Client Classification Policies for SLA Negotiation and Allocation in Shared Cloud Datacenters

Mario Macías and Jordi Guitart

Barcelona Supercomputing Center/Universitat Politecnica de Catalunya
Jordi Girona 29, 08034 Barcelona, Spain
{mario.macias, jordi.guitart}@bsc.es

Abstract. In Utility Computing business model, the owners of the computing resources negotiate with their potential clients to sell computing power. The terms of the Quality of Service (QoS) to be provided as well as the economic conditions are established in a Service-Level Agreement (SLA). There are situations in which providers must differentiate the SLAs in function of the type of Client that is willing to access the resources or the agreed QoS e.g. when the hardware resources are shared between users of the company that own the resources and external users. This paper proposes to consider the information of potential users when the SLA is under negotiation to allow providers to prioritize users (e.g. internal users over external users, or preferential users over common users). Two policies for negotiation are introduced: price discrimination and client-aware overselling of resources. The validity of the policies is demonstrated through exhaustive experiments.

1 Introduction

In recent years, the Utility Computing business model is increasing its acceptance in the Information Technology sector [19] thanks to the burst of Cloud Computing paradigm [8]. In Utility Computing, the users of the resources are not necessarily their owners: users run their applications or services in remote data centers and pay in function of the usage, as with other utilities such as water provision or the electric grid. The terms of the Quality of Service (QoS) to be provided and the economic conditions are established in a Service-Level Agreement (SLA). Utility Computing allows the users to economically benefit from economies of scale, because it minimizes the space and maintenance costs. However, despite of the economic benefits of using computing as a utility, there are still open security reasons to not submit the critical or confidential data to resources that are located in third parties [13].

Companies may decide to hire out the spare resources of their data centers to external users that do not have such security or confidentiality restrictions [11]. The price that external users pay to use the resources contributes to amortize the cost of the data centers. However, a binary classification of the users as internal/external is not accurate enough in many situations. For example, headquarters of a big company may classify the users of its data centers according to different levels: users from the headquarters that owns the resources are completely internal, users from other companies are completely external, and users from other headquarters of the same company have an intermediate range. Even multinationals could define more degrees of proximity for headquarters in the same country and headquarters in other countries. Whilst completely external users pay a fee and completely internal users use the resources for free, the users in between would pay a reduced fee that does not report profit, but encourages each location to only use resources from external locations when strictly necessary.

Another example of intermediate users are users from trusted entities that decide to share their computing resources for sharing risks and dealing with peaks of workload without the need of overprovision resources. Examples of trusted entities are different companies from the same business cluster [17].

Clients may be classified according to other criteria. Many service providers classify their clients according to the QoS that they have purchased. For example, Spotify [5] is an online music provider that classifies its clients in three categories (*free*, *unlimited* and *premium*) according to their monthly fee. The higher the fee the more services and QoS: unlimited streaming hours, highest quality of sound, available downloads, etc. The provider must consider the purchased QoS when allocating the resources.

The usage of the resources by external users can affect the QoS of internal users if the SLAs do not reflect priorities between clients in terms of pricing or allocation of resources. This paper suggests applying Client Classification to keep high QoS to internal users or users with high QoS requirements. Client Classification considers the information about the users when giving them access to the resources and prioritizes some SLAs according to two criteria:

QoS that the users are willing to acquire: the higher the QoS the higher the price. This is the traditional classification of services in Utility Computing.

Affinity between the client and the provider: clients from the same company as the provider or from entities that have a privileged relationship with the provider can hire the services at better prices, better QoS, or any other privilege. This novel approach was devised with the success of Cluster and Grid Computing, in which organizations share part of their resources with users from other organizations. By prioritizing users to which there is high affinity, organizations can ensure that their internal users will have enough resources or QoS when there is a peak of external demand.

According to previous considerations, our contributions are:

1. Proposal of new approaches to perform Client Classification in pricing and SLA allocation policies.
2. Demonstration of the validity of the model through fine-grained experiments that demonstrate how a provider can reach its Business-Level Objectives (BLO) without penalizing its internal users or external users with priority SLAs. The results are evaluated in terms of revenue and proportion of priority users in the system.

We propose policies to allocate SLAs by pursuing a main BLO: users differentiation according of their Affinity/QoS relationship with the provider. In addition, our model also considers the economic profit as secondary BLO when negotiating the SLAs: prices at peak hours are higher than prices at off-peak hours for all the users. In that way, costs are amortized faster and companies are encouraged to move part of their tasks (such as resource-intensive unattended batch executions) to hours with low demand, such as the late night.

