Int. J. Fuzzy Syst.
DOI 10.1007/s40815-015-0112-6

A New Trust Evaluation Algorithm Between Cloud Entities Based on Fuzzy Mathematics

Ali Mohsenzadeh[1] · Homayun Motameni[2] · Meng Joo Er[3]

Received: 31 January 2015 / Revised: 4 November 2015 / Accepted: 7 November 2015

Abstract High security of cloud computing is one of the most challenges to be addressed before the novel pas-as-you-go business paradigm is widely applied over the internet. Trust brings a novel means to improve the security and enable interoperability of current heterogeneous independent cloud platforms. However, there is no special trust evaluation model for cloud computing environment. Hence, this paper presents a new trust model based on fuzzy mathematics in cloud computing environment according to success and failure interaction between cloud entities based on the properties and semantics of trust. To compute trust in cloud systems, an algorithm based on proposed model is given. Simulation results show that the proposed model has some identification and containment capability in synergies cheating, promotes interaction between entities, and improves the performance of the entire cloud environment.

Keywords Cloud computing · Trust models · Fuzzy mathematics · Recommendation trust

✉ Ali Mohsenzadeh
ofu.mohsenzadeh@gmail.com;
A.mohsenzadeh@ustmb.ac.ir

Homayun Motameni
motameni@iausa.ac.ir

Meng Joo Er
emjer@ntu.edu.sg

[1] Department of Computer Engineering, Mazandaran University of Science and Technology, Babol, Iran

[2] Department of Computer, Islamic Azad University, Sari Branch, Sari, Iran

[3] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, Singapore

1 Introduction

Many of modem systems and applications such as pervasive computing, P2P networks, grid computing, and even web applications have employed the concepts of trust. Trust is a well-known social behavior; however, it is hard to have precise definition for it. The concept of trust in distributed computing environments comes from social science, such as psychology, economy, and sociology [1–3].

There are many researches in the literature which focus on trust. These works are generally placed under two main categories: trust measurement and trust management. The former deals with how to represent and evaluate the value of trust between two nodes, while the latter tries to find a way to make decision based on trust values. On the other point of view measuring (calculating), the trust may be centralized or decentralized. Centralized approach via a centralized authority or across multiple distributed participants often leads to simple solutions but if the centralized authority is not carefully designed, it can become a single point of failure for the entire system [4, 5]. Instead, most systems calculate trust in a fully distributed manner. Although these decentralized systems are inherently more complex, they scale well and avoid single points of failure in the system.

Several trust definitions have been given at psychology, economy, sociology, mathematics, etc. [3]. We have based our definition on Josang as the belief that an entity has about other entity, from past experiences, knowledge about the entity's nature and/or recommendations from trusted entities. This belief expresses an expectation on the entity's behavior, which implies a risk. Trust relationships are required to fulfill certain properties, such as reflexive, non-symmetrical conditionally transitive, and dynamic. In fact, trust is the most complex relationship among entities,

Published online: 10 December 2015

because it is extremely subjective, context-dependant, non-symmetric, uncertain, partially transitive, and difficult to evaluate and establish [6].

Today, there is no special trust evaluation model for cloud computing environment. Therefore, in order to improve the security of ultra-large-scale, heterogeneous, open, and totally virtualized cloud environments, a novel trust model based on evaluation similarity between entities for cloud computing environments is proposed. Our contributions can be summarized as follows:

- We present a trust model based on fuzzy mathematics in cloud computing environment according to direct and recommendation interaction between cloud entities so that the fuzzy direct trust relation is calculated based on direct experiences between trustor and trustee. Also, a trustor builds a fuzzy indirect trust relation with trustee through his acquaintances.
- The entity's trust value is not entirely consistent with the credibility of the recommendation. Therefore to resist malicious behavior, the global trust value for an entity computed by calculating the similarityweighted recommendations of the entities who have interacted with him according to adjusted cosine similar function.
- We have verified the trust and security of our proposed trust model by comparing our algorithm with others such as DMTC model [6] and give systematic analysis on how our proposed model can enhance the system trust. In [6], a trust evaluation model is studied. This paper discusses the integration of fuzziness and randomness of trust relation, analyzes the ways cloud models describe uncertain concepts and the cloud models transform algorithms between qualitative concepts and their quantitative expressions, and presents the direct and recommendation of trust between cloud entities based on cloud theory. The trust cloud model achieves a complete description of the concept of trust, and the trust values obtained in this model contain more semantic information. However, the evaluations of the model for all entities are equal, not matching the facts that in real life, to different people, the levels of trust are not the same. Because of this, the error of the trust information created by the model is large.

The rest of the paper is organized as follows. In Sect. 2, we present a trust model of choosing trusted entities based on the fuzzy relationship theory in fuzzy mathematics in cloud environment. We also calculate direct trust, indirect trust, recommendation trust, trust evaluation similarity, and total trust, respectively. Also, Sect. 3 describes algorithms on the basis of proposed trust model in detail. Experiment results in Sect. 4 show that proposed model can effectively prevent selfish entities. Finally, the summary and future work is presented in Sect. 5.

2 Proposed Trust Model

In general, trust can be classified into different categories according to different standards [7].

- According to attributes: identity trust and behavior trust
- According to obtaining way: direct trust and recommended trust
- According to role: code trust, third party trust and execution trust, etc.
- According to based theory: subjective trust and objective trust

In this paper, we use the second category for evaluating the trust.

Definition 1 (*Trust*) Trust is a level of subjective probability between two entities, a trustor (i.e., source entity) and a trustee (i.e., target entity), which is formed through the direct observation nature and/or recommendation from trusted entities, to fulfilling a particular service within a specific time and context [6, 8, 9].

It is supposed that the trustor is a cognitive entity with an ability to make assessments and decisions about the received information and past experiences. Trust is usually evaluated by trust degree and described with trust relation.

Definition 2 (*Trust degree*) Trust degree Td_{ij} is used to evaluate the degree of trust from a domain set of possible trust values that trustor e_i in views trustee e_j and denotes entity's e_i trust attitude (opinion) towards entity e_j in time t and context c_z. The trust degree can be expressed as the following relation:

$$Td_{ij} = \begin{cases} DT(e_i, e_j, c_z, t), \\ RT(e_i, e_j, c_z, t), \\ IDT(e_i, e_j, c_z, t), \\ \text{Otherwise} \end{cases} \quad (1)$$

where $Td_{ij} = DT(e_i, e_j, c_z, t)$, $Td_{ij} = RT(e_i, e_j, c_z, t)$, and $Td_{ij} = IDT(e_i, e_j, c_z, t)$ are the direct trust degree, recommendation trust degree, and indirect trust degree between trustor e_i (i.e., source entity) and trustee e_j (i.e., target entity) in context c_z and time t.

In real cloud environment, trust and reputation both depend on some context [10]. For example, entity A trusts entity B as multimedia provider, but it does not trust B as a storage provider. So in the context of requesting a multimedia service, B is trustworthy. But in the context of providing storage service, B is untrustworthy.

2.1 Fuzzy Mathematics

In real world, there are three types of objects. They are certainty, randomness, and fuzzy objects [11]. Certainty is proposition that is either true or false. The true value of this statement is 0 or 1. We may say this type of statement is crisp. But on the other hand, there are some statements you cannot make with such certainty. In the uncertainty object, there also have two types of concept, randomness and fuzzy object. The randomness object references to the chance of an event's result. Just because of our inadequate knowledge of the circumstances, an event may have many results, so the result is random.

There also have many fuzzy objects you cannot make with precision in real world. You may say some people are healthy or unhealthy, but there is not clear line between good health and bad health. There has a continue process from quantitative change to qualitative change between healthy and unhealthy. Trust and distrust just have the same fuzzy property as good health and bad health.

