leveraging spot VMs has recently attracted considerable attention. Andrzejak et al. [22] have proposed a probabilistic decision model to help users decide how much to bid for a certain spot instance type in order to meet a certain monetary budget or a deadline. Yi et al. [23] proposed a method to reduce monetary costs of computations using Amazon EC2s spot instances for resource provisioning. These works consider methods for increasing customers' benefit in using spot VMs, while we are interested in better resource provisioning policies for providers in the presence of spot VMs. Moreover, the problem of dynamic allocation of data center resources to different spot markets to maximize cloud provider's total revenue has been investigated by Zhang et al. [11].

A few works consider the application of market-oriented mechanisms in federated environments [24], [25]. These mechanisms mostly promote fairness and ensure mutual benefits for parties involved in the federation. Study and development of such techniques motivate both resource providers and resource consumers to join and stay in the market. Niyato et al. [26] study the cooperative behavior of multiple cloud providers and propose a cooperative game model. Our work, on the other hand, is focused on specific policies to be applied by Cloud IaaS resource providers to decide when to buy computational resources and how resources should be made available in the market for other IaaS providers.

VI. Conclusion and Future Work

We proposed policies to enhance IaaS providers' profit when the provider is member of a Cloud federation. Since each provider has restricted amount of capacity, increase in load may overload a provider's data center and may result in QoS

violation or users' request rejection. Providers that support different types of QoS and pricing scheme for VMs (e.g. on-demand and spot VMs) have the possibility of canceling their terminable less profitable VMs (e.g. spot VMs) in favor of more profitable requests (e.g. on-demand VMs). Providers can also benefit from federation by outsourcing requests to other members of the federation with lower load.

Our experiments compared the proposed policies to determine the impact of a provider's decision on its performance metrics. Evaluated parameters include the ratio of spot VMs to the total load, percentage of persistent spot VMs, number of providers in the federation and provider's load. Results showed that our policies help providers to enhance profit and to reject less requests, while they keep utilization at an acceptable level.

Experimental results also allow us to derive some guidelines for providers. Running on-demand requests locally is more profitable if the provider has high ratio of spot VMs and termination of spot VMs may lead to less discontinuation of service by customers. Moreover, outsourcing is more profitable when spot VMs are scarce and termination of them may result in discontinuation of using the services by customers. Furthermore, federation also helps underutilized providers in making profit by selling idling resources to other members.

We plan to investigate the impact of strategies that also include shutting down unused hosts of the data centers to save electric power consumption, in addition to termination of spot VMs and outsourcing on-demand VM requests. Furthermore, strategies that consider prediction of future resource availability to drive decisions will be explored as an extension of this work. Proposing policies for dynamic pricing of resources to offer idling capacity of the data center will also be explored.

REFERENCES

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

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, Jun. 2009.

[3] W. Voorsluys, J. Broberg, and R. Buyya, *Introduction to Cloud Computing*, R. Buyya, J. Broberg, and A. Goscinski, Eds. John Wiley & Sons, Inc., 2011.

[4] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich, and F. Galan, "The Reservoir model and architecture for open federated Cloud computing," *IBM Journal of Research and Development*, vol. 53, no. 4, pp. 1–11, Jul. 2009.

[5] R. Buyya, R. Ranjan, and R. N. Calheiros, "InterCloud: Utility-oriented federation of Cloud computing environments for scaling of application services," in *Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP'10)*, ser. Lecture Notes in Computer Science, vol. 6081. Busan: Springer, May 2010, pp. 13–31.

[6] K. Keahey, M. Tsugawa, A. Matsunaga, and J. A. B. Fortes, "Sky computing," *IEEE Internet Computing*, vol. 13, no. 5, pp. 43–51, Oct. 2009.

[7] "Amazon Elastic Compute Cloud (Amazon EC2)," http://aws.amazon.com/ec2.

[8] J. Varia, "Best practices in architecting Cloud applications in the AWS Cloud," in *Cloud Computing: Principles and Paradigms*, R. Buyya, J. Broberg, and A. Goscinski, Eds. Wiley Press, 2011, ch. 18, pp. 459–490.