The experiments have been performed with the Economically Enhanced Resource Manager (EERM) simulator [4]: a customizable Cloud market simulator that applies several Business policies and allows users to define new policies as JBoss Drools rules [3].

The remainder of this paper is structured as follows. After the discussion of the related work, Section 3 describes the scenario in which Client Classification is applied: its participants and some preliminary definitions. Section 4 introduces the proposed rules for Client Classification: their motivation and their concrete implementation. Next, Section 5 describes the simulation environment and shows the experimental results that demonstrate the validity of the rules. At the end, Section 6 describes the conclusions of this paper and states the future research.

2 Related Work

In this paper, we extend part our previous work in Negotiation Models [16] and Rule-Based SLA Management for maximizing BLOs [15]. Previous work introduced several policies for maximizing the revenue of providers in Cloud Computing Markets [9,14]: dynamic pricing, overselling of resources, dynamic scaling of resources, migration of Virtual Machines (VMs), etc. This paper introduces rules that are essentially similar, but focused in Client Classification from the provider side.

Many previous works classify SLAs by considering the client information. The innovation of this paper relies on the proposal of new rules for price discrimination and client-aware overselling of resources, and their exhaustive evaluation in terms of revenue, client affinity, QoS, and SLA fulfillment. In addition, whilst related works tend to classify users in function of their internal/external condition, this paper defines them in a continuous range between 0 (lowest preference) to 1 (highest preference).

Client Classification is a usual practice in many businesses, such as banking services [2]. These businesses categorize clients in function of their size, budget, etc. and establish policies that define clearly the priorities of the clients, their protection level, their assigned resources, Quality of Service, etc. In Cloud Computing, Amazon Elastic Computing Cloud (EC2) provides a set of predefined VM instances [1], each one with different performance profiles (CPU load, Memory, etc.), but a fixed Quality of Service: they promise that their machines have an annual availability of 99.5%. This approach may be economically suitable for huge resource providers, but not for smaller providers. With this paradigm, small providers should overprovision resources for minimizing risks and provide high availability. We try to channel the risk to the SLAs with the lowest priority according to the defined BLOs. In case of SLA violation, the Clients will receive an economic compensation proportional to the seriousness of the violation.

Previous papers introduced some policies similar to those introduced in this paper. Sulistio et al. [22] propose overbooking strategies for mitigating the effects of cancellations and no-shows for increasing the revenue. The overbooking policies used in this paper consider in addition the possibility of under-usage of the reserved resources of the client. Dube et al. [10] establish different ranges of prices for the same resource and analyze an optimization model for a small number of price classes. Their proposal is similar to our proposal about establishing Gold, Silver and Bronze ranges and optimizing their QoS performance giving priority to the contracts that report the highest economic profit. We extend this work by combining the QoS ranges with several other policies, such as Price Discrimination. Another main difference between this paper and the work from Sulistio et al. [22] and Dube et al. [10] is that the main BLO of our work is the Client Classification instead of the Maximization of the Revenue.

Püschel et al. [18] propose a scheme for Client Classification by means of price discrimination, different priorities in job acceptance and differentiation in Quality of Service. They adopt the architecture of an EERM. The EERM supports the optimization of SLA Negotiation and Management by dealing with both economic and technical information of Cloud Computing Markets. In addition, this paper extends the research of Püschel et al. [18] in Client Classification with the extension and detail of the policies, and deeper validation of them by means of a tailored simulation of Clients, Cloud Market, EERM, and Resource Fabrics.

3 Preliminary definitions

A Cloud Market has two main actors: Clients and Providers. Clients try to buy resources in the Market to host their services, by sending offers to providers to start a negotiation. Each provider owns a set of N physical machines. Each physical machine can host several VMs that execute single tasks, such as Web Services or Batch Jobs. The QoS terms of a task are described in $SLA = \{Rev(vt), C, \overrightarrow{S}, \Delta t\}$:

- $Rev(vt)$ is a revenue function that describes how much money the provider earns after finishing correctly or incorrectly a task. vt is the amount of time in which the provider has not provided the agreed QoS to the client. Let MP be the Maximum Penalty (can be seen as negative revenue: lower MP implies higher penalties), MR the Maximum Revenue, MPT the Maximum Penalty Threshold, and MRT the Maximum Revenue Threshold, Equation 1 describes the revenue function. If $vt < MRT$ the SLA is not violated (0 violations); if $vt > MPT$, the SLA is completely violated (1 violations). $MPT > vt > MRT$ implies a partial violation ($\frac{vt-MRT}{MPT-MRT}$ violations).

$$Rev(vt) = \frac{MP - MR}{MPT - MRT}\,(vt - MRT) + MR \qquad (1)$$

This equation allows a grace period where the provider can violate the SLA without being penalized. When vt surpasses the MRT threshold, the revenue linearly decreases (see Figure 1) in function of vt. The Maximum Penalty MP is defined for avoiding infinite penalties. Client and provider can negotiate the values of MRT, MR, MPT, MP for establishing different QoS ranges for the clients, which report different revenues and penalties for the providers [16].

