The Computer Journal Advance Access published July 17, 2015

doi:10.1093/comjnl/bxv053

An Autonomous Time-Dependent SLA Negotiation Strategy for Cloud Computing

Amir Vahid Dastjerdi* and Rajkumar Buyya

Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Melbourne, Australia
**Corresponding author: amir.vahid@unimelb.edu.au*

Cloud service level agreement negotiation is a process of joint decision-making between cloud clients and providers to resolve their conflicting objectives. With the advances of cloud technology, operations such as discovery, scaling, monitoring and decommissioning are accomplished automatically. Therefore, negotiation between cloud clients and providers can be a bottleneck if it is carried out manually. Our objective is to propose a state-of-the-art solution to automate the negotiation process for cloud environments and specifically infrastructure as a service category. The proposed negotiation strategy is based on a time-dependent tactic. For cloud providers, the strategy uniquely considers utilization of resources when generating new offers and automatically adjusts the tactic's parameters to concede more on the price of less utilized resources. In addition, while the previous negotiation strategies in literature trust offered quality of service values regardless of their dependability, our proposed strategy is capable of assessing reliability of offers received from cloud providers. Furthermore, to find the right configuration of the time-dependent tactic in cloud computing environments, we investigate the effect of modifying parameters such as initial offer value and deadline on negotiation outputs that include ratio of deals made, and inequality index. The proposed negotiation strategy is tested with different workloads and in diverse market conditions to show how the time-dependent tactic's settings can dynamically adapt to help cloud providers increase their profits.

Keywords: cloud computing; service level agreement; web service; service computing

Received 11 August 2014; revised 3 April 2015
Handling editor: Iain Stewart

1. INTRODUCTION

Cloud computing has transferred the delivery of IT services to a new level that brings the comfort of traditional utilities such as water and electricity to its users. The advantages of cloud computing platforms, such as cost effectiveness, scalability, and ease of management, encourage more and more companies and service providers to adopt cloud computing platforms and offer their solutions via cloud computing models. According to a recent survey of IT decision makers of large companies, 68% of the respondents expect that, more than 50% of their company IT services will be migrated to cloud platforms [1].

Service deployment in clouds can be considered as a process consisting of multiple phases [2, 3]. As depicted in Fig. 1 we consider five major phases for a cloud service deployment. During the Service Discovery phase, user requirements are used as input for discovery of the best suited cloud services among various repositories of cloud providers. In the Service Level Agreement Negotiation (SLAN) phase, discovered providers and the user negotiate on the quality of services. Finally, an SLA contract will be achieved if two parties reach an agreement on a set of quality of service (QoS) values. Then, the acquired service will be continuously monitored in the monitoring phase. If the monitoring service detects that predefined thresholds are reached, services are scaled dynamically in the scaling phase. Finally, in the decommissioning phase, last minute operations are carried out before the service is terminated. With the advances of cloud technology, operations such as discovery [4, 5], scaling [6, 7], monitoring [8, 9] and decommissioning are accomplished automatically [10]. Therefore, negotiations between cloud services clients and providers

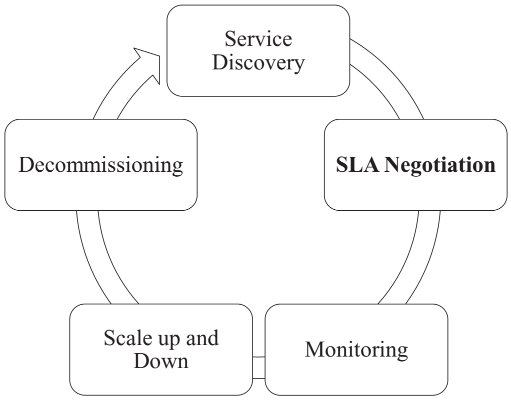

FIGURE 1. The service deployment life cycle.

can be a bottleneck if they are carried out manually. Hence, the objective of this work is to propose a solution that automates the negotiation process in cloud computing (specifically infrastructure as a service) environments.

Cloud SLAN is a process of joint decision-making between cloud users and providers to resolve their conflicting objectives. Cloud services have cost, availability, and other non-functional properties on one hand and generate profits on the other hand. In cloud environments, both clients and providers have cost–benefit models for negotiation and decision-making. Therefore, SLA negotiation automation requires mapping of the knowledge and objectives of policy makers to lower level decision-making techniques. The first step towards the automation is finding, capturing, and modeling goals and objectives of parties involved in the negotiation. The second step is finding a proper strategy to use the goals in the low-level negotiation process.

Automated SLAN has attracted a great deal of interest in the context of Service Oriented Architecture (SOA), grid computing and recently cloud computing. Studies in these contexts mainly focused on offering negotiation strategies that maximize the user's utility values and the number of signed contracts. However, they have not considered infrastructure management issues in the bargaining strategy. It means that cloud providers are willing to concede on the price of resources which are less utilized, and that has to be reflected in the negotiation tactics. In addition, previous works have not considered reliability in the negotiation process. These researches assume that service requestors would trust whatever QoS criteria values providers offer in the process of negotiation. Nevertheless, providers may offer a QoS value during the negotiation that was not fully achieved according to the monitored QoS data.

To address these challenges, *we propose* a negotiation strategy that acquires user's preferences and provider's resource utilization status and utilizes time-dependent tactic along with theory of statistics to maximize the cloud providers profit while adhering to deadline constraints of users and verifying providers offer reliability.

This paper is a significant extension of our previous work [11]. The new contributions reported in this paper are:

(i) We investigate the effect of modifying parameters of the time-dependent tactic such as initial offer value and deadline on negotiation outputs including social welfare (which is measured based on inequality index) and success of negotiation in cloud environments.
(ii) In addition, we investigate how the proposed negotiation strategy reacts to different market conditions (demand to supply ratio (DSR)) to increase the profitability of the negotiation strategy. The effect of the offered strategy on the utility of the whole system is investigated in that regards.
(iii) The experiments for this paper are completely revised and redesigned to cover the aforementioned novel aspects.

The rest of the paper is organized as follows: the next section narrows the scope of the paper while Section 3 highlights challenges in SLA negotiation in cloud which have not been given enough attention. Next, Section 4 aims at highlighting the uniqueness of the proposed negotiation framework by comparing its characteristics with related work. Then, while Section 5 provides a high-level description of the negotiation framework for the readers, Section 6 covers the detail of negotiation strategies for both cloud providers and users. In Section 7, the negotiation strategies are tested to evaluate their effectiveness, and applicability in diverse market conditions. Finally, the work is concluded with suggestions on future directions and conclusions.

2. SCOPE AND ASSUMPTIONS

In this work, the goal of users is acquiring virtual machine (VM) from an Infrastructure as a service provider via negotiation of the following parameters:

(i) Hard disk (functional requirement and fixed)
(ii) CPU (functional requirement and fixed)
(iii) RAM (functional requirement and fixed)
(iv) Cost (non-functional requirement and negotiable)
(v) Availability (QoS requirement and negotiable)
(vi) Deadline (non-functional requirement and fixed)

Users and cloud providers would like to maximize their utilities. Here, we assume that the utility of a provider may increase if in the SLA contract less service availability is guaranteed and higher price is achieved for the service. Besides, users have time constraints when they are participating in the negotiation. This is because if they do not acquire the required resources by a particular time they are not able to satisfy their end users expectations or reach their business objectives. Furthermore, as we have mentioned in the previous section, a client's negotiation service (NS) measures reliability of offers. We presume that there are sufficiently large number of observations (monitoring results for SLA contracts), which make inferences regarding reliability of offers more accurate.

It is worth mentioning that the major objective of this work is to show how differently cloud providers negotiate when they consider their infrastructure status. We consider data center resource utilization as a key aspect that can impact offers of providers during the negotiation. In Section 7, we elaborate on this and provide a detailed discussion. Our proposed negotiation strategy is capable of handling multiple criteria, However in this work, the majority of our experiments focuses on the price of service as the main negotiation issue. In addition, we introduce two main scenarios to explain experiment results in real life contexts:

(1) In the first scenario, a single third party is responsible for providing an NS for both cloud users and providers. The main objective of the negotiation process is maximizing the number of deals made and being fair to both clients and providers. The third party service consists of clients' negotiation agents and providers' negotiation agents which are the main players in the negotiation process and are the focus of the paper in both scenarios.
(2) In the second scenario, parties have their own NS. In this case, the objective of an individual NS (consists of negotiation agents) is maximizing its utility.

3. MOTIVATIONS

3.1. Offers reliability

Since in parallel negotiation, a party makes a decision based on the presented QoS values in SLA offers, there has to be a way to know how reliable the provider is in delivering those promised QoS values. The recorded data from monitoring services can be analyzed and converted to reliability information of offers. The monitoring is based on the copy of the signed SLA, which is kept in the SLA repository. To make inference from the observed data we use the theory of statistics (beta density function), which will be explained in Section 6.