[9] Í. Goiri, J. Guitart, and J. Torres, "Characterizing Cloud federation for enhancing providers' profit," in *Proceedings of the 3rd International Conference on Cloud Computing*. Miami: IEEE Computer Society, Jul. 2010, pp. 123–130.

[10] Y. C. Lee, C. Wang, A. Y. Zomaya, and B. B. Zhou, "Profit-driven service request scheduling in Clouds," in *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. Melbourne: IEEE Computer Society, May 2010, pp. 15–24.

[11] Q. Zhang, E. Gürses, R. Boutaba, and J. Xiao, "Dynamic resource allocation for spot markets in Clouds," in *Proceedings of the 2nd Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE '11)*. Boston: USENIX, Mar. 2011.

[12] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper, "Resource pool management: Reactive versus proactive or let's be friends," *Computer Networks*, vol. 53, no. 17, pp. 2905 – 2922, 2009, virtualized Data Centers.

[13] Í. Goiri, F. Julià, J. Fitó, M. Macías, and J. Guitart, "Resource-level QoS metric for CPU-based guarantees in Cloud providers," in *Economics of Grids, Clouds, Systems, and Services*, ser. Lecture Notes in Computer Science, J. Altmann and O. Rana, Eds. Springer Berlin / Heidelberg, 2010, vol. 6296, pp. 34–47.

[14] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of Cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[15] U. Lublin and D. G. Feitelson, "The workload on parallel supercomputers: modeling the characteristics of rigid jobs," *Journal of Parallel and Distributed Computing*, vol. 63, no. 11, pp. 1105 – 1122, 2003.

[16] L. Rodero-Merino, L. M. Vaquero, V. Gil, F. Galán, J. Fontán, R. S. Montero, and I. M. Llorente, "From infrastructure delivery to service management in Clouds," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1226–1240, Oct. 2010.

[17] R. V. den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in Hybrid IaaS Clouds for deadline constrained workloads," in *IEEE International Conference on Cloud Computing*. Los Alamitos, USA: IEEE Computer Society, 2010, pp. 228–235.

[18] H. Kim, Y. el Khamra, S. Jha, and M. Parashar, "Exploring application and infrastructure adaptation on hybrid Grid-Cloud infrastructure," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. Chicago: ACM, June 2010, pp. 402–412.

[19] M. D. de Assunção, A. di Costanzo, and R. Buyya, "A cost-benefit analysis of using Cloud computing to extend the capacity of clusters," *Cluster Computing*, vol. 13, no. 3, pp. 335–347, Sep. 2010.

[20] Y. Lee, C. Wang, J. Taheri, A. Zomaya, and B. Zhou, "On the effect of using third-party Clouds for maximizing profit," in *Algorithms and Architectures for Parallel Processing*, ser. Lecture Notes in Computer Science, C.-H. Hsu, L. Yang, J. Park, and S.-S. Yeo, Eds. Springer Berlin / Heidelberg, 2010, vol. 6081, pp. 381–390.

[21] M. Mihailescu and Y. M. Teo, "Dynamic resource pricing on federated Clouds," in *IEEE International Symposium on Cluster Computing and the Grid*. Los Alamitos, USA: IEEE Computer Society, 2010, pp. 513–517.

[22] A. Andrzejak, D. Kondo, and S. Yi, "Decision model for Cloud computing under SLA constraints," in *Proceedings of the IEEE International Symposium on Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS)*, Aug. 2010, pp. 257 –266.

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

[24] E. Gomes, Q. Vo, and R. Kowalczyk, "Pure exchange markets for resource sharing in federated Clouds," *Concurrency and Computation: Practice and Experience*, vol. 23, 2011.

[25] B. Song, M. M. Hassan, and E.-N. Huh, "A novel Cloud market infrastructure for trading service," in *Proceedings of the 9th International Conference on Computational Science and Its Applications (ICCSA)*. Suwon: IEEE Computer Society, Jun. 2009, pp. 44–50.

[26] D. Niyato, A. V. Vasilakos, and Z. Kun, "Resource and revenue sharing with coalition formation of Cloud providers: Game theoretic approach," in *Proceedings of IEEE CCGrid*. Newport Beach: IEEE, May 2011, pp. 215–224.