A Risk-Based Model for Service Level Agreement Differentiation in Cloud Market Providers

Mario Macías, Jordi Guitart

A Risk-based Model for Service Level Agreement Differentiation in Cloud Market Providers

Mario Macias and Jordi Guitart

Barcelona Supercomputing Center (BSC) and
Universitat Politecnica de Catalunya - Barcelona Tech (UPC)
Jordi Girona 29, 08034 Barcelona, Spain
{mario.macias, jordi.guitart}@bsc.es

Abstract. Cloud providers may not always fulfil the Service Level Agreements with the clients because of outages in the data centre or inaccurate resource provisioning. Minimizing the Probability of Failure of the tasks that are allocated within a Cloud Infrastructure can be economically infeasible because overprovisioning resources increases the cost and is economically inefficient. This paper intends to increase the fulfilment rate of Service Level Agreements at the infrastructure provider side while maximizing the economic efficiency, by considering risk in the decision process. We introduce a risk model based on graph analysis for risk propagation, and we model it economically to provide three levels of risk to the clients: moderate risk, low risk, and very low risk. The client may decide the risk of the service and proportionally pay: the lower the risk the higher the price.

1 Introduction

Cloud Computing arisen as a successful commercial solution to sell computing resources as a utility: clients dynamically size the resources according to their workloads, and pay only for what they use. Cloud resources are usually sold as Virtual Machines (VMs) that can run isolated in the same hardware as other VMs and scale at runtime. In current commercial Clouds, infrastructure providers price the resources and clients may decide to buy them or not. There is no negotiation. Our research, however, is framed in research Cloud Market implementations such as the SORMA [10] Market Middleware. In Cloud Markets, both resource users and providers are autonomous agents that negotiate the terms of the Quality of Service (QoS) and the price that the client will pay to the provider. When the negotiation is finished, the terms of the contract are established in a Service Level Agreement (SLA) that keeps contractual information about the terms of the QoS as well as the pricing information (price and penalty to pay in case of violation of the SLA).

Cloud providers may not always fulfil the SLAs they agree with the clients because of outages in the data centre or errors in the hardware. Not fulfilling the agreed SLAs would lead to economic penalties [8] and a loss of reputation that can cause clients with high reliability requirements to not allocate their

tasks in the provider [7]. Minimising the Probability of Failure (PoF) of the tasks that are allocated within a Cloud Infrastructure can be economically infeasible. Overprovisioning resources increases the cost and is economically and ecologically inefficient because the overbooked resources are underused most of the time.

This paper has a main goal: to increase the fulfilment rate of SLAs at the infrastructure provider side by providing risk-aware policies that maximize the economic efficiency. We introduce a risk model based on graph analysis for risk propagation, and we model it economically to provide three levels of risk to the clients: moderate risk, low risk, and very low risk. The client may decide the risk of the service and proportionally pay: the lower the risk the higher the price.

To achieve the stated goal, this paper introduces the following contributions:

1. Model of the PoF of a multi-tier service that is hosted in a Cloud data centre by means of the analysis of the links between virtual resources.
2. A new revenue model that will help providing different levels of risk at different prices, while adapting prices to the present value of the resources (that is, the rate the resources decrease their value over time).

This paper is structured as follows. After the related work section, Section 3 introduces the baseline negotiation and revenue model that is used as framework for this paper; Section 4 shows the risk model and Section 5 shows the revenue model; Section 6 describes the experiments and evaluates their results; at the end, Section 7 show the conclusions and the future work lines.

2 Related work

This paper extends our previous work [7,8], which demonstrated that differentiating SLAs according to their QoS level could lead to economic benefits in an open market with many competitors, because there is a variety in the objectives of clients. Some clients may require high QoS guarantees and other clients may prioritize lower prices. This paper extends the previous research by modelling the risk of complex cloud appliances and defining a revenue model that would allow to provide accurate prices as a function of the offered QoS level.

Bayesian networks[4] define a graph model for describing the probability of an event from the probabilities of the events that would cause it, in terms like *"if event A happen, event B will happen"*. That is inaccurate for Cloud appliances, because a failure in a node would not always involve a failure in the linked node. In addition, it is difficult to express some complex relations like redundancy. Our graph model extends the Bayesian Network model with the addition of weighted links and introduces two special types of node for representing union and intersection operations. These additions ease the expression of some complex Cloud appliances and the risk propagation through their nodes.

Djemame et al. [2] described an architecture to assess risk in computing Grids that allow providers to estimate the risk of agreeing a given SLA and use management techniques to maximize its fulfilment. They use risk assessment for

task scheduling. Our work intends to be an upgrade of some of their risk models to adapt them to the architecture of Clouds and using such risk assessment also for improving business objectives.