3.2. Maximizing utility of providers and users by adaptively react to changes in data center resource utilization and market conditions

If we consider a scenario where multiple providers, with differences in resource utilization, negotiate with users, with differences in resource and QoS requirement, providers might be interested in an SLA negotiation strategy that adaptively reacts to changes in data center resource utilization and market conditions. To achieve that, NSes can concede more on the price of resources that are less utilized (or have more free capacity) and less on the price of resources that are more utilized. Consequently, providers can offer more attractive prices in earlier stages of negotiation for clients whose requested VMs allocate less utilized resources. This has number of advantages: The clients would be attracted to less utilized resources; users pay less for required resources; and finally it creates an effective load balancing across providers that maximizes the utility of the whole system. For example, consider a scenario where a user needs number of VMs for memory-intensive applications, and a there are two data centers in the system one of which has considerably higher RAM capacity available and the other one has limited RAM available. Hence, the NS of the first data center (based on resource utilization) gives more discount to the user compared with other data center, therefore it is more likely that user reaches an agreement with the first data center. Consequently, load is redirected to less utilized data center and uses acquire resources with less price, which in theory should maximize the whole system utility.

Moreover, this strategy can work for SaaS or PaaS services that consume multiple types of resources with limited supplies. In other words, if there exist a range of services and different quantities of resources allocated to each service type, then this negotiation strategy can be effectively utilized. We provide more details on the aforementioned advantages in Section 7.

3.3. Investigating behavior of the time-dependent function in the cloud computing context

Time-dependent tactics [12] are proper candidates to be adopted for cloud computing environments as users have deadline for acquiring resources when they are participating in a negotiation. However, with the best of our knowledge, their applicability for the cloud context has not yet been deeply investigated. Therefore, first we are going to create a testbed that allows us to implement time-dependent functions for an environment consisting of multiple clouds and brokers which are negotiating on behalf of users, and then we modify negotiation parameters such as deadline of requests and initial offer values and type of tactic (polynomial or exponential) to study the behavior of time-dependent tactics for our problem.

4. RELATED WORK

SLAN has been investigated in details by researchers in the context of SOA and grid computing (either for a single service or for a service composition). Figure 2 shows the main categories of approaches for SLAN. In this section, first we briefly describe the common negotiation tactics. After that, we review SLA negotiation strategies used in cloud environments.

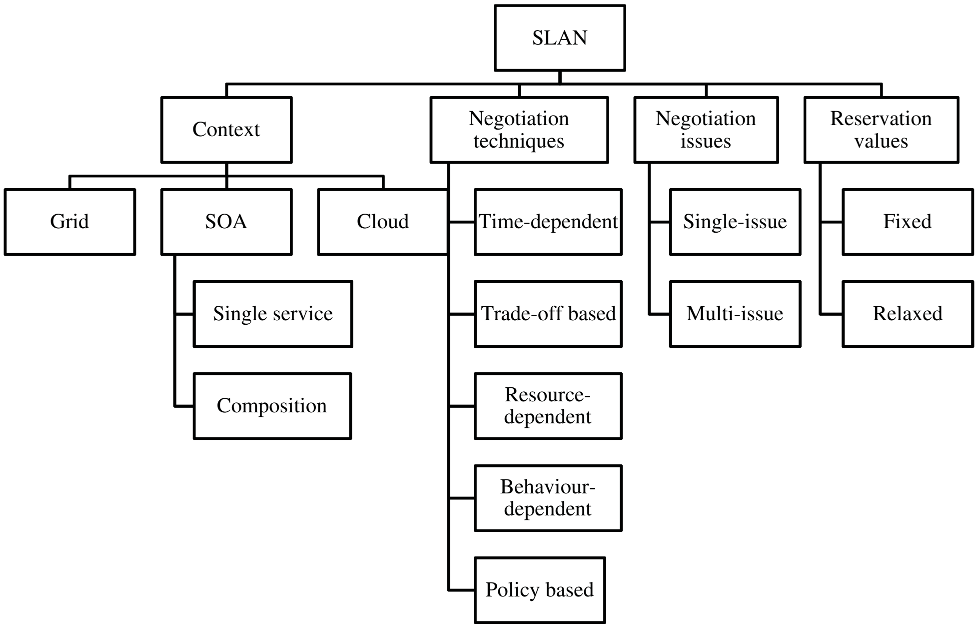

FIGURE 2. Negotiation tactics taxonomy.

4.1. Negotiation tactics

The following families are the most commonly used negotiation techniques in the literature: (1) *time dependent:* if parties have deadline in the negotiation, these techniques are the appropriate choice. This category of techniques concedes faster as the deadline approaches [12]. (2) *Resource-dependent:* this family of negotiation strategies are particularly helpful to reach a consensus, when resource constraints such as the remaining bandwidth (for providers) and the budget (for users) are imposed. This family generates offers and counter-offers based on the resource availability. (3) *Policy-based:* they aim at defining protocols and languages to capture user preferences in the form of policies, and then proposing transformation approaches to map high-level policies to low-level offer values. (4) *Behavior-dependent:* when there is no deadline in a negotiation, agents can adopt this class of techniques. This way, they can imitate behaviors of opponents (e.g. through prediction using regression analysis [13]) to perform at least as well as other parties in the negotiation. Both Axelrod [14] and Faratin [12] have studied this category. The amount of relaxation in this class is decided based on the opportunity function [15]. The opportunity function works out the probability of reaching an agreement based on the number of alternative negotiation parties and the difference between its offers and the received counter offers.

In addition, these tactics differ in number of criteria they deal with (single vs multiple) and number of parties involved in the negotiation process. Coehoorn and Jennings [16] proposed a multi-issue negotiation approach by gathering information regarding opponents' preferences across negotiation issues using kernel density estimation. After obtaining the preferences, the work uses fuzzy similarity [17] to create a counter offer. In addition, Rahwan *et al.* [18] proposed a negotiation tactic which addresses challenges of one-to-many negotiation, where buyer agents coordinates a set of sub-negotiators.

4.2. Cloud and grid computing SLA negotiation

In the context of Grid computing studies [19–21] generally applied the pure resource-dependent techniques or its combination with other techniques for SLA negotiation problem. For example, they concede slower when resources such as the bandwidth are scarce. Comuzzi *et al.* [22] proposed a resource dependent negotiation strategy for pricing of network services. Similar to our approach, the negotiation strategy is time dependent, bilateral, and considers multiple criteria. However, in contrast to our work the time-dependent function parameters are not adjusted automatically. Likewise, Sim [23] investigated market dynamics in their negotiation strategy for Grid computing environments. They considered a heterogeneous e-market where users and resource providers exercise diverse negotiation strategies. Compared with their study, in addition to the market condition, we used resource utilization to offer a negotiation strategy for providers, and reliability to improve the negotiation strategy of users.

Zulkernine and Martin [24] used time-dependent functions for SLA negotiation in the context of cloud and Software-as-a-Service. We argue that our approach is suitable for parallel negotiation. The reason is that we are discriminating regarding the pattern of concession when negotiating concurrently with multiple clients (to increase cloud providers profit) while Zulkernine *et al.* assume the same pattern of concession for all clients.

An SLA management framework for cloud computing environments is proposed by Chhetri *et al.* [25], which uses a policy-based model to support the automated establishment of SLA. The approach uses WS-policy [26] to create a set of rules that can be later queried by cloud users and providers to automatically choose the most appropriate interaction protocol in a given context. Nevertheless, the approach only considers the client side and lacks a negotiation strategy that maximizes providers' profit.

Copil *et al.* [27] proposed a negotiation protocol that involves performance-oriented cloud users and the cloud provider whose objective is saving energy. The process of negotiation determines the amount of overprovisioning a provider is allowed to make. The negotiation strategy employs Particle Swarm Optimization to reach maximum welfare for both negotiation parties. Contrary to our strategy market condition was not considered in their study.

Likewise, the technique that maximizes the objective functions of both cloud providers (Platform as a Service providers) and customers is proposed by Ranaldo and Zimeo [28]. They introduced a new approach for dynamic evaluation of the acceptable region of SLA offers when there exists a non-additive utility function. The effectiveness of the negotiation strategy is verified through queuing-based performance model. Similar to our strategy the proposed strategy takes into account information about providers' capacity. However, their strategy cannot estimate the reliability of providers' offers and is not capable of pricing cloud resources.

To reduce the SLA-gap between cloud providers and users, an SLA negotiation mechanism [29] for cloud environment is modeled using GENIUS [29] which is a generic negotiation platform. Similar to our work, they considered social welfare, social utility and mutual gain metrics. The strategy is a subclass of Tit-for-Tat (behavior-dependent) which aims at predicting opponent behavior. Although the strategy can achieve fair SLAs, it has not considered the market condition and capacity management.