- C is the client information. Let id be the client identifier and $\overrightarrow{CD}$ a vector that handles the description of the client, then $C = \{id, \overrightarrow{CD}\}$. The information contained in $\overrightarrow{CD}$ must be decided by the System Administrator and applied consequently in the policies.
- $\overrightarrow{S}$ describes the QoS of the purchased service: throughput, response time, and so on.
- Δt is the time period requested to allocate the task.

The revenue function $Rev(vt)$ (as well as all the revenue figures in the evaluation) subtracts the penalties from the incomes, so it indicates how profitable is the allocation and execution of a SLA with a given set of policies. However, it does not indicate the provider's net benefit because it does not consider other costs, such as infrastructure maintenance.

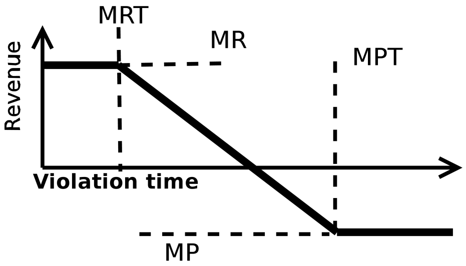


Fig. 1: Revenue of a SLA in function of the violation time (Equation 1)

3.1 Client Classification criteria

We propose the classification of clients according to the **priority** that the provider assigns to them. This priority can be described using two different criteria:

Client Affinity: The affinity (aff $\subseteq [0, 1]$) measures how the client is related to the provider. For example, aff $= 1$ for a completely internal user; aff $= 0.25 \sim 0.75$ for a client from a company with privileged relationship with the provider (e.g. in the same business cluster); aff $= 0$ for a completely external client. The calculation of the affinity may be different among different providers, depending on their business goals. How affinity is calculated is not important in this paper: the main topic is how to discriminate clients in function of their affinity.

Quality of Service: The same Cloud provider could host critical tasks and tasks that can tolerate lower QoS. For example, e-commerce applications may need extra QoS guarantees to avoid losing money on service unavailability. It is reasonable to allow critical clients to buy extra QoS guarantees at higher prices, and keep cheap prices (but fewer QoS guarantees) for non-critical tasks. The different ranges of QoS are defined by establishing different values for MRT, MR, MPT and MP in $Rev(vt)$ (Equation 1). We define three ranges of QoS, in descending order: Gold, Silver, and Bronze. The higher the QoS range, the higher MR and the lower MP, MRT and MPT (lower values of these three values imply higher penalties).

The policies for Client Classification are applied when the SLAs are negotiated between client and provider and allocated by the provider: the EERM gives priority to users to which the provider has high affinity when providing access to the resources by applying policies for Price Discrimination and Overselling of Resources.

4 Applying Client classification in negotiation time

To facilitate the reading of this paper, the names of the policies have been abbreviated according to the next notation: $PolicyName^{PriorityType}$. $PolicyName$ is an abbreviation of the policy name. The abbreviations of all the policies are shown below, enclosed in parentheses next to their names. $PriorityType$ is an abbreviation of the magnitude that is used for calculating the priority of the client: the affinity (Aff) or the Quality of Service (QoS). When the policies for Client Classification are compared with policies that prioritize the maximization of the revenue, the abbreviation for this last priority is RM (Revenue Maximization). As example, Price Discrimination policies that apply discount to clients according to their affinity are notated as $PrDsc^{Aff}$.

The proposed policies are:

Price discrimination (PrDsc): The price of a task varies in function of the time slot, the workload of the resources of the provider, and the amount of resources required for providing the agreed QoS [16]. In addition, we propose to apply discounts to clients proportionally to their affinity.

Overselling of Resources (Ovrs): Clients do not always use all the resources that they buy. In consequence, the spare resources are resold to other clients according to their priority. This policy will increase both the revenue of the provider and the average priority of the clients in the system.

4.1 Price Discrimination (PrDsc)

In our previous works, providers dynamically establish the prices for maximizing their revenue. They ask for high prices when the workload is high (peak hours) and low prices when it is low (off-peak hours) [16,15]. This maximizes the profit by attracting clients when the system is idle and maximizing prices when the demand is high.