To cope with certainty object, the traditional set theory and binary logic theory can be used. The probability and statistic theory is used to model randomness object. The type of fuzzy situation is what fuzzy mathematics was developed to model. Fuzzy mathematics deals with propositions that can be true to a certain degree—somewhere from 0 to 1, sounds like a probability, but it is not quite the same. Probabilities for mutually exclusive events cannot add up to more than 1, but the fuzzy values may. Suppose that the probability of a cup of coffee being hot is 0.8 and the probability of the cup of coffee being cold is 0.2. These probabilities must add up to 1.0. Fuzzy values do not need to add up to 1.0. The truth value of a proposition that a cup of coffee is hot is 0.8. The truth value of a proposition that the cup of coffee is cold can be 0.5. [12, 13]

Fuzzy mathematics is very fit for describing trust and distrust and now we use fuzzy mathematics to build our trust model, so the mathematical model of fuzzy trust should be firstly created [14, 15].

Suppose $E = \{e_1, e_2,\ldots,e_n\}$ is the problem domain of fuzzy trust model, where $e_i (i = 1, 2, \ldots, n)$ is an entity in the problem domain [14, 15]. A membership function μ_e defines the degree to which a fuzzy variable e is a member of a set. μ_e map e into the interval [0, 1]. Full membership is represented by 1 and no membership by 0. The values between 0 and 1 characterize fuzzy members, which belong to the fuzzy set only partially [13, 16]. Supposing the problem domain E is not the empty set, TR is a fuzzy set of Cartesian product of $E \times E$; E is the set that includes all the entity in cloud environment. There exists a mapping:

$$\begin{aligned} &TR : E \times E \rightarrow [0,1], \\ &(e_i, e_j) \rightarrow \mu_A(e_i, e_j) \in [0,1] \end{aligned} \tag{2}$$

To manage a collection of trust related activities across domains, we need to understand trust itself. From different points of views, trust can be categorized into different classes [15]: direct trust and indirect trust (the indirect trust relation is a composite fuzzy relation of recommending relation and direct trust relation).

2.2 Fuzzy Direct Trust Relation

When we say entity e_i is trustworthy or untrustworthy for entity e_j, there is a trust relationship between entity e_i and entity e_j. If this statement is based on entity e_i's direct experiences with entity e_j completely, this relationship is called the direct trust relation or direct trust model. Figure 1 shows fuzzy direct trust degree between entity e_i and entity e_j at context c_z and time t.

Direct trust relation is not a crisp binary relation that is either true or false. Direct trust relation just has fuzzy properties. We can use fuzzy relation to describe direct trust relation. Fuzzy direct trust degree between two entities can be denoted by fuzzy graph.

Definition 3 (*Trust graph*) The trust relations in the cloud computing environment of entities are represented as a trust graph G. It represents a directed graph with entities as nodes and edges as trust relations among them. The edges are directed and if an edge indicates a trust relation of entity e_i towards entity e_j, it is directed from entity e_i to entity e_j with the trust degree Td_{ij}. A possible way of representing a directed graph is a matrix defined as follows [10, 14, 15].

Definition 4: (*Trust matrix*) Interactions in a cloud computing environment of N entities are represented with a trust matrix M, where elements Td_{ij} indicate a trust relation of entity e_i towards entity e_j, and have values, where each value denotes the degree of trust. If a relation is not defined, it is indicated as zero (Note that $Td_{ij} = 0$ does not imply $Td_{ji} = 0$). The matrix represents trust in a cloud computing environment at a specific time $t \in T$ and specific context c_z denoted $M(t,c_z)$ as

$$M(t,c_z) = \begin{pmatrix} Td_{11} & \cdots & Td_{1n} \\ \vdots & \ddots & \vdots \\ Td_{m1} & \cdots & Td_{mn} \end{pmatrix} \tag{3}$$

Based on the assumption that the trust relation is reflexive, it follows that all the diagonal elements in diagonal are equal to one which indicates the maximum trust degree.

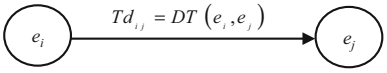

Fig. 1 Direct trust relation between e_i and e_j

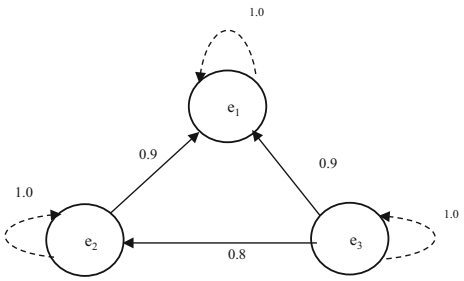

Fig. 2 Fuzzy graph

Figure 2 gives an example of trust propagation in a cloud computing environment of three entities at a specific time t and context c_z.

- $\frac{\mu_A(e_i,e_j)}{(e_i,e_j)}$ represents $Td_{ij} = DT(e_i,\ e_j,\ c_z,\ t)$, which is the trust degree between entity e_i and entity e_j at a specific time t and context c_z. $DT = \frac{1.0}{(e_1,e_1)} + \frac{0.9}{(e_2,e_1)} + \frac{0.9}{(e_3,e_1)} + \frac{1.0}{(e_2,e_2)} + \frac{0.8}{(e_3,e_2)} + \frac{1.0}{(e_3,e_3)}$
- Here $Td_{12} = Td_{13} = Td_{23} = 0$ that we do not write them out.
- The reflecting trust matrix $M_{DT}(t,c_z)$ is given below:

$$M_{DT}(t,\ c_z) = \begin{pmatrix} 1.0 & 0 & 0 \\ 0.9 & 1.0 & 0 \\ 0.9 & 0.8 & 1.0 \end{pmatrix}$$

When observing actual trust relationships TR between entities, the following properties will be assumed:

- TR is reflexive, an entity trusts itself implicitly, i.e., $\forall e \in E, eTRe$
- TR is not symmetric [7], because an entity may trust another entity within a specific context, while the opposite might not be true, i.e., $\exists\ e_i, e_j \in \mathrm{E};\ e_i$ TR $e_j \not\Leftrightarrow e_j$ TR e_i. It represents entity e_k recommend e_j to entity e_i at a specific time t and context c_z.
- TR is time-based variant [17], the trust degree that trustor ei in views trustee tj for specific context cz will decreases with the passage of time. As an entity behavior is not always constant but often changes with time, therefore, the recent experience is more credible than the general historical experience
- TR is context-dependant [10], a trustor ei may have different trust degree on trustee ej for different contexts

  If $\exists e_j,\ e_j \in E$, then $Td_{ij}\ (e_i,\ e_j,\ ck\,0,\ t)$ $= td \not\Leftrightarrow Td_{ij}\ (e_i,\ e_j,\ ck1,\ t) = td$ where $ck_0 \neq ck_1$

2.3 Fuzzy Direct Trust Degree Computing

Suppose in the past entity e_i has p times successful interactions and q times failure interactions with entity e_j at a specific time t and context c_z. We define the fuzzy direct trust relation membership function:

$$T_z d_{ij} = DT(e_i,\ e_j,\ c_z,\ t) = \frac{p_z}{p_z + q_z} \tag{4}$$

It is worth to mention again that, as an entity behavior is not always constant but often changes with time, therefore, the recent experience is more credible than the general historical experience. Therefore, we have considered the function to determine the successful experiences over time. This function calculates the successful interaction rate based on historical successful interaction between trustor e_i and trustee e_j at a specific time t and context c_z. This function is given below:

$$\begin{aligned} p(Tp_i) &= \alpha p(\Delta T_i) + (1-\alpha)p(T_{i-1}), \\ \Delta T_i &= Tp_i - Ts_{i-1}, \end{aligned} \tag{5}$$

where α is the adjustable parameter and presents the weight of successful interactions in different timescales (ΔT_i). $P\,(\Delta T_i)$ is recent successful interactions and $P(T_{i-1})$ is historical successful interaction. Moreover, T_p and T_s represent present time and start time, respectively. Also T_{s0} represents the first interaction between trustor e_i and trustee e_j at time t and context c_z.

