Economic model of a Cloud provider operating in a federated Cloud

Íñigo Goiri · Jordi Guitart · Jordi Torres

Abstract Resource provisioning in Cloud providers is a challenge because of the high variability of load over time. On the one hand, the providers can serve most of the requests owning only a restricted amount of resources, but this forces to reject customers during peak hours. On the other hand, valley hours incur in under-utilization of the resources, which forces the providers to increase their prices to be profitable. Federation overcomes these limitations and allows providers to dynamically outsource resources to others in response to demand variations. Furthermore, it allows providers with underused resources to rent them to other providers. Both techniques make the provider getting more profit when used adequately. Federation of Cloud providers requires having a clear understanding of the consequences of each decision. In this paper, we present a characterization of providers operating in a federated Cloud which helps to choose the most convenient decision depending on the environment conditions. These include when to outsource to other providers, rent free resources to other providers (i.e., insourcing), or turn off unused nodes to save power. We characterize these decisions as a function of several parameters and implement a federated provider that uses this characterization to exploit federation. Finally, we evaluate the profitability of using these techniques using the data from a real provider.

1 Introduction

Nowadays, providers have start offering applications, data, and computing resources over the Internet as utilities, which is being called Cloud Computing. A flavor of this paradigm (i.e., IaaS) consists of offering raw computing resources where the users can deploy and execute their services while only paying for the resources they use. In this way, the users can access the services anytime and anywhere, avoiding hardware acquisition costs, dealing with software licenses, upgrades management, etc.

These providers usually offer their computing resources in the form of Virtual Machines (VM). This allows that different users can share the provider's resources while being also isolated among them. Resource sharing is a key technique for provider's profitability. However, the provider has to ensure that, even sharing resources, the performance goals agreed with each customer can be met.

Resource sharing in Cloud providers has to deal with an important problem: the amount of users varies over time. Typically, there are peaks during daytime and valleys during night and weekend (Fig. 1). Former resource management approaches for Cloud providers hindered their market potential by considering a limited amount of resources.
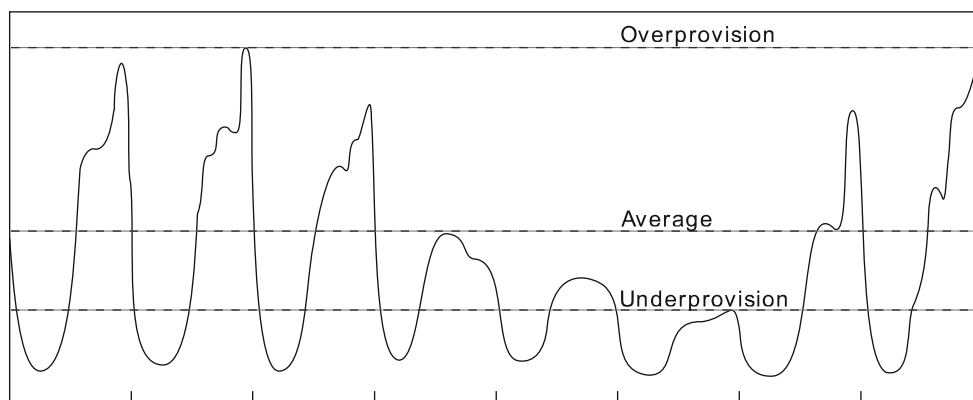
Í. Goiri (✉) · J. Guitart · J. Torres
Universitat Politecnica de Catalunya and Barcelona
Supercomputing Center, Jordi Girona 31,
08034 Barcelona, Spain
e-mail: igoiri@ac.upc.edu

J. Guitart
e-mail: jguitart@ac.upc.edu

J. Torres
e-mail: torres@ac.upc.edu

Fig. 1 Cloud provider's workload over a week

On the one hand, if the provider wants to provide service to all the users, it must have enough resources to support the peaks. In this situation, the provider is overprovisioned and its resources are highly underutilized during long periods. Although operational costs can be reduced during these underutilized periods by turning idle nodes off (Pinheiro et al. 2001; Elnozahy et al. 2002), this technique does not reduce capital costs related to the purchase and hosting of IT equipment, which need to be amortized. This forces the providers to increase their prices to be profitable, which makes them less competitive in the Cloud market. Hence, efficient resource management techniques that avoid overprovisioning are needed.

On the other hand, owning just enough resources to support the average number of users (i.e. underprovisioning) reduces underutilization and saves infrastructure costs. However, if a provider has not enough local resources to fulfill its customers' requirements, it should start denying the acceptance of new customers or canceling services that are already running on the system (e.g., those that are providing less revenue to the provider). This has further implications than just losing the revenue from some services, because it also implies a loss of reputation and therefore a loss of future customers (Armbrust et al. 2010).

The most successful approach to deal with these problems caused by load variability over time in the Cloud is based on dynamic resource sharing among providers, because this is the only way to reduce capital expenses and transform them into operational ones. Whereas this resource sharing can take many forms (Juan et al. 2011), in this paper we focus on federated Cloud scenarios (Rochwerger et al. 2009). Nevertheless, considering other scenarios involving several providers would only require slight modifications on our characterization.

In a federated Cloud, providers running services that have complementary resource requirements over time can mutually collaborate to share their respective resources and fulfill each one's demand. For instance, a provider can outsource resources to other providers when its workload cannot be attended with its local resources (peak hours). Thus, the provider would obtain higher profit because it can attend more customers and would not lose reputation, without facing with expensive capital costs to acquire IT equipment. Of course, the expected revenue from these customers should be higher than the cost of outsourcing the additional resources to be worth doing it.

Similarly, a provider that has underutilized resources could rent part of them to other providers. We refer to this situation as *insourcing* resources. Thus, the provider improves its benefit, exploits better its resources, and compensates the maintenance cost. Again, the expected benefit from renting its resources should be higher than the cost of maintaining them running. Otherwise, it would be preferable to turn them off to save power (and thus reduce costs).

From previous discussion, one could realize that the profitability of a Cloud provider in a federated scenario highly depends on a wide number of parameters, such as the provider's incoming workload, the cost of outsourcing additional resources, the revenue for renting unused resources, or the cost of maintaining the provider's resources operative. All these parameters must be considered to decide the best resource management action for the provider depending on current (and foreseen) environment conditions. Depending on their value, the provider can decide at every moment whether to outsource, insource, or turn off resources. This requires of complex resource management mechanisms that can dynamically manage the internal and external resources in the most cost-effective way while satisfying the QoS agreed with the users.

To obtain the maximum benefit in a federated Cloud scenario, and given the described complexity of federation decisions, it is important that the provider has a

clear understanding of the potential of each federation decision. Using this, the provider can take the most convenient decision depending on the environment conditions.

In this paper, we analyze the impact of federation as a mechanism to maximize Cloud providers' profit in a scenario that federates both Private and Public Clouds (Armbrust et al. 2010). By Private Clouds we mean, essentially, a private infrastructure dedicated to one organization having a limited capacity. We refer as Public Cloud to the utility computing made available to the general public in a pay-as-you-go manner. We present an analytical model that characterizes Cloud federation and can be used to drive provider's decisions about when to outsource, insource, and turn off idle resources.

This model is used to build the resource management core of a provider that offers Infrastructure as a Service (IaaS) (Armbrust et al. 2010), where users rent Virtual Machines (VMs) in a similar way it is done in Amazon EC2 (http://aws.amazon.com/ec2). In this environment, we study the effect of these decisions on the provider's profit and we evaluate the most appropriate provider's configuration depending on the environment conditions. For this purpose, we propose a scheduler able to support federation capabilities and we evaluate its profitability using information of actual providers.

The remainder of this paper is organized as follows: Section 2 presents a case study based on a real provider to motivate the benefit of using federation. Section 3 introduces the architecture of federated Cloud providers. Section 4 states the equations for characterizing Cloud federation. Section 5 describes how to implement a federated provider based on the proposed characterization. Section 6 describes the experimental environment and the evaluation. Section 7 presents the related work. Finally, Section 8 presents the conclusions of the paper and the future work.

2 Analyzing commercial providers' profitability

Low profitability brought by load variability is an important problem for commercial IaaS providers. These providers have been forced to deal with this problem by increasing their prices to ensure their profitability. In this section, we present a case study for a real commercial provider, where we study its basic profitability parameters. We have chosen RackSpace Cloud (http://www.rackspacecloud.com/) because they provide detailed information of their specifications and thus, we can make realistic assumptions about their infrastructure.

RackSpace use quad-core machines with 16 to 32 GB memory and 10 to 620 GB of disk to host the VMs. Using their standard VM with 1 GB memory and 40 GB of disk, which costs $0.06, we can calculate they can host up to 16 VMs per host. Hence, they have a potential benefit of $2.4 per hour per machine.

Regarding the infrastructure, they lease their datacenters from DuPont Fabros (http://www.dft.com/data-centers/location-information). This company provides information of some of their datacenters. We use as a reference their largest datacenter, which is also one of largest datacenters in the US: ACC4. This is located in the Ashburn Corporate Center campus (ACC) in Northern Virginia. It comprises approximately 348,000 gross square feet with 171,000 raised square feet and 36.4 MW of critical load.

To calculate the costs, we get the most relevant information for Ashburn location: the energy costs $0.0673 per KWh (Department of Energy. U.S. Energy Information, http://www.eia.doe.gov/), the land costs $95 per square foot (Showcase, http://www.showcase.com/), and the maximum temperature is 26°C (Weatherbase, http://www.weatherbase.com/).