An SLA negotiation mechanism for multi-issue negotiation was proposed by Son and Sim [30]. They considered time slot and price as negotiation criteria. The SLA negotiation is equipped with a technique that selects a proper data center from set of globally distributed locations to minimize the SLA violations. The novelty of the approach is related to the proposed utility function that models preferences for different time slots. The negotiation tactic is based on a trade-off algorithm and improves the negotiation speed and the aggregated utility function. The negotiation techniques results in a pricing approach that once compared with the three pricing models of Amazon EC2, shows support for faster agreements and achieves higher utilities. However, this strategy does not capture reliability of offers.

In summary, our approach has the following combined features that are contributions over previous research works: setting a deadline for negotiation while considering resource utilization and market condition; discriminating regarding the pattern of concession to maximize providers' profit; taking into account reliability metric to discard unreliable offers, and investigating social optimality of time-dependent functions in the negotiation process.

5. NEGOTIATION FRAMEWORK

Figure 3 describes the sequence of interaction between the cloud provider NS and the client NS. First, the service requester specifies hardware specifications like CPU, storage, memory as well as preferences on the QoS criteria. After that, functional and QoS requirements are used as input for discovering suitable cloud services. Afterwards, the client NS starts negotiating with the discovered service providers' NSes on QoS criteria (price and availability) based on the requester's preferences. Client's budget and deadline for acquiring resources are used by the client NS to make a decision on accepting or rejecting an offer. Client NS uses time-dependent tactic that takes the client's preferences as an input and automatically generates an initial and then consequent offers. Once cloud NS receives the offer, it uses request functional and QoS requirements and cloud resources utilization from the monitoring system to generate counter offers. On the arrival of providers' offers, the client NS uses the reliability evaluator component and the time-dependent tactic to accept or reject the offer, or otherwise reply with a counteroffer.

If the negotiation is successful, an SLA contract will be signed by both parties and the obtained contract, which includes a set of expected QoS values (service level objectives (SLOs)), is kept in the SLA contract repository. The SLA will be constantly monitored, and reliability evaluator will be notified in the case of violation of SLOs.

6. NEGOTIATION STRATEGY

Prior to explaining the negotiation strategies for each party, a brief description of the negotiation model and the applied negotiation tactics are given. Descriptions of symbols used for expressing the negotiation process are listed in Table 1.

6.1. Negotiation model

To create a negotiation model, we extended the model proposed by Raiffa [31] to incorporate the reliability of offers. In the model, the NS receives requestor preferences on the importance (W_i) of n negotiation issues, $\min_i$ and $\max_i$ (reservation values), which are the acceptable range of values for issue i (VI_i), and negotiation deadline ($t_{\max}$). The service then measures the utility of offers received from other NS based on

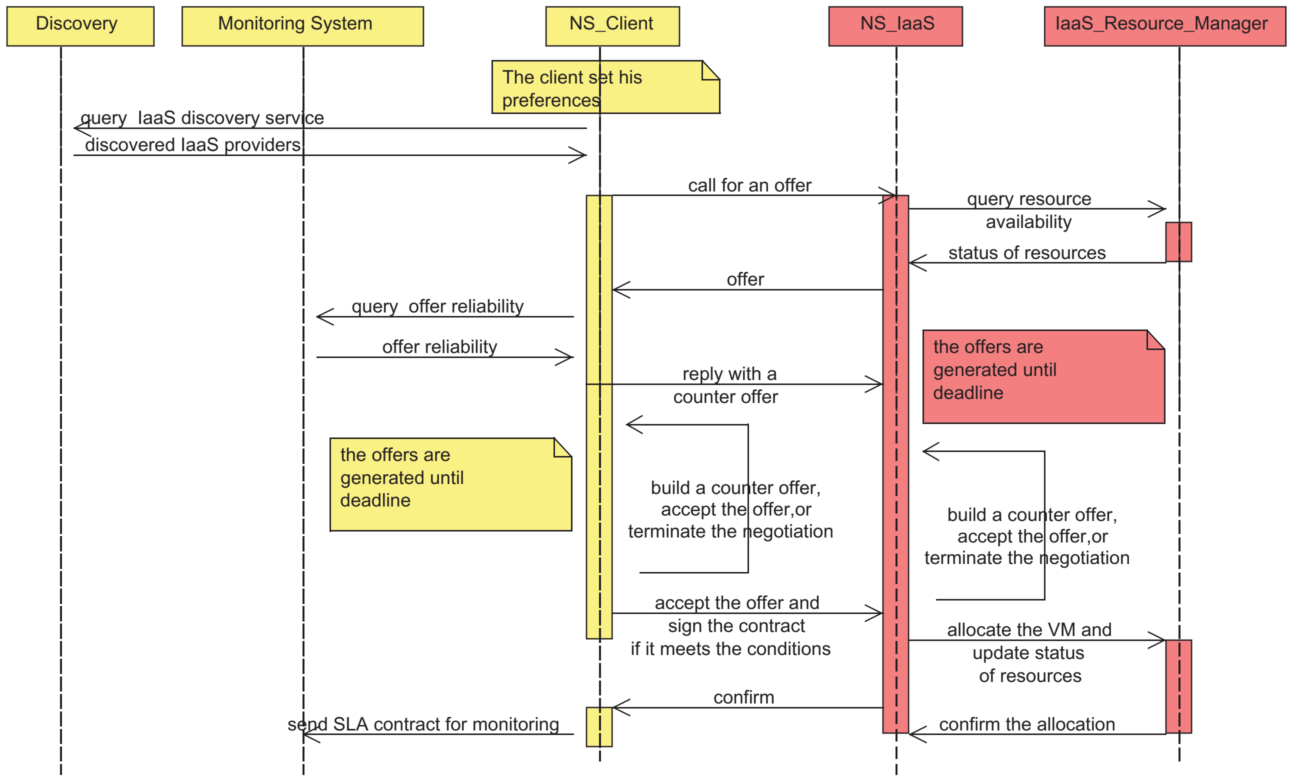

FIGURE 3. Negotiation sequence diagram.

Eqs. (1) and (2).

$$UV = \sum_{i=1}^{n} W_i VI_i(y_i) \quad \text{where} \quad \sum_{i=1}^{n} W_i = 1 \tag{1}$$

$$VI_i(y_i) = \begin{cases} \dfrac{\max_i - y_i}{\max_i - \min_i} & VI_i \text{ increases as } y_i \text{ decreases;} \\ \dfrac{y_i - \min i}{\max_i - \min_i} & VI_i \text{ decreases as } y_i \text{ decreases.} \end{cases} \tag{2}$$

Next, as shown in Eq. (3), the offer is accepted if its utility value is greater than or equal to the utility of the counter offer that will be sent by the NS. Otherwise, the NS generates a new counter offer. In addition, if the timestamp of the received offer (t_{offer}) is greater than the deadline the service terminates the negotiation.

$$\text{Response} = \begin{cases} \textit{terminate if} \quad t_{\text{offer}} > t_{\max}; \\ \textit{accept if} \quad UV_{\text{offer}} > UV_{\textit{counter offer}}; \\ \textit{new counter offer otherwise}. \end{cases} \tag{3}$$

6.2. Time-dependent negotiation tactic

As cited by Faratin [12], time-dependent negotiation tactics are a class of functions that compute the value of a negotiation issue by considering the time factor. Therefore, they are particularly helpful when the NS receives a deadline ($t_{\max}$) as an input, and has to concede faster as the deadline approaches. For this family of tactics, Eq. (4) is used by NS ‘a’, which represents either a cloud service requestor or a provider to generate a new counter offer for NS ‘b’ for negotiable issue i.

$$O^t_{a\rightarrow b}[i] = \begin{cases} \min_i^a + \alpha_i^a(t)(\max_i^a - \min_i^a) \\ \quad \text{if} \quad V_i^a \text{ is decreasing;} \\ \min_i^a + (1 - \alpha_i^a(t))(\max_i^a - \min_i^a) \\ \quad \text{if} \quad V_i^a \text{ is increasing.} \end{cases} \tag{4}$$

Numerous functions have been defined for calculation of α_i^a(t) such as polynomial and exponential [12]. As it can be figured out from Eq. (5), by changing the value of β (convexity degree) in both functions, the behavior of the negotiation tactic changes. If $\beta > 1$, the tactic reaches its reservation's value at the early stage of negotiation. On the contrary, in the case of $\beta < 1$, it concedes to its reservation value only when the deadline is approaching. In addition, K is determining the initial offer value. We adopt this family of the negotiation functions and change β and K dynamically to maximize the NS

TABLE 1. Description of symbols.

