This is a post-print version of the paper published at the followinf DOI: https://doi.org/10.1504/IJGUC.2017.085915

Combining reputation and QoS measures to improve cloud service composition

Fabrizio Messina* and Giuseppe Pappalardo

Department of Mathematics and Informatics,
University of Catania,
V.le Andrea Doria 6, 95125 Catania, Italy
Email: messina@dmi.unict.it
Email: pappalardo@dmi.unict.it
*Corresponding author

Antonello Comi, Lidia Fotia, Domenico Rosaci and Giuseppe M.L. Sarné

Department of Engineering (DIIES and DICEAM),
University Mediterranea of Reggio Calabria,
Loc. Feo di Vito, 89122 Reggio Calabria, Italy
Email: antonello.comi@unirc.it
Email: lidia.fotia@unirc.it
Email: domenico.rosaci@unirc.it
Email: sarne@unirc.it

Abstract: In order to provide a wide range of composite Cloud services, providers need to establish mutual agreements on large-scale distributed multi-cloud scenario. In such a way, providers can compose effective and efficient service workflows by taking resources of their own competitors and gain the capability to satisfy unexpected workload peaks. In this paper, we propose a reputation-based model capable to support the composition of complex Cloud services by taking into account both costs and measures of QoS which are collected by measuring both system measures and reputation feedbacks provided by the customers. The proposed model has been validated by a set of experimental results obtained by means of a number of simulations.

Keywords: cloud computing; reputation; QoS; service composition.

1 Introduction

In the last years, the business around on-demand computing has rapidly increased. This model is particularly suitable to address problems which require a large number of computing resources and/or a set of services of heterogeneous nature.

In the aforementioned context, Cloud Computing (Grossman, 2009; Armbrust et al., 2010) is the most widespread paradigm in the field of on-demand service composition (Jula et al., 2014), as it allows providers to deliver services over the internet through hardware and software deployed as third-party data/computing centres. This is made possible due to software stacks (Wen et al., 2012) built over virtualisation technologies (Barham et al., 2003; Kivity et al., 2007). IT enterprises store data and software, service providers have simplified software installation and maintenance, while end-users can access services, share data and collaborate easily.

In the last years, companies and public administrations have been moving applications into the Cloud (Leymann et al., 2011; Messina et al., 2013b), in order to overcome some limits in managing data and applications, and to reduce the costs related to data centres maintenance. The emerging practice of Cloud Service Composition (Jula et al., 2014) – i.e providing complex services composed by atomic services running on multiple IT centres – has encouraged the growth of the XaaS market (Anything as a Service), which allows users to take benefits of these new opportunities, i.e. composing their applications (e.g. workflows) easily in the cloud. As a consequence, services are provided to Citizens and Enterprises (PON, 2014) in an efficient way.

In particular, to compose such services from atomic ones, it is important to have a pool of available services (basic or composed) that can be mutually interfaced. Myriads of Cloud services are published worldwide every year (Anderson et al., 2013), which are the basis to compose complex services needed by customers. This scenario brings scientists to approach to the Cloud Computing Service Composition (CCSC) in order to deal with the providers' problem of selecting appropriate services (Messina et al., 2013a) from a service pool. The main challenge is represented by the rapid changes of service parameters and network properties, which impact on the measure of important Quality of Service (QoS) parameters (Jula et al., 2014).

As Service Composition is based on the availability of pool of interoperable services, providers publish services that a broker can select on the basis of functional and non-functional requirements. In particular, non-functional requirements are expressed and negotiated on the basis of QoS metrics (Stantchev and Schröpfer, 2009). To this aim, many different approaches have been proposed in the literature and most part of them have been summarised in (Jula et al., 2014). Usually, they are classified into five categories: graph-based algorithms (CGBAs), combinatorial algorithms (CAs), machine-based approaches (MBAs), structures (STs), and frameworks (FWs). For instance, the main approach of the first three categories consist in solving an optimisation problem based on QoS parameters and data measurements. The fourth category includes techniques devoted to help providers in managing the big amount of information about atomic services by exploiting a wide variety of data structures (Bayer and Unterauer, 1977). Finally, the last category includes those approaches aimed at improving automation during the composition process, e.g. requirements analysis and knowledge-based maintenance.

Such approaches take account of the reputation as a parameter to consider among others, while none of them considers the use of a reputation system to improve the services selection. To provide an effective approach for composing efficient services for a 'Cloud community', e.g. intercloud as cloud federations (Messina et al., 2014a; Comi et al., 2014; De Meo et al., 2015; Messina et al., 2016), we propose to combine reputation-based information with QoS measurements performed in the community. Our approach is based on a collaborative scenario (Grozev and Buyya, 2014) on which providers are able to expand their 'service catalog' on the base of the competitors resources availability (Messina et al., 2012; Messina et al., 2014b). Reputation measures allow customers and providers to improve service composition, if combined with estimation of QoS and costs. For this aim we discuss a simple heuristic with the goal of finding a trade-off between effective QoS measurements and reputation estimated by collecting feedbacks. Some experimental simulations based on the proposed approach confirm our expectations.

The paper is organised as follows. Related works are discussed in Section 2, while in Section 3 we present an analysis of the main requirements of complex services, e.g. workflows. In Section 4, we present the proposed reputation-based approach, and an experimental evaluation of the proposed approach is provided in Section 5. While in Section 6, we draw our conclusions and discuss possible developments of our ongoing research.

2 Related work

In this section, we briefly review some works which deal with the problem of Cloud Service Composition (SC). Given the rich literature on the subject, we only discuss those studies closely related to this work and/or could be integrated with it.