Using this information and the datacenter costs and models from Barroso and Hölzle (2009) and Turner and Seader (2006), we can calculate the datacenter details. A datacenter with a maximum outside temperature of 26°C will have a maximum PUE of 1.6. Hence, the datacenter would have around 23 MW to power IT equipment.

According to the literature (Barroso and Hölzle 2009), a medium size datacenter has a cost of $12 per Watt. Hence, the building for this datacenter costs around 437 million dollars. The cost of the land in the US will be around 3.3 million dollars. Amortizing these costs over 12 years, it is 3 millions and 230 thousand dollars per month respectively.

Regarding the servers, Rackspace uses quad core machines which match the specifications of a Dell PowerEdge 2970. It consumes a maximum power consumption of 300 W and according to the size of their datacenter, they can host around 80.000 of these machines. Each of these servers costs around $4000 dollars (amortizing over 4 years is $85 per month). Assuming the datacenter hosts 80 K servers and distributing the other costs among them, each server has an associated capital cost of $103 every month. On the other hand, operating and maintaining this datacenter has a typical cost of $0.05 per Watt per month, which is a total of $23 per machine.

Using the highest and lowest ranges for each cost, the error in these estimations is around 10%. Nevertheless, it is accurate enough for our purpose of giving a broad view of the provider's profitability.

If we assume the machines are fully used all the time, each one consumes 216 KWh each month and 345 KWh taking into account the PUE, which implies an energy cost of $23 every month. Hence, a fully used machine would have a total cost per month of $173 and would generate an income of $1728. Thus, the profit per month per machine would be $1555.

Nevertheless, literature suggests that the utilization of servers is around 30% (Barroso and Hölzle 2009). Servers at this utilization consume around 200 W which implies $16 per month. This implies a total cost of $165 per month, an income of $518 per machine, and thus, a profit per month of $353. In fact, with a 5% use, a host would imply a cost of $123 and an income of $90 per month and thus, the provider would lose money.

High prices vs. federation. Providers with large infrastructures are proportionally cheaper than small ones (Barroso and Hölzle 2009). As we have seen, this allows them to afford relatively low utilizations. However, in these situations, they are forced to have high prices for VM usage to maintain the big benefit margins they have.

In fact, if these providers would have lower prices with lower benefit margins, they could offer the same service to their customers at lower rates, and thus increase their competitiveness in the Cloud market. This would be especially important in a tight Cloud market where the others providers have lower VM prices.

In such a competitive scenario, the provider's profitability will suffer. In this situation, a more efficient usage of the provider's resources would make the difference, especially if the provider is able to take advantage of federation capabilities. Using federation, the workloads of multiple federated providers could consolidate their loads and avoid the purchase and maintenance of resources.

3 Federation in Cloud providers

To bring federation into IaaS providers, we envision a global scheduling layer on each provider. This is able to interact with other federated providers and exchange load according to the provider's requirements. In addition, as other common schedulers, it decides the placement of the VMs using the provider's local resources.

3.1 Architecture of a federated Cloud provider

To perform resource management in Cloud providers supporting federation, we use a typical architecture composed by three layers: *Scheduler*, *Resource Management*, and *Resource Fabrics*. This architecture is shown in Fig. 2.

The *Scheduler* layer comprises all the global resource management decisions, among both external providers and nodes in a single provider. This layer is in charge of deciding where a VM will run and periodically managing its location during the execution (e.g., migrations, cancellations, etc.). However, events such as the arrival

Fig. 2 Architecture of a federated Cloud provider
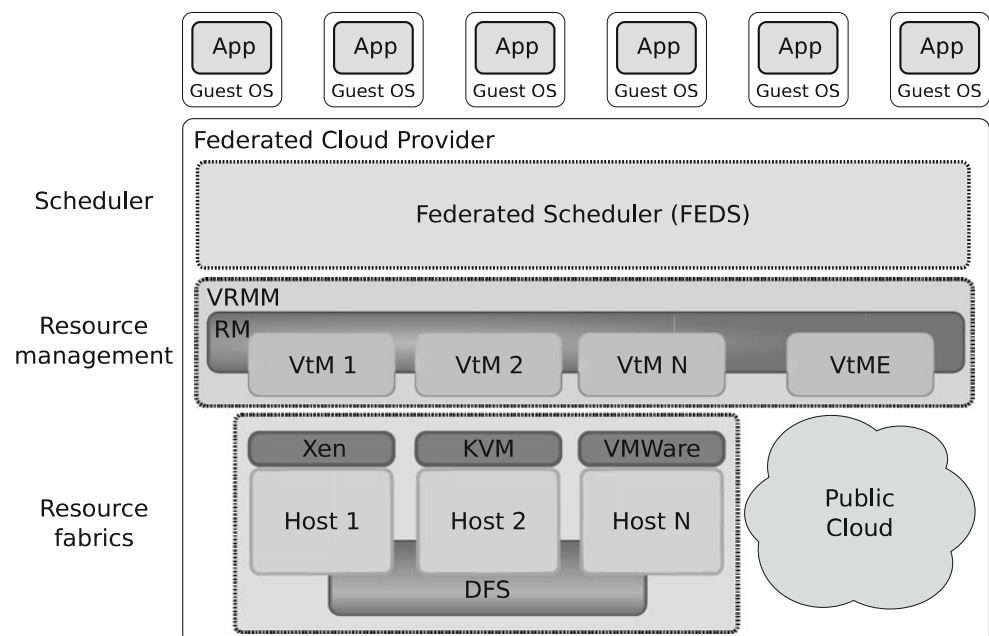
of a new task or an SLA violation of a running task can trigger the process too.

The *Resource Management* layer comprises all the local resource management decisions (i.e., in a single node). This layer is in charge of managing the physical resources in a node using virtualization and distributing them among the VMs running on that node by means of the *Virtualization Manager (VtM)*. In addition, this layer does not only offer local resources but it is also able to wrap external Clouds using an extension of the *VtM*: *VtME*. Using this approach, the *Scheduler* layer is able to manage external resources as locals. Finally, the *Resource Fabrics* layer comprises the physical resources where the VMs run.

An interaction with the system starts when a user sends a request for a VM to the *Scheduler* layer. This selects a node to run the task and forwards the request to the *VtM* in charge of this node, which creates the VM. The user can submit a task to this VM or just use it as a traditional machine. Finally, the *Resource Management* layer is also in charge of monitoring the execution to check the SLA status and finalizing the VM when is over. This architecture is implemented in EMOTIVE Cloud (http://www.emotivecloud.net) and it is described in detail in Goiri et al. (2009).

3.2 Federated Scheduler (FEDS)

In this paper, we analyze the impact of using federation in Cloud providers by characterizing the behavior of the *Scheduler* layer, from now on referred as *Federated Scheduler (FEDS)*.

FEDS is divided in two different parts: the management of the local resources and the management of external resources using Cloud federation. The scheduling policies for managing local resources are presented in Goiri et al. (2010). Notice that this can include migrating, pausing, or even canceling tasks if the cost of these actions is compensated with higher utility for the provider. When it manages local resources, *FEDS* can also decide to turn idle nodes off to save power.

Regarding external resources, *FEDS* is able to allocate additional resources from a federated Cloud provider or insource load from other federated providers when internal resources are unused. Outsourcing to other providers is performed by adding the other providers as internal resources where VMs can be executed.

For instance, Fig. 3 shows an example situation with three Federated Cloud Providers (FCP). In this example, *FCP 2* is running 2 VMs and it is already full. There is a third request to run a new one, *VM 3*, but *FCP 2* is not able to run it as it does not have enough resources. Nevertheless, *FCP 1* has free resources and offers them to *FCP 2*. At this point, *FCP 2* decides to outsource *VM 3* execution to *FCP 1*.

To decide the placement of the VMs, *FEDS* uses the characterization presented in next section to calculate the foreseen profit of all the allocation possibilities and then it distributes the resources using the configuration with maximum profit.

Fig. 3 Interaction of multiple federated Cloud providers
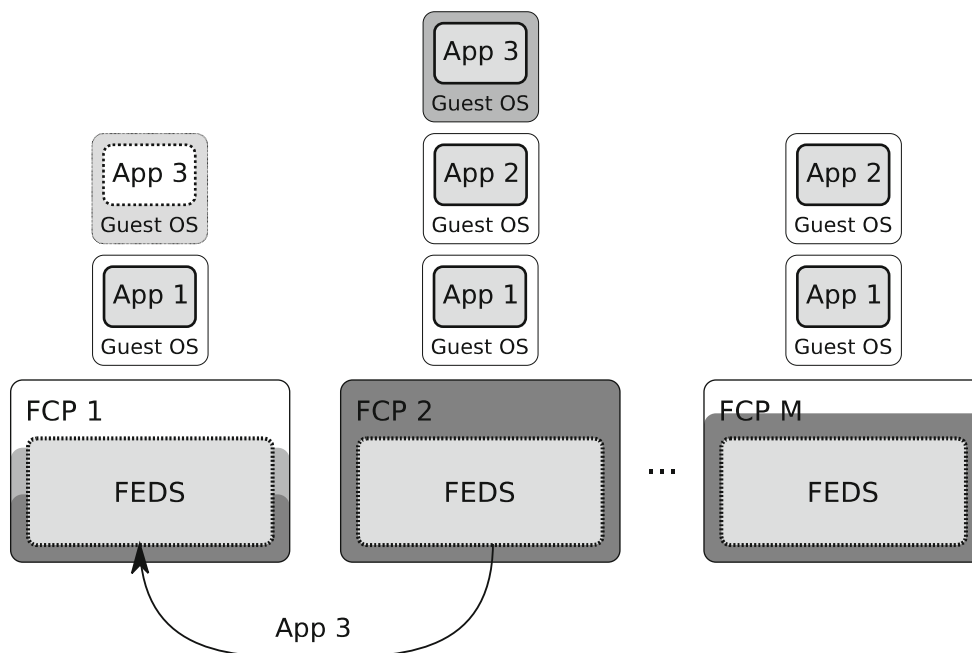
4 Characterizing a federated Cloud

To take the decision of outsourcing a VM to another provider or running it locally, *FEDS* uses a model to estimate the profitability of every action. This model evaluates the provider's utilization (number of total and used nodes), the pricing of the VMs, the capital costs (CAPEX), and the operational costs (OPEX). With all this information, this model uses the characterization of a federated environment to estimate the profitability of each situation.

