J Netw Syst Manage (2019) 27:66–92
https://doi.org/10.1007/s10922-018-9457-0

TMM: Trust Management Middleware for Cloud Service Selection by Prioritization

Mukalel Bhaskaran Smithamol[1] · **Sridhar Rajeswari**[1]

Received: 17 November 2016 / Revised: 3 April 2018 / Accepted: 16 April 2018 /
Published online: 21 April 2018

Abstract Cloud computing is a prominently distributed paradigm that offers a wide variety of infrastructure, platform, and software services over the internet on demand. However, the identification of trustworthy cloud services imposes difficulties due to the multiplicity and the resemblance in their functionality. The shortage of proficient trust management schemes for services prevents the large-scale adoption of cloud computing paradigm by the public. In this paper, we propose trust management middleware (TMM), a framework for trustworthy service selection in the cloud. TMM performs service selection by integrating subjective assessment from users and objective assessment from service monitors. We proposed a new covariance based algorithm to determine the credibility of user feedback. Also, in our model, a novel objective trust evaluation algorithm is proposed based on prioritization of quality of service parameters depending on the user preferences. The results show that the proposed framework improves the accuracy of trust evaluation considerably and is more efficient in identifying trustworthy cloud services as compared with the other relevant methods.

Keywords Service selection · Subjective trust · Objective trust · Satisfaction level · Covariance

✉ Mukalel Bhaskaran Smithamol
smithamolm@acm.org

Sridhar Rajeswari
rajisridhar@gmail.com

[1] Department of Computer Science and Engineering, Anna University, Chennai 600025, India

1 Introduction

Cloud services and cloud computing are the most commonly used facilities in the world of information and communication technology (ICT). National Institute of standards and technology (NIST) defines cloud computing as a ubiquitous on-demand access to a shared pool of configurable resources that are elastically provisioned over the Internet [1]. In the cloud, services are commoditized and offered in a mode similar to the traditional utilities such as electricity, telephony, and water [2]. A cloud service is made available to the users on demand via the internet from Cloud Service Provider (CSP).

The economic benefits offered by the cloud computing along with its ability to accelerate any technological enhancement has even forced the government and public sector business organizations to adopt cloud services [3–5]. On the other hand, with the augmentation of cloud computing popularity and service utilization, security and trust concerns associated with the services have also emerged. The lack of consumer trust is a key inhibitor to the adoption of cloud services [6–8]. The establishment of trust associated with any cloud service depends on the performance evaluation of that service. In-depth and accurate assessment of cloud service performance is necessary for both consumers and providers which form an active research area [9]. To select a cloud service, consumers are usually concerned with the functions and Quality of Service (QoS) [10].

State-of-the-art approaches for trust based cloud service selection focus on evaluating QoS of each service and recommending a service with highest matching QoS parameters as per the user request [11–15]. However, in practice, it is challenging to measure the QoS values accurately due to the dynamic nature of the network and varying user requirements [16–18]. The trustworthiness of a CSP depends on both subjective and objective assessment which is related to evaluation QoS parameters associated with a specific service delivery [19–21]. Subjective trust is related to user perceptions of CSP based on the preference and requirements which are determined by the feedback given by users on Service Level Agreement (SLA) and Quality of Experience (QoE) [22–24]. Objective trust assessments are primarily based on extracting QoS values from SLA to determine service performance and trust [25, 26]. Service selection methods integrating both subjective trust and objective trust are necessary, especially when sensitive information such as financial and health data are outsourced for computation. Healthcare service demand data privacy, security and after-sales services, which are hard to quantify and can only be assessed through subjective assessments. Besides, the credibility of subjective or objective assessments given by users should be evaluated to ensure service selection accuracy. However, the proposals based on the above two methods ignored the satisfaction level of user preferences over the various QoS parameters associated with service delivery.

In this study, we have identified the need for advanced multi-criteria based measurement of user preferences method for determining service performance and trust effectively. Prioritizing the selection criteria according to user preference is important in objective trust assessment. Most of the literature uses static weight

assignment to measure the relative importance of the QoS parameters and is not correct as satisfaction varies widely due to the network and geographical conditions even for the same service request. Developing algorithms which quantify both objective and subjective trust by capturing user preferences is required for enhanced service selection.

To achieve efficient cloud service selection, our work considers prioritizing selection criteria based on user preference and context of service delivery in trust evaluation. The proposed Trust Management Middleware (TMM) incorporates both subjective trust and objective trust for cloud service trust evaluation. TMM performs the dynamic evaluation of QoS arguments of a service using prioritization of user preferences and the context of the service delivery. The use of prioritized aggregation operator [27] enables TMM to assign dynamic weights to the QoS attributes contributing to the accuracy of objective trust calculation. Also, the model includes a novel covariance based method to determine the credibility of user feedback. The proposed algorithm identifies dishonest rating and determines subjective trust effectively. The small mean error rate and high transaction success rate validates the practicability and efficiency of TMM. The major contributions of this article are:

1. A novel dynamic Objective Trust Evaluation (OTE) algorithm is proposed and evaluated. OTE uses prioritized aggregation operator to assess the transaction trust dynamically based on cloud user QoS preferences.
2. A novel Subjective Trust Evaluation (STE) algorithm is proposed and implemented. The algorithm uses covariance based approach to check the user feedback credibility.
3. Using OTE and STE, an efficient trust management middleware is introduced that recommend top-k trusted cloud services.

The rest of the paper is structured as follows. Section 2 provides an overall view of related works on the subject of trust management in cloud services. Section 3 illustrates the proposed system architecture and algorithms in detail to perform dynamic trust computation and service selection. Results and discussions in Sect. 4 provide a claim to the performance enhancement of trust computation using the proposed algorithms. Finally, Sect. 5 provides a fruitful conclusion to work by giving valuable insight into the future improvements.

2 Related Works

The industry and academia have already identified the need for regulation, monitoring, and trust establishment in the cloud computing environments [11, 12]. In the literature, several frameworks are proposed for assessing trust relationship from different perspectives. These frameworks mainly quantify trust from either user perspective or service provider perspective. Most of the proposals in the

literature determine trust based on historical records, feedback and QoS monitoring [12, 28]. These models can be classified as subjective trust models, objective trust models and models integrating both.

Paper [23] proposes the design and implementation of a reputation-based trust management framework named Cloud Armor. The suggested framework is a reliable cloud service recommendation system which computes trust mainly based on the user feedback. The Cloud Armor platform evaluates the credibility of user feedback based on majority consensus and feedback density. However, the evaluation of uncertainty in the feedback is still a challenge. To manage the uncertainty in trust evaluation, authors [29] proposed Bayesian trust model. Paper [30] recommend a lightweight reputation measurement approach for cloud services based on user feedback. The authors use fuzzy set theory to calculate the reputation score of each serving. Paper [31] suggest a trust model based on SLA negotiation for cloud marketplace. The customer has to accumulate the information regarding a provider periodically from other customers to evaluate the new trust value and reputation. It is a system of trust propagation. Nevertheless, feedback based trust evaluation models need computationally effective algorithms to determine the credibility of user feedback since it is affected by malicious user rating. As well, it should be noticed that feedback based trust assessment alone is not sufficient to define the final trust value of a cloud service.

In the literature of objective trust models, authors [32] proposed a QoS prediction model for cloud service selection based on similarity ranking. A personalized ranking and selection method was proposed in [33] based on customer satisfaction. A multi-criteria decision-making technique to rank Cloud services is proposed in [34]. In reality, rating-oriented approaches have difficulty guaranteeing accurate ranking prediction due to missing or sparse rating data. To address the reliability issues of QoS, monitoring tools were used and paper [35] developed a trust evaluation framework for cloud providers. T-broker [36] uses a lightweight feedback mechanism for adaptive trust calculation. Here, authors use a hybrid and adaptive trust model to compute the overall trust degree of service resources. Authors [37] present and implement SLA based trust management system for the multi-cloud environment. Paper [38] evaluate trust by considering the influence of opinion leaders and removing troll effect in the cloud environment. Actual QoS of cloud services is affected by the unpredictable network conditions [39–41]. Also, it is challenging to monitor and quantify QoS parameters such as security and privacy. Therefore, merely using QoS monitoring to assess the trustworthiness may not provide a complete solution to the problem of trustworthy service selection.

There are also some studies which have explored the importance of integrating both subjective and objective trust. Authors [42] provide a framework for evaluating customer satisfaction. Here, the authors define trust as soft trust and hard trust. The model assesses hard trust using a certification scheme and soft trust using user feedback. Paper [43] suggests multi-dimensional trust-aware cloud service selection. The framework evaluates trust based on the evidential reasoning

approach that integrates both perception-based trust value and reputation-based trust value. Paper [44] present an adaptive trust management model which combines the rough set and Induced Ordered Weighted Averaging (IOWA) operator for evaluating the performance of cloud services based on multiple attributes. Paper [45] propose a hybrid model for dynamic evaluation of trust and reputation in cloud services. The authors calculate trust based on user feedback and broker monitor service, and malicious user feedback rating area is detected using control charts. Paul [46], recommends a trust model based on past credential and present capability of the service provider. The approach combines SLA parameters and service provider capability for the evaluation of trust parameters. Paper [7] propose a context-aware service selection based on subjective and objective assessment. Here, the authors used objective assessments as benchmarks to filter out biased, subjective evaluations. Paper [47] propose a framework, SelCSP, which combines the trustworthiness and competence to estimate the risk of interaction. The model determines service trust based on the context of service delivery and feedback. Paper [19] propose an integrated trust evaluation method via combining objective trust assessment and subjective trust assessment. A mathematical model for comprehensive trust evaluation is proposed [48] by extracting SLA parameters.