$PrDsc^{Aff}$ policy is built on top of the $PrDsc^{RM}$ policy: after calculating the best resource allocation for maximizing the economic profit according to the Dynamic Pricing policies introduced in our previous works ($PrDsc^{RM}$) [16,15], the calculated revenue is multiplied by $(1 - affinity)$. This allows users with some affinity to receive a discount that is proportional to their affinity. This policy combines Client Classification with Revenue Maximization as a secondary BLO and always considers affinity as the main priority. Multiplying price by $(1-affinity)$ will linearly prioritize users (a user whose affinity is 1 will have the double of priority than a user whose affinity is 0.5). However, other distributions such as $(1 - affinity^2)$ could be considered in function of the provider policies.

The $PrDsc^{QoS}$ policy is not considered because it would not have sense: Gold tasks must not be cheaper than Silver tasks, and Silver tasks must not be cheaper than Bronze tasks.

4.2 Overselling of resources (Ovrs)

Sometimes the clients do not use all the capacity that they have reserved because they tend to slightly overprovision the required computing resources that they finally use. Thanks to Cloud Computing elasticity mechanisms, the overprovisioning required by clients is very low [12]. However, the summation of the spare resources of all the clients may be sold to other clients to increase the resources usage.

We propose the sale of capacity that has been sold previously but the client is not using: when a client negotiates a SLA and there are not enough resources to allocate it, the scoring function in Equation 2 is calculated over the set $j = \{1 \dots N\}$ of N physical machines. The physical resource j with the highest positive score is selected as candidate for executing the task and the $PrDsc$ policy is triggered for establishing a price. If there are not physical resources whose score is positive, the job is rejected.

$$score_j = 1 - \frac{\int_{t_i}^{t_f} R'_{used}(t) + R_{req}(t)\, dt}{\int_{t_i}^{t_f} R_j(t)\, dt} \tag{2}$$

The terms of Equation 2 are described herewith:

- $R_{req}(t)$ is a constant function that represents the amount of bottleneck resources requested in the SLA under negotiation.
- $R_j(t)$ is a constant function that represents the amount of bottleneck resources in the physical resource j.
- Let $R_{used}(t)$ be a prediction of the bottleneck resources that the SLA under negotiation will use; let $\delta \subset [0, 1]$ be the maximum percentage to penalize or unpenalize the predicted workload in function of the client priority P. The priority-corrected prediction is defined as $R'_{used}(t) = (1 + \delta - 2\delta P)R_{used}(t)$. The prediction of the used resources is artificially increased when the priority of the client that negotiates the SLA is low and artificially decreased when the priority is high.

The amount of resources used by services at a given time can be obtained from monitoring information. The prediction of resource usage for a given service ($R_{used}(t)$) can be calculated statistically or by several Machine Learning algorithms [20] from the monitoring information. In this paper, we use the CPU usage from Resource Monitoring because it is the bottleneck resource for the majority of services to be executed in the Cloud Provider. Equation 2 is abstract enough to allow using any other type of resource, such as memory or network bandwidth.

The corrections made in $R'_{used}(t)$ will motivate a higher acceptance of clients to which the provider has high affinity. As example, $\delta = 0.4$ in the experiments. This value is only chosen for showing the tendency of the graphs. Higher values of δ would decrease the revenue and increase both the average affinity and the number of SLA violations. Lower values of δ would have the opposite effect.

5 Experimental results

This section describes the experimental environment and its configuration values. We have used the EERM Simulator [4] to execute and evaluate the policies that are introduced in this paper. The EERM Simulator is a fine-grained Cloud Market simulator, which simulates the complete cycle of a Cloud Resource sale and execution: services discovery, SLA negotiation process between provider and client, execution of web services or batch jobs and monitoring of the resources. It supports many features of Cloud Computing, such as elasticity of resources or migration of VMs. In addition, it integrates the Drools [3] Rule Engine to configure the SLA allocation and management policies in function of the BLOs (e.g. the Client Classification policies described in this paper).

The advantages of using a simulation environment instead of real machines is the possibility of generate more data with fewer resources in less time, so the evaluation is more accurate. For every experiment, a total of 64 CPUs working during a week have been simulated using real web workloads to acquire statistically representative data.

5.1 Simulation Environment

The constant values and the parameters of the simulation described are arbitrary because there are no real market traces to extract data from. Different real market scenarios could require different values, but the contribution of this paper is to show how Client Classification reports benefit **qualitatively** but not quantitatively. In other words, the paper shows how a given policy can improve the average affinity of the clients that use the system but not whether its values are optimum, because they would vary in function of the market status. In our future work, the provider will automatically adjust its parameters for self-adapting to changing market environments.