A framework for service composition is described in Pham et al. (2010). The authors adopt an agent-based framework where a composition agent receives and analyses requests for managing services. By using a knowledge base, the agent tries to identify all the service dependencies for proposing a service composition after that all of the required services are available; then it updates the knowledge base. By a packaging engine software, a package is generated on demand by using existing composition together with the new composition and then registers it in a suitable service catalog.

Chen et al. (2012) designed a framework for automatically detecting service conflicts, supplying policies and user's requirements. In a first phase this framework is able to detect conflicts by exploiting two components. The first to match policies and user's requirements, while the other provides to detect contradictions between the user's requirements and affiliate relationships. Besides, this framework can determine a set of appropriate atomic services based on the analysis of user's requirements and then publishing a composite service which can be considered as the best with respect to policies and requirements.

Another interesting work is Wu et al. (2012) which solves the SC problem by adopting the definition of trust as a conceptual probability by which a composite service is expected to execute a task as well as desired by the user. This trust-based method consists of three steps (i.e. components), namely trustworthy in service selection, trust in the composition processes, and trustworthy in binding the generated plan. The requirement analyser provides to classify user's requirements into their different elements, by including functional and non-functional requirements and expected input–output parameters. The service retriever restores information on the services coming from the resource pool by exploiting query requests. This component eliminates inappropriate services from the candidate list by a service filter, while the name and the type of the remaining services are identified by a WSDL analyser. Finally, the clustering component, template generator and binding optimiser check both the services composability and the math interfaces and then evaluate the binding plan trust.

CloudRecommender (Zhang et al., 2012) is a cloud-based service composition system structured on three layers, where the first is a configuration management layer in which a cloud service and a cloud QoS ontologies are located together for uncovering services based on their functionalities and QoS parameters, while services are mapped to a rational model and a data structure. The second is the Application Logic layer selecting single services in the form of SQL queries to include criteria, views, and stored procedures. The third is a widget layer dividing the user interface into four objects consisting of the recommendation and the computing, storage and network resources. This layer is implemented by using the Web Services and several JavaScript frameworks.

A novel framework for adaptive service selection in mobile cloud computing is in Wu et al. (2013). The framework extracts the QoS by the user's preferences immediately after a request has been received. Then, based on the Euclidean distance, some of the nearest user's service preferences are selected and suggested to the service adapter. Finally, the service adapter selects the best service among the services suggested for the user, with respect to the device context matching and the effectiveness of the service options. For reaching the context matching service basing on the input information, also a fuzzy cognitive map model is adopted in the service vice adapter module. Unfortunately, the proposed framework can only be used to select a single service at time.

Some approaches for SC also consider reputation into their models. For instance, Ye et al. (2011) for composing Cloud services adopt a genetic-algorithm-based technique to calculate QoS values of cloud services. The same authors classified their work as a combinatorial algorithm for Cloud Computing SC. Bao and Dou (2012) consider the correlations between services in service composition, i.e. the fact that services selected in cloud environment are not segregated and irrelevant with each other. They use Finite State Machine (FSM) to model the allowed invocation orders of services, and an improved Tree-pruning-based algorithm to create the Web Service Composition Tree (WSCT) and use a Simple Additive Weighting (SAW) to select an optimal path in the tree.

3 Service composition in multi-cloud environments

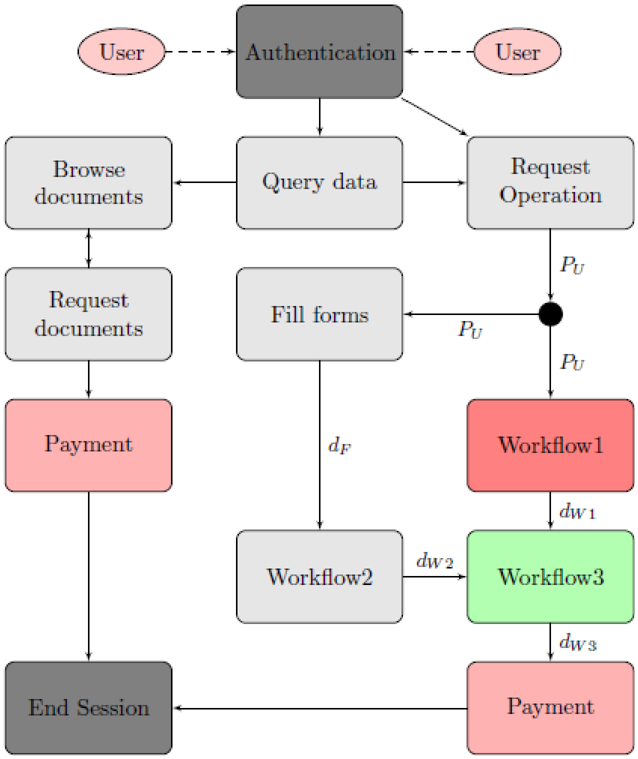
The generic 'Instance-Intensive Business Workflows' (hereafter IIBW) (Liu et al., 2012) takes account of a potentially large number of transactions involving short workflow instances having few steps. It is a significant example to discuss the main issues concerning Service Composition (SC) into multi-cloud environments. A relevant characteristic of a such service workflows is the *heterogeneity* of the applications composing them, i.e. the great differences among the several atomic services considered into workflows.

As an example, in Figure 1 an interactive workflow starting with a user authentication process is shown. For instance, social insurance services are typically designed as online banking services. Commonly, such authentication system processes release very short-term security tokens; in other words, there exists strict time-based constraints (not shown into Figure 1). A typical e-workflow may receive strong peaks of requests and therefore these systems need to scale very quickly when needed.