However, the above proposals did not consider the influence of the user preference, context, and unfair ratings on the service performance evaluation. From the literature review, it is evident that more works are needed on trust management system integrating both subjective and objective trust for the cloud model. Compared to the existing works, the proposed framework TMM considers various factors which can influence the effectiveness and accuracy of service evaluation and selection. TMM provides a realistic trust assessment, not biased to cloud users or CSPs while being computationally simple.

3 Design of the Proposed Trust Evaluation Framework: TMM

As mentioned earlier, the principal actors are Cloud Service Provider, Trust Management Middleware, and Cloud User. The system can be viewed as a three layer architecture where CSP is at the bottom layer, TMM the middle layer, and CU at the top. TMM interacts with both CU and CSP and selects trusted cloud service for the request. The architecture of the proposed model TMM is shown in Fig. 1. The main architectural requirements of the work depend on the three layers, namely CU, TMM, and CSP. From the CU perspective, the primary requirement is a simplified interface with the adaptability that should address QoS preference and service functions. Also, a user needs QoS enabled and scalable service selection framework ensuring trust. From the perspective of CSP, the requirements are service monitor tool and secure database to store and manage trust vectors associated with each service. Finally, TMM should have the capability of evaluating both objective and subjective trust to recommend the most

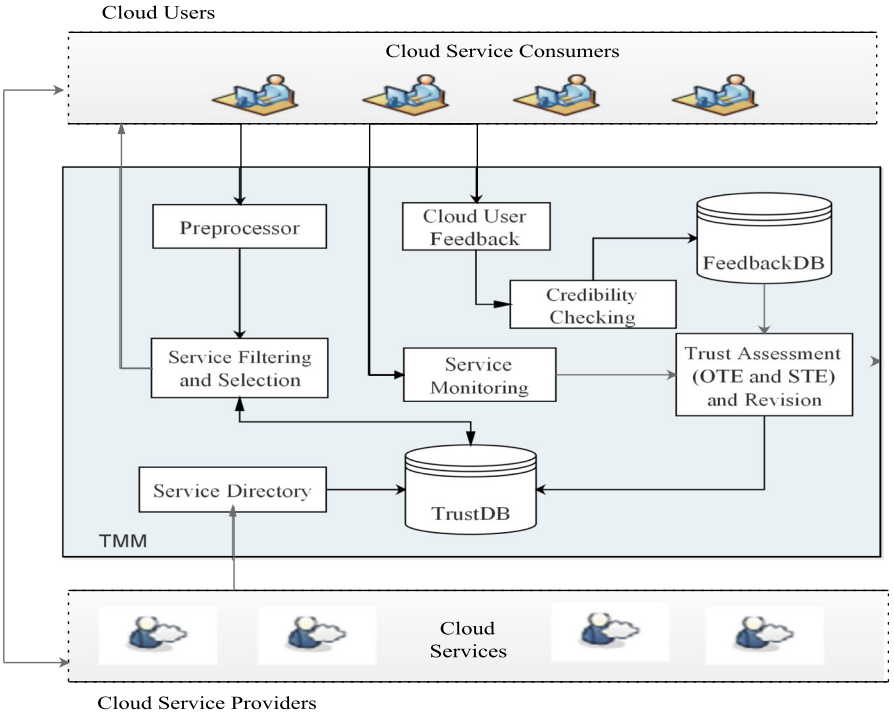

Fig. 1 Trust evaluation framework

trusted cloud services to the user. The details of the TMM are introduced as follows:

Preprocessor The preprocessor parses CU request and identifies the necessary QoS parameters required for the selection of cloud service. A CU may specify QoS parameters such as privacy, security, response time, availability, and cost along with their preference. The preprocessor does natural language processing at a minimal level using syntactic analysis to find QoS parameters and the associated preference value. To achieve this, tokens are extracted from the request by providing word boundaries. The extracted tokens are compared with QoS parameters stored in the array. Then the preprocessor prepares the service selection query using JavaScript Object Notation (JSON) [49]. We use JSON format compared to XML because JSON maps directly into data structures used by most of the prominent programming languages. A formal definition and format of the service selection query are given below.

Definition 1 A cloud service selection query Q is in the form $\{<q_1, v, p>, <q_2, v, p>, \ldots, <q_m, v, p>\}$, where q_i denotes ith QoS parameter, v denotes the expected value, and p denotes the priority or preference.

Service Selection and Filtering Cloud service selection maintains a trust database, TrustDB, which lists the services with the computed trust value. TrustDB has an entry for each cloud service registered. The proposed service filtering

algorithm filters the available cloud services based on the cumulative trust value for the recent time slots and returns top-k cloud services.

Service Monitoring We use the QoS metrics definition and calculation methods are given in [50] for the service monitor. The proposed trust evaluation framework using standard and well-established mechanism for service monitoring which monitors the QoS of service delivery. For cloud service monitoring, a variety of tools like CopperEgg (https://app.copperegg.com), Amazon CloudWatch (https://aws.amazon.com/cloudwatch) could be adapted.

Cloud User Feedback CU feedback is collected through a form with series of the questionnaire. For example, ease of use, customer support, failure meet, response time and others. Cloud user is given options to rate each QoS parameter as poor, average, good and excellent. Each answer is mapped into the real interval [0, 1], and is stored in the feedback database. User rating is subjective evidence towards the quality of service delivery [51] and is essential in a business model.

Credibility Checking The Credibility Checking module had an interface to collect feedback directly from users in the form of history records and stored them in the feedback database. In this paper, a novel covariance based approach to check the credibility of CU feedback is proposed. Similar users are identified using covariance and outliers are removed by Cumulative Sum (CUSUM) control charts [52].

Trust Assessment and Revision Trust assessment and revision is an important component of the framework. The module uses OTE and STE algorithms to determine the transaction level trust value. Each service invocation is recorded in the TrustDB along with evaluated aggregate trust value. The aggregate trust value of each service is maintained as a trust vector in the TrustDB. It contains the computed trust value using the proposed algorithms which can act as base for further trust related decisions. The format of the trust vector is (Transaction ID, Timestamp, Service ID, User ID, CSP ID, Trust Value). The proposed algorithm OTE in Sect. 3.1 uses a novel approach to calculate objective trust by assigning dynamic weight to the measured QoS using prioritization operator. Assigning weight according to the satisfaction level of the most prioritized QoS attribute in trust calculation is one of the important contributions in this paper. The calculation of subjective trust is based on CU feedback and the credibility of feedback is verified using the proposed covariance based algorithm.

Service Registration The CSPs deploy their services in the cloud and advertise through the web giving different QoS configurations along with cost. In general, a CSP may list various QoS parameters such as flexibility, transparency, security, tenant support, the level of control, ease of use, availability, scalability, billing procedure and other SLA parameters in the website. CSP can register services in the service directory, along with necessary information. The TMM evaluates the requesting CSP using the CSA [1, 53] recommendations. Once the evaluation procedure is completed, the TMM makes an entry in the TrustDB with an initial trust value to the services offered by the requesting CSP.The initial value gets modified after each invocation of the cloud service. Any new CSP must gradually build its trust value by offering good performance and QoS [54].

Database TMM maintains two databases namely TrustDB and FeedbackDB. TrustDB stores each transaction with its identity, the respective CSP ID, invoked service ID, CU location, CU QoS priority, timestamp, and transaction trust. Here, ID stands for identity. For each transaction, CU feedback is collected and stored in the feedback database. Each record has information of CU ID, service ID, transaction ID and a feedback rating of each QoS parameter. Feedback database is essential for checking the credibility of a CU current feedback.

3.1 Objective Trust Evaluation (OTE)

Service monitor evaluates objective trust using the evidence collected against QoS monitoring as per SLA. According to [46], while calculating objective trust the model considers many QoS attributes such as CPU percentage, disk read throughput, disk write throughput, and network bandwidth. Service monitor performs trust evaluation based on measuring the performance of these attributes as per the SLA. Server side QoS provide indications of the service capabilities and client-side QoS provide realistic quality of experience [32]. QoS metric that depends on user experiences such as response time, latency and failure needs to be measured on the client side. QoS metric like CPU percentage, disk read throughput, disk write throughput, and network bandwidth is measured from server side by monitoring VM usage. The trust evaluation models discussed in Sect. 2, assigned static weight to QoS parameters for all invocations of the service irrespective of the context of service delivery. The proposed TMM allocates dynamic weight to each QoS parameter by using prioritization operator [27], as CU preference of QoS parameter plays a vital role in the QoE.