A Cloud Market has two main actors: Clients and Providers. Providers offer VMs of variable sizes. Clients try to buy resources in the Market to host

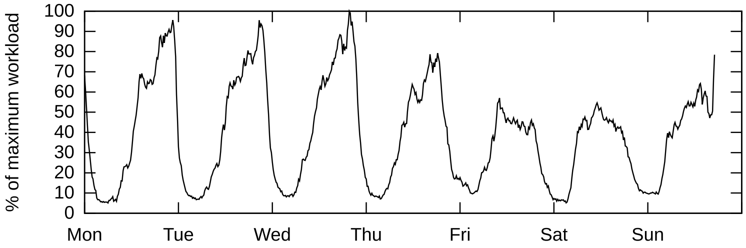


Fig. 2: Sample pattern of web workload

their services, by sending offers that contain $\{QoS, C, \overrightarrow{S}, \Delta t\}$, in which $QoS = \{Gold, Silver, Bronze\}$. For the same task in equal time and load conditions, the maximum price that the client is willing to pay for Gold QoS is 50% higher than the one for Silver QoS, and the maximum price that the client is willing to pay for Silver QoS is 20% higher than the one for Bronze QoS.

The Web workload is acquired from a real anonymous ISP (see figure 2), and varies in function of the hour of the day and the day of the week [7].

When the offer is in the market, the providers that accept it return a revenue function $Rev(vt)$, which specifies the prices and penalties to pay for the execution of that service. Finally, the client chooses the provider with a best price and time schedule for its interests and sends him a confirmation.

When a provider checks the offer from the client, it applies Machine Learning techniques to predict future workloads and verify whether the offered job can be executed correctly [20,21]. A bad prediction might entail a violation of the SLA.

In all the simulations, four different Cloud providers sell their services in a market during a week. For each experiment there is:

1. A provider that executes all the introduced policies until that subsection. It prioritizes users to which there is high affinity.
2. Same as Provider 1, but prioritizing tasks with high QoS.
3. A provider that executes all the introduced policies until the previous subsection. It prioritizes users to which the provider has high affinity in experiments that compare it with Provider 1 or tasks with high QoS in experiments that compare it with Provider 2. In the first section, it does not execute any policy and uses a fixed pricing schema as current Cloud providers [1,6].
4. A provider that executes the same policies as Providers 1 and 2 but without client classification as a main BLO. Its priority is the maximization of the economic profit [15].

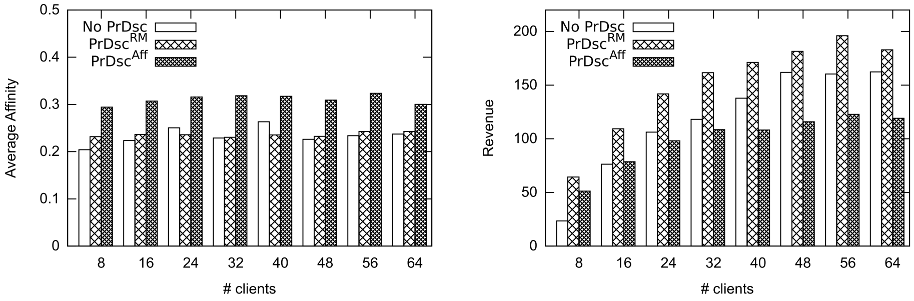
Every provider belongs to a different organization. All of them have the same number of resources: two 8-CPU physical machines. Every provider has an affinity higher than 0 to the 25% of the clients in the market, and equal to 0 to the other 75% of clients. The affinity of the clients of the same organization than the provider ranges from 0 (non-inclusive) to 1 (inclusive) following a uniform distribution. Summarizing, the average affinity of all the clients is ~ 0.21 for every provider.

Each client asks for Gold, Silver or Bronze QoS, independently of their organization. 1/6 of the clients ask for Gold QoS, 2/6 ask for Silver QoS, and 3/6 ask for Bronze QoS.

It is important to evaluate how the providers behave and how effective the policies are in different scenarios. For example, if there are many providers and few clients, the prices and the load of the system will be low; if there are too many clients and the providers cannot host all of them, prices and the system workload will be high. To evaluate the policies in all the scenarios, the experiments are repeated with different offer/demand ratios for each policy, gradually from low to high demand.

5.2 Price Discrimination (PrDsc)

Figure 3a compares the average affinity of the clients that buy services in providers that are competing in the market (see Section 5). Every provider has different policies for Price Discrimination: $NoPrDsc$ policy, $PrDsc^{RM}$, and $PrDsc^{Aff}$. The x axis shows the number of clients in each experiment, and the y axis represents the average affinity of the clients that used each resource. Each column group represents the obtained results of the providers in different experiments. The figure shows that the provider that implements $PrDsc^{Aff}$ increments the average affinity of its clients by 50%. The average affinity of clients in providers without $PrDsc^{Aff}$ is almost the same as the average affinity of all the clients in the market (~ 0.21).