The workflow depicted in Figure 1 represents a typical example on which heterogeneous services are assembled together in order to satisfy the needs of the customer. More in detail, once the authentication is executed, users can choose between (i) asking the system to provide information about its own profile (*Query Data*) or (ii) performing an operation (*Request Operation*). Finally, every process may include one or more transactions to pay a certain amount of fees (*Payment*).

In Subsection 3.1, we briefly discuss the main issues related to QoS measurements and, in Subsection 3.2 that of reputation for composite cloud services.

Figure 1 Example of workflow (see online version for colours)



3.1 *QoS measurements*

The components of the workflow depicted in Figure 1 can be further characterised on the basis of the QoS measurements relying on specific metrics used for monitoring purposes, as follows.

Fine-grained QoS measurements. A large set of QoS measurements is needed to obtain a fine-grained characterisation of the service components of simple or composite services. Nevertheless, this is possible when, for various reasons, the service is under full control of the customer. For instance, it is possible that the service is deployed on a private cloud, or the provider offers fine-grained, accurate measurements tools. For instance, the *Workflow2* depicted into Figure 1 is gray-coloured to give an indication that it is under the customer's full control.

Coarse-grained QoS measurements. Some services – for instance those boxes with a colour that is not gray, depicted in Figure 1 – do not enable the customer to retrieve fine-grained QoS measurements. Such an unavailability is due to the fact that the service does not enable, in nature, fine-grained measurements or the provider does not permit it. Consequently, for this class of activities, only a few QoS metrics can be extracted. This last concern is taken into account in Section 4.

3.2 *Scope and incidence of reputation measures*

Behind QoS measurements, the level of *trustworthiness* perceived by users could give a high relevance in providing cloud services (Habib et al., 2010). In particular, in the following we adopt the definition of reputation provided by Jøsang et al. (2007), where reputation is considered as '*a collective measure of trustworthiness associated with an individual and it is usually based on the referrals/ratings an individual got from the members living in her/his community*'.

However, we have to consider that a fine-grained characterisation of reputation might not be obtained, similarly to a fine-grained user perception of QoS (and costs). Furthermore, the incidence of the gained reputation, on the basis of each composite service provided by external actors, should be carefully evaluated and integrated in the reputation system. For instance, a specific system for *Payment* transactions, to which the community has assigned a high level of trustworthiness, has to contribute to the overall service reputation by weighting all the factors characterising it. At this regards, in Section 4, we introduce a number of normalised weights in order to consider the incidence of each service when evaluating feedbacks (reputation).

Based on the previous considerations, in Section 4, we outline a reputation-based approach able to support the process of improving service composition by considering both providers measurements/estimations and customers feedbacks.

4 The reputation-based model for cloud computing service composition

In this section, we illustrate the design of a reputation-based model aimed at assisting players in selecting a service composition (SC) in a multi-cloud environment (Grozev and Buyya, 2014; Comi et al., 2015). In Subsection 4.1, we establish basic definitions that will be used later. In Subsection 4.2, we discuss some basic issues related to the practical problem of collecting reputation of cloud services, while in Subsection 4.3 we outline similar considerations regarding cost and QoS measurements. Finally, in Subsection 4.4 we present a simple approach to combine reputation and costs/QoS measurements.

4.1 *Premises and basic definitions*

When addressing the Service Composition (SC) problem in a multi-cloud environment, it should be taken into account that the larger the number of available alternatives for service components, the larger the number of possible compositions. At the same time we take into account that QoS parameters have to be evaluated for all the available services.

Trivially, the purpose of the customer is to obtain the highest level of QoS in accordance to negotiated non-functional requirements (Alhamad et al., 2010). Therefore, the main problem to solve in service composition is choosing a suitable number of atomic services to be coupled together to form the composite service which is compliant with the negotiated SLA. Existing techniques for SC find their application when a pool of atomic and functionally equivalent services are available in a multi-cloud context. As already stated, SC techniques should take account of sudden changes in non-functional requirements given by end-users, i.e. SLA (re)negotiation, services not yet available and so on (Jula et al., 2014).

Let us denote the generic composite service as $W = \{S_1, \ldots, S_k, \ldots, S_n\}$, built by n atomic services belonging to different classes, with $k = 1, \ldots, n$. Moreover, let us assume that for each class of service S_k in W there are m_k different alternatives $\left(s_k^1, \ldots, s_k^{m_k}\right)$, therefore there exists $\prod_{k=1}^{n} m_k$ possible compositions for W. If we denote with A the whole community of users, we also suppose that, after a user $i \in A$ often consumed a composition W_j for the service W, he/she provides his/her individual satisfaction levels about cost and QoS of W_j by means of two specific feedbacks, named FC_{ij} and FQ_{ij}, which range in [0,1]. By composing such feedbacks in a personalised way, we provide to compute a 'global' *feedback* $F_{ij} \in [0,1]$, which represents the personal point of view assigned by i to the composition W_j of the service W.

4.2 Retrieving reputation

Reputation embraces a number of manifold aspects and, therefore, it has a multidimensional nature (Sabater and Sierra, 2001). In this context, the main objective of the reputation system is the same of the measuring system, i.e. to allow the organisation to increase the ratio between the QoS and the cost for the customer.