Yeo and Buyya [16] provide two methods for risk analysis: separate, which only evaluates the risk over a facet, and integrated, which evaluates the risk over multiple facets. In the integrated method, they assume all the facets are independent from each other. In our model, the facets are the multiple risks of all the independent resources in a multi-tier application, but we do not consider them as independent. The risk is propagated according how such resources are linked, and the effects of such uncertainty have different impact depending on the location of the risk within the resources graph.

Sawade et al. [12] consider that risk models may lose their validity over time for some reason (the environment changes, the inputs change, learning errors, ...). Our model can minimize these drawbacks, because it is dynamically built according to the SLA templates from the clients.

The pricing models from Becker et al. [1] consider the concept of Business Value: how much a client is willing to pay for any extra unity of QoS. We also consider this concept in our model. However, our intention is not to solve the issue of calculating it but consider it as an important part of a revenue model for providing multiple pricing and risk levels. They also calculate the penalty of the execution of multiple services, while distributing the risk between the different services. Our approach considers only a single project and calculates the risk of penalties by evaluating the internal topology of such service.

Simao et al. [14] propose a pricing model that allows clients and providers to negotiate the price and penalties, but also how much service degradation the client is willing to accept and how much it will pay proportionally to such degradation. As in this paper, they define three levels of QoS and propose a depreciation strategy to control the degradation of the SLAs for each profile when the resources are overloaded. Our model does not explicitly select the SLAs to degrade, but allocates them by risk levels according to the envisioned probability of failure. Both models are complementary and may coexist simultaneously.

Wee [15] profiles in detail the Amazon Spot Instances. He concludes that such model does not motivate enough users to move their workload to the off-peak hours. Our model provides an extra incentive to move because, in addition to the lower prices derived from the low market demand, users can benefit also of lower risks derived from the low workload.

Li and Gillam [6] apply risk assessment to the financial aspect of the grid. They provide node granularity risk assessment to calculate prices and penalties for the SLAs. Our approach combines assessments from several nodes and links them to consider a Cloud appliance topology.

3 SLA negotiation model

This paper focuses two stages of the Infrastructure as a Service (IaaS) provisioning: the negotiation of SLAs between the clients and an Infrastructure Provider

(IP), and the provisioning of resources to fulfil the terms of the agreement. We consider the OCCI Standard [9] to describe the infrastructure: the client can get three types of cloud resources (compute nodes, networks, and storage nodes) and define how they are linked by means of network interfaces and storage links.

When a client wants to host a service, it calculates how many Cloud resources it needs and sends an offer to the IP to start a negotiation. Each IP owns a set of N physical hosts. Each physical machine can host several VMs. The SLA of a set of VMs is described as $SLA = \{\overrightarrow{S}, \overrightarrow{L}, \Delta t, RL, Rev(vt)\}$. $\overrightarrow{S} = (s_1, \ldots, s_k)$ are the Service Level Objectives (SLOs) that describe the amount of resources to be purchased by the client, and $\overrightarrow{L}$ describes how the resources are linked according to its OCCI description. Each s_* term represents the amount of CPUs, Memory, Disk, network bandwidth, and so on. RL is the risk level (medium, low, very low) that the client selects for its service, having an impact in the price. Δt is the time period during which the VM will be allocated. If the IP provider has not enough free resources to allocate the SLA, it can reject it.

Current commercial Clouds do not compel to specify Δt and sell resources at fixed price/hour. In contrast, research Cloud Market middlewares [10] require to specify Δt when a client and a provider negotiate a price. Market mechanisms would motivate clients to distribute their workloads across time, if possible.

$Rev(vt)$, which was originally proposed in [7], is a revenue function that describes the revenue of a provider after the operation of a SLA. The violation time vt is the total time during which the agreed QoS has not been provided. Let MP be the Maximum Penalty (seen as negative revenue), MR the Maximum Revenue, MPT the Maximum Penalty Threshold, and MRT the Maximum Revenue Threshold, Equation 1 describes the revenue function.

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

Equation 1 allows a grace period where the provider can violate the SLA without being penalized. If $vt \leq MRT$, the provider will get all the negotiated revenue (MR); if $vt \geq MPT$, the provider will pay the maximum penalty (MP); for $MRT > vt > MPT$ the money to earn or the penalty to pay will be proportionally in between MR and MP as a function of vt (see Figure 1) . The Maximum Penalty MP is defined to avoid infinite penalties. Client and provider can negotiate the values of MRT, MR, MPT, MP for establishing different QoS ranges that would report different revenues and penalties. The client can assume that vt will be normally near zero and will have to pay MR most times.