(a) Average affinity

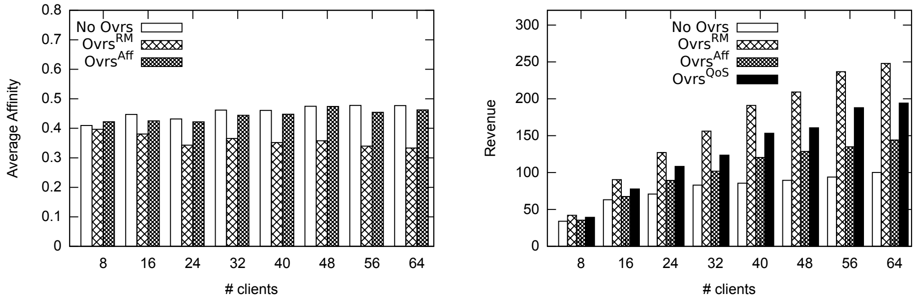
(b) Revenue

Fig. 3: Average affinity and revenue when using different $PrDsc$ policies

Figure 3b is structured similarly to Figure 3a, but instead of showing the average affinity for each provider in each experiment, it shows the revenue of the providers (y axis). It shows that revenue is noticeably decreased if compared with fixed-pricing and revenue maximization providers. It is demonstrated that the increment of the average affinity of the clients of $PrDsc^{Aff}$ penalizes the revenue. To compensate the impact in revenue of $PrDsc^{Aff}$, hardware resources may be oversold as explained in next section.

5.3 Resources Overselling (Ovrs)

Figure 4a has a similar structure to Figure 3a: it shows the average affinity of the clients according to the policy combination in the provider. The three providers apply $PrDsc^{Aff}$ but they differ in how they implement Overselling. The provider of the previous section is labeled $NoOvrs$, because it does not apply Overselling. To compare the usefulness of overselling based on affinity discrimination ($Ovrs^{Aff}$), the figure also includes the results of a provider that performs $PrDsc^{Aff}$, but its overselling policy is driven by revenue instead of affinity (labeled as $Ovrs^{RM}$). As the intention of $Ovrs^{QoS}$ is not to attract clients to which the provider has high affinity, this policy is not included in the figure. Figure 4a shows that not considering the client affinity in the overselling policy decreases average affinity of the clients. It is not caused by any type of penalization, but it is a statistical fact: more clients enter the system, regardless their affinity. $Ovrs^{Aff}$ maintains similar affinity levels to those of $NoOvrs$ but increasing the revenue of the provider, as in Figure 4b.



(a) Average affinity

(b) Revenue

Fig. 4: Average affinity and revenue when using different $Ovrs$ policies

Figure 4b compares the revenue of four providers. All four implement $PrDsc^{Aff}$, but different overselling policies. The provider labeled as $NoOvrs$ does not apply any overselling policy, as in previous section. The other providers apply overselling policies based on Revenue Maximization ($Ovrs^{RM}$), affinity discrimination ($Ovrs^{Aff}$), and QoS range ($Ovrs^{QoS}$). Figure 4b shows that all the overselling policies have a positive impact on earnings. The provider labeled as $NoOvrs$ is the lower bound and the provider labeled as $Ovrs^{RM}$ is the higher bound. $Ovrs^{Aff}$ and $Ovrs^{QoS}$ stay in the middle of both: the clients are classified without renouncing the revenue completely. The revenue with $Ovrs^{Aff}$ is lower than the revenue with $Ovrs^{QoS}$ because $Ovrs^{QoS}$ prioritizes Gold and Silver contracts, which report more revenue than Bronze ones.