Here, we discuss in detail the objective trust evaluation algorithm. For a CU request R_i, the preprocessor identifies a set of m QoS parameters and transforms the given request into service selection query as in Definition 1. The proposed algorithm OTE partitions the set Q_i into a h number of subsets using the CU priority of the QoS parameters. Each subset contains QoS parameters having equal priority, and are numerically ordered based on priority as $P_{i1} > P_{i2} > .. > P_{ih}$. The set P_{i1} contains higher priority QoS parameters than P_{i2}. Then the algorithm OTE maps priority value into the real interval [0, 1]. For each QoS parameter in the priority class P_{ih}, the algorithm determines the satisfaction level by the proposed Eq. (1). Satisfaction level, u, clearly indicates whether the quality of the intended QoS parameter meets the expected level according to SLA objectives against the monitored performance measurement.

$$u\left(q_{im}^{h}\right) = 1 - [E(q_{im}) - M(q_{im})] * 0.1 \tag{1}$$

Satisfaction level of a QoS parameter indicates the difference between anticipated performance $E(q_{im})$ and the monitored performance $M(q_{im})$, of the concerned QoS parameter. We multiply the difference with a percentage value of ten to normalize the satisfaction to the real interval [0, 1] for uniformity of further

calculation. The service monitor measures the quality of each parameter in the interval [1, 10], and the least difference between parameters will indicate better performance. For example, for the first invocation of a specified service assume that anticipated response time is 3 and monitored time is 4. Then the difference in the values of response time is 0.1, after converting to the real interval. Now, for the next invocation of the same service, assume the difference in the values of response time is 0.3. Here, the first value (0.1) indicates better performance compared to the second one (0.3). The better value should always result in good satisfaction level. To include this concept, subtract the resultant difference from numeral one to normalize it in the interval [0, 1]. Therefore, the final satisfaction for the response time in the above two examples is 0.9 and 0.7 respectively. Hence, according to the proposed approach, a larger satisfaction indicates better performance. After determining the satisfaction level, the algorithm OTE assigns weight to each QoS parameter.

The proposed dynamic weight assignment method depends on the satisfaction level and the priority class of each chosen QoS parameter for the service requested. The weight assigned to a lower priority QoS parameter depends on the satisfaction of the higher priority QoS parameter. To assign weight to the QoS parameters, at the beginning select the least satisfaction level from each priority class q by the proposed Eq. (2).

$$L_q = \left\{ min\left(u(q_{ij}^h)\right), 1 \le q \le h \quad \text{and} \quad 1 \le \mathrm{j} \le \mathrm{m} \right\} \tag{2}$$

For each priority class q, find the satisfaction value and this is the minimum satisfaction value of the QoS members of that class. Next, assign weight to each priority class by the proposed Eq. (3).

$$W_h = \begin{cases} 1 & \text{if} \quad h = 1 \\ \prod_{i=1}^{h-1} L_i & \text{if} \quad \mathrm{h} > 1 \end{cases} \tag{3}$$

The method allocates equal weight to all QoS parameters members of a priority class. As per the Eq. (3), always a high weight factor, one, is assigned to the high priority class. For any lower priority class, the weight factor will be depending on the satisfaction level of QoS parameters in all higher priority classes compared to the said QoS parameter considered. For example, the method gives weight one to the top priority class P_{i1} always. Now, assume that P_{i1} has two equal priority attributes and their satisfaction values are 0.7 and 0.9. Now, the weight of P_{i2} will depend on the least satisfaction level of P_{i1}, and it is 0.7. Finally, algorithm OTE quantifies objective trust as the sum of products of satisfaction factor and the weight factor of each priority class by the proposed Eq. (4). The details of the objective trust calculation procedure are formalized in Algorithm 1.

$$OT(S_j) \leftarrow sum\left(u\left(q_{ij}^h\right) \times w_{i,h}\right) \tag{4}$$

Algorithm 1: Objective Trust Evaluation (OTE)

Input: User request R_i and selected service S_j
Output: Objective trust for the service S_j

1 Initialize $Q_i = \{q_{i1}, q_{i2}, \ldots q_{im}\}$;
2 **foreach** $P_{ih} \in P_i$ **do**
3 **foreach** $q_{ij} \in p_{ih}$ **do**
4 $u\left(q_{im}^h\right) \leftarrow 1 - [E\left(q_{im}\right) - M\left(q_{im}\right)] * 0.1$;
5 **end**
6 $L_{ih} \leftarrow min\left(u\left(q_{im}^h\right)\right)$;
7 **end**
8 $Initialize \quad w_{i1} \leftarrow 1 \quad and \quad w_{i2} \leftarrow L_{i1}$;
9 **foreach** $q \in (3, h)$ **do**
10 $w_{iq} \leftarrow w_{i(q-2)} \times L_{(q-1)}$;
11 **end**
12 $Initialize \quad OT(S_j) \leftarrow 0$;
13 **foreach** $P_{ih} \in P_i$ **do**
14 $Temp \leftarrow 0$;
15 **foreach** $q_{ij} \in p_{ih}$ **do**
16 $Temp \leftarrow Temp + \left(u\left(q_{ij}^h\right) \times w_{i,h}\right)$;
17 **end**
18 $OT\left(S_j\right) \leftarrow OT\left(S_j\right) + Temp$;
19 **end**
20 $Return\left(OT\right)$

OTE works as follows. Lines 2–7 compute the satisfaction level of a QoS attribute (u) and each priority class (L) by the equations for $u\left(q_{im}^h\right)$, L_{ih}. Lines 8–11 updates the weight parameter value for each QoS attribute dynamically based on the satisfaction level for the present service invocation. Lines 12–19 do the calculation of objective trust through the substitution of previous values. We now summarize the effect of OTE algorithm and its reliability as follows.

Lemma 1 *Consider an invocation of* $OTE(R_i, Q_i, P_i, S_j)$ *where* S_j *is available. Let OT denote the value returned by OTE. Then, at the iteration number i of the outer loop of lines 12–19, the variable* $OT(S_j)$ *holds the sum of the weighted trust values of first i priority classes* P_1 *to* P_i.

Proof Let the variable i, represent the number of iterations of the loop over the number of priority classes h. The initial value of $OT(S_j) = 0$ is trivial, prior to the first iteration of the loop. If the loop invariant holds true before step i, then it will be true before step $(i + 1)$. Suppose that priority class i has r number of QoS attributes. In the ith iteration, the lines 15–17 compute the sum of trust values of QoS attributes in the priority class i as defined Eq. (4) by the following expression,

$$Temp_i = \sum_{j=1}^{j=r} u\left(q_{ij}^h\right) \times w_{i,h}$$

Then,

$$OT(S_j)_i = OT(S_J)_{i-1} + Temp_i$$

Hence, after i number of iterations, the variable $OT(S_j)$ holds the sum of weighted trust values of first i priority classes. Similarly, after $(i+i)$th iteration, the objective trust computation statement is $OT(S_j)_{i+1} = OT(S_J)_i + Temp_{i+1}$, which shows the maintenance of loop invariant. Consequently, at the end of $i = h$ iteration, the variable, $OT(S_j)$, holds the sum of h priority classes satisfying the loop invariant. □

Lemma 2 *Given a cloud service invocation (S_j, Q, P), algorithm OTE computes the objective trust by assigning optimal weight value to each QoS based on the satisfaction level.*

Proof The result of OTE depends on the cardinality of QoS set, $R \leftarrow |Q|$, and the number of priority classes, $S \leftarrow |P|$. The OTE algorithm consists of the following operations: first, the satisfaction level for each QoS attribute and priority class is determined. Second, dynamic weight is assigned to each priority class based on the satisfaction level. For each QoS parameter the satisfaction level is determined using the expression, $u(q_{im}^h) \leftarrow 1 - [E(q_{im}) - M(q_{im})] * 0.1$, which is the difference between experienced QoS value and anticipated value. Since the number of QoS chosen for each service is finite, line numbers 2–11 induces only finite computational cost. The core statements (line numbers 12–19) finding the objective trust is finite in terms of the input size as per the Lemma 1. The time complexity of line numbers 2–19 is quadratic and given by $O(R^2)$ which will happen when $S = R$. Consequently, the algorithm is scalable and reliable regarding the size of input data. □

3.2 Subjective Trust Evaluation (STE)

The CU feedback indicates the trustworthiness of a CSP based on QoE. Since cloud model has evolved as a business model, trust establishment through user feedback is necessary and essential. The system maintains cloud user feedback repository along with contextual information of service experience. A record in the feedback database has fields for storing transaction ID, service ID, user ID, region, timestamp, QoS parameter, experienced feedback for QoS parameter, the order of QoS parameters. The model differentiates a CU as known or unknown CU, depending on the past transaction history. A known CU has previous feedback in the repository for the same instance of the service invocation.

Definition 2 A similar cloud user is defined as one who has experience with the requested cloud service instance in the respective context.

Definition 3 The degree of similarity of feedback behavior is defined as affinity ratio which is calculated as the fraction of total similar users to total users in the feedback DB for the opted cloud service.

As per Definition 2, the similar user is one who has invoked the same cloud service in the past, and the feedback is recorded in the feedback DB. From the FeedbackDB, feedback of similar users is retrieved. Let UF_i, denotes the set of all similar user feedback for the service S_i and F_i denotes the set of feedback made by the user for the same service. The algorithm measures the variance of current feedback from both F_i and UF_i. Depending on the variance, credibility of current feedback is assessed, and subjective trust is evaluated accordingly. The approach considers each CU feedback as a continuous random variable, following a normal distribution. The joint distribution of any two normal random variables is also a normal distribution. The common expectation of any two random variables X_i and X_j, in general, is defined in the literature [55] as:

$$\left(X_i^m, X_j^n\right) = \iint_{\infty} x_i^m x_j^n f(X_i, X_j) dxdy \tag{5}$$

The covariance between X_i and X_j can be calculated as in [23]:

$$cov\left(X_i^m, X_j^n\right) = E\left[X_i^m X_j^n\right] - E\left[X_i^m\right] E\left[X_j^n\right] \tag{6}$$

Normalize the covariance to correlation coefficient ρ which depicts the relationship between any two users in the set and ρ is defined as per [55] as,

$$\rho = \frac{cov\left(X_i^m, X_j^n\right)}{\sqrt{var(X_i^m) var\left(X_j^n\right)}} \tag{7}$$