When the client wants to acquire resources from the IP, it starts the next negotiation: it sends to the IP an SLA template with values for $(\overrightarrow{S}, \overrightarrow{L}, \Delta t, RL)$. According to its envisioned status for Δt, if the IP has enough resources, it returns a complete SLA that specifies the values for MRT, MR, MPT and MP as a function of the number of resources, the market status, and the risk level requested by the client. A lower risk level would entail higher prices (MR), but higher penalties (lower MP and less tolerance to SLA violations (lower MRT

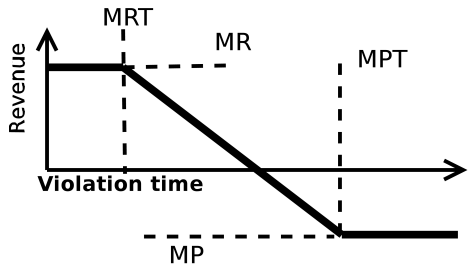


Fig. 1: Revenue of a SLA as a function of the violation time (Equation 1)

and MPT)). If the client agrees the terms proposed by the IP, it confirms the SLA or, otherwise, rejects it and looks for another provider in the market.

4 Risk Management

This paper considers risk as the effect of uncertainty on objectives [3]. Risk depends on two facets: the probability of an unwanted event and how it deviates the desired outcomes. Given a time frame, an unwanted event may occur. This may impact or not in the desired outcomes. For example, if a single disk fails within a storage system with redundancy, an unwanted event occurred but its impact is low (cost of replacement, but no data has been loss). In our work, the impact of risk will be economically determined by the penalties that are specified in the SLA. Calculating the risk is calculating the PoF of a complex system, and calculating how the failure can impact the fulfilment of the SLA.

Measuring risk in individual components. For each component based on OCCI types, we identify as failure each incident that causes this component to not work correctly. Although real computing resources may have multiple degrees of malfunction, our model adopts a binary definition of malfunction for single resources: working/failure. Our model does not care about the grade of performance for each individual component, but whether the propagation and aggregation of all the errors/misbehaviours of the individual resources will lead the system to fulfil the SLA or not.

The quantitative risk assessment for each component is based on the process proposed by Guitart et al. [3], which divides the risk assessment into the following stages: (1) Identify what weaknesses could prevent a component from functioning properly. In this paper we identify two: overload of resources and age of resources. (2) Identify which situations can exploit system vulnerabilities. Information from vulnerabilities and threats can be gathered from experts, historical databases and files. (3) The monitoring information is retrieved at different levels. We basically consider information from physical and virtual hosts. (4) Identify the likelihood of a threat acting over a vulnerability. This information is retrieved from historical facts that take place in a specific context. And (5) calculate the PoF of a single component as a function of the current monitoring status, given

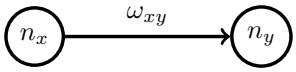
a time frame (e.g. calculate the PoF of a network during the next 24 hours). In this paper, we use statistical information from monitoring history (e.g. check the historic of failures when resources reach a given load). In our future work, we will explore alternative methods: machine learning, non-linear regressions, etc.

Measuring risk in complex appliances. In compliance to OCCI, our risk model is composed by nodes that have dependencies between them. A node n_x is failing when it is not providing the agreed QoS (e.g. a disk is not able to read or write data, a compute resource is not providing all the promised computation power, a network fails...). The PoF of n_x is notated as $P(n_x)$ and it can be measured according to the steps as previously described.

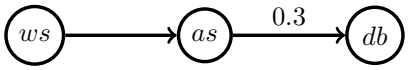
Let n_x and n_y be two nodes that are linked to work together as a composite system. We consider that n_x has a risk link of weight ω_{xy} to n_y when the failure of n_y prevents n_x to work correctly (for example, n_x is an application server that uses n_y as a database). The weight $\omega_{xy} \in [0, 1]$ is the probability that a failure in n_y is propagated to n_x. In consequence, n_x can fail because an internal failure on n_x or a failure in n_y that is propagated to n_x with probability ω_{xy}. Equation 2 defines $P'(n_x)$ as the propagated probability of failure of n_x.

$$P'(n_x) = P(n_x) + \omega_{xy}P(n_y) - \omega_{xy}P(n_x)P(n_y) \tag{2}$$

Equation 2 is based on the formula for union of probabilities, which assumes that $P(n_x)$ and $P(n_y)$ are independent (unlike $P'(n_x)$ that depends on both $P(n_x)$ and $P(n_y)$). The graphical notation for such risk relation is the next:



The aforementioned notation is used as a primitive for calculating the risk of complex systems. For example, let ws be a web server that handles requests from clients and contacts the application server as. We measured that the 30% of the times that as is invoked it accesses a database (db). If the database fails, the error will be propagated to as and, in consequence, to ws. In this example we assume $P(ws) = 0.05$, $P(as) = 0.01$, and $P(db) = 0.03$.