Symbols	Description
a,b	negotiation parties
W_i	importance of issue i
VI_i	offer value for issue i
UV	utility value of the offer
t_{offer}	offer timestamp
$t_{\max}$	negotiation deadline
$O^t_{a\to b}$ [i]	offer sent from a to b for issue i
$\min_i^a$	minimum acceptable value of issue i for a
$\max_i^a$	maximum acceptable value of issue i for a
y_i	defines the range of values for an issue i
α_i^a (t)	time-dependent function of issue i for a
V_i^a	value offered for issue i by a
K_i^a	initial offer value for issue i by a
β	convexity degree
P_t	price of virtual machine instance at t
RP_{jt}	price of a resource j (e.g. RAM) at t
αRP_j	time-dependent function for price of resource j
IRP_j	initial price for resource j
A_j	portion of resource j that is available
β_j	convexity degree for price of resource j
RA	resource aware tactic
PO	priority oriented tactic
γ_1	relative importance of RA
γ_2	relative importance of PO
$RC_{offerVI_i}$	reliability constraint for issue i
$R_{offerVI_i}$	reliability of an offer's value of issue i
COD	consensus desirability
CF	conceding factor
ρ, τ	beta distribution parameters

utility function

$$\alpha_i^a(t) = \begin{cases} k_i^a + (1-k_i^a)\left(\dfrac{\min(t, t_{\max})}{t_{\max}}\right)^{1/\beta} & \textit{Polynomial}; \\ e^{(1-\min(t,t_{\max})/t_{\max})^{\beta}\ln k_i^a} & \textit{Exponential}. \end{cases} \tag{5}$$

6.3. Providers strategy

For providers, the NS input is composed of the cloud resource utilization, minimum and maximum resource prices, and amounts of requested resources. The output of NS can be an SLA contract with a detailed description of a provider, a client, a service and SLOs. Providers are interested in an SLA negotiation strategy that gives the attractive offers while keeps their utility functions high. If providers concede more (by adjusting time-dependent function parameters) on the price of the resources that are less utilized (or have more free capacity) and less on the resources that are more utilized, the utility of whole system increases. This is because this strategy can resemble a load balancer that distributes user requests to the least-expensive resources that are offered by the least utilized providers [32]. In addition, as we show in Section 7, it can improve the utilization of data centers.

Unlike the majority of works that require time-dependent function parameters to be given explicitly, Zulkernine and Martin [24] proposed a method to derive the parameters from the high-level negotiation policy. Inspired by their work, we propose an approach to derive a price for the next offer based on the cloud resource utilization. In comparison with their work, we argue that our approach is more suitable for the cloud context as we consider the resource management in the negotiation. As shown in Eqs. (6–10), we first define a total price of a VM instance as the sum of prices of its individual resources Eq. (6). In the next step, for each resource, a time-dependent function (Eqs. (7) and (8)) is defined, and its parameters are adjusted (Eqs. (9) and (10)) based on its underutilized capacity compared with average resources' idle capacity ($\bar{A}$) for m type of resources.

$$P_t = \sum_{j=1}^{m} RP_{jt} \tag{6}$$

$$RP_{jt} = \text{Min}\, RP_j + \alpha RP_j(\text{Max}\, RP_j - \text{Min}\, RP_j) \tag{7}$$

$$\alpha RP_j = IRP_j + (1 - IRP_j)\left(\frac{\min(t, t_{\max})}{t_{\max}}\right)^{1/\beta_j} \tag{8}$$

$$\bar{A} = \frac{\sum_{j=1}^{m} A_j}{m} \tag{9}$$

$$\begin{gathered} \beta_j = CF \times e^{C(A_j - \bar{A})} \\ \text{where } CF = \omega \times COD \text{ and,} \\ \omega \text{ and } C \text{ are constants and } c, \omega > 0. \end{gathered} \tag{10}$$

As shown in Eq. (10), when the idle capacity of a resource is greater than the average free capacity of resources in the data center, $A_j - \bar{A} > 0$ and $\beta_j > 1$, and therefore the negotiation strategy is conceding on the price of that resource. As a result, providers offer a more attractive price in earlier stages of negotiation for clients whose requested VMs' allocations would balance resource utilization. This increases the chance of reaching an agreement with the preferred request. However, in this tactic β is calculated based on the resources utilization and does not reflect the preferences of provider regarding the importance of price and guaranteed availability criteria. The tactic based on [24] is adopted in Eq. (11) to derive β from provider's preferences. When NS deals with multiple criteria, weights that are presented in Eq. (10) are same as the one used in Eq. (1) as they show how important is an issue compare with the others. Consequently, in order to satisfy all providers' objectives, the negotiation strategy has to be built as a mixture of those aforementioned tactics as shown in Eq. (12).

$$\beta_j = \mathrm{e}^{C(1/n - W_i)} \tag{11}$$

where n is the number of criteria in the negotiation, C is a constant and W_i is the importance of issue i and $\sum_{i=1}^{n} W_i = 1$.

$$O^t_{a\to b}[i] = \gamma_1 RA^t_{a\to b}[i] + \gamma_2 PO^t_{a\to b}[i] \tag{12}$$

where $\gamma_1 + \gamma_2 = 1$, $0 \le \gamma_1, \gamma_2 \le 1$, and RA, PO are offers' issue values generated by Resource Utilization Balancing Oriented tactic and Preference Oriented tactic, respectively.

6.3.1. Discussion

When the negotiation scenario consists of multiple criteria and the mixture of strategies as shown in Eq. (12) is used, one of the following three conditions holds:

(i) $\beta > 1$ for RA, that means the provider is interested in the client's offer, and $\beta < 1$ for the issue of price in PO, that means the price is the most important issue for the provider. Therefore, it is likely that the provider reaches consensus with clients (with desirable resource demand) who are interested in higher guaranteed availability.

(ii) $\beta > 1$ for RA, that means the provider is interested in the client's offer, and $\beta > 1$ for the issue of price in PO, that means the availability is the most important issue in the negotiation for the provider. Then, it is likely that the provider reaches consensus with clients (with desirable resource demand) who are less concerned regarding the lower guaranteed availability.

(iii) $\beta < 1$ for RA, that means the provider is not interested in the client's offer, and $\beta < 1$ for the issue of price in PO, that means the price is the most important issue in the negotiation for the provider. Then, it is likely that the provider reaches consensus with clients who are willing to pay considerably higher for the desirable guaranteed availability.

(iv) $\beta < 1$ for RA, that means the provider is not interested in the client's offer, and $\beta > 1$ for the issue of price in PO, that means the availability is the most important issue in the negotiation for the provider. Then, it is likely that the client receives neither an attractive offer on the price nor on the availability issue.

The focus of this work is on cases where providers and users are purely interested in the price factor and the remaining scenarios will be investigated in our future works.

6.3.2. Extending provider strategy to support negotiation for auto-scaling

As mentioned in Section 1, with recent advances in cloud technology, cloud users are capable of scaling their resources capacity up or down automatically based on conditions they define on performance metrics. Auto-scaling is an important feature for both applications that experience steady demand patterns or those have daily or hourly surges in workload.

However, both cloud providers and users have to first reach a consensus on SLA conditions for the required capacity. Our negotiation strategy can be extended via two different methods to support auto-scaling negotiation: (i) on-demand when scaling-up is required and (ii) at deployment time before scaling-up is required.

Negotiation process triggered on-demand when scaling-up is required: for this case, the same proposed negotiation strategy can be employed and negotiation is triggered by user-defined thresholds on performance metrics (e.g. CPU utilization). However, this approach suffers from two drawbacks. First, the time required to reach consensus may affect the SLA of a business services which are deployed on the cloud resources. In addition, there may not be enough resources available in the market when scaling up is required. In this condition, cloud users either have to pay considerably higher price for the resources or may not be able to acquire the capacity they need.

Negotiation process triggered at deployment time before scaling is required: to address the challenges associated with the first approach, the negotiation process can be started at deployment time. This means that clients negotiate for a reserved capacity that later can be utilized when scaling up is needed. To support this, our proposed negotiation strategy is required to be extended. *First*, the provider strategy has to consider a separate resource pool that is built to reserve resources for auto-scaling purposes. *Secondly*, two new negotiation criteria along with the one described in Section 2 have to be considered by negotiation strategies:

(1) *Upfront payment:* amount to be paid by clients for reserving a capacity in the provider data center for auto-scaling.

(2) *Utilization pattern:* it defines how the reserved capacity is utilized (for example, 30% of the time).

Thirdly, the provider strategy (which was a mixture of a time-dependent and resource-dependent) has to be further updated to additionally accommodate a trade-off tactic (as described in Section 4) that balances the amount of upfront payment and effective hourly price of VM based on the utilization pattern to both attract more customers and maximize the cloud provider's profit.[1] This has two advantages: through negotiation, clients can minimize their expenses once they gain knowledge on their utilization pattern, and providers can benefit from the utilization pattern information supplied by clients to perform and efficient capacity planning.

Although this approach solves the challenges of the first extension, it is more complex as it requires a decision to be made on how to partition resources between on-demand and reserved pools.

[1] For example in Amazon EC2 for light-utilization reserved instance type, clients pay less upfront and pay more hourly. http://aws.amazon.com/ec2/purchasing-options/reserved-instances/

6.4. Cloud client NS

The client NS receives user preferences on budget, deadline and QoS criteria importance and maps them to low-level time-dependent parameters as described in the previous section and based on Eq. (11) [24]. It means that β is defined in a way that the NS concedes less if the criteria are more important to the user and concedes more otherwise. To capture the importance of the criteria for the user, analytic hierarchy process [33] is adopted. Similar to the provider NS, the output can be an SLA contract with full specification of services, provider, client and SLOs. In this strategy, our contribution lies in the probabilistic assessment of offers reliability in negotiation.