We have considered the weights of the past negative behavior β which can be regulated to punish the selfish entity action. Then the fuzzy direct trust relation can be revised as

$$T_z d_{ij} = DT(e_i,\ e_j,\ c_z,\ t) = \frac{p_z(Tp_i)}{p_z(Tp) + \beta q_z} \tag{6}$$

It is difficult to decide whether an entity is bad or good based on only few interactions. In determining trust, it is important that an entity has sufficient experience on which to calculate trust [15, 17, 18]. So, we define the confidence level in the experience for a particular context c_z as an interaction threshold value co_z of interaction times.

Thus, if the interaction times are too small (i.e., $p_z + q_z <= co_z$) between trustor e_i and trustee e_j, this computing as defined in relation 4 may be an arbitrary decision and the following equation can be used.

$$T_z d_{ij} = DT(e_i,\ e_j,\ c_z,\ t) = 0.5 + \frac{p_z(Tp_i) - \beta q_z}{2 \times Co_z} \tag{7}$$

In general, fuzzy direct trust at context cz and time t between trustor ei and trustee ej is calculated as follows:

$$T_z d_{ij} = DT(e_i, e_j, c_z, t) = \begin{cases} 0.5 + \dfrac{p_z(Tp_i) - \beta q_z}{2 \times Co_z} & \text{if} \quad p_z + q_z \le Co_z \\ \dfrac{p_z(Tp_i)}{p_z(Tp_l) + \beta q_z} & \text{if} \quad p_z + q_z > Co_z \end{cases} \quad (8)$$

2.4 Fuzzy Indirect Trust Relation

Only one entity as trustor e_i always has limited direct interaction experiences with trustee e_j. If he wants to get a more accurate trust degree, a natural way for trustor e_i is to ask its acquaintances about their opinions at specific context c_z. Therefore, even trustor e_i has not any direct experience with trustee e_j in the past, trustor e_i can build a trust relation with trustee e_j through his acquaintances. We call the trust relation built by its acquaintances indirect trust relation, which is shown in Fig. 3. Actually an indirect trust relationship is built from recommendations by a trusted third party (i.e., acquaintances) or a chain of trusted partied, which create an indirect trust path, which has fuzzy properties. In other words, the indirect trust integrates the recommendation trust and direct trust model.

As shown in Fig. 3, entity e_k has directed interaction experiences with trustee e_j, so there has a direct trust relation between entity e_k and trustee e_j noted as DT_{kj}. There also has a recommending relation between entity e_k and trustor e_i. Entity e_k recommends its direct experiences to trustor e_i noted as RT_{ki}, and then these experiences become indirect experiences for trustor e_i noted as IDT_{ij}. Entity k has directed interaction experiences with entity j, and there has a direct trust relation between k and j noted as DT_{kj}. There also has a recommending relation between k and i. Entity k recommends its direct experiences to i, and then these experiences become indirect experiences for i, but maybe k is not a very familiar friend of entity i, or k has recommended i inaccurate experiences in the past, entity i does not think k's recommendation is completely right.

2.5 Fuzzy Recommendation Trust Relation

In Fig. 3 entity e_k recommends its direct experiences to trustee. But maybe entity e_k is not a very familiar friend of entity e_i, or e_k has recommended dishonest in the past, and

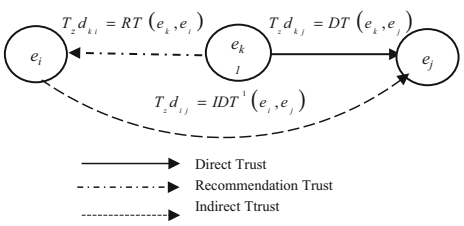

Fig. 3 One-level fuzzy indirect trust model

entity e_i does not think e_k's recommendation is completely right. Thus, the recommending relation also has fuzzy properties. We also can use fuzzy relation to describe the recommending relation.

The fuzzy relation membership function defines a degree of recommending relationship between entity e_i as trustor and entity e_k as recommender, which is similar to fuzzy direct trust relation membership function:

$$T_z d_{ki} = RT(e_k, e_i, c_z, t) = \begin{cases} 0.5 + \dfrac{r_z(Tr_i) - \beta s_z}{2 \times \mathrm{Co}_z} & \text{if} \quad r_z + s_z \le \mathrm{Co}_z \\ \dfrac{r_z(Tr_i)}{r_z(Tr_i) + \beta s_z} & \text{if} \quad r_z + s_z > \mathrm{Co}_z \end{cases}, \quad (9)$$

where r represents the number of successful recommendation interactions and s represents the number of failure recommendation interactions between entity e_j and entity e_k at a specific time t and context c_z.

Intuitively, it seems reasonable that the higher the trust value of the entity, the more important the recommendation view. However, the entity's trust value is not entirely consistent with the credibility of the recommendation. On the other hand, some malicious entity may exist in the system. In such cases, different types of attacks can be considered (such as bad-mouthing and on–off) [8, 19, 20]. In all attacks, malicious one tries to be keeping herself as a trusted entity using misleading actions or reputation. Parts of selfish entity through camouflage get the higher trust values, while they give the higher recommendations to their acquaintances, but those recommendations are obviously incredible. So, the credibility of the recommendation of an entity is different from that of itself, especially under some collective or disguised selfish entity. Therefore, every proposed model for trust must be able to consider these attacks and also should be able to verify the system against them. To resist malicious behavior, we put forward a trust model based on the fuzzy recommendation similarity in cloud environments, which aims at preventing the synergistic effect of selfish entity.

In this paper, we use the adjusted cosine similar function to determine the similarity between two entities in cloud environment [21]. In this case, similarity between two entity e_i an e_k is measured by computing the Pearson correlation. To calculate the correlation, we must first isolate the co-rated entities (i.e., entities that both two entity e_i and entity e_k has direct trust relation with them denoted $CE(e_i, e_k)$). In the case of using basic cosine measure (that is used in some p2p papers [22, 23]), the difference in rating scale between different entity is not taken into account. The adjusted cosine similarity offsets this drawback by subtracting the corresponding node average from each co-rated pair.

For any entity e_i and entity e_k, the similarity between entity e_i and entity e_k at time t and context c_z, denoted as Sim(e_i, e_k, c_z, t), is given by

$$\mathrm{Sim}(e_i,e_k,c_z,t) = \frac{\sum_{e_c\in CE(e_i,e_k)}(DT_{e_i,e_c}-\overline{DT_{e_i,e_c}})(DT_{e_k,e_c}-\overline{DT_{e_k,e_c}})}{\sqrt{\sum_{e_c\in CE(e_i,e_k)}(DT_{e_i,e_c}-\overline{DT_{e_i,e_c}})^2}\sqrt{\sum_{e_c\in CE(e_i,e_k)}(DT_{e_k,e_c}-\overline{DT_{e_k,e_c}})^2}} \tag{10}$$

Sim(e_i, e_k, c_z, t) describes the similarity of evaluation between entity e_i and entity e_k. $\overline{DT_{e_i,e_c}}$ and $\overline{DT_{e_k,e_c}}$ represent the average direct trust value between two entity e_i and e_k with co-rated pair $CE(e_i, e_k)$, respectively. Then, the fuzzy recommendation trust relation can be revised as

$$\begin{aligned} T_z d_{ki} &= RT(e_k,e_i,c_z,t) \\ &= RT(e_k,e_i,c_z,t)\times \mathrm{Sim}(e_i,e_k,c_z,t), \end{aligned} \tag{11}$$

where $RT(e_i, e_k, c_z, t)$ represents recommendation trust value between entity e_i and e_k at time t and context c_z and Sim(e_i, e_k, c_z, t) describes the similarity of evaluation between entity e_i and entity e_j.