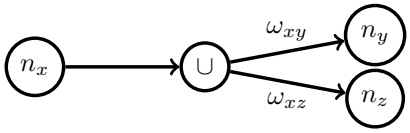


If the arrow between nodes does not show any number, we assume a weight value = 1 between risk nodes. From the client side, if the node ws fails, the complete web application is failing. The PoF of the complete super system is $P'(ws)$, which is calculated as shown in equation 3. Resolving it, the probability that the complete system fails (that is, the client cannot access ws) is ~ 0.068. It is always true that $P'(n_x) \geq P(n_x)$.

$$\begin{cases} P'(ws) = P(ws) + P'(as) - P(ws)P'(as) \\ P'(as) = P(as) + 0.3P(db) - 0.3P(as)P(db) \end{cases} \tag{3}$$

In the previous example, the probability of failure of node as that will be propagated to ws is actually the probability of failure of the subsystem formed by as and db. By this reason, Equation 3 calculates $P'(ws)$ as a function of $P'(as)$ instead of $P(as)$. Our model allows simplifying complex systems by grouping many of their nodes and treats them as a single node.

In our model, a node can also have risk dependencies to many other nodes. We introduce two types of meta nodes to represent unions and intersections between risk probabilities. The next system is interpreted as follows: the system headed by n_x will fail when there is a failure in n_x OR there is a failure in n_y (with probability w_{xy}) OR there is a failure in n_z (with probability w_{xz}).

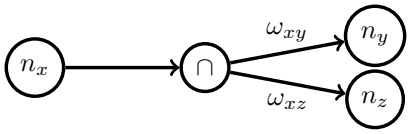


The node labelled as '$\cup$' (union operator) is a meta node to which $P(\cup) = 0$. It is used to allow grouping the subsystem formed by n_y and n_z and treating it as a single node when calculating the risk propagation to n_x (calculated in $P'(\cup)$). In consequence, calculating $P'(n_x)$ is solving the next equations:

$$\begin{cases} P'(n_x) = P(n_x) + P'(\cup) - P(n_x)P'(\cup) \\ P'(\cup) = \omega_{xy}P(n_y) + \omega_{xz}P(n_z) - \omega_{xy}P(n_y)\omega_{xz}P(n_z) \end{cases} \tag{4}$$

As example, imagine n_x is a VM that executes a disk-intensive task against a RAID-0 disk system which distributes the data chunks within two disks (n_y and n_z) for improving performance. If only one disk fails in a RAID-0 system, the complete system will fail, since there is no redundancy for recovering the data.

Our model also introduces the intersection operator '$\cap$' to model redundancy in fault tolerant systems:



The probability of failure of the subsystem headed by the node '$\cap$' is the intersection of probabilities of failure for nodes n_y and n_z, assuming that they are independent: $P'(\cap) = \omega_{xy}\omega_{xz}P(n_y)P(n_z)$. For example, imagine a RAID-1 disk system that mirrors two disks.

The combination of the union and intersection operators may also be used to model systems to which the redundancy is partial. For example, a master node M sends tasks to slave nodes A, B, and C. If one of the slave nodes fails, the other two nodes can handle the work; if two slave nodes fail, the complete system will fail.

Although we focus on the hardware failures at infrastructure level, our model allows also expressing software components as nodes within the graph, or simply considering the software failure within the PoF in the hardware node.

Risk-aware cloud operation. Risk must be considered during the allocation and operation of cloud appliances. For example, a client that needs high availability would negotiate SLAs with a high penalty for the provider in case of SLA violation. In such scenario, the Cloud provider has to minimise the PoF of the application according to two complementary strategies:

- For each node n_x, minimizing $P(n_x)$, which is caused by risk in the node (not propagated). This paper considers two factors that influence in $P(n_x)$: hardware lifetime and workload [13]. The failure rate of hardware resources is high both at the beginning and the end of the components lifetime. There is also direct correlation between the workload and the failure rate, being higher during peak hours and lower during off-peak hours. We use statistical analysis based on historical data to calculate $P(n_x)$.
- Consider decreasing $P'(n_x)$ for each node n_x. Analysing risk graphs and providing redundancy in the critical points of the graph would noticeably reduce the risks of the system with reasonable economical performance.