Using this model, *FEDS* picks the most profitable situation and performs the scheduling actions to achieve it:

– Run VMs using local resources.
– Run VMs on federated providers (outsource).
– Offer idle resources to other federated providers (insource).
– Turn on off nodes.
– Turn off idle nodes.

Next sections present the characterization of a federated provider according to its capabilities: (1) only with local resources, (2) using external providers, (3) offering idle resources to external providers, or (4) using all the federation capabilities.

4.1 Allocation within the provider

Currently, researchers are seeking to find effective solutions to make Cloud data centers reduce power consumption while keeping the desired quality of service. One approach consists of consolidating the maximum number of tasks in a single node to maximize its usage. This allows applying different techniques for reducing the power consumption of the provider, such as Dynamic Voltage/Frequency Scaling (DVFS) and turn on/off nodes (Horvath et al. 2007; Heo et al. 2007).

In particular, *FEDS* can turn nodes off that remain unused to lower power consumption, thus saving the costs of keeping them in idle state. *FEDS* continuously monitors the node utilization in the provider, and turns nodes on and off (i.e., using Wake-on-LAN) to fit the customers' demand. Our scheduler is very extensive, and if needed, it could be extended with other power saving techniques like Raghavendra et al. (2008), which proposes merging some of the basic ideas and coordinating them in a multi-level power management for data centers.

The expected profit for the provider drives the allocation decisions of *FEDS*. We define the profit obtained from executing tasks in a provider p during in a certain period Δt as $Profit_p(\Delta t) = Revenue_p(\Delta t) - Cost_p(\Delta t)$. As in this scenario we are only consider-

ing the nodes of this single provider, its total profit $Profit(\Delta t)$ is equal to $Profit_p(\Delta t)$.

$Revenue_p(\Delta t)$ is obtained by multiplying the number of VMs running in the provider during that period of time $VM_p(\Delta t)$, which depends on the provider's incoming workload, with its corresponding price (e.g., *Price_VM_Hour* of a small instance in Amazon EC2 is €0.085 per hour). Note that the characterization could also include the cost of data storage and network bandwidth for that VM. This is part of our future work. Anyway, we believe that our work is still valuable for characterizing federated Clouds, in the sense that it is generic enough to include those costs (and other operational costs) just including them within the *Price_VM_Hour* parameter.

$$Revenue_p(\Delta t) = VM_p(\Delta t) \cdot Price\_VM\_Hour \cdot \Delta t \quad (1)$$

$Cost_p(\Delta t)$ is defined as the cost of maintaining all the nodes in the provider up ($Nodes_p \cdot Cost\_Node\_Hour_{var}$) during a certain period Δt. In addition, since turning off idle nodes would reduce the costs for the provider, we add a factor to the formula ($C_p(\Delta t)$), which indicates the capacity of the system (understood as the ratio of nodes that are up), to reflect this. If all the nodes in the system are up, capacity is 1. If the provider turns off half of the nodes, capacity is 0.5. Finally, we add also some fixed costs per node ($Cost\_Node\_Hour_{fix}$), which include the costs of acquiring the nodes and the physical space they occupy, taking into account their amortization.

$$Cost_p(\Delta t) = C_p(\Delta t) \cdot Nodes_p \cdot Cost\_Node\_Hour_{var} \cdot \Delta t$$
$$+ Nodes_p \cdot Cost\_Node\_Hour_{fix} \cdot \Delta t$$
$$(2)$$

To normalize the provider's incoming workload (i.e., the number of VMs to be executed), we define the provider's utilization ($U_p(\Delta t)$). It is calculated using as reference the maximum number of VMs that the provider can host, which depends on the number of nodes in the provider ($Nodes_p$) and the number of VMs that a node can host (VM_Node), in the following way:

$$U_p(\Delta t) = \frac{VM_p(\Delta t)}{Nodes_p \cdot VM\_Node} \quad (3)$$

As discussed before, a single provider is profitable when $Revenue_p(\Delta t) > Cost_p(\Delta t)$. Using previous equations and operating on this formula, we obtain Eq. 4, which establishes the relationship between the utilization (i.e., the amount of VMs to execute) and the capacity (i.e., the ratio of nodes that are operative) for provider's profitability. Obviously, this and the subsequent equations require $C_p(\Delta t)$ to be greater or

equal than $U_p(\Delta t)$. This equation will allow *FEDS* to determine the number of nodes to turn off ($C_p(\Delta t)$)

$$C_p(\Delta t) < \frac{U_p(\Delta t) \cdot VM\_Node \cdot Price\_VM\_Hour - Cost\_Node\_Hour_{\text{fix}}}{Cost\_Node\_Hour_{\text{var}}} \quad (4)$$

4.2 Outsourcing to federated Clouds

As described in Armbrust et al. (2010), outsourcing resources to federated providers can be preferable to overprovisioning a private data center when demand varies over time. In addition, it also allows the provider to insource its resources to other providers if these are not being used. The decision of using these capabilities is based on their economic viability. Armbrust et al. (2010) introduces an equation that evaluates whether outsourcing resources to an external provider is profitable or not. It essentially compares the profit (resulting from *Revenue − Cost*) for the provider when outsourcing external resources with respect to executing in its own resources.

Our analysis starts from this formula to decide grabbing additional resources when there is a resource demand that cannot be fulfilled using local resources. In particular, the additional revenue obtained when outsourcing resources $Revenue_o(\Delta t)$, which is shown in Eq. 5, is calculated in the same way as Eq. 1 and depends on the number of VMs that are outsourced ($VM_o(\Delta t)$). Notice that, in this scenario, the total revenue for the provider is $Revenue(\Delta t) = Revenue_p(\Delta t) + Revenue_o(\Delta t)$.

$$Revenue_o(\Delta t) = VM_o(\Delta t) \cdot Price\_VM\_Hour \cdot \Delta t \quad (5)$$

given the current workload ($U_p(\Delta t)$) to get the best profit.

The total cost for the provider in this scenario is $Cost(\Delta t) = Cost_o(\Delta t) + Cost_p(\Delta t)$. The cost of outsourcing ($Cost_o(\Delta t)$) could be calculated also from Eq. 5. We apply a factor α to the cost of the VM, obtaining Eq. 6, to reflect that the provider can buy the VMs with a different price than the revenue it obtains for selling them. In general, the provider will try to build cheaper VMs to ensure its profitability, although in some situations, it can suffer temporary losses to maintain the anticipated availability.

$$Cost_o(\Delta t) = VM_o(\Delta t) \cdot \alpha \cdot Price\_VM\_Hour \cdot \Delta t \quad (6)$$

In this scenario, the provider is profitable when $Revenue_p(\Delta t) + Revenue_o(\Delta t) > Cost_p(\Delta t) + Cost_o(\Delta t)$. Notice that in this case $U(\Delta t)$ must include both the VMs executed in the provider and the VMs outsourced to other providers. According to this, we define $U_{\text{ratio}}(\Delta t) = \frac{U_p(\Delta t)}{U(\Delta t)}$, which represents the ratio of incoming workload that is executed locally in the provider. Both $U_p(\Delta t)$ and $U(\Delta t)$ can be calculated using Eq. 3, the former using only $VM_p(\Delta t)$ and the latter considering both $VM_p(\Delta t)$ and $VM_o(\Delta t)$. Using previous equations, we can derive Eq. 7, which allows to determine the number of nodes to turn off ($C_p(\Delta t)$) and the distribution of local and outsourced VMs ($U_{\text{ratio}}(\Delta t)$) to get the best profit. The tradeoff when choosing these two values is demonstrated in the experimentation section.

$$C_p(\Delta t) < \frac{U(\Delta t) \cdot (1 - \alpha + U_{\text{ratio}}(\Delta t) \cdot \alpha) \cdot VM\_Node \cdot Price\_VM\_Hour - Cost\_Node\_Hour_{\text{fix}}}{Cost\_Node\_Hour_{\text{var}}} \quad (7)$$

4.3 Insourcing from federated Clouds

As commented before, in a federated environment the provider can offer its unused resources to other providers (i.e., insourcing). In this case, the total cost for the provider does not vary ($Cost(\Delta t) = Cost_p(\Delta t)$). This means that there are not additional costs if the provider rents its free resources. The number of potential VMs that could be sold is defined as follows:

$$VM_{\text{free}} = (C_p(\Delta t) - U_p(\Delta t)) \cdot Nodes_p \cdot VM\_Node$$

The total revenue is expected to increase ($Revenue(\Delta t) = Revenue_p(\Delta t) + Revenue_i(\Delta t)$). To

calculate $Revenue_i(\Delta t)$, we use again the α factor to reflect that the price of VMs offered for insourcing could be different from the price of VMs for regular users (it should be probably cheaper if we want to be competitive in the market). In addition, we include another factor (β) that represents the ratio of free resources that are really offered for insourcing from the total amount of free resources. This serves to model the market demand of resources, since the occupation of idle resources depends on the external utilization of the rest of providers in the federation. This factor also makes the provider able to reserve some idle resources to react to variations in its workload.