2.6 Fuzzy Indirect Trust Degree Computing

As mentioned, the fuzzy indirect trust relation IDT_{ij} is a composite fuzzy relation of fuzzy recommending relation and fuzzy direct trust relation. In this paper, we have used min–max composition to composite fuzzy direct trust value and fuzzy recommendation value. Therefore, the fuzzy indirect trust relation for Fig. 3 is given by

$$\begin{aligned} T_z d_{ij} &= IDT^1(e_i,\ e_j,\ e_z,\ t) = RT\circ DT \\ &= RT(e_k,\ e_j,\ e_z,\ t)\circ DT(e_k,\ e_j,\ e_z,\ t) \\ &= \{\mathrm{Max}_{ek}\mathrm{Min}(RT(e_k,\ e_j,\ e_z,\ t),\ DT(e_k,\ e_j,\ e_z,\ t))\} \\ &= \vee_{ek\in E}(RT(e_k,\ e_j,\ e_z,\ t)\wedge DT(e_k,\ e_j,\ e_z,\ t)) \end{aligned} \tag{12}$$

In the above equation, we calculated one-level fuzzy indirect trust value which includes one-level recommendation based on Fig. 3. Figure 4 shows the two-level fuzzy indirect trust which includes two-level recommendations. In Fig. 4, entity e_{k2} has the direct interaction experiences with entity e_j, and there has a direct trust relationship with them. Entity e_{k2} recommends its direct experiences to e_{k1}, then entity e_{k1} recommends its indirect experiences to trustor e_i, and then these experiences become indirect experiences for trustor e_i. The two-level fuzzy indirect trust is computed as follows:

$$\begin{aligned} T_z d_{ij} &= IDT^2(e_i,\ e_j,\ e_z,\ t) \\ RT^2 &= RT(e_{k2},\ e_{k1},\ e_z,\ t)\circ RT(e_{k1},\ e_j,\ e_z,\ t) \\ &= \{\mathrm{Max}_{ek}\mathrm{Min}(RT(e_{k2},\ e_{k1},\ e_z,\ t),\ RT(e_{k1},\ e_j,\ e_z,\ t))\} \\ &= \vee_{ek1\in E}(RT(e_{k2},\ e_{K1},\ e_z,\ t)\wedge RT(e_{k1},\ e_j,\ e_z,\ t)) \end{aligned} \tag{13}$$

If entity e_i continues in this manner, there have three, four… n-levels indirect trust relation and it can get more and more accurate trust degree with entity e_j in context c_z.The multi-level composite fuzzy indirect trust is calculated as

$$\begin{aligned} T_z d_{ij} &= IDT^n(e_i,\ e_j,\ e_z,\ t) = RT\circ RT\circ\ldots\circ RT\circ DT \\ &= RT^n\circ DT \\ RT^n &= RT^{n-1}\circ RT, (n=1,\ 2,\ 3,\ \ldots) \end{aligned} \tag{14}$$

If there is some trust path between trustor ei and trustee ej, the indirect trust value between ei and ej calculates from the union of all indirect trust value in different path (one-level, two-level,…):

$$\begin{aligned} T_z d_{ij} &= IDT(e_i,\ e_j,\ e_z,\ t) = IDT^1\cup IDT^2\cup\ldots\cup IDT^n \\ &= \bigcup_{i=1}^{n} IDT^i \end{aligned} \tag{15}$$

Usually trustor e_i has not only direct interaction experiences with trustor e_j (in context c_z), but also indirect experiences from asking its acquaintances. Then there are two fuzzy trust relations (i.e., fuzzy direct trust relation and fuzzy indirect trust relation as shown in Fig. 5) between trustor e_i and trustor e_j. If trustor e_i wants to get more accurate trust value with trustor e_j, it must integrate the direct and indirect experiences. The fuzzy global trust relation is a union of fuzzy direct trust relation and indirect trust relation that is obtained from relation 16.

$$T_z d_{ij} = DT\cup IDT^1\cup IDT^2\cup\ldots\cup IDT^n = DT\cup\bigcup_{i=1}^{n} IDT^i \tag{16}$$

2.7 Total Trust Computing in all Context

After calculating the global trust value in each of the context between trustor e_i and trustee e_j, the trustor e_i needs to calculate the total trust value in all context. The total trust value will be combined closely with the value assignment of each evaluation context. The nature of

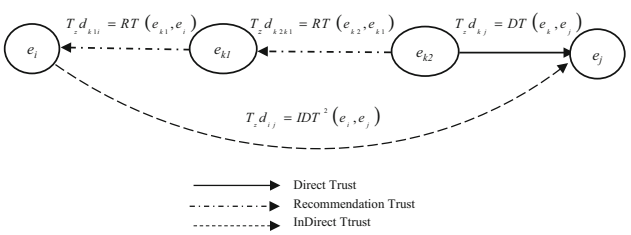

Fig. 4 Two-level fuzzy indirect trust model

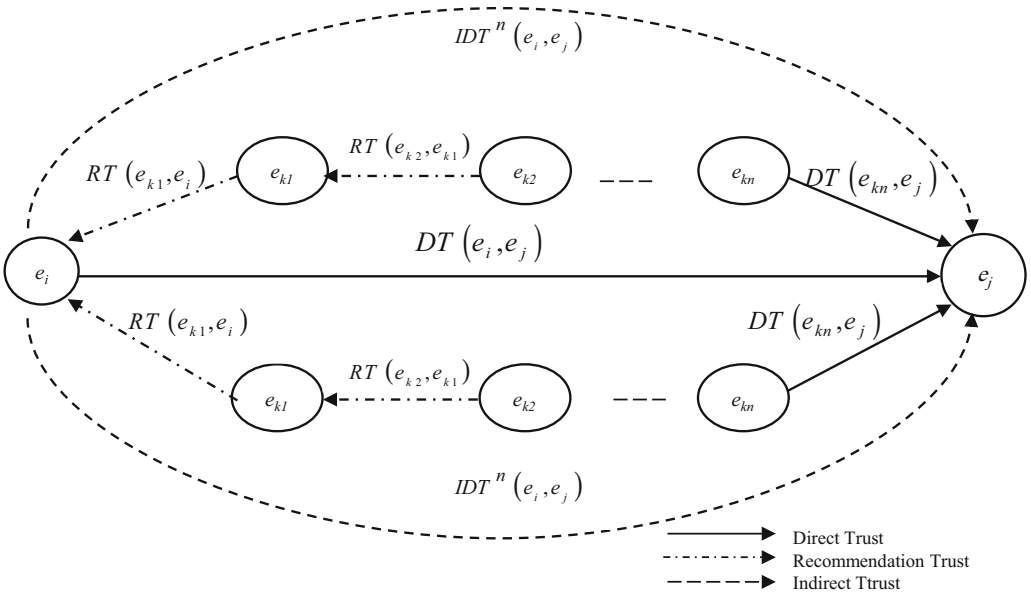

Fig. 5 Fuzzy global trust model

weight is shown in the quantity of different context on objects at different levels, i.e., the different influence from all contexts on the trust in view of trustor e_i. Suppose $w = (w_1, w_2,\ldots,w_n)$ is the weight of the context c_z such that w_i is in [0,1] for all i and $\sum_1^n w_i = 1$. So, the total trust Td_{ij} can be gotten by the following fuzzy mapping:

$$Td_{ij} = W_i(w_1,\ w_2,\ \ldots w_n) \circ \begin{matrix} c_1 \\ c_2 \\ \\ c_n \end{matrix} \begin{pmatrix} T_1d_{ij} \\ T_2d_{ij} \\ \vdots \\ T_nd_{ij} \end{pmatrix}^{T_zd_{ij}} = w_1 \times T_1d_{ij} + w_2 \times T_2d_{ij} + \ldots + w_n \times T_nd_{ij}, \quad (17)$$

where Td_{ij} represents the total trust value in all context that trustor e_i interacts with trustee e_j and t_zd_{ij} (t_1d_{ij}, t_2d_{ij},…,-t_nd_{ij}) represents the global trust value in each of the context between trustor e_i and trustee e_j.