To this purpose, the user i that has consumed the composition j for the service W (i.e. W_j) should give his/her feedbacks, respectively identified by FC_{ij} and FQ_{ij} and belonging to [0,1], which represent his/her level of satisfaction about cost and QoS for W_j. And, in order to represent it with a unique value, named $F_{ij} \in [0,1]$, the two feedbacks FC_{ij} and FQ_{ij} are combined together – in order to take into account their reciprocal relevance – by means of a coefficient named $\alpha_{ij} \in [0,1]$ that for the service W the user i assigns to the cost with respect to the QoS of W_j. In other words, we define $F_{ij} = \mathcal{F}\left(FC_{ij}, FQ_{ij}\right)$. This value can be written as a linear combination of FC_{ij} and FQ_{ij}, then F_{ij} can be expressed as:

$$F_{ij} = \alpha_{ij} \cdot FC_{ij} + \left(1 - \alpha_{ij}\right) \cdot FQ_{ij} \tag{1}$$

For sake of clarity, note that each service W_j is composed by a number of selected services s_k belonging to the different classes of service S_k with $k \in \{1, \ldots, n\}$. Furthermore, the feedbacks FC_{ij} and FQ_{ij} take into account each of such service components and, therefore, they can be written as:

$$FC_{ij} = \sum_{k=1}^{n} w_{i,k}^{C} \cdot FC_{ij}^{k}$$

$$FQ_{ij} = \sum_{k=1}^{n} w_{i,k}^{Q} \cdot FQ_{ij}^{k}$$

where FC_{ij}^k and FQ_{ij}^k are the contributes of each of the n selected service $s_k \in S_k$ forming the composed service W_j, with $w_{i,k}^C$ and $w_{i,k}^Q$ real coefficients ranging in [0,1] to weight the relevance of each service component of W_j in computing FC_{ij} and FQ_{ij}, with $\sum_{k=1}^{n} w_{i,k}^C = 1$ and $\sum_{k=1}^{n} w_{i,k}^Q = 1$.

Similarly,

$$\begin{aligned} F_{ij} = \sum\nolimits_{k=1}^{n} F_{ij}^{k} = \alpha_{ij} \cdot \sum\nolimits_{k=1}^{n} w_{i,k}^{C} \cdot FC_{ij}^{k} + \\ + \left(1 - \alpha_{ij}\right) \cdot \sum\nolimits_{k=1}^{n} w_{i,k}^{Q} \cdot FQ_{ij}^{k} \end{aligned} \tag{2}$$

However, we can observe that users rarely have the opportunity to interact with all the atomic services which compose W_j. This implies that the task of collecting the users' feedbacks about costs and QoS for each single atomic service of W_j and weighting their individual contributes might not be possible to carry out. As a consequence, each of the weights $w_{i,k}^C$ and $w_{i,k}^Q$, referred to a service chosen by the service class S_k to form W_j, will not be computable by users. Therefore, we can assume that, usually, only the feedbacks FC_{ij} and FQ_{ij} will be provided by users in an indivisible manner.

In order to compute reliable values for the reputation perceived by the user i for the composed service W_j (i.e. R_{ij}, RC_{ij} and RQ_{ij}) in terms of cost and QoS, we need: (1) to collect a statistically relevant number of observations (i.e. feedbacks) given by each user i and (2) to take into account that feedbacks can also significantly change in time. As a consequence, reputation values need to be periodically recomputed in order to have updated values.

To this purpose, we consider a time window $\tau = [t_1, t_2]$, which is large enough to obtain a sufficient number of FC_{ij} and FQ_{ij} feedbacks for periodically computing reliable reputation values, e.g. by averaging them in the time interval τ. In this way, the reputation terms RC_{ij}, RQ_{ij} and R_{ij} for the composed service W_j can be computed, respectively, as:

$$RC_{ij}^{\tau} = \frac{1}{N_i^{\tau}} \sum_{l=1}^{N_i^{\tau}} FC_{ij}^{l}$$

$$RQ_{ij}^{\tau} = \frac{1}{N_i^{\tau}} \sum_{l=1}^{N_i^{\tau}} FQ_{ij}^{l}$$

$$R_{ij}^{\tau} = \frac{1}{N_i^{\tau}} \sum_{l=1}^{N_i^{\tau}} F_{ij}^{l} = \alpha_{ij} \cdot RC_{ij}^{\tau} + \left(1 - \alpha_{ij}\right) \cdot RQ_{ij}^{\tau}$$

where N_i^τ is the number of feedbacks issued by the i-th user in the interval τ and α_{ij} is the same coefficient provided by i that we introduced above.

In a multi-cloud scenario where r users, belonging to the community A, provide feedbacks about W_j in the time interval τ then the overall reputations $\overline{RC}_j^\tau$, $\overline{RQ}_j^\tau$ and $\overline{R}_j^\tau$ of W_j in the user community A can be computed as:

$$\overline{RC}_j^\tau = \frac{1}{r}\sum_{i=1}^{r} RC_{ij}^\tau$$

$$\overline{RQ}_j^\tau = \frac{1}{r}\sum_{i=1}^{r} RQ_{ij}^\tau$$

and

$$\overline{R}_j^\tau = \frac{1}{r}\sum_{i=1}^{r} R_{ij}^\tau = \alpha_j^\tau \cdot \overline{RC}_j^\tau + \left(1-\alpha_j^\tau\right)\cdot \overline{RQ}_j^\tau \tag{3}$$

where α_j^τ is computed as the average in the time interval τ as:

$$\overline{\alpha}_j^\tau = \frac{1}{r}\sum_{i=1}^{r}\alpha_{ij} \tag{4}$$

Periodically, reputation values and α coefficient are recomputed based on users' feedbacks for each new time interval τ in order to take into account changes occurred in the users' evaluations (Buccafurri et al., 2013b; Buccafurri et al., 2013a).