According to this, $Revenue_i(\Delta t)$ can be calculated using Eq. 8.

$$Revenue_i(\Delta t) = \beta \cdot VM_{\text{free}}(\Delta t) \cdot \alpha$$
$$\cdot Price\_VM\_Hour \cdot \Delta t \qquad (8)$$

Having the option to offer unused resources to other providers, or to turn them off to reduce power consumption, the provider could doubt on which is the more profitable decision. The answer comes from resolving the following inequation: $Revenue_p(\Delta t) + Revenue_i(\Delta t) > Cost_p(\Delta t)$. Again, using previous equations and operating on this formula, we obtain Eq. 9, which allows to determine the number of nodes to turn off ($C_p(\Delta t)$) and the ratio of free resource to insource (β) given the current workload ($U_p(\Delta t)$) to get the best profit.

$$K = Cost\_Node\_Hour_{\text{var}} - \alpha \cdot \beta \cdot VM\_Node \cdot Price\_VM\_Hour$$

$$g(\Delta t) = \frac{U_p(\Delta t) \cdot (1 - \alpha \cdot \beta) \cdot VM\_Node \cdot Price\_VM\_Hour - Cost\_Node\_Hour_{\text{fix}}}{Cost\_Node\_Hour_{\text{var}} - \alpha \cdot \beta \cdot VM\_Node \cdot Price\_VM\_Hour}$$

$$f(\Delta t) = \begin{cases} C_p(\Delta t) < g(\Delta t) & \text{if } K > 0 \\ C_p(\Delta t) > g(\Delta t) & \text{if } K < 0 \end{cases} \qquad (9)$$

4.4 Insourcing and outsourcing in federated Clouds

The final step is putting all together. In this case, profitability occurs when $Revenue_p(\Delta t) + Revenue_o(\Delta t) + Revenue_i(\Delta t) > Cost_p(\Delta t) + Cost_o(\Delta t)$. After operating on this formula, we obtain Eq. 10. Using this equation, *FEDS* can decide whether outsourcing resources, renting free resources to others providers, or turning off nodes is profitable for the provider.

$$K = Cost\_Node\_Hour_{\text{var}} - \alpha \cdot \beta \cdot VM\_Node \cdot Price\_VM\_Hour$$

$$g(\Delta t) = \frac{U(\Delta t) \cdot (1 - \alpha + U_{\text{ratio}}(\Delta t) \cdot \alpha \cdot (1 - \beta)) \cdot Price\_VM\_Hour \cdot VM\_Node - Cost\_Node\_Hour_{\text{fix}}}{Cost\_Node\_Hour_{\text{var}} - \alpha \cdot \beta \cdot VM\_Node \cdot Price\_VM\_Hour}$$

$$f(\Delta t) = \begin{cases} C_p(\Delta t) < g(\Delta t) & \text{if } K > 0 \\ C_p(\Delta t) > g(\Delta t) & \text{if } K < 0 \end{cases} \qquad (10)$$

5 Implementing federation

We have already presented what the provider's basic components to support federation are, and the characterization of the provider's profitability in this environment. In this section, we present some other considerations that must be taken into account when implementing federation among different providers, including how *FEDS* uses the presented characterization to take decisions.

5.1 Capacity planning

When a provider wants to start offering its service, it must decide the amount of resources it will need to provide it. This depends on the federation capabilities it will support (i.e., outsourcing, insourcing, none...)

and the expected number of users. To estimate this expected number of users, the provider can use former workloads and apply business models to predict their growth.

Using the previously presented characterization and the expected workload, the provider can decide the amount of resources (i.e., number of nodes) required. In addition, with this information the provider can fix the pricing scheme of the VMs.

5.2 Policy for federated resource management

FEDS can use the characterization presented in previous section to implement a resource management policy that allows handling the provider's resources and placing the VMs locally or in federated providers. In this paper, we propose a potential policy with this aim.

The algorithm that implements the policy is shown in Fig. 4.

When using this policy, *FEDS* basically instantiates Eq. 10 from the characterization with the provider's current parameters. Notice that some of these parameters (α, β, *Price_VM_Hour*) have been fixed during the capacity planning phase (see previous subsection) by using also the presented characterization (see Section 6 to check the effect of these parameters).

The policy is executed when there is a change in the provider's utilization (i.e., a new VM arrives). The provider's resources can be full and it would require to use idle resources or use external resources to run more VMs. If the provider has no free resources, it uses Eq. 10 to decide whether to:

– Cancel the execution of new or already running VMs.
– Outsource VMs to other federated providers with a given pricing (α).

Otherwise, if the provider has idle resources, it uses Eq. 10 to:

– Turn off part of the idle resources.
– Offer part of the idle resources (β) to federated providers with a lower price (α).

The output decisions of the *FEDS* policy are then carried out by using the actuators provided by the *Resource Management* layer.

5.3 Interconnecting providers

To communicate to other providers, Amazon EC2 is becoming the standard de facto. However, there are still a number of providers supporting only proprietary APIs. Some efforts have been also carried out to converge in a standardized API (e.g., OCCI Interface 2011). Either way, providers in a federated Cloud must be able to interact with providers with different APIs, each one with its own features.

To solve this problem, we propose the usage of a wrapper (i.e., *VtME* component) that offers a unified interface to *FEDS* independently of the external provider used to outsource and internally implements the translation to all the supported APIs. *VtME* offers basically the same interface to *FEDS* than the *VtM* component. In this way, it offers a transparent way to operate with other providers in a federation as they were big resources with special features (creation times, limited operation, special pricing). A similar approach has been adopted by OpenNebula (http://www.opennebula.org), which currently supports transparent usage of Amazon EC2 and ElasticHosts (http://www.elastichosts.com/) resources.

The current implementation of the *VtME* component demonstrates its capabilities by supporting the EC2 (which covers Amazon EC2 and Eucalyptus-based providers) and the OCCI APIs. Thanks to its design, it could be easily extended to support other typical interfaces such as those from RackSpace (http://www.

Fig. 4 Policy implemented by FEDS

```
1.   Apply local scheduling policies
2.   Get provider's parameters:
        alpha: factor that applies to the price of outsourced VMs
        beta: ratio of idle resources offered to other providers
        Price_VM_Hour: price of a VM per hour
        VM_Node: number of VMs allowed in a node
        Cost_Node_hour_fix: capital cost of a node per hour
        Cost_Node_hour_var: operational cost of a node per hour
        U: provider's utilization
        U_ratio: ratio of outsourced VMs
        C: provider's capacity (ratio of nodes that are up)
3.   Instantiate Equation 10 with provider's current parameters
4.   While inequation is not fulfilled and there are actions to do:
5.     If local resources are full:
6.        If outsourcing is supported:
7.           If profitability outsourcing VM > profitability running VM with a local node:
8.              Outsource VM to a federated provider
9.           Else:
10.             Turn on off node
11.       Else:
12.          If C<100:
13.             Turn on off node
14.          Else
15.             Reject VMs that cannot be accepted
16.    Else:
17.       If insourcing is supported:
18.          If profitability insourcing > profitability turning off idle node:
19.             Insource VM from federated provider
20.          Else:
21.             Turn off idle node
22.       Else:
23.          Turn off idle node
```

rackspacecloud.com/) and GoGrid (http://www.gogrid. com/).

VtME is in charge of mapping its methods to the particular API functions of the different external providers. In addition, it is also in charge of dealing with the semantic differences of each provider. For example, the standard VMs in Amazon EC2 are categorized in: small, large, and extra large; while RackSpace distributes them according to the memory size. Therefore, *VtME* must deal with the heterogeneity when defining the VM size.

Apart from the API issue, interoperability also requires considering VM images compatibility. Different providers can run different hypervisors, and the VM images could be incompatible. This problem can be overcome using image converters or deploying the needed software on the images offered by the external providers.

5.4 Resource availability

A provider is able to know the status of its own resources and evaluate if it has enough room to host a new VM. In a federated environment, a provider should be also able to know the availability of the other providers to host a given VM. For this reason, when a provider is federated it must provide the capability to ask for the availability of its resources.

There are two possibilities: knowing the utilization of the provider or checking if it is able to host a given VM. The first approach would expose the internals of the provider, which can be not desired. The second approach provides more confidentiality but it can imply performance issues.