Analysing the risk propagation graphs is itself a large research field that would require to deep within the research of machine learning and pattern recognition algorithms, and how to apply them to this problem. The aim of this paper is to keep the focus in the risk and revenue model. We simplify the graph analysis by experimenting only with one template of application. The graph analysis has been done off line and the risk minimization policies always apply the same action with the graph: to add redundancy to the nodes whose failures would entail a failure to the rest of the application.

Both strategies for minimizing risk would entail an increment in the cost of operation. Next section describes a model for the management of the revenue during both SLA negotiation and operation that would allow providers providing differentiated risk levels consistently according to its business objectives.

5 Revenue Modelling

This paper uses Equation 5 to establish the price of a set of Cloud resources, given a time frame. MR is the price for a service (Maximum Revenue, as previously defined in Equation 1).

$$MR = RP + DO + BV \tag{5}$$

In Equation 5, RP is the Reservation Price: the minimum price the provider can sell a resource without losing money. DO and BV are subjective terms that may depend on several conditions. DO is the demand/offer overprice: a client may be willing to pay more when there is more demand than offer. DO will tend

to 0 when the demand is much lower than the offer. BV is the Business Value: the amount of money a client is willing to pay for an extra unit of QoS.

Our model calculates RP as the cost of amortization of all the resources that the client will use during a given period: the more amortized is a resource the lower is RP. Equation 6 shows how to calculate the amortization cost of a single Cloud resource that is allocated within a physical resource. The RP for a cloud appliance is the addition of the amortization costs for all its resources.

$$Cost_{Am} = (TCO - Amort)\frac{\Delta t}{(LT_{total} - LT_{now})H}\rho \tag{6}$$

TCO is the Total Cost of Ownership, the cost of the initial investment plus the common expenses in electricity and maintenance during the whole lifetime of a resource. $Amort$ is the sum of all the income associated to the provisioning of virtual resources for the given physical resource. Δt is the time that the client is willing to use the resource. LT_{total} is the Life Time that is planned for a group of resources: the time since it is provisioned until it is disengaged from the data centre. LT_{now} is the Life Time since a resource is provisioned until now. Finally $\rho = [0, 1]$ is a density function that indicates the proportion of a group of resources to which the cost is being calculated. For example, if a VM requires 4 CPUs from a node with 16 CPUs, $\rho = 0.25$. Finally, H is the percentage of usage of the resources as envisioned by the provider to this time. If $H = 1$, the provider would consider that all the resources that are assigned to a VM are at full occupation during this time. If the resources are underutilized, the value H would proportionally increase the reservation price that is needed for actually amortizing completely a resource at the end of its lifetime.

To avoid inequalities in the amortization of individual resources with the same age, we group all resources from the same type and age into an accounting group. Then the values TCO, $Amort$, ρ and LT apply to the total of resources instead of individual ones. Equation 6 differs from the traditional way to calculate the amortization cost, TCO/LT_{total}, because this formula assumes full load and does not consider how the value of a resource decreases over time.

To calculate DO overprice, our previous work [7] demonstrated that the DO must be low when the offer/demand ratio is high enough to allow users to choose from a big enough set of providers, and high only in peak hours, when most resources are busy.

Calculating BV is difficult because it may rely on several hidden variables that depend on the client, the market status, the reputation of the provider, etc. Instead of trying to synthesize them in a mathematical formula, Machine Learning techniques can allow providers estimating this value. However, those techniques are out of the scope of this paper. We apply a fixed overprice for the SLAs in our experiments, according to their level of QoS and Risk.

We account DO and BV within the total of amortized cost, which are overprices that accelerate the amortization of the resource and cause $Cost_{Am}$ value to decrease over time. That will allow the provider using different prices depending on the age of the resources that are being sold.

6 Evaluation

In our experiments, we used a Cloud Market simulator (available online [11]) that adopts the simulation architecture and methodology from our previous works [7,8]. We simulate 36 months of a IaaS provider that initially owns 50 hosts with 16 CPUs each one. The number of deployed services initially oscillates between 5 and 60 services/hour, according to a web workload that varies as a function of the hour of the day and the day of the week. To simulate the consolidation of the business of the provider, the average number of requests is linearly increased until it doubles its initial number at the end of the simulation. Because of the increase of the number of requests, the Cloud provider doubles its number of resources at month 18. From the point of view of Equation 6, there is initially an accounting group of resources and at the end of the simulation there are two accounting groups: the initial bunch of resources, and the new resources that were introduced at month 18.