Outliers are removed using CUSUM control charts. CUSUM chart depicts the deviation of each variable from the target variable. The method determines pairwise correlation coefficient of all random variables involved in the scene. The variable ρ gives the linear relationship between random variables. A positive ρ indicates the relationship is growing in the positive quadrant and negative ρ in negative quadrant. Now, find the total user affinity ratio as the count of users in the positive quadrant to total users in the entire scene. Let P_r denotes the count of the positive relationship of user i and N_u be a total number of users in the scene. As given in Definition 3, user affinity ratio is calculated as

$$\delta = \frac{P_r}{N_{user}} \tag{8}$$

It may be noted that $0 \leq \delta \leq 1$. The affinity ratio gives a percentage of how many similar users have made the same opinion as for the current user. A user affinity ratio, $\delta < 0.5$, indicates that user feedback has more deviation from similar users and will have a negative effect on the user trust value and feedback. Over k timeslots, the aggregation of past feedback of δ users as,

$$f_p = \sum_{i=1}^{P_r} \sum_{t=1}^{k} uf_{it} * e^{rt} \tag{9}$$

An exponential decay function is used to reduce the impact of accumulated past feedback over the period. An unknown user does not have a history of feedback, and the trust value will depend only on the affinity ratio. For a known user the framework may have past feedback. Here, both user affinity ratio and the past feedback will have an influence on the trust value computed.

$$ST(S_j) = \begin{cases} \phi uf \pm \delta f_p & \text{unknownuser} \\ \phi uf + (1-\phi)\dfrac{\sum_{t=1}^{k} F_{ij}^{t}}{k} \pm \delta & \text{knownuser} \end{cases} \tag{10}$$

Here, *uf* stands for the current feedback of the CU and ϕ denotes the significance of current feedback. The ideal value of ϕ is determined by running the experiment many times, and the value which gives high performance is taken as an optimum value. The second term in Eq. (10) denotes the aggregate of CU feedback for the same service over k timeslot. The third term indicates how similar users are given feedback for the same service which is the product of the percentage of similar users and their aggregate feedback. If δ falls below the threshold value, then there will be a negative impact on the subjective trust value and the product is subtracted. Subjective trust is cumulative and malicious feedback cannot have a great impact on the final trust and reputation of a cloud service. The discussions regarding subjective trust evaluation are summarized in Algorithm 2.

Algorithm 2: Subjective Trust Evaluation (STE)

Input: User request R_i, the QoS set Q_i, Priority P_i, selected service S_j, user current feedback f_c, and feedback database
Output: Subjective trust for the service S_j

Initialize $UF, F, \delta, P_r, f_p = 0.0$;
/* K is the set of time slots considered */
foreach $CU \in P_r$ *and* $t \in K$ **do**
  $f_p \leftarrow f_p + sum\left(uf_t^{CU}\right) \times e^{-rt}$;
end
foreach $f \in F$ **do**
  $f_u \leftarrow sum\,(f)$;
end
$\bar{f}_u \leftarrow \frac{f_u}{|F|}$;
$\delta \leftarrow P_r / |UF|$;
if $\delta > 0.5$ **then**
  $\bar{f}_p \leftarrow \delta \times f_p$
else
  $\bar{f}_p \leftarrow -(\delta \times f_p)$
end
if $KnownCU$ **then**
  $ST\,(S_i) \leftarrow \phi f + (1-\phi)\,\bar{f}_u + \bar{f}_p$
else
  $ST\,(S_i) \leftarrow \phi f + \bar{f}_p$
end
return $ST\,(S_i)$

The proposed covariance based approach to find similar CUs is efficient because covariance is a proven statistical approach to find similarity. Also, everything in

nature is supposed to follow Normal distribution and outliers can be easily detected using CUSUM control chart.

Lemma 3 *Given a service invocation S_j and associated feedback f_c, algorithm STE effectively computes subjective trust.*

Proof The preprocessing steps remove the outliers regarding the feedback distribution and computes affinity ratio based on Definitions 2 and 3. Suppose that feedback history of each similar user is randomly partitioned into t time slots $(1 \leq t \leq K)$ to regulate the significance of recent feedback. Statements from 2 to 4 determines the sum of similar user feedback as defined Eq. (9) by the expression,

$$f_p = \sum_{j=1}^{r} \sum_{t=1}^{K} \left(uf_t^r\right) \times e^{-rt}$$

As long as outliers in similar feedback are eliminated properly by δ and temporal decay function, the number of similar users r is finite and induces only finite amount of computation cost of quadratic. Requester's similar feedback is also determined and sum is calculated by the expression

$$f_u = \sum_{j=1}^{l} f_j$$

where l denotes the self-similar service invocation history. Since users are classified in a fine-grained manner based on Definition 1 and malicious feedback is eliminated using CUSUM charts, bad feedback sequences are very rare. The user affinity ratio δ decides whether the sum of similar user feedback has a positive or negative impact and fix the value $\bar{f_p}$ accordingly (line number 10–14). A malicious positive feedback or negative feedback identified based on both similar user and own feedback history by $f_p \leftarrow f_p + sum\left(uf_t^{CU}\right) \times e^{-rt}$ and $f_u \leftarrow sum(f)$. The user affinity ratio determines the impact of malicious feedback as: $\bar{f_p} \leftarrow \delta \times f_p$ or $\bar{f_p} \leftarrow -\left(\delta \times f_p\right)$. Based on previous computation results, the subjective trust for present invocation of the service S_j is determined as one of the ways: $ST(S_i) \leftarrow \phi f + (1-\phi)\bar{f_u} + \bar{f_p}$ (if the user has a previous feedback history) or $ST(S_i) \leftarrow \phi f + \bar{f_p}$ (if the user is making his first invocation) satisfying the postcondition. Since the bad sequences from the feedback history is eliminated properly, the algorithm computes subjective trust effectively. Hence, algorithm STE is reliable as it transforms the input condition to the postcondition of the algorithm. □

3.3 Trust Assessment and Revision

Trust is associated with a service and thereby with its provider. Trust can be combined from different sources for making a better decision [42]. In this framework, trust is computed as the sum of the objective trust which reflects the monitored performance and subjective trust based on the CU feedback. The model finds the trust value *T*of *i*th service S_i from *j*th provider CSP_j at timestamp t as,

$$T_{S_i}^t = \beta \times ST(S_i) + (1 - \beta) \times OT(S_i) \tag{11}$$

where ST refers to subjective trust and OT refers to objective trust. From the simulation experiments, TMM finds the optimal value of β as 0.6. A non-zero value of β implies that user feedback always has a positive impact on the final trust value. Transaction level trust is evaluated using Eq. (11) and this will contribute to the final aggregate trust of the concerned cloud service and the respective CSP.

3.4 Service Selection

For a CU request, the service selection algorithm uses transaction level trust and context information, to find the most suitable subset of service providers. First, transactions with matching spatial context are retrieved from the database. From the reduced search space, the algorithm considers past interactions in the interval, $1 \leq t \leq k$. Now, assume that there exists S providers offer the requested service S_i. The algorithm calculates the Compound Trust Value (CTV) of each cloud service provider CSP_j on a specific cloud service S_i using the newly defined equation,

$$CTV_{CSP_j}(S_i) = \frac{1}{n}\left(\sum_{t=1}^{k} T_{S_i}^t\right) \tag{12}$$

The value of CTV is changed after each time window k. We assume that within each time window n invocations of cloud service S_i. CTV reflects the global trust value for the CSP in the recent time window. The transaction level trust for matching CSPs are retrieved from the repository. After calculating the compound trust rating of all available S CSPs who offer service S_i, TMM has at most S alternatives. Based on the CTV, alternatives can be ranked. Hence, TMM recommends top-k trusted cloud services. If CSPs maintain a stable performance, then they will have a stable trust value over the time slots and will have a good reputation.

4 Experiments

This section provides an evaluation of accuracy and efficiency of the proposed trust evaluation framework TMM through simulation and comparison of the performance of few relevant schemes. Accuracy shows how accurate was the trust calculation and efficiency indicates the competency in performance compared to relevant works in the literature.

Experimental Setup Java based simulation environment is built by using NetBeans IDE 8.1. The simulation environment is run on the machine with 14.0 GB RAM, Intel core i7-4790 CPU@ 3.6 GHZ.

Dataset Description To evaluate the trust calculation accuracy and efficiency of service selection, TMM uses WSDREAM dataset#2 [32]. This data set has a response time and throughput from 142 users on 4500 services in 64 different time slots. The format of the dataset is (User ID, Service ID, Time Slice ID, Response Time (sec), Throughput (kbps)). The main limitation of this dataset is that it

includes only two attributes namely response time and throughput. To assess the computational efficiency and scalability of TMM varying the number of QoS attributes, we have generated a synthetic dataset following the format of WSDREAM dataset with ten QoS attributes using faker package of Python. CUs give their feedback of QoS parameters for the opted service, and the SLA monitoring data is randomly generated from a certain range of values depending on the industry practice. All simulation experiments were performed and the results are validated using the WSDREAM dataset#2.

4.1 TMM Evaluation and Discussion

The performance of TMM is analyzed from two directions [36], accuracy and efficiency. A set of experiments under various conditions were performed to evaluate the accuracy and effectiveness of the proposed framework TMM. The principal aim of the implementation is to assess how accurately TMM recommends the top-k trusted services matching with CU request.

TMM analyzes the transactional behavior of both CUs and CSPs. The evaluation of OTE and STE algorithms are carried out within a fixed and consecutive time intervals. According to the SLA, a CSP assures a certain level of performance quality of its service. We consider the most relevant QoS parameters as per the literature [8], such as the availability, reliability, response time, security, privacy, and customer support, to evaluate objective trust. The OTE algorithm is evaluated based on measuring these parameters as per the SLA.