5.5 Service level agreements (SLAs)

Current Cloud providers typically offer SLAs to their customers (Amazon EC2 Service Level Agreement, http://aws.amazon.com/ec2-sla/) (GoGrid Service Level Agreement, http://www.gogrid.com/legal/sla.php). They generally support very simple metrics based on resource availability. There are however proposals to support fine-grain resource-level QoS guarantees on Cloud SLAs (Goiri et al. 2010). The problem is that different providers can support different SLAs. Hence, this should be considered when outsourcing resources in a Cloud federation. Note that when a provider outsources resources to other providers, it depends on them to fulfill the guarantees in its SLAs. According to this, the provider should perform a previous triage to discard those providers with incompatible SLAs, or be able to translate the guarantees supported in the SLAs of federated providers to its own guarantees.

Even if this problem is overcome, the provider must monitor continuously the fulfillment of the SLAs of the VMs it has accepted. If any SLA violation is detected, an adaptation process for requesting more resources to the provider is started, first locally in each node, then globally in the provider (Ejarque et al. 2008), and finally outsourcing resources to other providers. This includes the VMs that it has outsourced to other providers. If any of these VMs is not fulfilling the SLA, the provider could react, for instance, allocating additional VMs for that service.

6 Evaluation

The evaluation of our proposal comprises two parts. First, we evaluate the proposed characterization by presenting an analytical study using parameters from real providers about the profitability of Cloud providers when doing outsourcing and insourcing and shutting down nodes. Second, we evaluate our policy for resource management in federated Clouds using a real Cloud workload.

6.1 Experimental environment

We follow the pricing idea for Grids presented in Opitz et al. (2008). We use reference values from real providers for revenues, costs, and virtualization parameters just to demonstrate how our equations can drive resource allocation decisions, though the particular values of these parameters will highly depend on the particular provider's characteristics. Anyway, the presented equations remain valid. As base node, we use a mid-range server with a direct consumption of 638 W in mean (see Koomey 2007, 2008). These mid-range servers support in mean a maximum amount of 6 VMs per node, assuming small EC2 instances, which have a cost of 0.085 €/h (EC2 pricing in Europe).

Nevertheless, power consumption is not just the server direct consumption. It must also take into account all the related energy costs such as cooling and other infrastructure consumptions. This is evaluated using the Power Usage Effectiveness (PUE), which is defined as the ratio of data center power to IT power draw (Belady et al. 2007). According to historical trends, site infrastructure consumes 50% of all data center energy, which corresponds to a PUE of 2.0 (this means that the data center must draw 2 W for every 1 W of power consumed by the IT equipment). Therefore, we assume an average consumption of 1276 W per

(a) Variable VM price

(b) VM Price = 0.09 €/hour
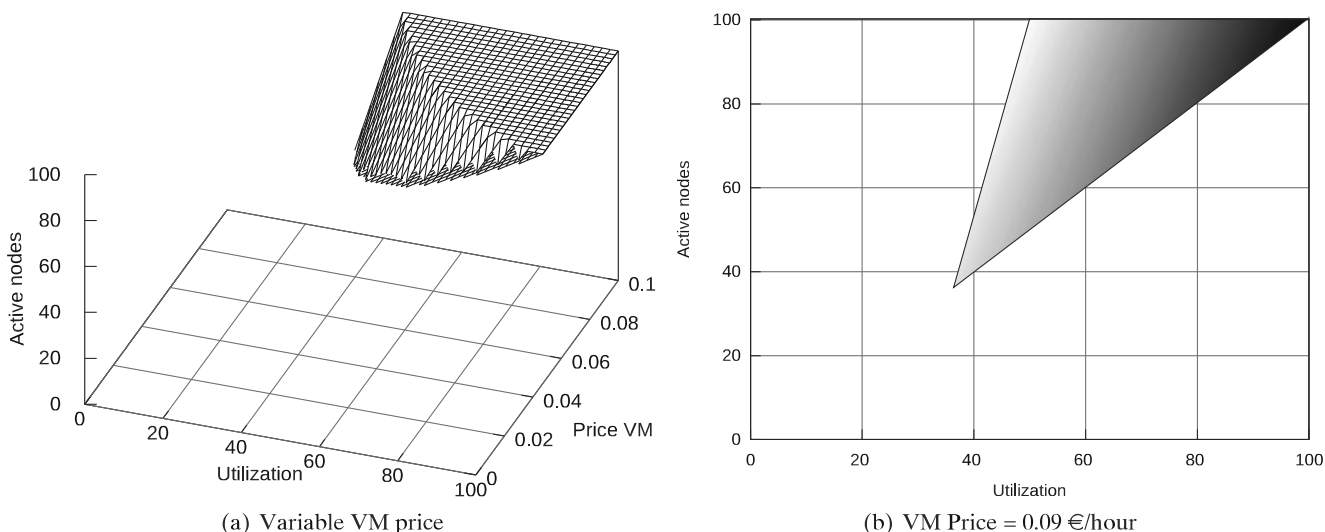
Fig. 5 Relation between utilization (%) and active nodes (%) in a single provider (Eq. 4). *Darker* is more profitable

node. The pricing used for the electricity is the Spanish one, which corresponds to 0.09 €/KWh (Europe's energy portal, http://www.energy.eu).

Finally, to calculate the cost of the nodes, we also take into account the amortization of the servers (in 3 years) and the space (in 10 years) required to deploy them using a price of 4000 €/node and 2000 €/m^2, respectively.

6.2 Profitability analysis in a federated Cloud

In this section, we present an analytical study of our characterization of a provider operating in a federated Cloud. We first calculate how a provider should be

dimensioned (i.e., its capacity) according to its utilization applying Eq. 4. Figure 5a shows the number of local nodes active (C_p in the equation) as a function of the price of the VMs (*Price_VM_Hour*) and the utilization (U_p). Figure 5b shows the profitability for a specific price of 0.09 €per VM (darker is more profitable, outside of the drawn area results are not possible or not profitable). Notice that a given provider needs a minimum utilization (greater than 40%) and a minimum price per VM (higher than 0.05 €) in order to be profitable, because it needs to amortize its capital expenditures. Below these values, the provider is not profitable as fix costs are too high. In addition, the greater the number of nodes the provider wants to

(a) Variable provider's utilization (U)

(b) Provider's utilization (U) = 80%
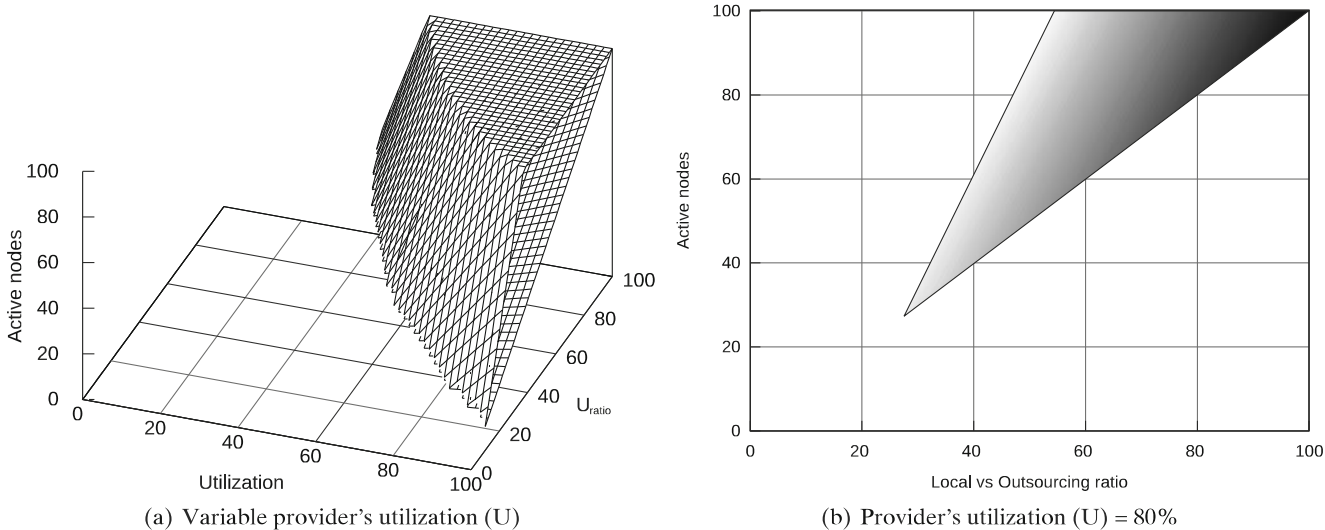
Fig. 6 Relation between utilization (%) and active nodes (%) using outsourcing (Eq. 7). *Darker* is more profitable
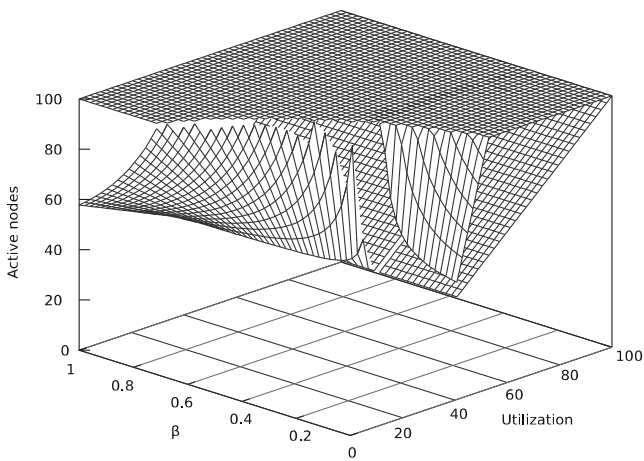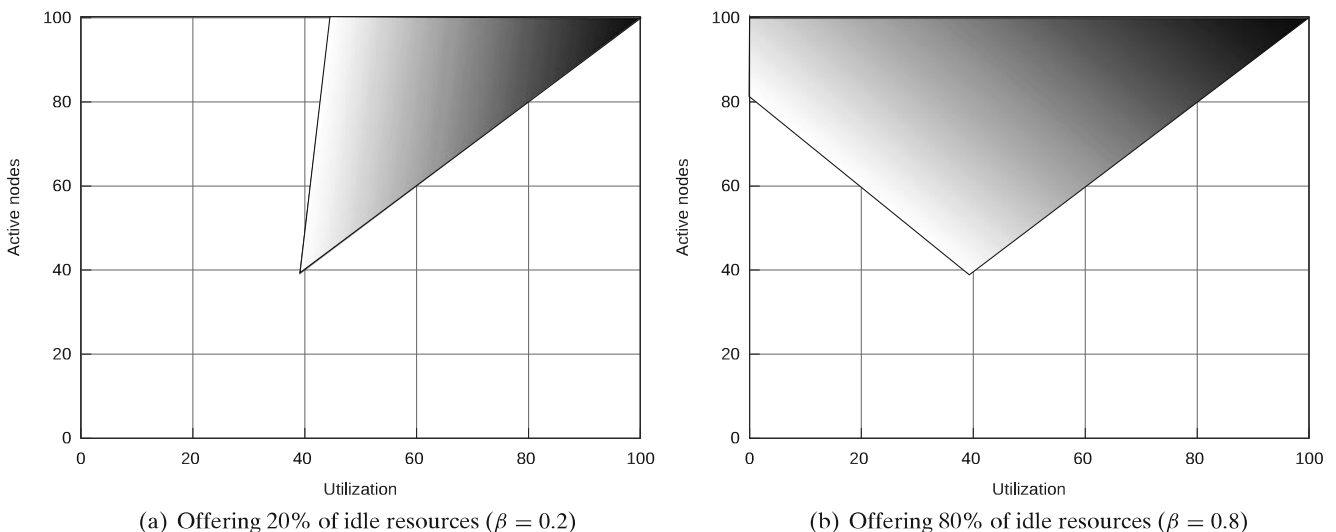
Fig. 7 Relation between utilization (%) and capacity (%) using insourcing (Eq. 9)