3 The Algorithm Design

According to the analysis above, the core of our proposed algorithm is divided into three parts, the direct trust algorithm, the recommendation trust algorithm, and the indirect trust algorithm.

3.1 Direct Trust Algorithm

The core part of the direct trust algorithm is described as Algorithm 1. As shown in Algorithm 1, the inputs of the direct trust algorithm are the number of direct interactions (i.e., successful interactions p and failure interactions q) in each context and all time between two entities, e_i and e_j; the weight of successful interactions α in different timescales; the weights of the past negative behavior β; the timescale ΔT to calculate the successful interaction rate based on historical successful interaction and threshold value co_z for direct trust relation. Also the initial direct trust value is equal to one when there is no interaction between entities. Steps 6 to 9 show, if the input entities are the same then, the number of successful and failure interactions is equal to zero. Therefore, the direct trust value will be equal to one. But if the two entities are not the same, then, steps 11 to 15 calculate the sum of successful and failure interactions. Step17 calculates the number of timescales c according to ΔT. After that, steps 18 to 29 calculate the successful interaction rate based on historical successful interaction in each ΔT. Finally, steps 30 to 35 calculate the direct trust value between entity e_i and entity e_j in context c_z and time t (according to relation 6 and 7).

Algorithm1. Direct Trust (e_i,e_i)

Input: a trustor e_i ; a trustee e_j ; number of direct transaction (Tp & Tq) in each context and all time between e_i,e_j ; ΔT ; α ; β ; co_z
Output: direct trust value between e_i,e_j
begin
  for each context. c_z **do**
  begin
    if e_i=e_j **then**
      $P_z[e_i,e_j]$=0 // number of successful interactions
      $q_z[e_i,e_j]$=0 // number of failure interactions
      $DT[e_i,e_j,c_z,t]$=1 // Direct trust value
    else
      for t=1 to present time.t **do**
      begin
        $P_z[e_i,e_j]=T_p[e_i,e_j, c_z,t]+P_z[e_i,e_j]$ // $T_p[e_i,e_j, c_z,t]$ is number of successful interactions at time t
        $q_z[e_i,e_j]=T_q[e_i,e_j, c_z,t]+q_z[e_i,e_j]$ // $T_q[e_i,e_j, c_z,t]$ is number of failure interactions at time t and context c_z
      end for
      $t=t_p$ // $t_p=t_{present}$
      Calculate the number of timescales c according to ΔT
      t_{pc} = t
      while c>= 0 **do**
      begin
        $ts_{c-1}=tp_c-\Delta T$ // $t_s=t_{start}$
        $tp_{c-1}=ts_{c-1}$ // Ts_0 represents the first interaction between trustor and trustee at time t and context c_z.
        for t=ts_{c-1} to tp_c **do**
        begin
          $P(\Delta T_c)z[e_i,e_j]=Tp[e_i,e_j,c_z,t]+P(\Delta T_c)z[e_i,e_j]$
        end for
        Calculate the successful interaction rate based on historical successful interaction by relation 5.
        c --
      end while
      if $p_z[e_i,e_j]+q_z[e_i,e_j] <= co_z$ **then**
        Calculate Direct trust value by relation (7)
      else
        Calculate Direct trust value by relation (6)
      end if
    end if
  end for
end

3.2 Recommendation Trust Algorithm

As shown in Algorithm 2, recommendation trust algorithm is similar to direct trust algorithm but in this algorithm, we have considered similarity of evaluation between entity e_i and entity e_j. Also, the inputs of the recommendation trust algorithm are the number of successful recommendation interactions that is shown T_r and the number of failure recommendation interactions that is shown T_s. Other inputs include the weight of successful recommendation interactions α in different timescales; the weights of the past negative behavior β for recommendation interactions; the timescale ΔT to calculate the successful recommendation interaction rate based on historical successful recommendation interaction and threshold value co_z for recommendation trust relation.

Algorithm2. Recommendation Trust (e_i,e_j)

```
1:  Input: a trustor e_i; a trustee e_j; number of recommendation transaction(Tr & Ts) in each context and all time
    between e_i,e_j; ΔT ; α ; β ;
2:  Output: recommendation trust value between e_i,e_j
3:  begin
4:    for each context.c_z do
5:    Begin
6:      if e_i=e_j then
7:        r_z[e_i,e_j]=0
8:        s_z[e_i,e_j]=0
9:        for all time.t do
10:       begin
11:         RT[e_i,e_j,c_z,t]=1
12:       end for
13:     else
14:       for t=1 to present time.t do
15:       Begin
16:         r_z[e_i,e_j]=T_r[e_i,e_j,c_z,t]+r_z[e_i,e_j]
17:         s_z[e_i,e_j]=T_s[e_i,e_j,c_z,t]+s_z[e_i,e_j]
18:       end for
19:       t=t_p
20:      Calculate the number of timescales c according to ΔT
21:       t_pc = t
22:       while c>= 0 do
23:       begin
24:         ts_c-1=tp_c- ΔT
25:         tp_c-1=ts_c-1
26:         for t=ts_c-1 to tp_c do
27:         begin
28:           P( ΔT_c )z[e_i,e_j]=Tp[e_i,e_j,c_z,t]+ P( ΔT_c )z[e_i,e_j]
29:         end for
30:       Calculate the recommendation successful interaction rate based on historical successful interaction
          by relation 5.
31:         c --
32:       end while
33:      Calculate Recommendation trust value according to relation (11) // ( i.e relation (9) × relation (10))
34:     end if
35:   end for
36: end
```

3.3 Indirect Trust Algorithm

As shown in Algorithm 3, the inputs of indirect trust algorithm are the number of trust chain and the number of recommenders in each trust chain. In this algorithm, indirect trust value is calculated for all trust chain (or trust path) in each context c_z between trustor e_i and trustee e_j because may exist several indirect trust chain between them. Steps 9–10 shows that direct trust relation between two entities should be calculated between trustor e_i and trustee e_j if there is no recommender between them.

Algorithm3. Indirect Trust (e_i,e_j)

```
1:  Input: a trustor e_i ; atrustee e_j; number of trust chain and number of recommenders in each trust chain
2:  Output: indirect trust value between e_i,e_j
3:  begin
4:    for each context.c_z do
5:    begin
6:      for each trust chin.j do
7:      begin
8:       if number of recommender == 0 then
9:          Direct Trust (e_i,e_j)
10:      else
11:      Calculate indirect trust value in each trust chain between ei and ej by relation (15)
12:      end if
13:         Calculate indirect trust value in all trust chain between ei and ej by relation (16)
14:     end for
15:   end for
16: end
```

4 Results and Discussions

In order to evaluate the performance of the proposed model in this paper, simulation environment and parameters set are firstly discussed in this section, and then precise performance evaluation results are given.