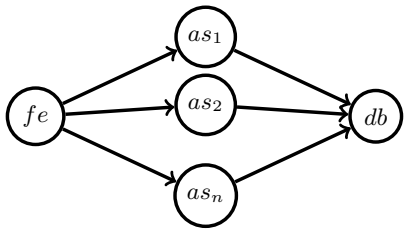


Fig. 2: Basic architecture of a web application

The clients can deploy several types of applications. In our experiments, the clients deploy web services according to the structure in Figure 2: a web front-end (fe) balances the job across a set of n application servers ($as_1, \ldots, as_n$) that use a database node (db) as persistence layer. The number of application servers varies from 2 to 4. The number of CPUs of each node follows a folded normal distribution [5] with both minimum value and variance equal to 1. The same distribution is used to determine the duration of the deployments, with minimum value and variance of 1 hour.

The allocation process of the SLA is the same as described in Section 3. The IP considers three different SLA allocation strategies, which offer three levels of risk for the SLA, from medium to lowest risk:

– **Cost Minimization** (CMin). The provider prioritizes the allocation of VMs in the hosts given two equally-weighted criteria: high consolidation, to save energy costs in hosts that are already running tasks and keep switched off those hosts that are idle [3]; and amortization, to allow lower prices according to the model in Equations 5 and 6. Because of high consolidation and resources age, SLAs allocated according this policy have the higher risk.

– **Node Risk Minimization** (NRMin). The provider prioritizes the allocation of VMs in the hosts according to two equally-weighted criteria: low consolidation, to lower the risks derived from overload in resources that would entail to not provide the agreed QoS; and resource age, trying to avoid the resources that are new and those resources that are near the end of their lifetime [13]. Figure 3 shows the used distribution of failures as a function of the age of the resource.
– **Graph Risk Minimization** (GRMin). The provider applies Node Risk minimization but, in addition, it analyses the OCCI links to try to detect single point of failures. Given the model in Section 4, the provider would detect that a failure in the database node would entail a failure in the whole application, so it decides to replicate it.

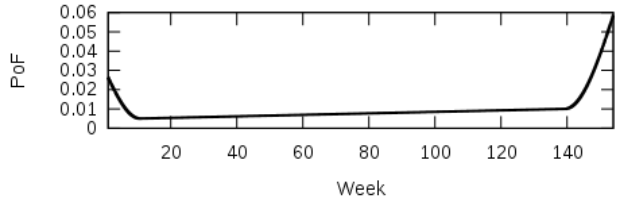


Fig. 3: Probability of Failure of resources over time

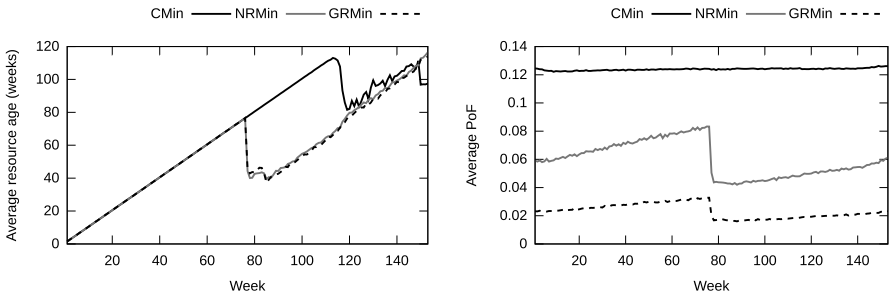
	CMin	NMin	GRMin
MP	-MR	-1.5MR	-2MR
MRT	$0.15\Delta t$	$0.1\Delta t$	$0.05\Delta t$
MPT	$0.75\Delta t$	$0.5\Delta t$	$0.3\Delta t$

Table 1: Revenue function values for each group of SLAs

The client selects the type of risk minimization strategy as a function of the risk needs of its application. When the IP calculates the price, it applies a fixed overprice of 50% to the NRMin SLAs and 100% to the GRMin SLAs. In addition to the overprice, that determines the MR value of Equation 1, the risk level also determines MP, MPT and MRT. Lower values for MP, MPT and MRT imply that there is less tolerance to failures for low-risk SLAs (see Table 1), because vt will reach MRT sooner (see Figure 1). These fixed values, as well as the other constants that the simulation relies on, are not intended to reflect real market data but to evaluate the model in terms of relative results and tendencies.

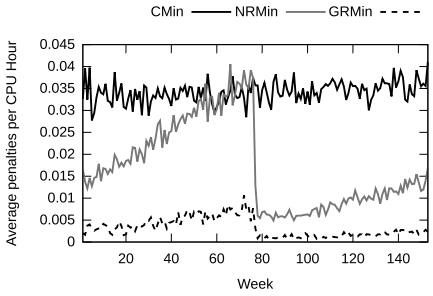
Evaluating risk minimization policies. The graphics of this section show weekly average values to make them more clear and understandable, because

hourly or daily averages are highly influenced by the workload oscillations. The weekly granularity for the values is also accurate enough because the simulation is long-term enough (36 months) to show clearly the tendencies of the metrics used to evaluate the effectiveness of the policies.