4.3 Measuring QoS and costs

On the provider sides, let us assume to measure costs and QoS for the atomic services which compose the generic service W_j in the time interval $\tau = [t_1, t_2]$. Since $S_k = \left(s_k^1, \ldots, s_k^{m_k}\right)$ is the k-th class of services, let be $C_k^l = c\left(s_k^l\right)$ the cost associated with the l-th service of class k and $Q_k^l = qos\left(s_k^l\right)$ the measured QoS.

Let be $C_{min}^k = min\left\{C_k^1, \ldots, C_k^{m_k}\right\}$ and $Q_{max}^k = max\left\{Q_k^1, \ldots, Q_k^{m_k}\right\}$ the minimum and the maximum values, respectively, computed over the costs and the measured QoS associated with the services of class k. Then, for a given service composition W_j, measures of costs and QoS can be normalised with respect to the minimum cost (i.e. C_{min}^k) and the maximum measured QoS (i.e. Q_{max}^k).

Furthermore, let be $1 \le l \le m_k$, such that $C_{j,k} = C_k^l$ and $Q_{j,k} = Q_k^l$, i.e. the measured cost and QoS for the current selection (identified by l) of the class of service k for the composition j. Therefore, we can write:

$$C_{j,k}^* = \frac{C_{min}^k}{C_{j,k}} \qquad Q_{j,k}^* = \frac{Q_{j,k}}{Q_{max}^k} \tag{5}$$

and, as in the computation of $\widehat{RC}_j$, $\widehat{RQ}_j$ and $\hat{R}_j$, the overall values $\widehat{C}_j$ and $\widehat{Q}_j$ can be computed as follows:

$$\widehat{C}_j = \sum_{k=1}^{n}\omega_{j,k}^C C_{j,k}^* \qquad \widehat{Q}_j = \sum_{k=1}^{n}\omega_{j,k}^Q Q_{j,k}^*$$

where $\sum_{j=1}^{n_j}\omega_{j,k}^C = \sum_{j=1}^{n_j}\omega_{j,k}^Q = 1$ with $\omega_{j,k}^C, \omega_{j,k}^Q \in [0,1] \in \mathbb{R}$. The weights ω assume the same meaning of weights w in the computation of FC_{ij} and FQ_{ij}, but differently from them the weight ω are computed on the providers sides.

Finally, for each service a global parameter H_j^τ can be computed over a given time interval $\tau = [t_1, t_2]$, as follows:

$$H_j^\tau = \overline{\alpha}_j^\tau \cdot \widehat{C}_j^\tau + \left(1-\overline{\alpha}_j^\tau\right)\cdot \widehat{Q}_j^\tau \tag{6}$$

The dependency on the parameter τ in equation (6) is due to the fact that Q^k and C^k are collected during the generic time interval $[t_1, t_2]$. We remark that the relation between equations (3) and (6) is represented by the parameter $\overline{\alpha}_j^\tau$, which is based on the users' preferences and computed as defined by equation (4).

In the following, we discuss a simple algorithm aimed at combining reputation R and measures H in order to improve service W_j with a different composition. It should be triggered as measures have been updated or one or mode services becomes unavailable.

4.4 Combining R-measures and H-measures

Let us suppose that, for a given service W, a number $r_n \le \sum_{k=1}^{n} m_k$ of reputation scores R and $h_n = \sum_{k=1}^{n} m_k$ of measurements H have been respectively collected and calculated. Remember that scores R derive from users' feedbacks while measures H are obtained by linearly combining normalised costs and QoS (see expression 6); therefore, $r_n \le \sum_{k=1}^{n} m_k$ when feedbacks for some instances of the composed service W are not present and, consequently, some reputation scores may be not computed.

Generally it cannot be stated that, for all compositions W_1 and W_2, it holds $H_1 \le H_2 \Rightarrow R_1 \le R_2$. In particular, by considering the case $H_1 \ge H_2$:

$$I)\quad \frac{\overline{\alpha}^\tau}{1-\overline{\alpha}^\tau}\Delta C > -\Delta Q$$

$$II)\quad \frac{1-\overline{\alpha}^\tau}{\overline{\alpha}^\tau}\Delta Q > -\Delta C$$

$$III)\quad \Delta Q > 0 \wedge \Delta C > 0$$

where $\Delta C = C_{j_1} - C_{j_2}$ and $\Delta Q = Q_{j_1} - Q_{j_2}$. More in detail, in the case:

I the (weighted) reduction of cost (ΔC) is greater than the reduction of QoS ($-\Delta Q$);

II the (weighted) increasing in terms of QoS is higher than the increasing of cost ($-\Delta C$);

III the composition W_2 is more efficient that W_1, i.e. higher QoS and lower/equal costs.

Now, supposing that $H_{j_1} \geq H_{j_2} \wedge R_{j_1} \leq R_{j_2}$, and that case *I* holds, it means that users, in average, do not accept the loss of QoS (ΔQ) in exchange of the cost reduction (ΔC). Conversely, if *B* holds, it means that users, in average, do not accept the increase of cost (ΔC) in exchange of the increase of QoS (ΔQ). Therefore, as a consequence, it means that, in average, for the users the only QoS improving without a contemporary cost reduction (or vice versa) is not sufficient to accept a service change.

Procedure *P1* reported below is rather simple and synthesises the considerations above. More in detail, let *HS* (*RS*) be the list containing the indexes h_j and r_j of the collected measures $H_{r_j}\left(R_{r_j}\right)$, ordered in ascending order. The first time that a composition has to be selected, the composition having the best measure *H* is selected (lines 2–3). Otherwise, if no measures *H* are still available, the composition with the best measure *R* is selected (lines 4–5). Then, the loop in lines 8–12 represents an attempt to improve the current choice *c*. Clearly instructions in lines 8–12 are based on case (I) and (II) discussed above.