The client NS assesses providers offers' in a probabilistic approach based on their past adherence level to SLA contracts. Therefore, as shown in Eq. (13), the client NS only accepts offers when similar previous accepted offers have achieved certain level of reliability (based on the monitored data) for each issue. For example, if in a multi-criteria negotiation a provider concedes in availability, and its reliability in such criteria is not high, users should not consider that as an attractive offer.

$$\text{Offer acceptance conditions} = \begin{cases} UV_{\text{offer}} > UV_{\text{counter offer}} & \text{and} \\ \text{for each } VI_i & R_{\text{offer } VI_i} > RC_{\text{offer } VI_i}. \end{cases} \tag{13}$$

We used the β reputation system [34] to assess the reliability of offers. The reason is that monitoring outcome (MO) of a particular SLA contract SLO can be modeled as in Eq. (14), and therefore is a binary event. For example, a cloud service can be either available for more that 80% of time in a year or not. Therefore, for each contract and for the availability criterion (which is our focus here) you can brake down your monitoring results to binary events. The MO is measured per specific Service Level Objectives. It is important to mention that the objective is not to measure the reliability of a provider, but to measure the reliability of an offer by counting the number of time an SLO was achieved for all similar contracts. Consequently, the beta density function, which is shown in Eq. (15), can be efficiently used to calculate posteriori probabilities of the event. The higher the number of monitoring observations the higher the accuracy of offers reliability. The mean or expected value of the distribution can be represented by Eq. (16).

$$MO = \{\text{SLO not achieved, SLO achieved}\} \tag{14}$$

$$f(x \mid \rho, \tau) = \frac{\Gamma(\rho+\tau)}{\Gamma(\rho)\Gamma(\tau)} x^{\rho-1}(1-x)^{\tau-1}$$
$$\text{where} \quad 0 \le x \le 1, \rho > 0, \tau > 0 \tag{15}$$

$$\mu = E(x) = \rho/(\rho+\tau) \tag{16}$$

As mentioned in Section 5, in our architecture a component is responsible for monitoring SLA contracts. If we assume that the monitoring component has detected that SLA violation occurred v times for provider p (for a total number of n monitored SLAs). Considering that $\rho = n - v + 1$ and $\tau = v + 1$, the reliability is equal to probability expectation of SLA is not going to be violated and is calculated as shown in Eq. (17). Once $R_{\text{offer } VI_i}$ is calculated for all issues, NS can only accept the offer if for all the issues $R_{\text{offer } VI_i}$ is greater than $RC_{\text{offer } VI_i}$.

$$R_{\text{offer } VI_i} = \frac{n-v+1}{n+2} \tag{17}$$

In our study, we consider availability as a criteria that is part of the SLA and can be measured by the third party monitoring services or in the case of IaaS, that is our focus, even by a client-owned monitoring service. In addition, negotiating for an SLA that consists of QoS criteria that cannot be monitored is not realistic. Hence, if cloud providers believe that offering services under certain SLA can attract more customers, they need to facilitate the monitoring data collection. Currently, monitoring of existing valid SLA for metrics such as availability is not a challenge, however, the problem arises when clients have no history of SLA monitoring data and have not used the service before. In this case, the provider has to provide the SLA monitoring data with the motivation to attract more customers by being transparent and loyal to the agreement. Nevertheless, if no historical data are available, then approaches such as sentiment analysis on social media data [35] and consensus-based reliability analysis [36] can be used to estimate the reliability of the offer.

6.5. Discussion

Techniques based on the game theory assume that all agents are aware of the possible strategies of their opponents. Therefore, they are the best fit for cooperative problem-solving negotiation scenarios such as the one where a third party NS is responsible to maximize the utility of both users and providers. Conversely, our proposed negotiation strategy does not require prior knowledge of an opponent's strategy to operate. However, this does not mean that it is entirely immune to malicious negotiation attempts. Here we investigate two malicious negotiation scenarios:

(1) A malicious client submits arbitrary offers to gain the knowledge regarding utilization of resources and preferences of a cloud provider. Then, the client can concede faster in the particular negotiation criteria that is more important for the provider and thus allocate the cloud service earlier (but only if the utilization of resources and provider preferences do not vary frequently). This may not decrease the utility providers as the client ultimately has to pay for what they acquire. However, the client may be able to win the competition in the market if there is not enough resources to satisfy demands of all the clients.

(2) In another scenario, a provider can act as a malicious client to consume scarce resources of other providers

to stop them serving their client efficiently. However, if all providers start following the same approach, then the utility of the whole system decreases, not only of a particular provider.

7. PERFORMANCE EVALUATION

For our performance evaluation, we extended CloudSim, a discrete event cloud simulator [37], to build a new environment for testing negotiation techniques for cloud computing environment. The inter-arrival time of requests does not affect the performance of the negotiation strategies. Therefore, it is simply considered as a uniform distributed value between 0.0 and 1.0 s. The simulation period is 1 h and when the DSR (Eq. (18)) in the experiment is <1, the data centers capacity is set to 100 000 Hosts. When DSR is >1, data center capacity is set to 10 000 Hosts. Each Host has 12 CPU cores, each 1.7 GHz; 12 GB RAM and disk capacity of 4 TB. Reservation values for clients are set to \$5 per resource unit for the min price and \$15 for the maximum. For clouds, however, reservation values are \$10 and \$20 per resource unit for minimum and maximum price, respectively. In our work, we have not considered different payment plans as the price of services have no impact on our proposed negotiation strategy. Using the request generator class, brokers in CloudSim (which represent clients) send requests simultaneously to data centers. The request generator randomly generates requests with different deadlines and required instance types (defined as a tuple with three elements as represented by Eq. (19)).

$$DSR = \frac{\textit{total resource requirements}}{\textit{total resource supply}} \tag{18}$$

$$Instance = (N_{CU}, N_{RU}, N_{HDU}) \tag{19}$$

where N_{CU} is the number of CPU units requested, N_{RU} is the number of RAM units requested and N_{HDU} is the number of hard disk units requested. If we assume that providers offer same instance types as Amazon EC2 does, then requests generated for experiments can be classified into two classes, namely balanced and unbalanced. In a balanced request, $N_{CU} = N_{RU} = N_{HDU}$, while in an unbalanced requests $N_{CU} \neq N_{RU} \neq N_{HDU}$. Requests for our experiments have been designed according to Amazon EC2 instances types [38]. Examples of unbalanced instances are the ones from memory, storage or compute optimized family offered by EC2. An example of balanced instance type is the small instance type from the general purpose family. In addition, they can be further categorized to requests with tight (from 20 to 40 rounds), moderate (from 40 to 50 rounds) and loose deadline (from 50 to 100 rounds). These categories of deadlines are created to investigate the impacts of deadlines on negotiation outcomes as explained in Section 7.2. For example, when we mention that from 20 to 40 rounds are categorized as a tight deadline, we mean that NSes have comparatively less time to reach consensus and this has impacts on the negotiation outcome. We generate the workload by assigning a probability to each category of requests. For example, to generate a workload to investigate the effects of tight deadline, we can set the probability of a request having tight deadline to 0.8. The experiments described in the following subsections are repeated 30 times.

In addition, we introduce two main scenarios to explain experiment results in real life contexts:

(1) In the first scenario, a single third party system is responsible for providing an NS for both cloud users and providers. The main objective of the negotiation process can be maximizing the number of deals made or being fair to both clients and providers.
(2) In the second scenario, parties have their own negotiation systems. In this case the objective of each individual negotiation system is maximizing its utility.

The conducted experiments investigate:

(i) How modifying deadline of requests, initial offer values and time-dependent function type affect the consensus rate and social welfare (Sections 7.1 and 7.2);
(ii) How successful the proposed strategy for cloud NS is in increasing cloud providers' profits, which is calculated based on the number of VM allocated and the achieved price in the negotiation (Sections 7.3); and
(iii) How to react to different market conditions to increase the profitability of negotiation strategy (Section 7.4).