4.1 Simulation and Configuration Parameter

The platform of simulation environment is CloudSim toolkit [24] which is a simulation platform based on Java, which supports modeling and simulation of large-scale cloud computing data centers. Therefore, it is feasible to simulate our proposed model of cloud computing environments by CloudSim. We create ten data centers in the simulation environment (the detailed configuration is shown in Table 1). We set 500 virtual machines. Moreover, we submit 1000 tasks to the 500 virtual machines. Moreover, we submit 1000 tasks to the 500 virtual machines, each task is submitted according to Poisson distribution after its previous task, and the length of each task is considered as a random number within the range of [10,000, 20,000] MI. Also recommenders are divided into three types:

1. Virtuous recommenders who provide honest service and recommendation
2. Random recommenders who provide random service and recommendation
3. Malicious recommenders who provide malicious service and recommendation.

Tables 1 and 2 show the main parameters used in this set of experiments.

4.2 Results for the proposed method

In this section, we discuss the effect of the proposed trust model, according to the parameters of Table 2. We set the number of timescale 2 (i.e., ΔT_1 and ΔT_2) and then we evaluate the successful interactions rate $p_z(Tp_i)$ at time t and context c_z. Figure 6 shows the effect of different number of successful transactions in ΔT_1 and ΔT_2 when $\alpha = 0.7$. As shown in Fig. 6, there is a direct relationship between the number of successful interactions and the interactions time.

Then, we considered the effect of different $P(\Delta T_i)$ in direct trust value. We set p and q to 1, 5, …, 50 separately. The results are shown in Figs. 7, 8, and 9. With the increase of failure interactions q, the direct trust value will decrease. Also, a significant difference in direct trust value of $p = 20$–$p = 25$ is observed, and this is because the threshold value is considered to 25 (in other words $\mathrm{Co}_z = p + q = 25$). Thus, when p set to 25, with the increase of q, the number of interaction will be more than the threshold value. This indicates that the entities have sufficient experience.

Table 1 Detailed configuration of data centers

Data center ID	Machine number	PE per machine	Processing ability (MIPS)	Architecture and OS
Data center 0–1	10	1–4	100–200	X86/Linux
Data center 2–3	20	1–4	200–400	X86/Linux
Data center 4–5	30	4–8	200–600	X86/Solaris
Data center 6–7	40	4–16	400–600	X86/Solaris
Data center 8–9	50	8–16	400–800	X86/Solaris

Table 2 Configuration parameters

Direct trust relation	
0.7	α: the weight of successful direct interactions in different timescales
1.1	β: the weights of the past negative behavior for direct interactions
30	ΔT: the timescales determine the number of successful interactions
25	Co_z: the threshold value for direct trust relation
1	The initial direct trust when there is no interaction between entities
Recommendation trust relation	
0.6	α: the weight of successful recommendation interactions in different timescales
1.1	β: the weights of the past negative behavior for recommendation interactions
30	ΔT: the timescales determine the number of successful recommendation interactions
40	Co_z: the threshold value of the recommending times

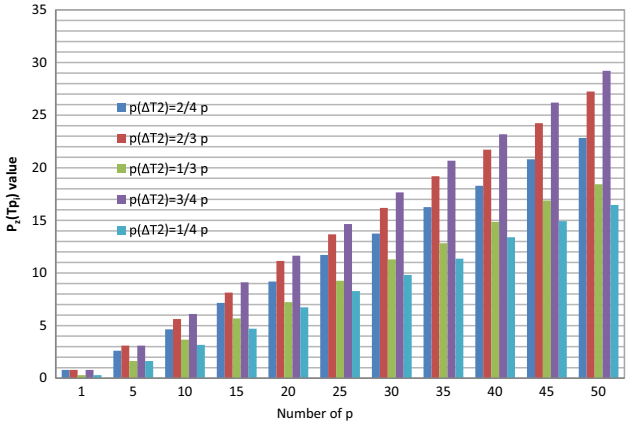

Fig. 6 Successful interactions rate $p_z(Tp_i)$ in two interval time ΔT_1, ΔT_2 and $\alpha = 0.7$

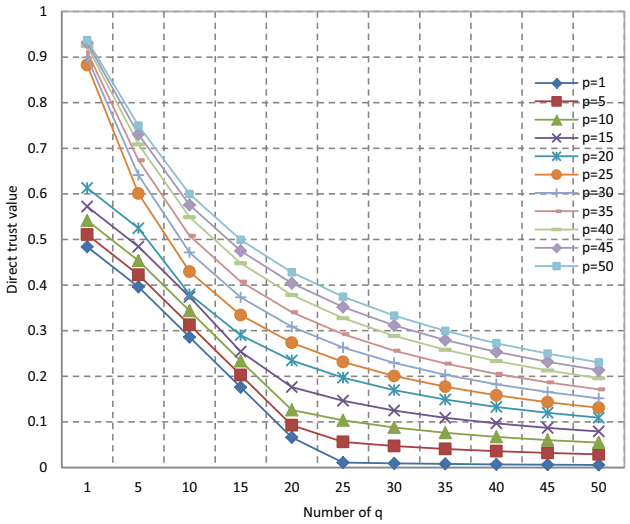

Fig. 7 Direct trust value when $p(\Delta T_2) = 1/4\ p$

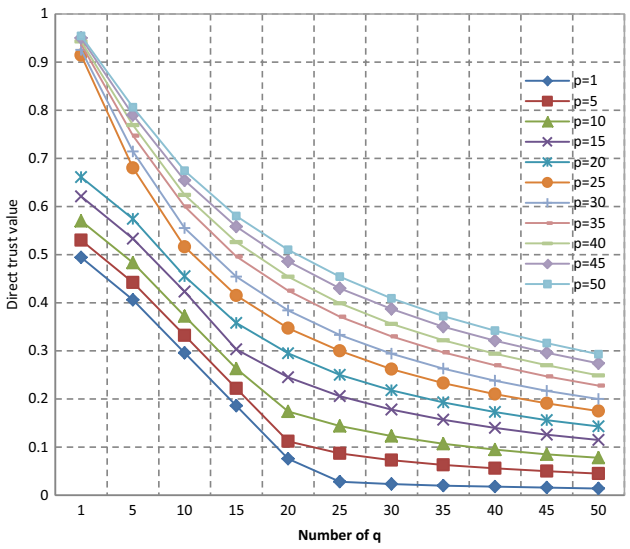

Fig. 8 Direct trust value when $p(\Delta T_2) = 2/4\ p$

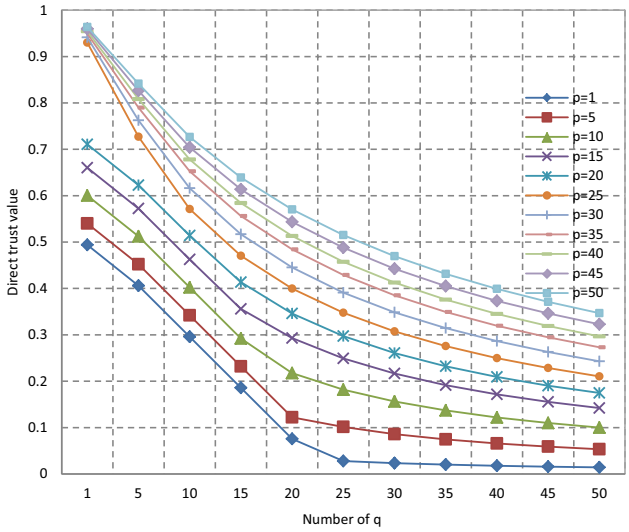

Fig. 9 Direct trust value when $p(\Delta T_2) = 3/4\ p$

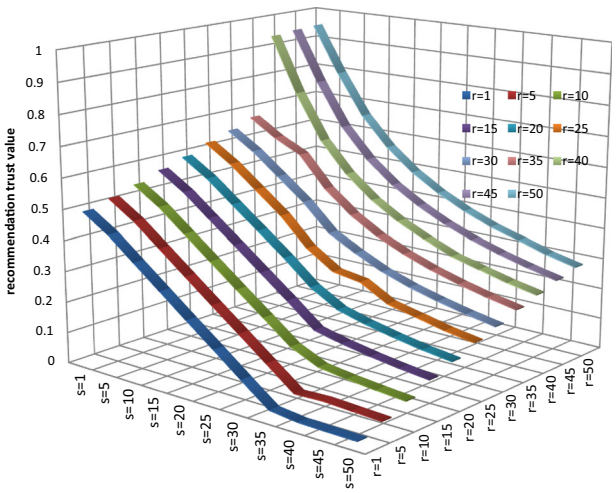

Fig. 10 Recommendation trust value without similarity between trustor e_i and trustee e_j