[P1] Input: $HS = \{h_1, \ldots, h_n\}$, $RS = \{r_1, \ldots, r_n\}$,
c index of the current composition.
Output: index *c* updated

1: **Update** *HS* and *RS* with new measures and feedbacks
2: **if** $(c == Nil)$ **then**
3: **if** $HS \neq \varnothing$ **then**
4: $c = r_n$
5: **else**
6: $c = h_n$
7: **end if**
8: **else**
9: **for all** $h_k : H_{h_k} > H_c$ **do**
10: **if not** $(\Delta Q \geq 0 \wedge \Delta C \geq 0) \wedge R_{h_k} \geq R_c$ **then**
11: $c = h_k$
12: **end if**
13: **end for**
14: **end if**

We remark that parameter $\overline{\alpha}^{\tau}$, which is used to compute measures *H* and *R*, is constructed on the basis of provided users' feedbacks (see equation (4)), i.e. basing on the users feedbacks. Since parameter $\overline{\alpha}^{\tau}$ enables the binding of measures *R* with measures *H*, it allows users to take advantage of a unique global measure concerning users' perceptions about the cost and the QoS and the different measurements performed by the providers.

5 Experiments

In this section, we present the results of a set of simulations performed to test the Procedure *P1* discussed in Section 4 with a set of generated data.

We considered a scenario involving a provider which offers a composed service to its clients (i.e. users). The composed service consists of four independent, atomic services, each of them can be chosen among different interchangeable services, basing on different values of QoS and cost.

We assumed that for each atomic service there exist ten possible alternatives (i.e. a total of 10,000 services can be composed) and the provider is capable to measure QoS and calculate costs on his/her side. Furthermore, in order to generate suitable values of cost we assumed that on the provider side the actual cost of each atomic service will depend linearly on the QoS. Note that the QoS and cost of the composed service is computed from those of its components by taking into account that they contribute, respectively, for the 10%, 20%, 30% and 40% of the whole. Therefore, for each composed service it is possible to compute the associated *H* measure by taking into account the parameter $\overline{\alpha}$ (see Section 4), measured QoS and calculated cost.

Since, as stated in the previous section, on the client side a user is not always enabled to have single interactions with the components of a composed service, then he/she only evaluates the QoS and the cost of the whole composed service based on his/her personal point of view. Furthermore, we remark that the ratings assigned by each user to services will have, in general, different order relationships with respect to those measured by the provider. Therefore, in order to simulate this behaviour we have associated with each customer a different level of 'perception' about QoS and cost, and about their reciprocal weight in computing *R* (i.e. the value to assign to the parameter α). An additional constraint consists of the fact that each user is aware of a fraction of possible composed services. As a consequence, in the performed simulations, the algorithm proposed in the previous section will produce different results, which are based on the global number of composed services rated (i.e. known) by the users. To this purpose, in our experiments the algorithm run with different amount of users' ratings in order to evaluate their influence on the algorithm performance.

We considered a population of 50,000 users, each one provided with different behavioural parameters. In particular, users are randomly selected and release a rating *R* to a number of services randomly chosen among the 10k considered.

Furthermore, in order to produce increasing amounts of users' ratings we used to build several different $\overline{R}$ sets. In particular, we considered ten different sets of composite services and, therefore, let $\overline{R}_1, \overline{R}_2, \ldots, \overline{R}_{10}$ be the correspondent ratings assigned by the users. In particular, $\overline{R}_i$ will contain the rating of the first $500 \times i$ services, e.g. $\overline{R}_5$ will contain the user rating for the first 2500 services.

Then, we tested the proposed algorithm with each $\overline{R}_x$ (with $x \in \{500, 1000, \cdots, 5000\}$) together with a correspondent set of *H* measures. Note that in order to make comparable the experimental results, we fixed a priori the number of rating for each specific $\overline{R_x}$ set by neglecting that the model fixes a time interval τ and not the number of ratings stored in each set. This expedient does not invalidate the obtained results, but it avoids to have sets containing different rating populations and allow to correctly carry out the result comparison.

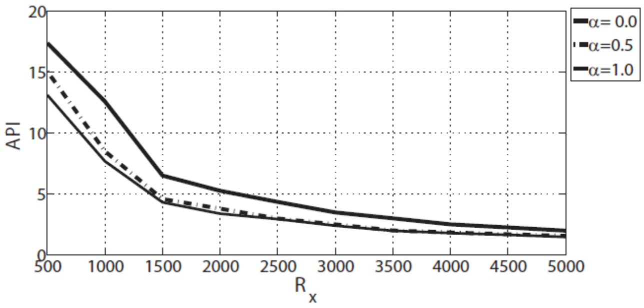
Table 1 The *Average Precision Index* (*API*) for each $\overline{R}$ set class

$\overline{R}$ set class	500	1000	1500	2000	2500
API	15.046	8.483	4.550	3.763	2.955
$\overline{R}$ set class	3000	3500	4000	4500	5000
API	2.470	1.970	1.803	1.661	1.523

In particular, for each $\overline{R}_x$, the simulation managed to select (at random) an already rated composed service — for which measures *H* always exist, as stated in the previous section – and searched to identify another composed service able to satisfy the users in terms of improved QoS and reduced cost by means of the procedure *P*1. The *Average Precision Index* (*API*) is defined as the ratio between the composed services for which procedure *P*1 has actually improved their subjective evaluation and the number of composed services for which there was no improvements. Related results are described in Table 1. In order to obtain such results we set $\alpha = 0.5$. Table 1 shows that API improves along with the percentage of rated services stored in the $\overline{R}_x$ sets. In particular, the increasing of *API* from a set to the larger one span from 43.61% to 8.03%. We can note how such percentages suddenly change between $\overline{R}_{1000}$ and $\overline{R}_{1500}$ and between $\overline{R}_{3000}$ and $\overline{R}_{3500}$.