7.1. Effect of strategies and negotiation parameters on negotiation outcome

The designed negotiation settings consists of one broker (which represent clients) and one data center (cardinality of negotiation for cloud service providers and clients is one-to-one), with negotiation parameters (CF and K) equally set for both parties. We consider the first scenario and, as shown in Eq. (20), we use the inequality index (II) to test the fairness of negotiation strategies. We have used Eq. (1) to calculate the utility value of each party to measure II. The closer the values of II gets to '0' the higher the fairness of the strategy. Not surprisingly, when lower values are given to Consensus Factor and initial offer, the ratio of successful negotiation decreases (Fig. 4b). In contrast, higher values for CF and initial offer increase the chance of reaching an agreement (Figs 4d, f, 5d and f). However, when they are set to extreme values (as shown by Fig. 4e when the K factor reaches 0.8), the offers received from a data center are accepted in a first round of negotiation and there is no time for a broker to concede. Therefore, the broker has comparatively higher utility value in this case and II increases drastically.

$$II = \left| \frac{UV_{BR}}{UV_{DC} + UV_{BR}} - \frac{UV_{DC}}{UV_{DC} + UV_{BR}} \right| \tag{20}$$

In addition, as illustrated in Fig. 5, when the polynomial function is used, the chance of reaching an agreement, even if

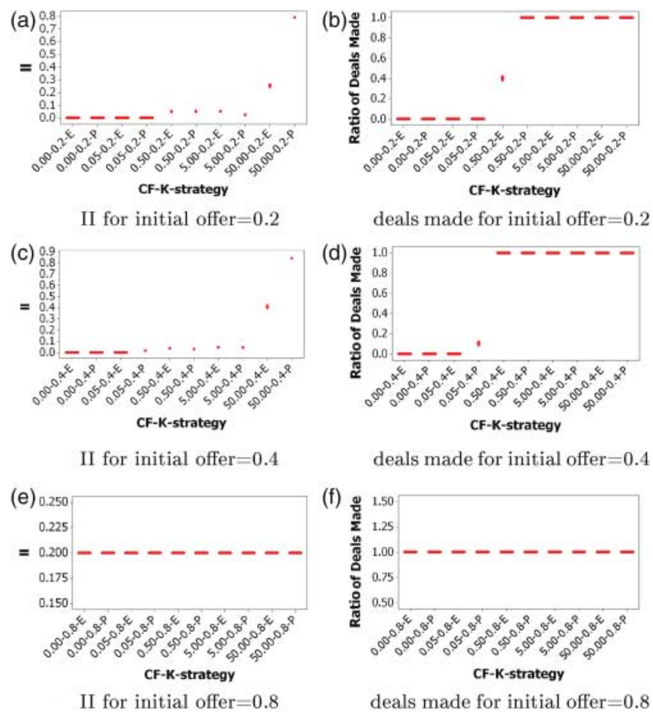

FIGURE 4. Impact of initial offer on II and negotiation success rate. In the dot plots P and E refer to polynomial and exponential tactics. (a) II for initial offer = 0.2, (b) deals made for initial offer = 0.2, (c) II for initial offer = 0.4, (d) deals made for initial offer = 0.4, (e) II for initial offer = 0.8 and (f) deals made for initial offer = 0.8.

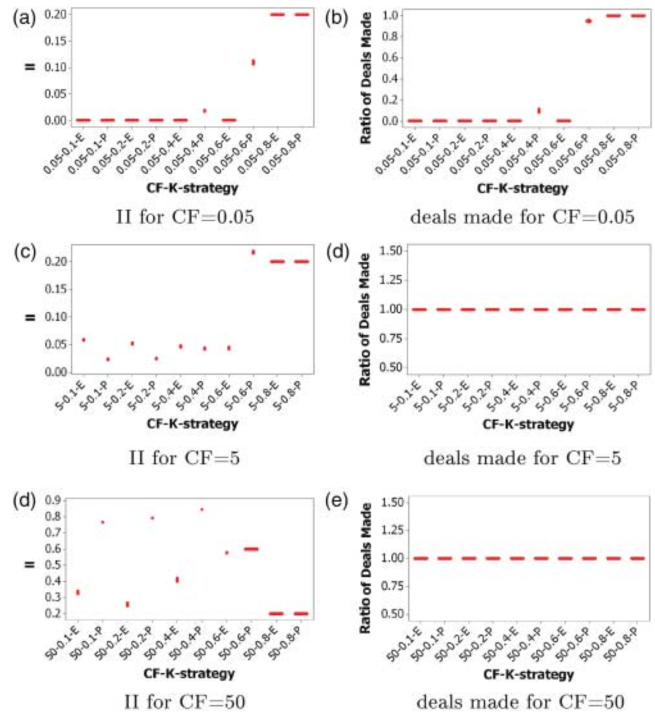

FIGURE 5. Impact of CF on II and negotiation success rate. In the dot plots P and E refer to polynomial and exponential tactics. (a) II for CF = 0.05, (b) deals made for CF = 0.05, (c) II for CF = 5, (d) deals made for CF = 5, (e) II for CF = 50 and (f) deals made for CF = 50.

the initial offer and CF is set to lower values, increases. Nevertheless, as depicted in Fig. 5e, in a majority of the cases, when CF is set to the highest value (50), the exponential function reaches lower II. Moreover, adoption of CF of 5 (and 0.5 although it is not depicted in Fig. 5) or K of 0.4 and 0.2 (as depicted in Figs 4 and 5) results in a lower inequality. This means that if the objective of the negotiation is to achieve higher fairness as described in the first scenario, initial offers should be set at maximum below the half of the overall concession that one party is going to make, and then it should not concede either very quickly or too slowly.

7.2. Impact of change in deadline on the ratio of deals made

In this experiment, one broker (which represent clients) and a data center participate in the first negotiation scenario (cardinality of negotiation for cloud service providers and clients is one-to-one). The request generator builds negotiation messages based on a given deadline-type probability. For the majority of cases illustrated in Fig. 6, when probability of a request having tight deadline increases, the ratio of agreements made decreases until no or a few deals are achieved. However, when CF and K leastwise set to 0.5 and 0.1 for polynomial function and 5 and 0.2 for exponential function, the deadline has no impact on the number of deals made, and both strategies reach 100% of consensus rate. This shows the dominance of the polynomial function in reaching higher number of deals when the deadline is tight. Therefore, for the first scenario, and if users have tight deadlines, it is best for the negotiation system to set its strategy to polynomial to achieve the goal of maximizing the number of deals.

7.3. Performance of the negotiation strategy

This experiment can be considered in the context of the second scenario where a data center owns an NS to maximize its utility function. However, we will still investigate the impact of our strategy on the whole system (including brokers) utility. The experiment was designed with four brokers and one data center (cardinality of negotiation for cloud service providers and clients is one-to-one). The reason for having four brokers is to increase the chance of having parallel negotiation between data centers and brokers. Therefore, we can examine whether our negotiation strategy can act as an efficient load balancer (refer to Section 3.2). For the data center in this experiment, the profit is calculated based on the number of VMs allocated and the achieved price in the negotiation. All parties adopted the aforementioned polynomial function. The data center first adopted a pure time-dependent function and concurrently negotiated with

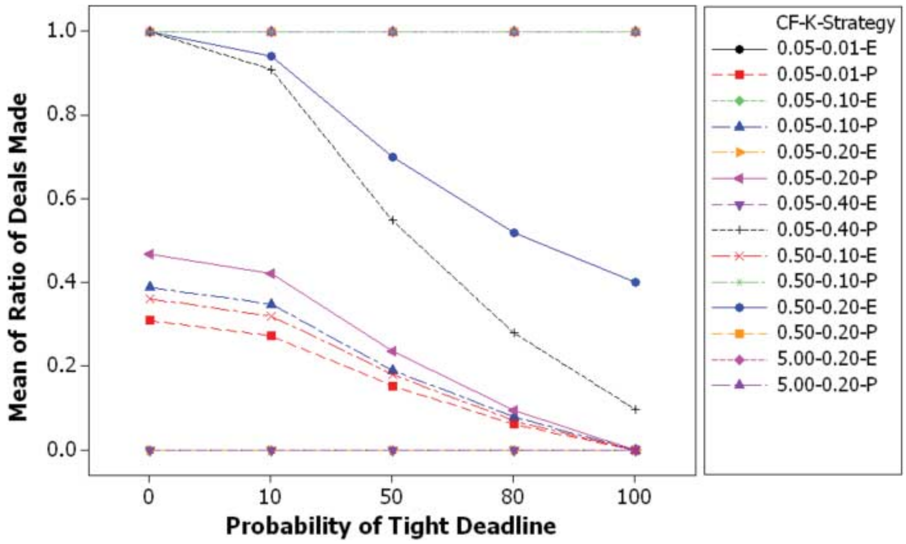

FIGURE 6. Impact of deadline on the success rate of negotiation. By '0.05-0.01-E', we mean CF, K and time-dependent function are set to 0.05, 0.01 and exponential, respectively.

four brokers (which use pure time-dependent function), then we repeated the experiment with the same configuration but this time we replaced the strategy with ours. For this experiment, we generated the workload in such a way that it is more likely to cause resource fragmentation if there is no load balancing in place. Then, we utilize our economy-based negotiation strategy (giving more discount when resources are available) to investigate its effects on achieved profit for data centers. The main objective is to investigate whether we can achieve more effective approach for distributing requests among data centers.