We also considered the effect of different similarity for evaluating the recommendation trust value. According to the parameters of Table 2, we set $\beta = 1.1$, $\alpha = 0.6$ and $Co_z = 40$. Also we set r and s to 1, 5, …, 50 separately. Figure 10 shows recommendation trust value without similarity between two entities. Figures 11 and 12 show recommendation trust value for similarity 0.5 and 0.8 separately. The result of simulation can be summarized as follows: when considering the credibility of the recommendations, an entity trusts more on the entities whose rating opinions are similar to him rather than those with high global trust values. That is, the higher degree of similarity entity e_i and entity e_j, the more consistent evaluation of the trust between entity e_i and entity e_j to other nodes.

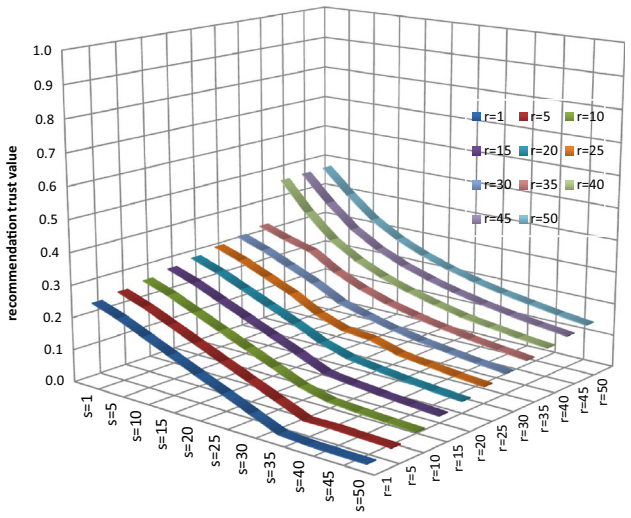

Fig. 11 Recommendation trust value with sim(e_i, e_j) = 0.5

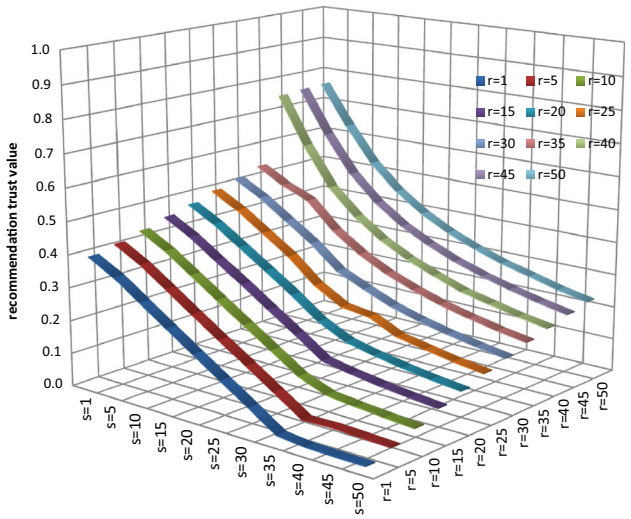

Fig. 12 Recommendation trust value with sim(e_i, e_j) = 0.8

4.3 Comparison among Trust Models

An important application of the proposed trust analysis is to facilitate comparison among different trust establishment methods. There are some trust schemes proposed for cloud environment, so, it is difficult to list all the trust models to compare with each other. In the Section, we make a comparison with two trust models, i.e., DMTC model [6] and proposed model without similarity.

4.4 Trust Accuracy Rate

We use absolute error metrics for evaluating the accuracy. Absolute error is the difference between the actual value of trust for an edge and the calculated value from a method [25].

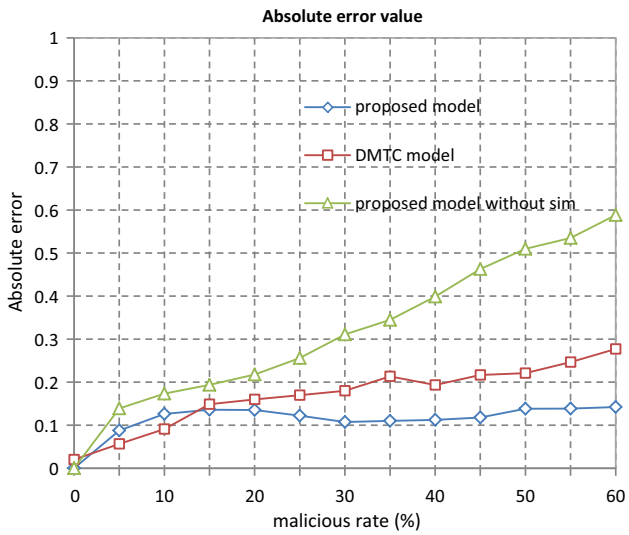

Fig. 13 Absolute error value

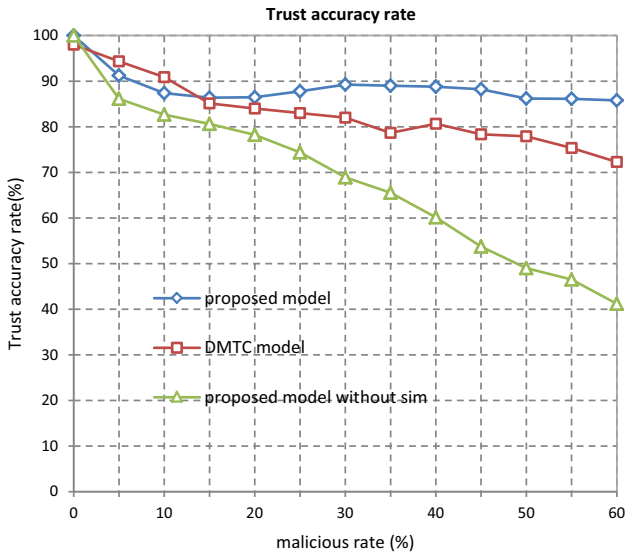

Fig. 14 Trust accuracy rate

$$\text{Absolute error} = |\text{trust calculated} - \text{actual trust}| \tag{18}$$

In the above equation, trust calculated is the trust value that is calculated by the proposed method and actual trust is the union of global trust value of all entities that interact with trustee. Therefore, the trust accuracy rate was calculated according to relation 19. The results are shown in Figs. 13 and 14

$$\text{Trust accuracy rate} = (1 - \text{Absolute error}) \times 100 \tag{19}$$

As shown in Fig. 14, in the first simulation time, when there is no interaction between entities, we set direct trust equal to one. Therefore, absolute error is set one. The

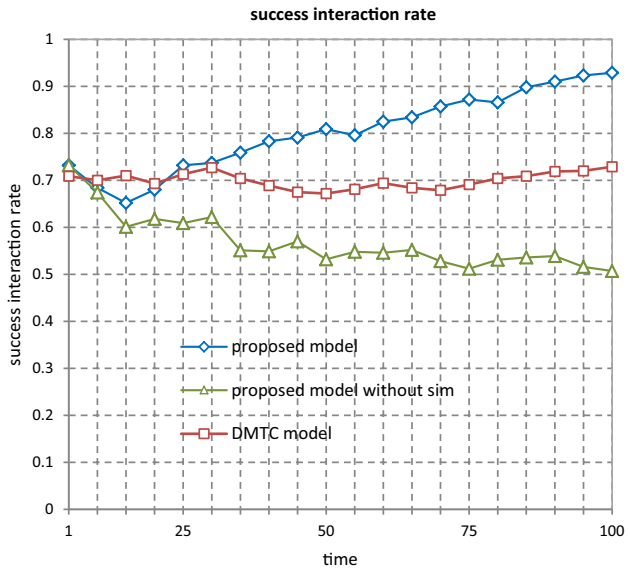

Fig. 15 Success interaction rate

success interaction rate declines with malicious interactions at the beginning. After a time, the success interaction rate keeps rising. Also, with the increase of the malicious rate, the proposed model can ensure trust accuracy rate in a high level.