keep operative, the higher the utilization and the price per VM it needs to fulfill its operational expenditures. Finally, the provider has more profit when its utilization is higher (as expected).

Figure 6 shows the maximum capacity a provider should have to be profitable when using outsourcing according to Eq. 7. This figure relates the provider's active nodes (C_p) with its global utilization (U) (i.e., the incoming workload) and the ratio of VMs that are locally executed (U_{ratio}), assuming that $\alpha = 0.75$ and $Price\_VM\_Hour = 0.085$. Figure 6b shows the profitability for a specific utilization of 80% (darker is more profitable, outside of the drawn area results are not possible or profitable). Again, the provider requires a minimum utilization to be profitable, and the higher

the utilization, the higher number of active nodes are allowed. In addition, executing the VMs locally is preferred (high U_{ratio}), as this allows maintaining more nodes active and maximizes profit. This applies especially when the provider has a medium utilization. When the utilization is high, the provider can outsource more VMs and still be profitable. However, as shown in the figure, at a given point, the more VMs the provider decides to outsource, the more nodes it must turn off to maintain its profitability. In addition, a minimum amount of VMs running locally is always needed to be profitable.

Figure 7 shows the maximum capacity a provider should have to be profitable if it is able to insource resources according to Eq. 9. It relates the provider's active nodes (C_p) with its utilization (U_p) and the factor β (the ratio of spare VMs that it sells), assuming an α factor of 0.75 and $Price\_VM\_Hour = 0.085$. Figure 8 shows the profitability when offering the 20 and the 80% of the idle resources (darker is more profitable, outside of the drawn area results are not possible or not profitable). When the provider sells less than the 30% of its free resources (β is low), high capacities are only profitable when the utilization is greater than 50%. The rationale in this case is similar to the situation described in Fig. 5. On the other side, when the value of β is high, high capacities are always profitable. In this case, when the utilization is low, the surplus resources can be sold to other providers. However, as shown in Fig. 8, it is preferable (more profitable) for the provider to have high utilization than selling spare resources to other providers. This happens because spare resources are sold at a lower price.

(a) Offering 20% of idle resources ($\beta = 0.2$)

(b) Offering 80% of idle resources ($\beta = 0.8$)

Fig. 8 Relation between utilization (%) and active nodes (%) using insourcing (Eq. 9). *Darker* is more profitable
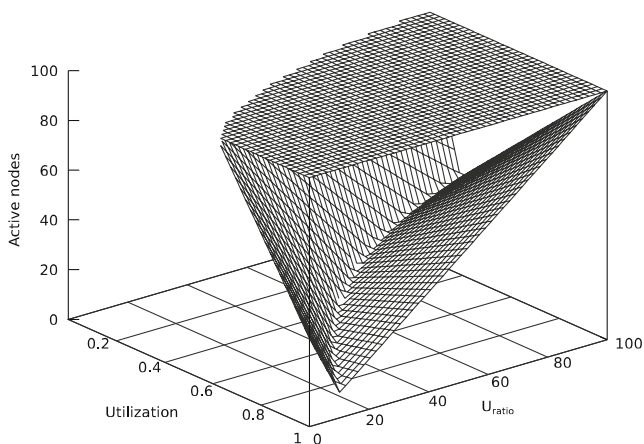
Fig. 9 Relation between utilization (%) and active nodes (%) using in and outsourcing (Eq. 10)

Finally, using Eq. 10, which takes into account both outsourcing and insourcing, we get Fig. 9. It relates the provider's active nodes (C_p) with its utilization (U) and the ratio of local VMs (U_{ratio}) assuming $\alpha = 0.75$, $\beta = 0.5$, and $Price\_VM\_Hour = 0.085$. Figure 10 shows the profitability for a specific utilization of 80% and when outsourcing the 40% of the load (darker is more profitable, outside of the drawn area results are not possible or not profitable). The figures show that the provider requires a minimum utilization to be profitable. Executing the VMs locally is preferred (high U_{ratio}), as this allows maintaining more nodes active (this is especially noticeable when the utilization is medium) and maximizes profit. Although it is not most the profitable option, the provider can outsource part

of its VMs and still be profitable when the utilization is high, because resources freed in this way can be insourced. However, when the utilization is medium or the provider outsources a lot of VMs, it must maintain a minimum number of active nodes to be profitable, because the incomes obtained by selling the free VMs must be used to cover the operational expenses. When the utilization is low, even having all the nodes operative is not enough to be profitable, because the revenue obtained insourcing free VMs cannot cover the provider's expenses. Notice that different values of α and β could force the provider to act in a different way. This is an important advantage of our characterization, because it is totally configurable depending on the provider's actual environment.

### 6.3 Evaluating the FEDS policies for federation

In this section, we evaluate the potential benefit of a provider that incorporates our *FEDS* scheduler when receiving a real Cloud workload during a week. This workload, which is shown in Fig. 11, is extracted from an anonymous ISP's log during the week from Monday 27th of April until Monday 4th of May 2009.

The presented policy is tested on top of a provider able to operate within a Cloud federation. As it is difficult to build such testbed, we use a simulator (Goiri et al. 2010), where we have configured a provider with 100 nodes.

However, as shown in Fig. 11, sometimes the customers' demand is higher. Using a traditional resource management approach, the provider has to reject all the services that exceed its maximum capacity. There-
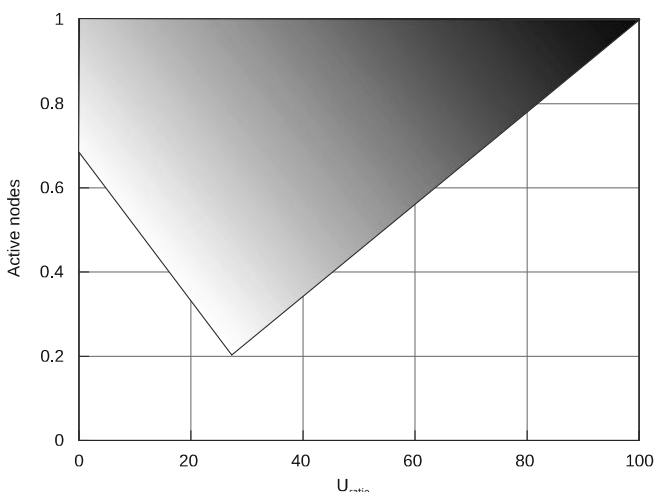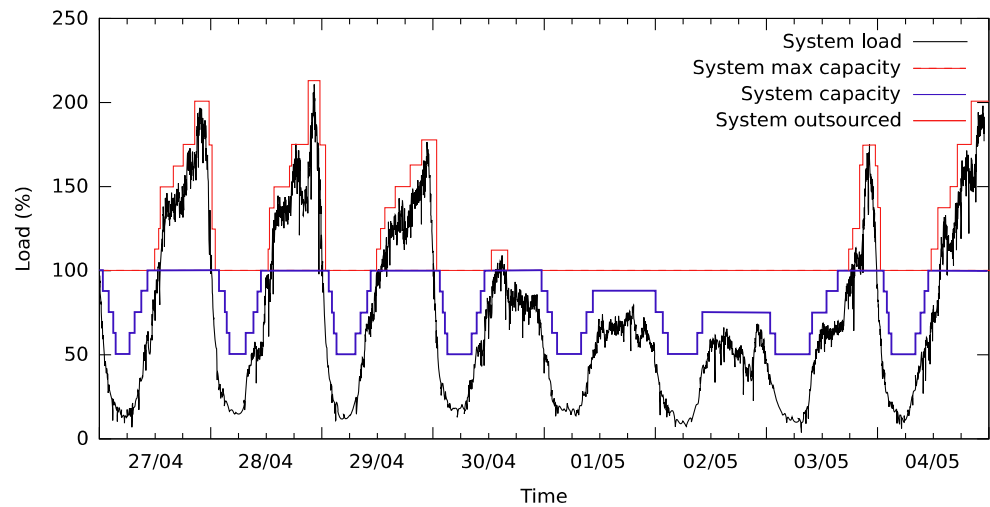