The model measures transaction trust value from the two aspects as explained in Sect. 3, objective trust and subjective trust. The inclusion of CU priority in the calculation of objective trust augments the performance of TMM. Dynamic weight is assigned to each QoS attribute monitored as per the satisfaction of the performance. As subjective trust depends on the trustworthiness of CUs, more priority is given to service monitor feedback. The parameter β fixes the relative significance of objective trust and subjective trust in final transaction trust computation. The value of β lies in the range, $0 \leq \beta \leq 1$. The ideal value of β is 0.3 at which trust evaluation has the highest accuracy as shown in Fig. 2a, and it indicates that inclusion of user opinion always improves the accuracy of trust

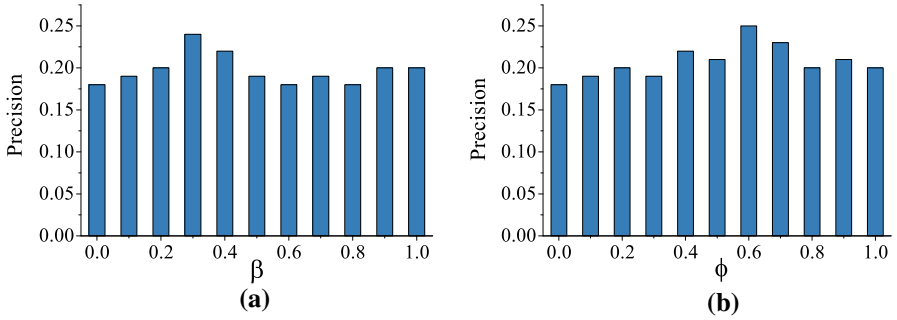

Fig. 2 Optimal value of regulating parameters. **a** Regulating parameter beta. **b** Regulating parameter Phi

calculation. This indicates that 70% priority is given to objective trust value and 30% to subjective trust value.

Subjective trust is evaluated based on the CU feedback. More priority is given to current feedback as the real assessment is reflected more in the present usage of service. The parameter ϕ determines the relative priority of current user feedback and previous feedback. The model decides the optimal value of the variable ϕ through repeated experiments. Figure 2b displays the various values of ϕ and observe that typical value is 0.6 giving optimal performance. Subjective trust evaluation shows better accuracy at this value. This indicates that 60% of priority is given to user current feedback and 40% to previous feedback. The optimal values for both regulating parameters are obtained by performing the trust calculation procedure using both WSDREAM dataset and synthetic dataset. Since, the values of both phi and beta are independent of the number of QoS arguments, the outcome was same.

Trust values are mapped into the interval [0,1]. The uncertainty of trust value increases as it approaches 0.5 and decreases as it reaches 0 or 1. The trust value closer to 0 or 1 is treated as a certain value. Otherwise, the trust value has more uncertainty and maximum uncertainty at 0.5 trust value. Therefore, we have defined three levels of trustworthiness of a CSP namely level 0 (trusted), level 1 (non-trusted), and level 2 (arbitrary) during the simulation. As mentioned earlier, SLA monitoring values were synthetically generated depending on the trust level of the selected CSP. The simulation is carried out with 100 fixed time slots. The interactions of a CU follows uniform distribution in [0,25] for each time slot. CUs give their feedback as a rating and is converted into the real interval [0,1]. We simulate 10,000 CU requests and out of which 80% trustworthy, 10% untrustworthy, and 10% arbitrary. We have opted 80% of CSPs as trustworthy because cloud computing is a business model and most of the CSPs are likely to be trustworthy.

Table 1 shows the percentage of CU interactions with the three types of CSPs. With trustworthy CSP the percentage of successful interaction will be more. This indicates the significance of finding a trustworthy CSP for the CU request. TMM identifies between level 0, levels 1 and 2 CSPs in the simulation. Also, with the use of algorithms OTE and STE, TMM maintains trust scores for each level of CSPs accurately. We have simulated the service selection with TMM and without TMM, for evaluating the rate of successful transaction. The results in Fig. 3 shows that there is a linearly increasing rate of successful interaction with the proposed middleware TMM because with an increase in time slots TMM gradually builds and maintains stable trust scores with trusted CSPs. This shows the effectiveness of our method.

Table 1 Interaction with CSP

% Interactions	Level 0	Level 1	Level 2
Successful	92	a = [0–49]	b = [25–75]
Uncertain	5	14	20
Unsuccessful	3	100-a-14	100-b-20

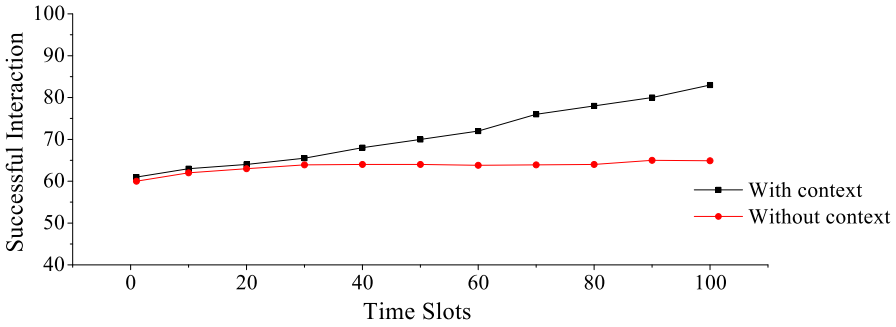

Fig. 3 Rate of successful interactions

Similarly, we classify CUs into three categories: honest, uncertain and non-honest. An honest CU is trustworthy and provides 90% of the time true and honest feedback. A non-honest CU is not trustworthy, and the fraction of true feedback and malicious feedback given by him falls in the range [0–50]. An uncertain CU is randomly categorized as honest and non-honest; the percentage of true and malicious feedback falls in the range[25–75]. For example, consider the number of transaction is 10. Then, an honest CU will give true feedback for nine transactions, and non-honest CU will give five true feedback. Uncertain CU will be randomly switching between honest and non-honest behavior. With this assumptions, our covariance based algorithm detects malicious CU feedback effectively as shown in Table 2 and FB stands for feedback.

During the simulation of TMM, we have calculated both objective trust and subjective trust of a transaction. Experiments were carried out with all levels of CSPs and introduced malicious rating to check the accuracy of TMM. Figures 4 and 5 show the results in detail. From the results shown in Fig. 4, it is clear that objective trust of trustworthy CSPs has a stable and high value and untrustworthy CSPs have low objective trust value. For arbitrary CSPs, the trust value oscillates between trustworthy and untrustworthy as per the real environment. The calculation of subjective trust as in Fig. 5, for trustworthy CSP always have a high subjective trust value, and untrustworthy CSP has very low trust value (<0.2). The results show the accuracy of TMM in handling all types of CSPs and CUs.

Service selection process is done by calculating the CTV of matching cloud services. The aggregate value (CTV) of trust enables TMM to rank similar cloud services. For example, while processing a CU request matching CSPs can be ranked

Table 2 CU feedback credibility

Type of CU	% True FB	% Malicious FB	% Malicious FB detected
Honest	90	10	90
Uncertain	[25–75]	[25–75]	70
non-honest	[0–50]	[0–50]	75

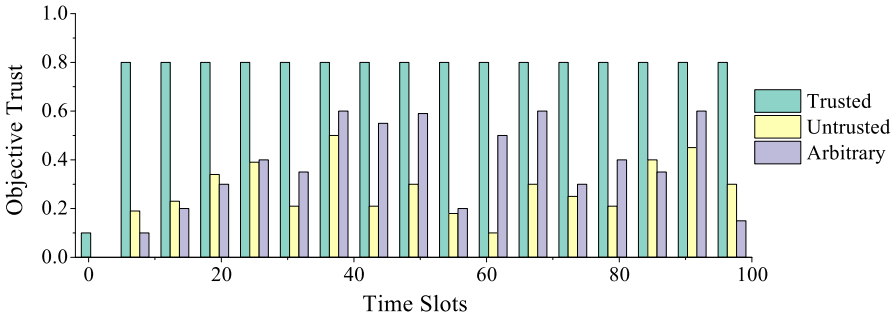

Fig. 4 Objective trust

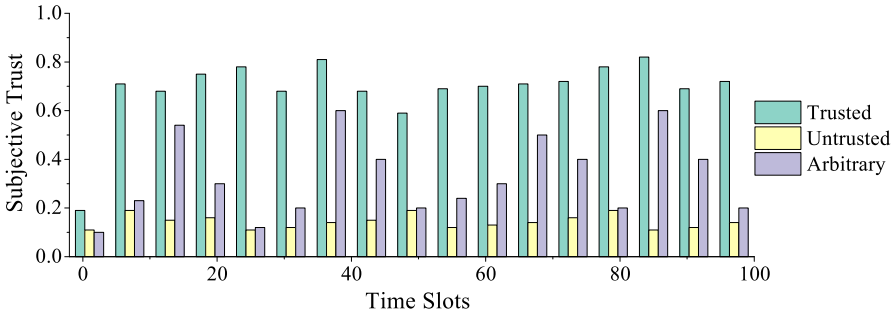

Fig. 5 Subjective trust

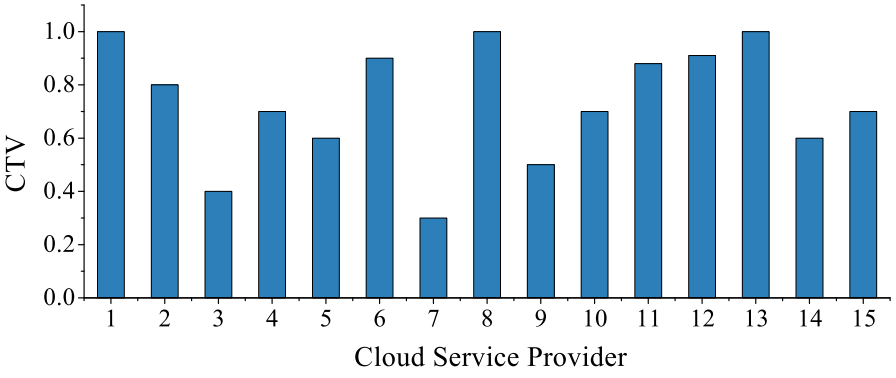

Fig. 6 Compound trust value (CTV)

using CTV, and the results for a particular request is shown in Fig. 6. TMM filtered fifteen CSPs offering the same instance of the requested cloud service, and CTV of each CSP is determined. Now, TMM will be recommending only those CSPs whose CTV is higher compared to candidate CSPs, and it is CSP1, CSP8, CSP13, CSP12, and CSP6 in the case of top-5 recommendation.