As second set of experiments, we evaluated the incidence of the parameter α by testing the algorithm with different values. The values of API for $\alpha = \{0, 0.5, 1\}$ are shown in Figure 2. From such results it can be observed that the relevance that users give, in average, to the QoS with respect to the cost will impact on the precision of the proposed algorithm. The mentioned effects are maximum for $\overline{R}_{500}$ and decreases once the services rated by the users increase. It can be observed also that the value of precision will converge to the unit value. This means that procedure *P*1 is able to improve the perceived QoS of composite services, in average, in the 50% of cases.

Figure 2 API for different $\overline{\alpha}$ and $\overline{R}_x$



6 Conclusions and future work

In this work we presented an approach to combine a reputation model with measures of QoS and cost, in order to measure to help customers in selecting an optimal composed service in multi-cloud environments.

The presented approach tries to improve the current composition by combining two different set of measures named *H* and *R*. The former is computed by weighting feedbacks about QoS and costs, while the second by combining similar measures calculated by the providers on their sides. The meaning of *R* and *H* measures is different, as *R* gives an indication of the user's appreciation about costs and performance of the service, while *H* is composed by 'objective' measures performed by providers. The important relation between measures *R* and *H* is the parameter α which is inferred from the same *R* measures and it is used for combining QoS and costs when computing *H*. In our approach, we tried to use both the *R* and *H* measures by assuming that *H* measures are always available, while *R* measures can be collected over time. More in detail, when a service is chosen and offered by a provider based on *H* measures then this selection is improved as *R* measures become available. An important assumption is that Cloud providers are able to collect feedbacks from users and share them with the customer, such that the user's appreciation can be profiled over time.

To test our model, we performed an experimental campaign by which we verified the correctness of the proposed approach satisfying our expectations. The current limitation of the approach seems to be that, even the service selection is improved by considering the average perception of the users, the actual improvement due to this selection may not reflect the evaluation of the previous users feedbacks.

As future work, we planned to perform some studies on the influence of the weights used for balancing *R* and *H* measures, as well as those used to obtain such measures also by taking into account reliability measures on the basis of other reputation models as TRR (Rosaci et al., 2012). We also planned to perform a number of experiments by means of WS-DREAM (Zheng, 2014), which is a web service research data sets offered as real-world data freely available for research purposes, containing several different data sets about response times and throughput collected from thousands of real web services.

Acknowledgement

This work has been supported by project PRISMA PON04a2 A/F funded by the Italian Ministry of University and the NeCS Laboratory: Department DICEAM, University Mediterranea of Reggio Calabria.

References

Alhamad, M., Dillon, T. and Chang, E. (2010) 'Conceptual SLA framework for cloud computing', *2010 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, IEEE, Dubai, United Arab Emirates, pp.606–610.

Anderson, E., Lam, L-L., Eschinger, C., Cournoyer, S., Correia, J., Wurster, L., et al. (2013, February) *Forecast Overview: Public Cloud Services, Worldwide, 2011–2016, 4q12 Update*, Gartner Inc., Stamford, CT.

Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., et al. (2010) 'A view of cloud computing', *Communications of the ACM*, Vol. 53, No. 4, pp.50–58.

Bao, H. and Dou, W. (2012) 'A qos-aware service selection method for cloud service composition', *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW)*, 21–25 May, IEEE, Shanghai, China, pp.2254–2261.

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I. and Warfield, A. (2003) 'Xen and the art of virtualization', *ACM SIGOPS Operating Systems Review*, Vol. 37, No. 5, pp.164–177.

Bayer, R. and Unterauer, K. (1977) 'Prefix b-trees', *ACM Transactions on Database Systems (TODS)*, Vol. 2, No. 1, pp.11–26.

Buccafurri, F., Fotia, L. and Lax, G. (2013a) 'Allowing non-identifying information disclosure in citizen opinion evaluation', *Technology-Enabled Innovation for Democracy, Government and Governance*, Springer, Berlin, Heidelberg, pp.241–254.

Buccafurri, F., Fotia, L. and Lax, G. (2013b) 'Allowing privacy-preserving analysis of social network likes', *2013 Eleventh Annual International Conference on Privacy, Security and Trust (PST)*, IEEE, Tarragona, Spain, pp.36–43.

Chen, C., Yan, S., Zhao, G., Lee, B.S. and Singhal, S. (2012) 'A systematic framework enabling automatic conflict detection and explanation in cloud service selection for enterprises', *2012 IEEE 5th International Conference on Cloud Computing (CLOUD)*, IEEE, Honolulu, HI, USA, pp.883–890.

Comi, A., Fotia, L., Messina, F., Pappalardo, G., Rosaci, D. and Sarne, G.M. (2015) 'An evolutionary approach for cloud learning agents in multi-cloud distributed contexts', *2015 IEEE 24th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, IEEE, Larnaca, Cyprus, pp.99–104.

Comi, A., Fotia, L., Messina, F., Rosaci, D. and Sarnè, G.M. (2014) 'A qos-aware, trust-based aggregation model for grid federations', *On the Move to Meaningful Internet Systems: OTM 2014 Conferences*, Springer, Berlin, Heidelberg, pp.277–294.

De Meo, P., Messina, F., Rosaci, D. and Sarné, G.M. (2015) 'An agent-oriented, trust-aware approach to improve the qos in dynamic grid federations', *Concurrency and Computation: Practice and Experience*, Vol. 27, No. 17, pp.5411–5435.