(a) Provider's utilization (U) = 80%
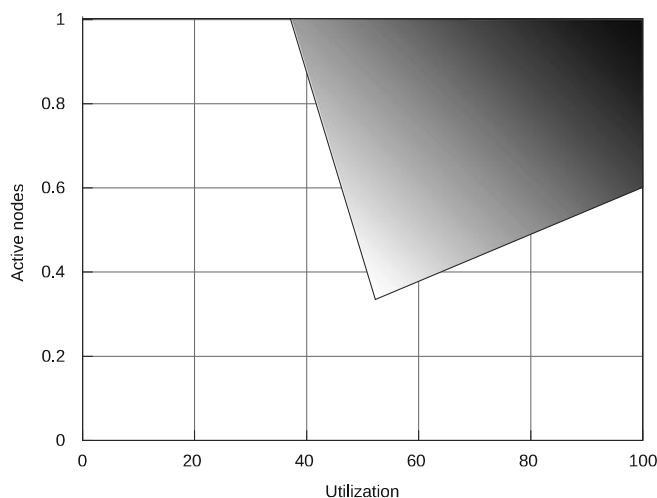
(b) Outsourcing the 40% of the load ($U_{ratio} = 0.6$)

Fig. 10 Relation between utilization and active nodes when introducing insourcing and outsourcing (Eq. 10). *Darker* is more profitable

Fig. 11 Service provider's workload during a week

fore, it loses many clients during rush hours. These lost clients can represent a great amount of money that is being wasted. In addition, the reputation of this provider is going down since customers stop trusting on it. For this reason, outsourcing resources to external providers can increase the provider's capacity when it is not enough to satisfy the demand.

On the other hand, the provider's capacity is underused during some periods. This reduces its total profit, since underused nodes are also consuming power. To avoid this, the provider can shut down those machines it guesses will not be required during a long period, for instance, during night. This decreases the power consumption during that period. Alternatively, the provider can also offer these unused resources to other providers, so they can execute their services on them. This option will be profitable for the provider when the obtained revenue is enough to compensate the cost of maintaining all these nodes up.

Of course, turn nodes off, outsourcing, and insourcing can be jointly applied to maximize the provider's profit. According to this, we have enabled this provider to use our *FEDS* scheduler. In this section, we evaluate

the impact on the profit when using several versions of our policy, which depend on whether the provider is able to turn idle nodes off, outsource, or insource.

The results, which assume $\alpha = 0.75$ and $\beta = 0.5$, are displayed in Table 1 and demonstrate the benefit of outsourcing resources, which is inversely proportional with the α factor. In addition, outsourcing allows the provider maintaining its reputation by being always available to give service to its customers. The second part of the table presents the profit in case the provider decided to increase its maximum capacity (up to 200 nodes) to support the whole workload without using outsourcing. It shows that the revenue has increased regarding the previous table, but also the fixed costs, such as hardware and maintenance. For this reason, global profit is lower in this case. In fact, only the *Insource-Nodes always up* configuration is profitable in this case.

The values in these tables are graphically represented in Fig. 12a and b. The benefit from shutting down nodes can be clearly appreciated in the 'Typical' and the 'Outsourcing' configurations in Fig. 12a. The same argument applies in the 'Typical' configuration

Table 1 Service provider's profit in Euros

	Nodes always up			Shutting down nodes		
	Revenue	Cost	Profit	Revenue	Cost	Profit
100 nodes						
Typical	5997.60	5185.84	811.76	5997.60	4793.41	1204.19
Outsource	7270.56	6140.56	1130.00	7270.56	5748.13	1522.43
Insource	8843.40	5185.84	3657.56	6771.29	4793.41	1977.88
Outsource & insource	10116.36	6140.56	3975.80	8044.25	5748.13	2296.12
200 nodes						
Typical	7270.56	10371.68	−3101.12	7270.56	8048.17	−777.61
Insource	11888.1	10371.68	1516.42	8044.25	8048.17	−3.92
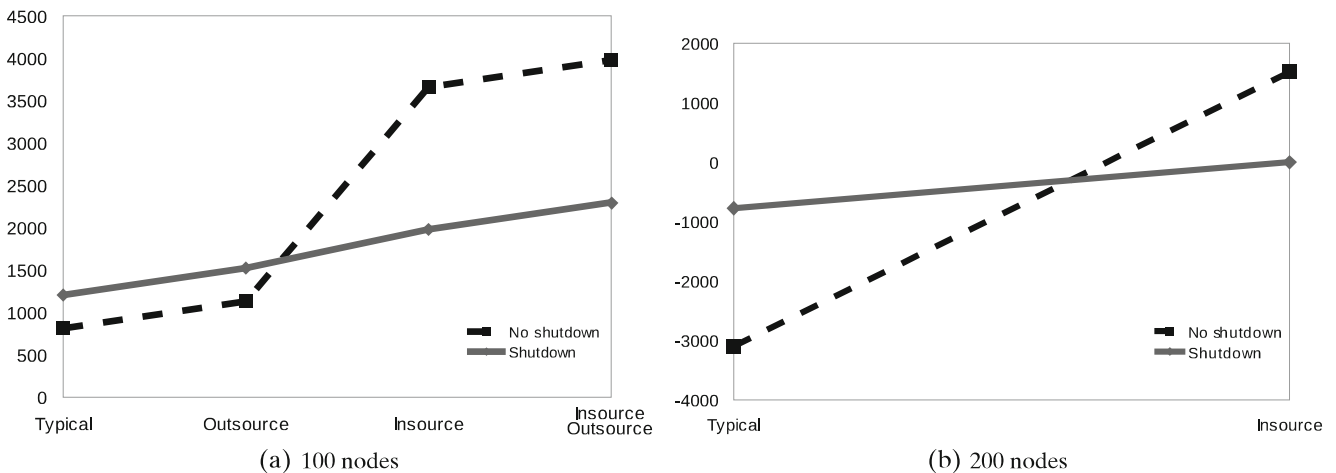
Fig. 12 Comparison of provider's profit with different capacities

in Fig. 12b. In the configurations with 'Insourcing' in Fig. 12a, it is more profitable not to shut down nodes, because the provider has more resources to offer to other providers. This also applies to the 'Insourcing' configuration in Fig. 12b, though in this case the profit is noticeable smaller due to the increased fixed costs.

7 Related work

Cloud computing has rapidly spread in recent times. Armbrust et al. (2010) present some key concepts of this paradigm such as the illusion of infinite computing resources available on demand and the ability to pay for use of computing resources on a short-term basis as needed. This allows companies to have a small set of resources that can be increased according to their needs, saving costs.

Jha et al. (2009) establish how Clouds can be viewed as a logical continuation from Grids by providing a higher-level of abstraction. Similarly, Buyya et al. (2009) define Cloud computing and provides the architecture for creating Clouds with market-oriented resource allocation by leveraging technologies such as Virtual Machines (VMs). It also proposes ideas for interconnecting Clouds to perform global exchanges. Following the use of service-oriented architecture, Jung (2011) proposes a model for businesses to define and discover services on top of this architecture.

The idea of federating systems was already present in the Grid. For instance, other authors (Boghosian et al. 2007; Sobolewski and Kolonay 2006) use federation to get more resources in a distributed Grid environment. The application of federation in the Cloud was initially proposed within the Reservoir project. In particular,

Rochwerger et al. (2009) describes the difficulty to merge different providers with different APIs and features. Nevertheless, they do not present any model to decide when to move tasks to a federated provider based on economic criteria. A first approach introducing this idea is presented by Campbell et al. (2009) who state some factors such as provider occupation and maintaining costs to dimension a Cloud provider and when to outsource to a federated provider.

One of the strengths of Cloud federation is the possibility to add public Clouds within the federation. There are a number of providers that offer VMs on demand, being Amazon EC2 (http://aws.amazon.com/ec2) the most popular. Nevertheless, it is a private implementation and it does not allow working with low-level aspects. Other public Cloud solutions with similar capabilities are GoGrid (http://www.gogrid.com/), 3Tera (http://www.3tera.com/), and ElasticHosts (http://www.elastichosts.com/).