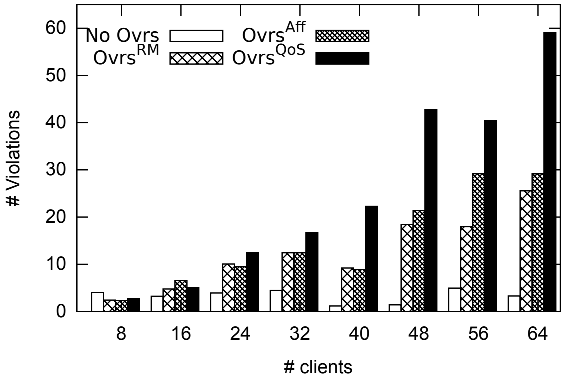


Fig. 5: Number of violations when using *Ovrs* policies

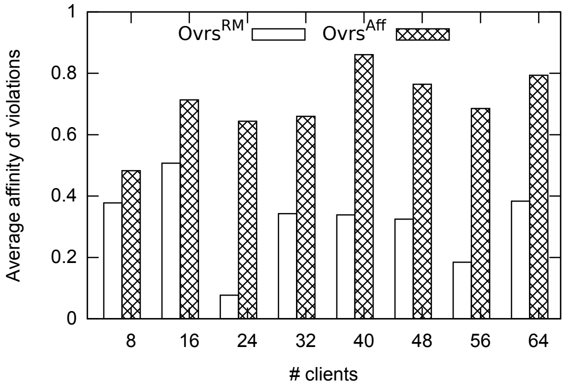


Fig. 6: Average affinity of the violations when using $Ovrs^{Aff}$

However, overselling considerably increases the number of SLA violations (Figure 5) because of two reasons: the associated error to the predictor component, and the permissiveness with clients to which the provider has high affinity in terms of workload that makes the provider to violate a highest proportion of SLAs of this kind of clients (Figure 6). The provider that applies $Ovrs^{QoS}$ reports the highest number of violations because Gold and Silver SLAs have

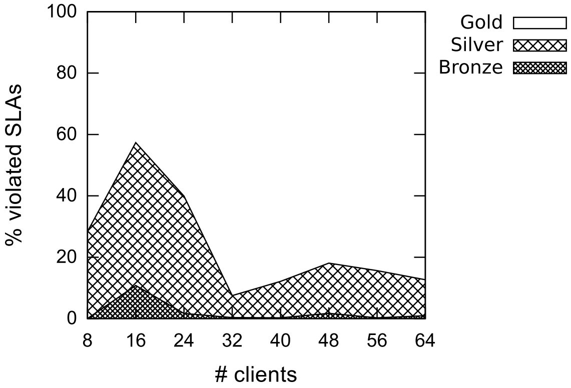


Fig. 7: Proportion of violations by QoS range when using $Ovrs^{QoS}$

stricter requirements that are more difficult to accomplish, as shown in Figure 7. Despite the increase in the number of violations, the proportion of violated SLAs over the total of allocated SLAs remains below 2% in the worst case. Figure 7 is a stacked chart that shows the percentage of each QoS range from the total of violations for $Ovrs^{QoS}$ provider in several market simulations with different number of clients. It shows that the higher the QoS rank, the higher the percentage of violated SLAs. More violations of high-QoS SLAs do not mean that the QoS for Gold SLAs is lower than the QoS for Silver SLAs: it is more difficult to achieve the QoS requirements of Gold SLAs because the QoS requirements are higher, but an achievement of 90% of the QoS for Gold is still higher than the 100% of the QoS for Silver or Bronze.

The negative effects of $Ovrs^{Aff}$ and $Ovrs^{QoS}$ can be minimized by applying policies for runtime management of resources from our previous work [15]: usage of VMs elasticity, selective SLA violation and live migration of VMs for dynamically reconfiguring the Cloud data centers and minimize the number of violations.

6 Conclusions and future work

In this paper, we have introduced a set of policies for managing SLAs in a Cloud provider considering the classification of clients. Two facets can be used to classify the clients: client affinity and QoS. These policies have been evaluated through experiments that show the improvement of adding each policy to the set of previously introduced policies. We have introduced $PrDsc$ and $Ovrs$ for increasing the proportion of high-priority SLAs in the provider. After the application of all the policies, the EERM increases the number of priority clients (high-affinity or Gold and Silver, depending on the chosen type of priority), keeping a reasonable compromise between giving access preference to priority users and keep a high revenue from non-priority users.

We conclude that Client Classification policies in SLA negotiation achieve their objectives: the percentage of priority users is increased when applying them in negotiation time. Classification by QoS is suitable for a pure Cloud provider whose business is only based on selling its resources (it does not use them for its internal applications). Classification by affinity is more suitable for organizations that mix internal and external applications on their resources.

The policies presented in this paper rely on some constant values that may not lead to the optimum achievement of the BLOs. However, getting the optimum results is not the main objective in this paper. The key value of this work is to show the tendencies of applying the explained policies in terms of increment of high-priority clients that use the system. The aim to further improve brings a research opportunity for future work: adding dynamism to rules to allow them self-adapting at runtime. Dynamic Rules will allow providers to autonomously adapt to the changes in the environment and achieve the optimum results according to their own BLOs.

Applying the policies enhances client classification but also increases the number of SLA violations and the proportion of violations of high-priority SLAs. Since the violations percentage is acceptable (below 2% of the allocated SLAs), we can state that the advantages of applying $PrDsc$ and $Ovrs$ outweight the disadvantages for achieving the BLO for which both policies have been designed.