The accuracy of the service selection process with the inclusion of contextual information is compared to one without context, which is shown in Fig. 7. Here, context mainly focuses on the CU priority and dynamic weight assignment scheme. From each time slot, the algorithm picks only those alternatives which meet the minimum performance criteria as per the Eq. (11). The above selection considerably reduces the runtime of the algorithm.

Another set of experiments were carried out to determine the time efficiency and scalability of the proposed model TMM. The computation time of TMM is the sum of preprocessing of the user request, OTE, and STE. The complexity of OTE algorithm is $O(hm)$, where h denotes the number of priority classes and m denotes the number of QoS attributes. The worst case complexity is $O(m^2)$, where each QoS belong to a unique priority class. The complexity of STE is $O(Ku)$, where K denotes the number of time slots and u denotes the number of user feedback in each slot. We first measured the mean execution time varying the number of CU request profiles with fixed services (1000) and attributes (10). For each value, the experiment is carried out repeatedly fifty times, and the average is taken as the final value. Next, we carried out experiments for measuring mean execution time varying the number of cloud services with a fixed number of QoS attributes (10) and requests (500). These numbers are limited by the dataset used for the experiment. The results are shown in Fig. 8 which proves the scalability and time efficiency of TMM.

To evaluate the robustness of the proposed STE concerning malicious behaviors, TMM measured the percentage of honest rating and malicious rating correctly identified. Malicious users may give their feedback values with a mean value (of the Normal distribution) different from the honest users. Figure 9 shows the percentage of honest and malicious rating correctly identified using the proposed algorithm STE.

The above results show the accuracy and effectiveness of TMM as a middleware in performing trusted cloud service selection.

4.2 Performance Comparison

The proposed model TMM is compared with three well-known and similar frameworks mentioned in Sect. 2: (i) SLA based trust evaluation framework [31],

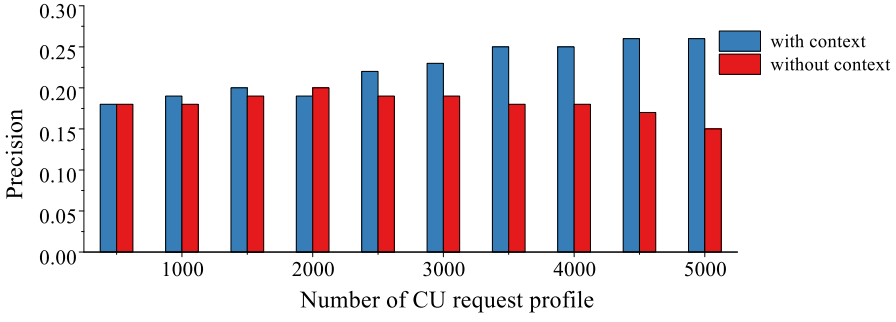

Fig. 7 Number of CU request profiles versus precision

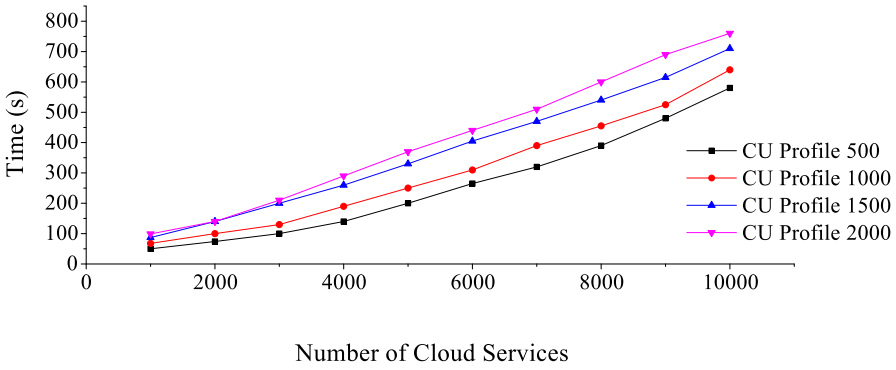

Fig. 8 Mean execution time of TMM varying with the number of CU profile and number of cloud services

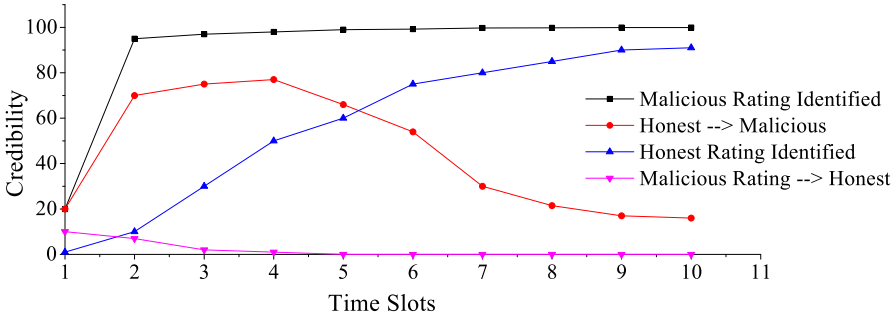

Fig. 9 Credibility of feedback identified

(ii) Adaptive trust management model using induced weighted averaging operator and rough set (AIOWA) [44], and (iii) QoS-based approach for finding the trust value of a service [46]. Paper [44] determines transaction trust by using two adaptive modeling tools, rough set and AIOWA operator. However, both methods assign weight function statically. Authors in [31] used SLA-aware model to determine the trust and reputation of cloud services. Trust is calculated based on QoS attributes such as availability (AV), reliability (RE), turnaround efficiency (TE), and data integrity (DI) in [46]. Author determines transaction trust value as:

$$QT = w_1 * AV + w_2 * RE + w_3 * DI + w_4 * TE \tag{13}$$

where w_i denotes weight assigned and $w_1 + w_2 + w_3 + w_4 = 1$.

The major difference is TMM employs dynamic weight assignment to QoS attributes based on CU context while calculating objective trust and the covariance method for checking user credibility. To compare the performance of proposed TMM with the well-known frameworks [31, 44, 46], experiment is carried out with $t = 1, \ldots, 25$ fixed time slots. CU requests are randomly generated using exponential distribution. For each timeslot, requests are generated in the range [5, 25]. The experiment is repeatedly carried out for both trustworthy and untrustworthy CSP.

The performance of TMM is compared using two significant metrics namely Mean Error Rate (MER) and Transaction Success Rate (TSR). MER is calculated as the root of squared error between calculated trust and Real Trust (RT), for the set of N CSPs.

$$MER = \frac{1}{N}\sqrt{\sum_{i=1}^{N}\left(RT - T_{SP_j}^{i}\right)} \tag{14}$$

Real trust is the ideal trust value which has the ideal performance level as per the SLA and is calculated for each transaction. In the literature, few works calculate the ideal trust value by prediction procedures using historical records and calculate the root mean square error using this value. But, we find the ideal trust by giving the maximal value for each QoS and the best feedback. The ideal trust value falls in the range [0.9,1]. We simulate both the level 0 and level 1 CSPs. In the case of a level 0 CSP, 90% of the total transactions are trusted and have successful interaction, whereas level 1 CSPs the percentage of successful interactions is [0, 49]. For arbitrary CSPs, the trustworthiness is not clearly defined and will be fluctuating between trustworthy CSP and untrustworthy CSP. Hence, we consider only level 0 and level 1 CSPs for the comparison. Figures 10 and 11 shows that TMM is more stable and has low error percentage in the final trust calculation compared with the other two methods.

In the case of trusted CSP, the error rate is $< 16\%$ for trusted cloud users and $< 25\%$ for untrusted cloud users. Trusted CSPs and trusted CUs are the perfect combinations which give optimum result. Each time slot shows a variation in MER due to the difference in user preference and network condition.

Figures 12 and 13 illustrates the variation in MER in the event of untrustworthy CSP considering both trusted and untrusted CUs. The difference in MER for trustworthy CSP and untrustworthy CSP shows that TMM differentiates between trusted and non-trusted CSPs effectively. Here, MER will be naturally high as untrustworthy CSPs are more in number. TMM outperforms the other methods with error percentage $< 25\%$ for trusted CUs and $< 29\%$ for untrusted CUs.