Grossman, R.L. (2009) 'The case for cloud computing', *IT Professional*, Vol. 11, No. 2, pp.23–27.

Grozev, N. and Buyya, R. (2014) 'Inter-cloud architectures and application brokering: taxonomy and survey', *Software: Practice and Experience*, Vol. 44, No. 3, pp.369–390.

Habib, S.M., Ries, S. and Muhlhauser, M. (2010) 'Cloud computing landscape and research challenges regarding trust and reputation', *2010 7th International Conference on Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC)*, IEEE, Xi'an, Shaanxi, China, pp.410–415.

Jøsang, A., Ismail, R. and Boyd, C. (2007) 'A survey of trust and reputation systems for online service provision', *Decision Support Systems*, Vol. 43, No. 2, pp.618–644.

Jula, A., Sundararajan, E. and Othman, Z. (2014) 'Cloud computing service composition: a systematic literature review', *Expert Systems with Applications*, Vol. 41, No. 8, pp.3809–3824.

Kivity, A., Kamay, Y., Laor, D., Lublin, U. and Liguori, A. (2007) 'KVM: the Linux virtual machine monitor', *Proceedings of the Linux Symposium*, Vol. 1, pp.225–230.

Leymann, F., Fehling, C., Mietzner, R., Nowak, A. and Dustdar, S. (2011) 'Moving applications to the cloud: an approach based on application model enrichment', *International Journal of Cooperative Information Systems*, Vol. 20, No. 3, pp.307–356.

Liu, X., Yuan, D., Zhang, G., Li, W., Cao, D., He, Q., Chen, J. and Yang, Y. (2012) *The Design of Cloud Workflow Systems*, Springer, Heidelberg.

Messina, F., Pappalardo, G., Rosaci, D., Santoro, C. and Sarné, G. (2016) 'A trust-aware, self-organizing system for large-scale federations of utility computing infrastructures', *Future Generation Computer Systems*, Vol. 56, pp.77–94.

Messina, F., Pappalardo, G., Rosaci, D. and Sarné, G.M. (2014a) 'A trust-based, multi-agent architecture supporting inter-cloud vm migration in iaas federations', *Internet and Distributed Computing Systems*, Springer International Publishing, Cham, pp.74–83.

Messina, F., Pappalardo, G. and Santoro, C. (2012) 'Decentralised resource finding in cloud/grid computing environments: a performance evaluation', *2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, IEEE, Hammamet, Tunisia, pp.143–148.

Messina, F., Pappalardo, G. and Santoro, C. (2013a) 'Exploiting gpus to simulate complex systems', *2013 Seventh International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*, IEEE, Taichung, Taiwan, pp.535–540.

Messina, F., Pappalardo, G. and Santoro, C. (2013b) 'Integrating cloud services in behaviour programming for autonomous robots', *Algorithms and Architectures for Parallel Processing*, Springer International Publishing, Cham, pp.295–302.

Messina, F., Pappalardo, G. and Santoro, C. (2014b) 'Decentralised resource finding and allocation in cloud federations', *2014 International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, IEEE, Salerno, Italy, pp.26–33.

Pham, T.V., Jamjoom, H., Jordan, K. and Shae, Z-Y. (2010) 'A service composition framework for market-oriented high performance computing cloud', *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, HPDC '10*, ACM, New York, NY, USA, pp.284–287.

PON (2014) *Prisma Project Official Website*. Available online at: http://www.istc.cnr.it/project/prisma-piattaforme-cloud-interoperabili- smart-government

Rosaci, D., Sarnè, G.M.L. and Garruzzo, S. (2012) 'Integrating trust measures in multiagent systems', *International Journal of Intelligent Systems*, Vol. 27, No. 1, pp.1–15.

Sabater, J. and Sierra, C. (2001) 'REGRET: reputation in gregarious societies', *Proceedings of the Fifth International Conference on Autonomous Agents*, ACM, Montreal, Quebec, pp.194–195.

Stantchev, V. and Schröpfer, C. (2009) 'Negotiating and enforcing QoS and SLAs in grid and cloud computing', *Advances in Grid and Pervasive Computing*, Springer, Berlin, Heidelberg, pp.25–35.

Wen, X., Gu, G., Li, Q., Gao, Y. and Zhang, X. (2012) 'Comparison of open-source cloud management platforms: OpenStack and OpenNebula', *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, IEEE, Sichuan, China, pp.2457–2461.

Wu, Q., Zhang, M., Zheng, R., Lou, Y. and Wei, W. (2013) 'A QoS-satisfied prediction model for cloud-service composition based on a hidden Markov model', *Mathematical Problems in Engineering*, doi:10.1155/2013/387083.

Wu, X., Li, B., Song, R., Liu, C. and Qi, S. (2012) 'Trust-based service composition and optimization', *2012 19th Asia-Pacific Software Engineering Conference (APSEC)*, Vol. 1, pp.67–72.

Ye, Z., Zhou, X. and Bouguettaya, A. (2011) 'Genetic algorithm based qos-aware service compositions in cloud computing', in: Yu, J., Kim, M. and Unland, R. (Eds): *Database Systems for Advanced Applications*, *Vol. 6588 of Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp.321–334.

Zhang, M., Ranjan, R., Nepal, S., Menzel, M. and Haller, A. (2012) 'A declarative recommender system for cloud infrastructure services selection', in: Vanmechelen, K., Altmann, J. and Rana, O. (Eds): *Economics of Grids, Clouds, Systems, and Services*, *Vol. 7714 of Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, pp.102–113.

Zheng, Z. (2014) *Ws-dream – Web Service Qos Datasets*. Available online at: http://www.wsdream.net/dataset.html