Our previous work in Rule-based SLA Management [15] introduced several policies for mitigating the negative effects of $Ovrs$ policies. Our future work will enhance existent policies for resources elasticity, selective SLA violation and live migration of VMs, by adding them client awareness to allow highest achievement of Client Classification BLOs.

Acknowledgements

This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2007-60625, by the Generalitat de Catalunya under contract 2009-SGR-980, and by the European Commission under FP7-ICT-2009-5 contract 257115 (OPTIMIS).

References

1. Amazon EC2 instances (last visit: Feb. 2011), http://aws.amazon.com/ec2/instance-types/
2. Client classification and reclassification policy of rabobank polska sa (last visit: Feb. 2011), http://goo.gl/AKu86
3. Drools rule engine, http://www.jboss.org/drools
4. Economically Enhanced Resource Manager (last visit: Aug. 2011), http://www.sf.net/projects/eerm
5. Spotify, http://www.spotify.com
6. Windows azure (last visit: Feb. 2011), http://www.microsoft.com/windowsazure/

7. Barford, P., Crovella, M.: Generating representative web workloads for network and server performance evaluation. In: 1998 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems (SIGMETRICS '98/PERFORMANCE '98). vol. 26, pp. 151–160. ACM Press, Madison, Wisconsin, USA (June 1998)
8. Buyya, R., Yeo, C.S., Venugopal, S.: Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In: 10th IEEE Intl. Conf. on High Performance Computing and Communications (HPCC 2008). pp. 5–13. IEEE Computer Society, Dalian, China (September 2008)
9. Choong, H.L., Jinwoo, S., Kyoungmin, P.: Grid and p2p economics and market models. In: Society, I.C. (ed.) 1st IEEE International Workshop on Grid Economics and Business Models (GECON 2004). pp. 3–18. Seoul, South Korea (April 2004)
10. Dube, P., Hayel, Y., Wynter, L.: Yield management for IT resources on demand: analysis and validation of a new paradigm for managing computing centres. Journal of Revenue and Pricing Management 4:1, 24–38 (2005)
11. Foster, I.: The anatomy of the grid: Enabling scalable virtual organizations. Cluster Computing and the Grid, IEEE International Symposium on 0, 6 (2001)
12. Goiri, I., Julia, F., Fito, J.O., Macias, M., Guitart, J.: Resource-level qos metric for cpu-based guarantees in cloud providers. In: 7th International Workshop on the Economics and Business of Grids, Clouds, Systems, and services (GECON 2010). vol. 6296, pp. 34–47 (August 2010)
13. Kaufman, L.M.: Data security in the world of cloud computing. IEEE Security and Privacy 7, 61–64 (2009)
14. Lee, H.Y., Choo, T.T., Khee-Erng, J.L., Wong, W.: A grid market framework. In: 3rd International Workshop on Grid Economics and Business Models (GECON 2006). pp. 70–79. Singapore (May 2006)
15. Macias, M., Fito, O., Guitart, J.: Rule-based sla management for revenue maximisation in cloud computing markets. In: 2010 Intl. Conf. of Network and Service Management (CNSM'10). pp. 354–357. Niagara Falls, Canada (October 2010)
16. Macias, M., Guitart, J.: Using resource-level information into nonadditive negotiation models for cloud market environments. In: 12th IEEE/IFIP Network Operations and Management Symposium (NOMS'10). pp. 325–332. Osaka, Japan (April 2010)
17. Porter, M.E.: Clusters and the new economics of competition. Harvard Business Review 76(6), 77–90 (Nov-Dec 1998)
18. Püschel, T., Borissov, N., Macias, M., Neumann, D., Guitart, J., Torres, J.: Economically enhanced resource management for internet service utilities. In: WISE. Lecture Notes in Computer Science, vol. 4831, pp. 335–348. Springer (2007)
19. Rappa, M.A.: The utility business model and the future of computing services. IBM Syst. J. 43(1), 32–42 (2004)
20. Reig, G., Alonso, J., Guitart, J.: Prediction of job resource requirements for deadline schedulers to manage high-level slas on the cloud. In: 9th IEEE Intl. Symp. on Network Computing and Applications. pp. 162–167. Cambridge, MA, USA (July 2010)
21. Sandholm, T., Lai, K.: Evaluating demand prediction techniques for computational markets. In: 3rd International Workshop on Grid Economics and Business Models (GECON 2006). pp. 3–13. Singapore (May 2006)
22. Sulistio, A., Kim, K.H., Buyya, R.: Managing cancellations and no-shows of reservations with overbooking to increase resource revenue. In: Intl. Symp. on Cluster Computing and the Grid (CCGRID 2008). pp. 267–276. IEEE Computer Society, Lyon, France (May 2008)