(a) Average age of resources

(b) Average PoF



(c) Average violation percentage

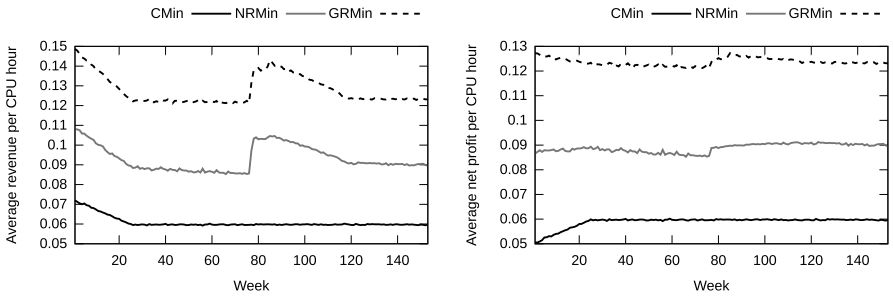
Fig. 4: Evaluation of risk metrics for different SLA policies

Figure 4a shows the behaviour of the policies with respect to the age of the selected resources. In the first half of the experiment all the resources are the same age. When a new bunch of resources is introduced at month 18 (week 77), the CMin policy still selects the older resources, which have the highest amortization rates. NRMin and GRMin progressively move their workloads to the new resources, after a short period in which new resources have higher risks than older resources (as shown in Figure 3). As explained before, the number of requests linearly increases over time. Around week 115 resources are highly loaded because of the high number of requests, and the provider has less possibility to choose resources for the different risk levels. This will influence the risk of the SLAs, as shown in Figures 4b and 4c.

Figure 4b shows the weekly average PoF of the SLAs, differentiated by the three different allocation policies (CMin, NRMin and GRMin). While CMin is near to constant over time, NRMin keeps much lower risk than CMin while is

noticeably influenced by the load of the resources. The PoF for NRMin SLAs increases linearly over time as the number of application executions also increases because the possibility to choose is reduced. When the number of resources is doubled, the PoF of NRMin is reduced again, while the PoF of CMin is kept constant, because the policy still chooses the older resources. GRMin SLAs are also sensible to the load of resources because the allocation policy is the same as NRMin, but the elimination of the single point of failure causes the system keeping much lower risk rates.

The PoF has a direct impact in the economic penalties as consequence of the violations of the SLAs. Figure 4c shows the strict correlation of the economic penalties with the probability of failure. The economic impact of failures is higher in SLAs allocated with low-risk policies (NRMin and GRMin), because both the prices and the penalties are higher for these SLAs (see Table 1). Figure 4c, as well as the rest of figures with economic information in our evaluation, does not show absolute economic values, but values divided by CPU hours to facilitate the comparison of data from appliances with different size and different time.



(a) Average revenue per CPU hour (b) Average net profit per CPU hour

Fig. 5

Evaluating the modelling of the revenue. Figure 5a shows the average price per CPU hour for the different types of SLAs. During the initial part of the experiment, the price decreases because the resources are being amortized which, applying the pricing model of Equation 5, reduces RP. Figure 5b shows that during this period the profit (the revenue minus the reservation price minus the penalties) for CMin policy can be improved because the market allows higher profit margin: RP decreases but DO and BV may increase in a lower proportion to decrease prices while increasing the profit margin. On the other side, profit for risk minimization policies keeps near constant during this period (despite the price reduction) because the business value (BV) of low-risk SLAs is higher (the users are willing to pay more for this extra QoS). In consequence, the provider has more room to increase its margin profit with risk minimization policies.

Figure 5a shows that, when the resources are doubled at the half of the experiment, prices increase for risk minimization policies because they tend to allocate tasks in newer resources with a lower amortization rate than older resources. CMin policy keeps constant prices because this policy still allocates the services in the older resources.

Although the penalty rate increases for minimization policies in the first half of the experiment (from week 1 to week 77), as shown in Figure 4c, the impact of penalties in the net profit is proportionally low because of the different scaling of Figures 5b and 4c. For this reason, the profit of NRMin and GRMin policies slightly decreases over time. However, when new resources are added, the incurred price increment and the reduction of SLA penalties have some positive impact in the net profit. The influence of penalties in CMin SLAs remains quite stable during the whole simulation.