As Fig. 7 shows, when the percentage of unbalanced requests (PUR) increases, the revenue difference between strategies presented in works such as [12, 24] (purely time-dependent Eq. (4)) and our work grows. The results show that for the cases where only a small percentage of incoming requests are unbalanced (20%), data centers can still increase their profits by almost 10% on average. In addition, if the chance of a request to be unbalanced is 50%, then the profit growth increases to 20% on average. For the case that PUR is set to 100, our strategy can dominate previous works' strategies by nearly 27%. This is because, via a economy-based load balancing strategy, we can map requests to data centers in a way that causes less resource fragmentation. For example, consider a request which requires more CPU units and less RAM, it is best to send it to a data center which has more CPU available, compare with the one which has less CPU and more RAM available. This way clients acquire the least-expensive resources and cause less resource fragmentation. If opposite happens, it causes resource fragmentation and then the data center with more RAM is less capable of hosting a new request. The proposed strategy not only increases the revenue of the data center, but also, as demonstrated in Fig. 8, increases the combined utility of the whole system (as mentioned earlier the utility of each party in the negotiation is calculated based on Eq. (1)). This means the strategy increases profit of the data center and the utility of the whole system (including brokers).

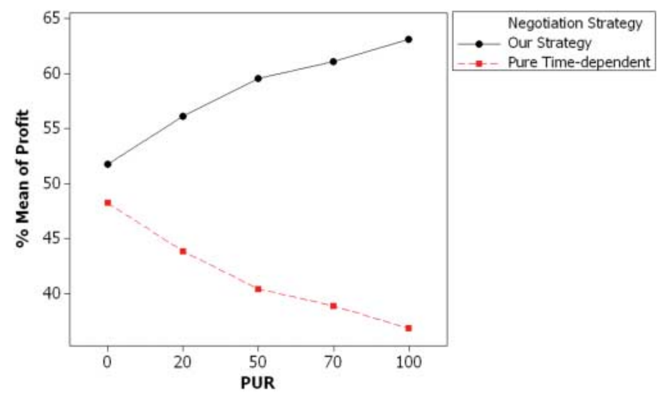

FIGURE 7. Impact of request type on the performance of the strategy. Workloads are built with different percentage of unbalanced requests (PUR).

7.4. Effect of DSR and consensus desirability on data centers revenue

To show how our proposed strategy can increase its competency when another data centers participate in the negotiation, this experiment is designed with four brokers and two data centers (cardinality of negotiation for clients and cloud service providers is one-to-many and clients terminate pending negotiations processes once they reach a consensus with a provider).

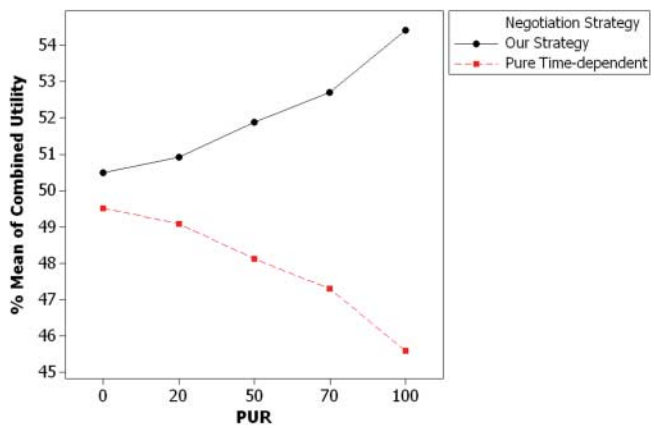

FIGURE 8. Impact of request type on the combined utility of the strategy.

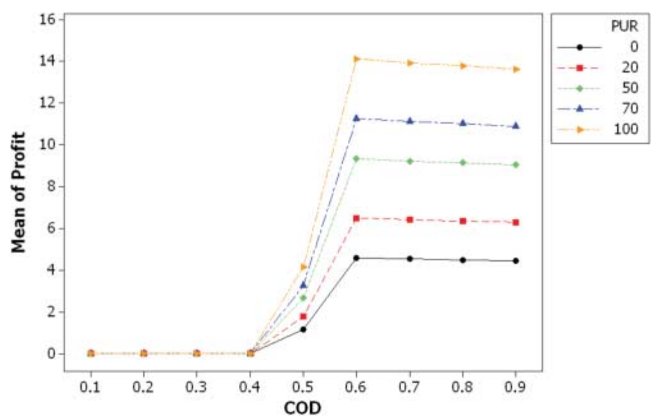

FIGURE 9. Impact of consensus desirability (COD) of data center on the data center profit when DSR is <1.

The polynomial function is adopted for all parties except a data center, which uses our strategy. We investigated the performance of the proposed strategy under different market conditions by varying DSR.

DSR is a single numerical measure of the gap in supply and demand for resources in the market. Fairly precise estimation of DSR can be calculated by a methodology proposed by Macias and Guitart [39]. When DSR is <1, competition among providers increases, and they try to win a larger share of markets by attracting as many VM requests regardless the request specification. Therefore, conceding faster by rising consensus desirability (COD) of data centers increases the chance of attracting more requests, and hence improves the revenue. The experiment results (Fig. 9) show that when DSR is <1, data centers have higher revenue if COD is set to higher value. However, after a certain point ($COD = 0.6$), increasing COD of data centers results in no gain, but a slight loss in revenue. In contrast, as illustrated in Fig. 10, when DSR is >1, data centers with lower COD earn higher revenue.

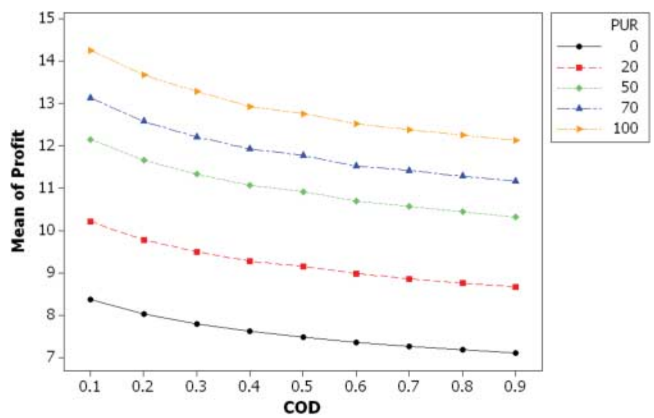

FIGURE 10. Impact of consensus desirability (COD) of data center on the data center profit when DSR is >1.

Furthermore, the gap between data center revenue increases when DSR is low, because a failure in reaching an agreement means those providers have permanently lost a chance of increasing their data centers utilization to other providers. However, when DSR is high, even if providers do not win an agreement at the beginning, their chances increase as the other providers utilization increases and there is no room for new requests.

8. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we proposed a time-dependent negotiation strategy capable of assessing the reliability of offers to fill the gap between decision-making and bargaining. To select an appropriate configuration for different negotiation objectives (e.g. number of deals made), we investigated the consequences of modification of parameters such as deadline, initial offer and type of time-dependent tactic (polynomial or exponential). Although many of the works in the literature apply the same pattern of concession for all clients when negotiating in parallel, we argued that discriminating regarding the pattern of concession helps cloud providers to accommodate more requests and thus increase their profit. Our approach was tested against purely time-dependent approaches, and it shows its dominance in generating more profit for providers. Furthermore, we show how providers could dynamically and based on market condition increase or decrease the COD to raise their revenue.

Future research can investigate effects of considering heterogeneous strategies on the achieved profit, II and ratio of deals made. Besides, consequences of variation in reliability constraints on the number of successful negotiations can be examined. Applying a combination of fuzzy similarity and time-dependent function is another area that can be explored. The current research is mainly designed for on-demand instances. However, investigating profitability of a third party

NS that operates on top of spot markets and offers resources with different reliability and prices is yet another promising research topic.

REFERENCES

[1] Narasimhan, B. and Nichols, R. (2011) State of cloud applications and platforms: the cloud adopters' view. *Computer*, **44**, 24–28.

[2] Khasnabish, B., Chu, J., Ma, S., So, N., Unbehagen, P., Morrow, M., Hasan, M., Demchenko, Y. and Meng, Y. (2013) Cloud reference framework. *Internet Eng. Task Force*, **7**, 1–47.

[3] Fortis, T.-F., Munteanu, V. and Negru, V. (2012) Towards a Service Friendly Cloud Ecosystem. *2012 11th Int. Symp. Parallel and Distributed Computing (ISPDC)*, Anchorage, Alaska, USA, June 25–29, pp. 172–179. IEEE Computer Society, Washington, DC, USA.

[4] Redl, C., Breskovic, I., Brandic, I. and Dustdar, S. (2012) Automatic SLA Matching and Provider Selection in Grid and Cloud Computing Markets. *Proc. 2012 ACM/IEEE 13th Int. Conf. Grid Computing (GRID)*, Beijing, China, September 20–23, pp. 85–94. IEEE Computer Society, Washington, DC, USA.

[5] Dastjerdi, A.V., Tabatabaei, S. and Buyya, R. (2010) An Effective Architecture for Automated Appliance Management System Applying Ontology-based Cloud Discovery. *Proc. 2010 10th IEEE/ACM Int. Conf. Cluster, Cloud and Grid Computing (CCGrid)*, Melbourne, Australia, May 17–20, pp. 104–112. IEEE Computer Society, Washington, DC, USA.