To set up private Clouds, it has appeared multiple solutions that implement the EC2 API, such as Eucalyptus (Nurmi et al. 2009) or Nimbus (http://workspace.globus.org/clouds/nimbus.html). Similarly, Aneka (Chu et al. 2007) is a .NET-based service-oriented resource management platform, which is based on the creation of containers that host the services and it is in charge of initializing services and act as a single point for interaction with the rest of the Aneka Cloud. Moreover, it provides SLA support such that the user can specify QoS requirements such as deadline and budget. Other open-source alternatives, such as AbiCloud (http://www.abiquo.com), EMOTIVE Cloud (http://www.emotivecloud.net), and OpenNebula (http://www.opennebula.org), also add outsourcing capabilities by adding external resources.

8 Conclusions

In this paper, we have studied how to implement federation in Cloud providers to increase their profitability by saving capital and operational costs. For this purpose, we have described and implemented a scheduler, *FEDS*, able to operate with external providers. In addition to the managing of local resources, this scheduler is also able to decide when to use other providers resources or when to offer them the own idle resources. This is performed using a characterization of Cloud federation aimed at enhancing providers' profit. Our characterization includes equations that assist decisions in a federated Cloud, namely when to outsource resources to other providers, when to insource free resources to other providers, and when to turn unused nodes off to save power.

Our experimentation has evaluated these equations with realistic data to determine the impact of some parameters in the providers' profit. Evaluated parameters include the provider's incoming workload, the cost of outsourcing additional resources, the ratio of outsourced resources, the ratio of unused resources to be sold, and the cost of maintaining the provider's resources operative.

Our results demonstrate that the provider requires a minimum utilization and a minimum price per VM in order to be profitable when all the nodes are operative. In addition, local resources are preferred over outsourced resources, though the latter can enhance the provider's profit when the workload cannot be supported locally. Furthermore, when the utilization is low, the best option for the provider is insourcing the unused resources (though this is not always possible). We can summarize that all the described actuations can have a positive impact on the provider's profit depending on the environment conditions.

Our future work includes considering further costs, such as network bandwidth and storage, for VM pricing. We also plan to work on SLAs for federated Clouds and analyze the effects of outsourcing on the Quality of Service the users perceive (e.g. they can perceive different latencies for outsourced VMs).

References

Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., et al. (2010). A view of Cloud computing. *Communications of the ACM, 53*(4), 50–58.

Barroso, L., & Hölzle, U. (2009). The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis Lectures on Computer Architecture, 4*(1), 1–108.

Belady, C., Rawson, A., Pfleuger, D., & Cader, S. (2007). The green grid data center power efficiency metrics: PUE and DCiE. White paper, The Green Grid Consortium.

Boghosian, B., Coveney, P., Dong, S., Finn, L., Jha, S., Karniadakis, G., et al. (2007). NEKTAR, SPICE and vortonics: Using federated grids for large scale scientific applications. *Cluster Computing, 10*(3), 351–364.

Buyya, R., Yeo, C., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems, 25*(6), 599–616.

Campbell, R., Gupta, I., Heath, M., Ko, S., Kozuch, M., Kunze, M., et al. (2009). Open cirrusTM cloud computing testbed: Federated data centers for open source systems and services research. In *USENIX workshop on hot topics in Cloud computing (HotCloud), 14–19 June 2009*. San Diego, CA, USA.

Chu, X., Nadiminti, K., Jin, C., Venugopal, S., & Buyya, R. (2007). Aneka: Next-generation enterprise grid platform for e-Science and e-Business applications. In *3rd IEEE international conference on e-science and grid computing, 10–13 Dec 2007* (pp. 151–159). Bangalore, India.

Ejarque, J., de Palol, M., Goiri, I., Julia, F., Guitart, J., Badia, R., et al. (2008). SLA-driven semantically-enhanced dynamic resource allocator for virtualized service providers. In *4th IEEE international conference on e-science, 7–12 Dec 2008* (pp. 8–15). Indianapolis, USA.

Elnozahy, E., Kistler, M., & Rajamony, R. (2002). Energy-efficient server clusters. In *2nd workshop on power-aware computing systems, PACS 2002, 2 Feb 2002* (pp. 179–196). Cambridge, MA, U.S.A.: Springer.

Goiri, I., Fitó, J., Julià, F., Nou, R., Berral, J. L., Guitart, J., et al. (2010). Multifaceted resource management for dealing with heterogeneous workloads in virtualized data centers. In *11th ACM/IEEE international conference on grid computing (Grid 2010), 25–29 Oct 2010* (pp. 25–32). Brussels, Belgium.

Goiri, I., Guitart, J., & Torres, J. (2009). Elastic management of tasks in virtualized environments. In *Proccedings of the XX jornadas de paralelismo* (pp. 671–676).

Goiri, I., Julia, F., Fitó, J., Macías, M., & Guitart, J. (2010). Resource-level QoS metric for CPU-based guarantees in Cloud providers. In *7th international workshop on economics of grids, Clouds, systems, and services (GECON 2010)* (pp. 34–47).

Heo, J., Henriksson, D., Liu, X., & Abdelzaher, T. (2007). Integrating adaptive components: An emerging challenge in performance-adaptive systems and a server farm case-study. In *28th IEEE international real-time systems symposium (RTSS 2007), 3–6 Dec 2007* (pp. 227–238). Tucson, AZ, USA.

Horvath, T., Abdelzaher, T., Skadron, K., & Liu, X. (2007). Dynamic voltage scaling in multi-tier web servers with end-to-end delay control. *IEEE Transactions on Computers, 56*(4), 444–458.

Jha, S., Merzky, A., & Fox, G. (2009). Using Clouds to provide grids with higher levels of abstraction and explicit support for usage modes. *Concurrency and Computation: Practice and Experience, 21*(8), 1087–1108.

Juan, A., Hernandez, F., Tordsson, J., Elmroth, E., Ali-Eldin, A., Zsigri, C., et al. (2011). OPTIMIS: A holistic approach to Cloud service provisioning. *Future Generation Computer Systems*. doi:10.1016/j.future.2011.05.022.

Jung, J. (2011). Service chain-based business alliance formation in service-oriented architecture. *Expert Systems with Applications: An International Journal, 38*(3), 2206–2211.

Koomey, J. (2007). *Estimating total power consumption by servers in the U.S. and the world, 15 Feb 2007*. Oakland, CA: Analytics Press.

Koomey, J. (2008). Worldwide electricity used in data centers. *Environmental Research Letters, 3*(3). doi:10.1088/1748-9326/3/3/034008.

Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., et al. (2009). The eucalyptus open-source Cloud-computing system. In *9th IEEE/ACM international symposium on cluster computing and the grid (CCGRID), 18–21 May 2009* (pp. 124–131). Shanghai, China.

Open Grid Forum (2011). Open Cloud Computing Interface (OCCI) Infrastructure, GFD-P-R.184, Version 1. Retrieved August 30, 2011 from http://www.ogf.org/documents/GFD.184.pdf.

Opitz, A., König, H., & Szamlewska, S. (2008). What does grid computing cost? *Journal of Grid Computing, 6*(4), 385–397.

Pinheiro, E., Bianchini, R., Carrera, E., & Heath, T. (2001). Load balancing and unbalancing for power and performance in cluster-based systems. In *Workshop on compilers and operating systems for low power, 9 Sept 2001* (pp. 182–195). Barcelona, Spain.

Raghavendra, R., Ranganathan, P., Talwar, V., Wang, Z., & Zhu, X. (2008). No power struggles: Coordinated multi-level power management for the data center. In *13th international conference on architectural support for programming languages and operating systems (ASPLOS 08), 1–5 Mar 2008*. Seattle, WA, USA.

Rochwerger, B., Breitgand, D., Levy, E., Galis, A., Nagin, K., Llorente, I., et al. (2009). The RESERVOIR model and architecture for open federated Cloud computing. *IBM Journal of Research & Development, 53*(4), 535–545.

Sobolewski, M., & Kolonay, R. (2006). Federated grid computing with interactive service-oriented programing. *Concurrent Engineering, 14*(1), 55.

Turner, W., & Seader, J. (2006). Dollars per kW plus dollars per square foot are a better data center cost model than dollars per square foot alone. Uptime Institute white paper, Santa Fe.

Íñigo Goiri obtained his MS and PhD degrees in Computer Science at the Universitat Politècnica de Catalunya (UPC), in 2008 and 2011 respectively. Previously he got his bachelor degree in computer science in the University of Lleida in 2006. He also collaborated at Barcelona Supercomputing Center (BSC) as researcher staff in the eDragon Research Group. Currently, he helds a postdoctoral researcher position in the Department of Computer Science at Rutgers University.

Jordi Guitart received the MS and PhD degrees in Computer Science at the Universitat Politècnica de Catalunya (UPC), in 1999 and 2005, respectively. Currently, he is an associate professor at the Computer Architecture Department of the UPC and an associate researcher at Barcelona Supercomputing Center (BSC). His research interests are oriented towards innovative resource management approaches for modern distributed computing systems. He is involved in a number of European projects.

Jordi Torres has a Masters degree in Computer Science from the Universitat Politècnica de Catalunya (UPC, 1988) and also holds a Ph. D. from the same institution (Best UPC Computer Science Thesis Award, 1993). Currently he is a full professor in the Computer Architecture Department at UPC. He has more than twenty years of experience in research and development of advanced distributed and parallel systems in the High Performance Computing Group at UPC.