The performance of TMM against the transaction success rate as the number of malicious feedback increases is verified using TSR. TSR is defined as the ratio of the number of successful transactions over the total number of transactions in the

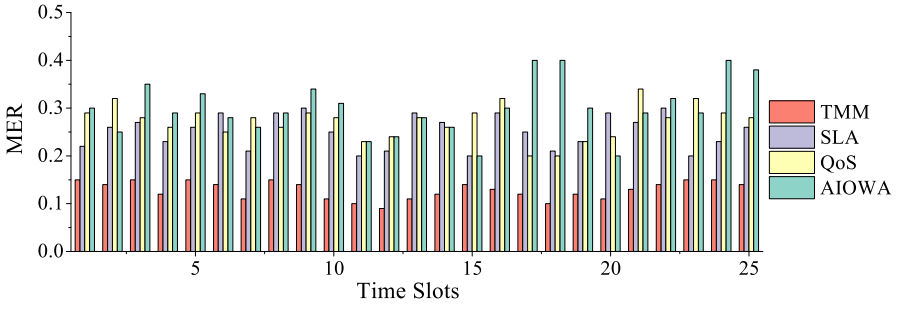

Fig. 10 MER with trustworthy CSP given trusted CU

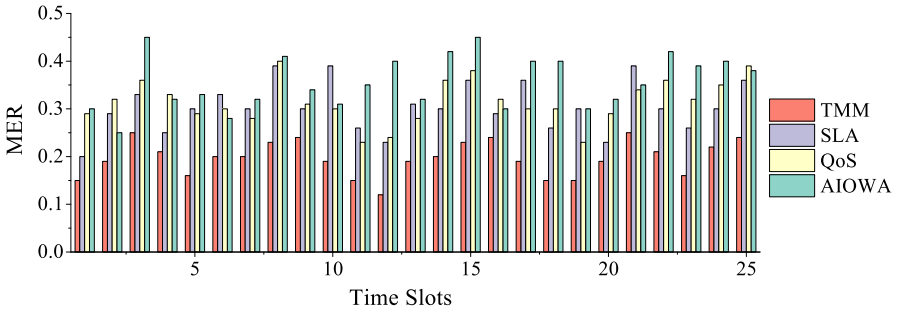

Fig. 11 MER with trustworthy CSP given untrusted CU

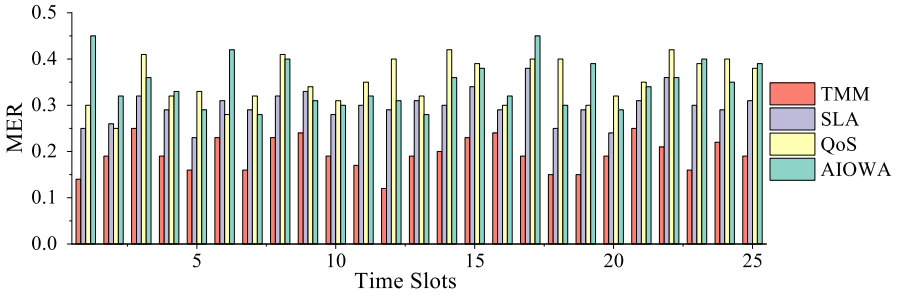

Fig. 12 MER with untrustworthy CSP given trusted CU

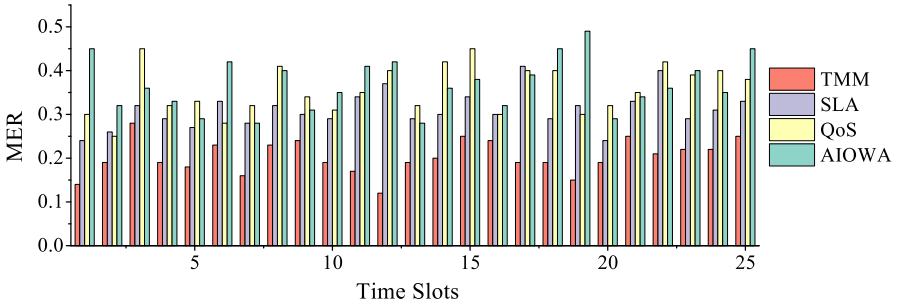

Fig. 13 MER with untrustworthy CSP given untrusted CU

period considered., which is formally defined in Eq. (15). Let COUNT is a function to determine the count of successful transactions in the period t, then TSR is defined as

$$TSR = \frac{COUNT(T_{succ})_t}{M_t} \tag{15}$$

where T_{succ} represent a successful transaction and M_t denotes the total number of transactions in the period t.

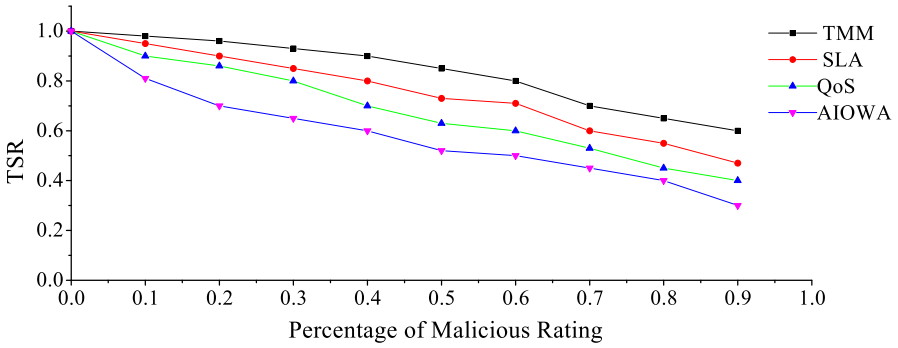

Fig. 14 Transaction success rate

Simulation results shown in Fig. 14 illustrates the efficiency of TMM compared to the existing works by achieving high TSR. As the percentage of malicious feedback increases to 90%, TSR of the proposed model is 60%, which is much better compared to the other two methods($< 40\%$). TMM achieves better performance as it effectively verifies user credibility using STE algorithm.

Simulation results show that TMM has the lowest MER and highest TSR compared with the other three methods as TMM assigns weight dynamically as per the functional performance of the cloud service. Hence, the proposed work TMM improves the accuracy of trust computation by using OTE and STE algorithms. From the above results, we can see that our method has a more stable performance compared to the other two methods. Also, the experimental results indicate that with adequate user interactions the quality of the recommendation is improved as compared with the other two methods. The performance can be further improved by considering both CU context and CSP context.

5 Conclusions and Future Work

This paper proposes a dynamic trust management middleware TMM, which is selecting trustworthy cloud services using the context of service delivery. Experimental results show that TMM yields result with better accuracy through the inclusion of contextual information. Two algorithms are proposed in this paper namely OTE to determine objective trust and STE to determine subjective trust. Mathematical formulations are provided to quantify the baseline values for both objective and subjective trust assessment. Algorithm OTE uses dynamic weight assignment with CU preferences. The credibility of a CU feedback is verified by user affinity ratio, which is computationally efficient. Moreover, the proposed TMM is compared with other analogous trust evaluation models. The experimental results indicate that TMM is more stable and has a low percentage of error in trust calculation compared with the other two frameworks.

In future, we need to enhance the performance of TMM by including both CSP and CU context for trust evaluation. Implementing and evaluating the proposed model in the multi-cloud environment is another direction of future research.

Acknowledgements This research work is supported in part by the Quality Improvement Programme of All India Council for Technical Education, India. The authors are very grateful to the editors and the anonymous referees for their detailed comments and suggestions regarding this paper.

References

1. Mell, P., Grance, T.: The NIST definition of cloud computing. Commun. ACM **53**(6), 50 (2011)
2. Ghanbari, S., Othman, M.: A priority based job scheduling algorithm in cloud computing. Proced. Eng. **50**, 778–785 (2012)
3. Pearson, S.: Privacy, security and trust in cloud computing. In: Privacy and Security for Cloud Computing, pp. 3–42. Springer, London (2013)
4. Kirthica, S., Sridhar, R.: Cit: a cloud inter-operation toolkit to enhance elasticity and tolerate shut down of external clouds. J. Netw. Comput. Appl. **85**, 32–46 (2017)
5. Somu, N., Kirthivasan, K., Shankar, V.S.: A rough set-based hypergraph trust measure parameter selection technique for cloud service selection. J. Supercomput. **73**(10), 1–25 (2017)
6. Noor, T.H., Sheng, Q.Z.: Web service-based trust management in cloud environments. In: Advanced Web Services, pp. 101–120. Springer, New York (2014)
7. Lie, Q., Wang, Y., Orgun, M.A., Liu, L., Liu, H., Bouguettaya, A.: Cccloud: Context-aware and credible cloud service selection based on subjective assessment and objective assessment. IEEE Trans. Serv. Comput. **8**(3), 369–383 (2015)
8. Jayapriya, K., Mary, A.B.N., Rajesh, R.S.: Cloud service recommendation based on a correlated Qos ranking prediction. J. Netw. Syst. Manag. **24**, 1–28 (2016)
9. Duan, Q.: Cloud service performance evaluation: status, challenges, and opportunities-a survey from the system modeling perspective. Digit. Commun. Netw. **3**(2), 101–111 (2017)
10. Fan, W., Perros, H.: A reliability-based trust management mechanism for cloud services. In: 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1581–1586. IEEE, New York (2013)
11. Habib, S.M., Hauke, S., Ries, S., Mühlhäuser, M.: Trust as a facilitator in cloud computing: a survey. J. Cloud Comput. Adv. Syst. Appl. **1**(1), 1 (2012)
12. Noor, T.H., Sheng, Q.Z., Zeadally, S., Yu, J.: Trust management of services in cloud environments: obstacles and solutions. ACM Comput. Surv. (CSUR) **46**(1), 12 (2013)
13. Sun, L., Dong, H., Hussain, F.K., Hussain, O.K., Chang, E.: Cloud service selection: state-of-the-art and future research directions. J. Netw. Comput. Appl. **45**, 134–150 (2014)
14. Kumar, R.R., Mishra, S., Kumar, C.: Prioritizing the solution of cloud service selection using integrated MCDM methods under fuzzy environment. J. Supercomput. **73**, 4652–4682 (2017)
15. Ding, S., Li, Y., Wu, D., Zhang, Y., Yang, S.: Time-aware cloud service recommendation using similarity-enhanced collaborative filtering and arima model. Decis. Support Syst. **107**, 103–115 (2018)
16. Lin, D., Squicciarini, A.C., Dondapati, V.N., Sundareswaran, S.: A cloud brokerage architecture for efficient cloud service selection. IEEE Trans. Serv. Comput. (2016). https://doi.org/10.1109/TSC.2016.2592903
17. Casalicchio, E., Cardellini, V., Interino, G., Palmirani, M.: Research challenges in legal-rule and qos-aware cloud service brokerage. Future Gener. Comput. Syst. **78**, 211–223 (2018)
18. Meng, Y., Huang, Z., Zhou, Y., Ke, C.: Privacy-aware cloud service selection approach based on p-spec policy models and privacy sensitivities. Future Gener. Comput. Syst. **86**, 1–11 (2018)
19. Tang, M., Dai, X., Liu, J., Chen, J.: Towards a trust evaluation middleware for cloud service selection. Future Gener. Comput. Syst. **74**, 302–312 (2017)
20. Jatoth, C., Gangadharan, G.R., Fiore, U.: Evaluating the efficiency of cloud services using modified data envelopment analysis and modified super-efficiency data envelopment analysis. Soft. Comput. **21**(23), 7221–7234 (2017)