7 Conclusions and future work

This paper introduces a model to differentiate SLAs as a function of multiple risk levels and adapt the provisioning of resources to multiple client profiles in a Cloud Market. We introduce three policies for each differentiated risk profile that we present in the paper: cost minimization, risk minimization at individual node level, and risk minimization at graph level. In addition, we introduce an accounting model that allows the provider adjusting prices to the risk as a function of the amortization of the resources. Our model also helps adapting prices at the long-term, allowing the provider to estimate how the price of the resources decays over time.

The risk propagation model will be improved in the future with bidirectional dependencies and allowing cyclic graphs. The node-level risk analysis will also be improved by exploring alternative methods to the statistical analysis: machine learning, non-linear regressions, etc.

The other main line for future research is related to the automated analysis of graphs. New pattern recognition techniques must be introduced to allow the provider to automatically identify critical points of failure and suggest corrective actions that would minimize the risk only in the required points of the graph to avoid soaring the costs due to the excess of redundancy.

Regarding revenue modelling, we will work on techniques to discover the Business Value of resources. In other words, it is important that the provider estimates accurately how clients are willing to pay the additional QoS to maximize the profit of the provider without losing clients.

Acknowledgement

This work is supported by the Ministry of Science and Technology of Spain and the European Union (FEDER funds) under contract TIN2012-34557 and by the Generalitat de Catalunya under contract 2009-SGR-980.

References

1. Becker, M., Borrisov, N., Deora, V., Rana, O.F., Neumann, D.: Using k-pricing for penalty calculation in grid market. In: 41st Hawaii International International Conference on Systems Science (HICSS-41 2008). pp. 97–106. IEEE Computer Society, Waikoloa, Big Island, HI, USA (January 2008)
2. Djemame, K., Padgett, J., Gourlay, I., Armstrong, D.: Brokering of risk-aware service level agreements in grids. Concurr. Comput. : Pract. Exper. 23(13), 1558–1582 (Sep 2011), http://dx.doi.org/10.1002/cpe.1721
3. Guitart, J., Macias, M., Djemame, K., Kirkham, T., Jiang, M., Armstrong, D.: Risk-driven proactive fault-tolerant operation of iaas providers. In: 5th IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2013). Bristol, Uk (December 2013)
4. Jensen, F.V.: An introduction to Bayesian networks, vol. 210. UCL press (1996)
5. Leone, F., Nelson, L., Nottingham, R.: The folded normal distribution. Technometrics 3(4), 543–550 (1961)
6. Li, B., Gillam, L.: Risk informed computer economics. In: Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid. pp. 526–531. CCGRID '09, IEEE Computer Society, Washington, DC, USA (2009), http://dx.doi.org/10.1109/CCGRID.2009.18
7. Macias, M., Guitart, J.: Client classification policies for SLA negotiation and allocation in shared cloud datacenters. Lecture Notes on Computer Sciences (LNCS) 7150, 90–104 (December 2011)
8. Macias, M., Guitart, J.: Client classification policies for SLA enforcement in shared cloud datacenters. In: 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. pp. 156–163. Ottawa, Canada (May 2012)
9. Metsch, T., Edmonds, A.: Open Cloud Computing Interface - Infrastructure. Tech. Rep. GFD-P-R.184, Open Grid Forum (2011)
10. Neumann, D., Stoesser, J., Anandasivam, A., Borissov, N.: SORMA - building an open grid market for grid resource allocation. In: 4th international conference on Grid economics and business models (GECON'07). pp. 194–200. Springer-Verlag, Berlin, Heidelberg (2007)
11. Risk-aware cloud market simulator. Online, https://github.com/mariomac/riskCloud
12. Sawade, C., Landwehr, N., Bickel, S., Scheffer, T.: Active risk estimation. In: Proceedings of the 27th International Conference on Machine Learning (ICML10). pp. 951–958. Omnipress, Haifa, Israel (June 2010)
13. Schroeder, B., Gibson, G.A.: A large-scale study of failures in high-performance computing systems. In: Proceedings of the International Conference on Dependable Systems and Networks. pp. 249–258. DSN '06, IEEE Computer Society, Washington, DC, USA (2006), http://dx.doi.org/10.1109/DSN.2006.5
14. Simao, J., Veiga, L.: Flexible slas in the cloud with a partial utility-driven scheduling architecture. In: Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on. vol. 1, pp. 274–281 (Dec 2013)
15. Wee, S.: Debunking real-time pricing in cloud computing. In: 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing 2011. pp. 585 –590 (may 2011)
16. Yeo, C.S., Buyya, R.: Integrated Risk Analysis for a Commercial Computing Service in Utility Computing. Journal of Grid Computing 7(1), 1–24 (Mar 2009)