[6] Xiao, Z., Chen, Q. and Luo, H. (2014) Automatic scaling of internet applications for cloud computing services. *IEEE Trans. Comput.*, **63**, 1111–1123.

[7] Hasan, M., Magana, E., Clemm, A., Tucker, L. and Gudreddi, S. (2012) Integrated and Autonomic Cloud Resource Scaling. *Proc. 2012 IEEE Network Operations and Management Symp. (NOMS)*, Hawaii, USA, April 16–20, pp. 1327–1334. IEEE Computer Society, Washington, DC, USA.

[8] Alhamazani, K., Ranjan, R., Mitra, K., Jayaraman, P., Huang, Z., Wang, L. and Rabhi, F. (2014) Clams: Cross-layer Multi-cloud Application Monitoring-as-a-Service Framework. *Proc. 2014 IEEE Int. Conf. Services Computing (SCC)*, Anchorage, Alaska, USA, June 27–July 2, pp. 283–290. IEEE Computer Society, Washington, DC, USA.

[9] Jehangiri, A., Yahyapour, R., Wieder, P., Yaqub, E. and Lu, K. (2014) Diagnosing Cloud Performance Anomalies using Large Time Series dataset Analysis. *Proc. 2014 IEEE 7th Int. Conf. Cloud Computing (CLOUD)*, Anchorage, Alaska, USA, June 27–July 2, pp. 930–933. IEEE Computer Society, Washington, DC, USA.

[10] Joshi, K., Yesha, Y. and Finin, T. (2014) Automating cloud services life cycle through semantic technologies. *IEEE Trans. Serv. Comput.*, **7**, 109–122.

[11] Dastjerdi, A.V. and Buyya, R. (2012) An Autonomous Reliability-aware Negotiation Strategy for Cloud Computing Environments. *Proc. 2012 12th IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing (CCGrid 2012)*, Ottawa, Canada, May 13–16, pp. 284–291. IEEE Computer Society, Washington, DC, USA.

[12] Faratin, P. (2000) Automated service negotiation between autonomous computational agents. PhD Thesis, University of London.

[13] Hou, C. (2004) Predicting Agents Tactics in Automated Negotiation. *Proc. IEEE/WIC/ACM Int. Conf. Intelligent Agent Technology*, Beijing, China, September 20–24, pp. 127–133. IEEE Computer Society, Washington, DC, USA.

[14] Axelrod, R. (2006) *The Evolution of Cooperation: Revised Edition*. Basic Books, New York, USA.

[15] Sim, K. and Wang, S. (2004) Flexible negotiation agent with relaxed decision rules. *IEEE Trans. Syst. Man Cybernet. B: Cybernet.*, **34**.

[16] Coehoorn, R.M. and Jennings, N.R. (2004) Learning on Opponent's Preferences to Make Effective Multi-issue Negotiation Trade-offs. *Proc. 6th Int. Conf. Electronic Commerce*, October 20–22, pp. 59–68. ACM, New York, NY, USA.

[17] Faratin, P., Sierra, C. and Jennings, N. (2002) Using similarity criteria to make issue trade-offs in automated negotiations. *Artif. Intell.*, **142**, 205–237.

[18] Rahwan, I., Kowalczyk, R. and Pham, H.H. (2002) Intelligent Agents for Automated One-to-Many E-commerce Negotiation. *Proc. Twenty-fifth Australasian Conf. Computer Science*, Melbourne, Victoria, Australia, January 28–February 1, pp. 197–204. Australian Computer Society, Inc., Darlinghurst, Australia.

[19] Xiao, Z. and Cao, D. (2010) A Policy-based Framework for Automated SLA Negotiation for Internet-based Virtual Computing Environment. *Proc. 16th Int. Conf. Parallel and Distributed Systems (ICPADS)*, Shanghai, China, December 8–10, pp. 694–699. IEEE Computer Society, Washington, DC, USA.

[20] Lawley, R., Luck, M., Decker, K., Payne, T. and Moreau, L. (2003) Automated negotiation between publishers and consumers of grid notifications. *Parallel Process. Lett.*, **13**, 537–548.

[21] Sim, K. (2010) Grid resource negotiation: survey and new directions. *IEEE Trans. Syst. Man Cybernet. C: Appl. Rev.*, **40**, 245–257.

[22] Comuzzi, M., Francalanci, C. and Giacomazzi, P. (2008) Pricing the quality of differentiated services for media-oriented real-time applications: a multi-attribute negotiation approach. *Comput. Netw.*, **52**, 3373–3391.

[23] Sim, K.M. (2006) Grid commerce, market-driven g-negotiation, and grid resource management. *IEEE Trans. Syst. Man Cybernet. B: Cybernet.*, **36**, 1381–1394.

[24] Zulkernine, F. and Martin, P. (2011) An adaptive and intelligent SLA negotiation system for web services. *IEEE Trans. Serv. Comput.*, **4**, 31–43.

[25] Chhetri, M.B., Vo, Q.B. and Kowalczyk, R. (2012) Policy-based Automation of SLA Establishment for Cloud Computing Services. *Proc. 2012 12th IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing (CCgrid 2012)*, Ottawa, Canada, May 13–16, pp. 164–171. IEEE Computer Society, Washington, DC, USA.

[26] Bajaj, S. *et al.* (2006) Web services policy 1.2-framework (WS-policy). *W3C Member Submission*, **25**, 12.

[27] Copil, G., Moldovan, D., Salomie, I., Cioara, T., Anghel, I. and Borza, D. (2012) Cloud SLA Negotiation for Energy Saving-a Particle Swarm Optimization Approach. *Proc. 2012 IEEE Int.*

Conf. Intelligent Computer Communication and Processing (ICCP), Cluj, Romania, August 30–September 1, pp. 289–296. IEEE Computer Society, Washington, DC, USA.

[28] Ranaldo, N. and Zimeo, E. (2013) Capacity-aware Utility Function for SLA Negotiation of Cloud Services. *Proc. 2013 IEEE/ACM 6th Int. Conf. Utility and Cloud Computing (UCC)*, Dresden, Germany, December 9–12, pp. 292–296. IEEE Computer Society, Washington, DC, USA.

[29] Yaqub, E., Yahyapour, R., Wieder, P., Kotsokalis, C., Lu, K. and Jehangiri, A. (2014) Optimal Negotiation of Service Level Agreements for Cloud-based Services Through Autonomous Agents. *Proc. 2014 IEEE Int. Conf. Services Computing (SCC)*, Anchorage, Alaska, USA, June 27–July 2, pp. 59–66. IEEE Computer Society, Washington, DC, USA.

[30] Son, S. and Sim, K.M. (2012) A price- and-time-slot-negotiation mechanism for cloud service reservations. *IEEE Trans. Syst. Man Cybernet. B: Cybernet.*, **42**, 713–728.

[31] Raiffa, H. (1982) *The Art and Science of Negotiation*. Belknap Press, MA, USA.

[32] Wancheng, N., Lingjuan, H., Lianchen, L. and Cheng, W. (2007) Commodity-market Based Services Selection in Dynamic Web Service Composition. *Proc. 2nd IEEE Asia–Pacific Service Computing Conf.*, Tsukuba Science City, Japan, December 11–14, pp. 218–223. IEEE Computer Society, Washington, DC, USA.

[33] Dastjerdi, A., Tabatabaei, S. and Buyya, R. (2011) A dependency-aware ontology-based approach for deploying service level agreement monitoring services in cloud. *Softw.: Practice Exp.*, **42**, 501–518.

[34] Jsang, A. and Ismail, R. (2002) The Beta Reputation System. *Proceedings of the 15th Bled Electronic Commerce Conference*, Slovenia, June 17–19, pp. 41–55. Bled Electronic Conference Proceedings, Slovenia.

[35] Pang, B. and Lee, L. (2008) Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, **2**, 1–135.

[36] Sharifi, M., Manaf, A., Memariani, A., Movahednejad, H. and Dastjerdi, A. (2014) Consensus-based Service Selection using Crowdsourcing under Fuzzy Preferences of Users. *Proc. 2014 IEEE Int. Conf. Services Computing (SCC)*, Anchorage, Alaska, USA, June 27–July 2, pp. 17–26. IEEE Computer Society, Washington, DC, USA.

[37] Calheiros, R., Ranjan, R., Beloglazov, A., De Rose, C. and Buyya, R. (2011) Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw.: Practice Exp.*, **41**, 23–50.

[38] Amazon. Amazon elastic compute cloud (amazon ec2). http://aws.amazon.com/ec2 (accessed september 2012).

[39] Macias, M. and Guitart, J. (2010) Using Resource-level Information into Nonadditive Negotiation Models for Cloud Market Environments. *Proc. Network Operations and Management Symposium (NOMS)*, Osaka, Japan, April 19–23, pp. 325–332. IEEE, Washington, DC, USA.