21. Xiahou, J., Lin, F., Huang, Q.H., Zeng, W.: Multi-datacenter cloud storage service selection strategy based on ahp and backward cloud generator model. Neural Comput. Appl. **29**(1), 71–85 (2018)
22. Qian, H., Medhi, D., Trivedi, K.: A hierarchical model to evaluate quality of experience of online services hosted by cloud computing. In: 2011 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 105–112. IEEE, New York (2011)
23. Noor, T.H, Sheng, Q.Z, Ngu, A.H.H., Alfazi, A., Law, J.: Cloud armor: a platform for credibility-based trust management of cloud services. In: Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, pp. 2509–2512. ACM, New York (2013)
24. Yamato, Y.: Server selection, configuration and reconfiguration technology for IaaS cloud with multiple server types. J. Netw. Syst. Manag. **45**, 1–22 (2017)
25. Kumawat, S., Tomar, D.: Sla-aware trust model for cloud service deployment. Int. J. Comput. Appl. **90**(10), 10–15 (2014)
26. Yadav, N., Goraya, M.S.: Two-way ranking based service mapping in cloud environment. Future Gener. Comput. Syst. **81**, 53–66 (2018)
27. Yager, R.R.: Prioritized aggregation operators. Int. J. Approx. Reason. **48**(1), 263–274 (2008)
28. Huang, J., Nicol, D.M.: Trust mechanisms for cloud computing. J. Cloud Comput. Adv. Syst. Appl. **2**(1), 1 (2013)
29. Jin, B., Wang, Y., Liu, Z., Xue, J.: A trust model based on cloud model and bayesian networks. Proced. Environ. Sci. **11**, 452–459 (2011)
30. Wang, S., Sun, L., Sun, Q., Wei, J., Yang, F.: Reputation measurement of cloud services based on unstable feedback ratings. Int. J. Web Grid Serv. **11**(4), 362–376 (2015)
31. Macías, M., Guitart, J.: Analysis of a trust model for sla negotiation and enforcement in cloud markets. Future Gener. Comput. Syst. **55**, 460–472 (2016)
32. Zheng, Z., Xinmiao, W., Zhang, Y., Lyu, M.R., Wang, J.: Qos ranking prediction for cloud services. IEEE Trans. Parallel Distrib. Syst. **24**(6), 1213–1222 (2013)
33. Ding, S., Wang, Z., Wu, D., Olson, D.L: Utilizing customer satisfaction in ranking prediction for personalized cloud service selection. Decis. Sup. Syst 93: 1–10 (2017)
34. Sun, L., Ma, J., Zhang, Y., Dong, H., Hussain, F.K.: Cloud-fuser: fuzzy ontology and MCDM based cloud service selection. Future Gener. Comput. Syst. **57**, 42–55 (2016)
35. Kanwal, A., Masood, R., Shibli, M.A: Evaluation and establishment of trust in cloud federation. In: Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication, pp. 12. ACM, New York (2014)
36. Li, X., Ma, H., Zhou, F., Yao, W.: T-broker: a trust-aware service brokering scheme for multiple cloud collaborative services. IEEE Trans. Inf. Forensics Secur. **10**(7), 1402–1415 (2015)
37. Fan, W., Perros, H.: A novel trust management framework for multi-cloud environments based on trust service providers. Knowl.-Based Syst. **70**, 392–406 (2014)
38. Matin Chiregi and Nima Jafari Navimipour: A new method for trust and reputation evaluation in the cloud environments using the recommendations of opinion leaders' entities and removing the effect of troll entities. Comput. Hum. Behav. **60**, 280–292 (2016)
39. Dalmazo, B.L., Vilela, J.P., Curado, M.: Performance analysis of network traffic predictors in the cloud. J. Netw. Syst. Manage. **25**(2), 290–320 (2017)
40. Li, J., Squicciarini, A.C., Lin, D., Sundareswaran, S., Jia, C.: Mmbcloud-tree: authenticated index for verifiable cloud service selection. IEEE Trans. Dependable Secur. Comput. **14**(2), 185–198 (2017)
41. Al-Faifi, A.M. Song, B., Hassan, M.M., Alamri, A., Gumaei, A.: Performance prediction model for cloud service selection from smart data. Future Gener. Comput. Syst. **85**, 97–106 (2018)
42. Habib, S.M., Varadharajan, V., Mühlhäuser, M.: A framework for evaluating trust of service providers in cloud marketplaces. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, pp. 1963–1965. ACM, New York (2013)
43. Fan, W.-J., Yang, S.-L., Perros, H., Pei, J.: A multi-dimensional trust-aware cloud service selection mechanism based on evidential reasoning approach. Int. J. Autom. Comput. **12**(2), 208–219 (2015)
44. Li, X., Junping, D.: Adaptive and attribute-based trust model for service level agreement guarantee in cloud computing. IET Inf. Secur. **7**(1), 39–50 (2013)
45. Rajendran, V.V., Swamynathan, S.: Hybrid model for dynamic evaluation of trust in cloud services. Wireless Netw. 1–12 (2015)
46. Manuel, P.: A trust model of cloud computing based on quality of service. Ann. Oper. Res. **233**(1), 281–292 (2015)

47. Ghosh, N., Ghosh, S.K., Das, S.K.: Selcsp: a framework to facilitate selection of cloud service providers. IEEE Trans. Cloud Comput. **3**(1), 66–79 (2015)
48. Alhanahnah, M., Bertok, P., Tari, Z., Alouneh, S.: Context-aware multifaceted trust framework for evaluating trustworthiness of cloud providers. Future Gener. Comput. Syst. **79**, 488–499 (2018)
49. Crockford, D.: The application/json media type for javascript object notation (json) (2006), RFC 4627
50. Garg, S.K., Versteeg, S., Buyya, R.: Smicloud: a framework for comparing and ranking cloud services. In: 2011 Fourth IEEE International Conference on Utility and Cloud Computing (UCC), pp. 210–218. IEEE, New York (2011)
51. Li, Z., Liao, L., Leung, H., Li, B., Li, C.: Evaluating the credibility of cloud services. Comput. Electr. Eng. **58**, 161–175 (2016)
52. Mason, R.L., Tracy, N.D., Young, J.C.: Decomposition of t2 for multivariate control chart interpretation. J. Qual. Technol. **27**(2), 99–108 (1995)
53. Gonzalez, N., Miers, C., Redigolo, F., Simplicio, M., Carvalho, T., Näslund, M., Pourzandi, M.: A quantitative analysis of current security concerns and solutions for cloud computing. J. Cloud Comput. Adv. Syst. Appl. **1**(1), 1 (2012)
54. Krautheim, F.J., Phatak, D.S., Sherman, A.T.: Introducing the trusted virtual environment module: a new mechanism for rooting trust in cloud computing. In: International Conference on Trust and Trustworthy Computing, pp. 211–227. Springer, Berlin (2010)
55. Joreskog, K.G.: Structural analysis of covariance and correlation matrices. Psychometric society presidential address. Technical report, Research Report 78: 10, Department of Statistics, University of Uppsala, 1978

Mukalel Bhaskaran Smithamol has received B.Tech. from Governemnt Engineering College Thrissur and M.Tech. from National Institute of Technology Karnataka. She started her career as Assistant Professor in LBS College of Engineering Kasaragod (Govt. undertaking) and currently pursuing Ph.D. in Anna University Chennai. Her research interest include Cloud Computing, Data Structures, and Virtualization Technologies. She is a member of ACM, CSI and ISTE.

Sridhar Rajeswari has a Ph.D. in Computer science and Engineering. She is an Assistant Professor (Senior Grade) at the Department of Computer Science in College of Engineering, Guindy, Anna University. She has more than 60 publications in various journals and conferences. Her areas of interests include Language technologies, NLP, Information retrieval, Music Signal processing, Data structures and Algorithms, Cloud Computing, Big data and Compilers.