4.5 Success Interaction Rate

The good entities can be differentiated from the misbehavior entities by their trust values after a few interactions. At the beginning, all entities have the same initial trust value, trustors randomly select an entity, and after a few numbers of interactions, the normal entities can get the higher trust value than the other selfish entities. With a help of the trust computing based on proposed model, we can identify the malicious entities efficiently. We can restrict the interaction of malicious entities further. It can help increase the success interaction rate of the system.

Success interaction rate is the ratio of successful interactions to overall interactions in the simulation time, which is denoted as

$$\text{success interaction rate} = \frac{p}{p+q} \tag{20}$$

The experiment results are shown in Fig. 15. Results show that the success interaction rate with proposed model is higher than DMTC model. From Fig. 15, we can see that the changing of success interaction rate is divided into two stages: decline stage and rise stage. The success interaction rate declines with malicious interactions at the beginning. After a time, the success interaction rate keeps rising. It is because that the system with trust computing has begun to identify the malicious entities and refuse to provide service for them.

5 Conclusion and Future Work

This paper presents a trust evaluation model based on fuzzy mathematics in cloud computing environment according to direct and recommendation interactions between cloud entities. The entity's trust value is not entirely consistent with the credibility of the recommendation. Therefore to resist malicious behavior, the global trust value for an entity computed by calculating the similarity-weighted recommendations of the entities who have interacted with him according to adjusted cosine similar function.

Also, we have verified the trust and security of our proposed model by comparing our algorithm with others, and give systematic analysis on how our proposed model can enhance the system trust. Experimental results demonstrate that the proposed approach is well comparable with the well-known techniques, and in some cases, superior performances are achieved. Simulation experiments show that it can effectively identify malicious entities, and provide reliable information to correctly make the security decisions for the system. Also, the proposed model has some identification and containment capability in synergies cheating, promotes interaction between entities, and improves the performance of the entire cloud environment.

This paper discusses only one type of situation when selfish entities attack. In the future, other types of entities attack as well as the possible impacts on proposed model will be extended. In addition, we plan to develop a complete trust management framework based on our model. We also will offer scheduling algorithm according to proposed model for cloud computing.

References

1. Huang, J., Nicol, D.M.: Trust mechanisms for cloud computing. J. Cloud Comput.: Syst. Appl. **2**(9), 1–14 (2013)
2. Habib, S.M., Hauke, S., Ries, S., Muhlhauser, M.: Trust as a facilitator in cloud computing: a survey. J. Cloud Comput.: Adv., Syst. Appl. **1**(19), 1–19 (2012)
3. Josang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decis. Support Syst. **43**(2), 618–644 (2007)
4. Khorshed, M.D.T., Shawkat Ali, A.B.M., Wasimi, S.A.: A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing. J. Future Gener. Comput. Syst. **28**(6), 833–851 (2012)
5. Sun, D., Chang, G., Sun, L., Wang, X.W.: Surveying and analyzing security, privacy and trust issues in cloud computing environments. J. Proced. Eng. **15**, 2852–2856 (2011)
6. Sun, D., Chang, G., Sun, L., Li, F., Wang, W.: A dynamic multidimensional trust evaluation model to enhance security of cloud computing environments. Int. J. Innov. Comput. Appl. **3**(4), 200–212 (2011)

7. Li, W., Ping, L.: Trust Model to Enhance Security and Interoperability of Cloud Environment, vol. 5931, pp. 69–79. Springer, New York (2009)
8. Bidgoly, AJ., TorK Ladani, B.: Trust Modeling and Verification Using Colored Petri Nets. 2011 8th International ISC Conference on Information Security and Cryptology (ISCISC), September 14–15, pp. 1–8 (2011)
9. Kovac, D., Trcek, D.: Qualitative trust modeling in SOA. J. Syst. Archit. **55**(4), 255–263 (2009)
10. Liao, H., Wang, Q., Li, G.: A Fuzzy Logic-based Trust Model in Grid. International Conference on Networks Security, Wireless Communications and Trusted Computing. pp. 608–614 (2009)
11. Yang, L., Gao, Y.: The Theory and Application of Fuzzy Mathematics, 3rd edn. The Press of South China University of Technology, Guangzhou (2001)
12. Rao, V.B.: C ++ Neural Networks and Fuzzy Logic. M&T Books, IDG Books Worldwide, Inc., Foster City (1995)
13. Piegat, A., Landowski, M.: Horizontal membership function and examples of its applications. Int. J. Fuzzy Syst. **17**(1), 22–30 (2015)
14. Azzedin, F., Ridha, A., Rizvi, A.: Fuzzy trust for peer-to-peer based systems. Proceedings of World Academy of Science, Engineering and Technology PWASET. pp. 123–127 (2007)
15. Griffiths, N., Chao, K.M., Younas, M.: Fuzzy trust for peer-to-peer systems. 26th IEEE International Conference on Distributed Computing Systems Workshops, 04–07 July 2006, pp. 73–73 (2006)
16. Schmidt, S., Steele, R., Dillon, T.S., Chang, E.: Building a fuzzy trust network in unsupervised multi-agent environments. On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops, vol 3762, pp. 816–825. (2005)
17. Luo, J., Liu, X., Fan, M.: A trust model based on fuzzy recommendation for mobile adhoc networks. J. Comput. Netw. **53**(14), 2396–2407 (2009)
18. Yu, F., Zhang, H., Yan, F.: A Fuzzy Relation Trust Model in P2P System. International Conference on Computational Intelligence and Security, 3–6 Nov, pp. 1497–1502 (2006)
19. Li, J., Wang, X., Liu, B., Wang, Q., Zhang, G.: A reputation management scheme based on global trust model for peer-to-peer virtual communities. Adv. Web-Age Inf. Manag. **4016**, 205–216 (2006)
20. Li, J.T., Jing, Y.N., Xiao, X.C., Wang, X.P., Zhang, G.D.: A trust model based on similarity-weighted recommendation for p2p environments. J. Softw. **18**(1), 157–167 (2007)
21. Sarwar, B., Karypis, G., Konstan,J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th international conference on World Wide Web., pp. 285–295 ACM (2001)
22. Zhang, S., Lu, D., Yang, Y.: A Fuzzy set based trust and reputation model in P2P networks. Intelligent Data Engineering and Automated Learning–IDEAL, vol 3177, pp. 211–217 (2004)
23. Wang, X., Liang, P., Ma, H., Xing, D., Wang, B.: A P2P trust model based on multi-dimensional trust evaluation. Bio-Inspir. Comput. Intell. Appl. **4688**, 347–356 (2007)
24. Buyya, R., Ranjan, R., Calheiros, R.N.: Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: challenges and opportunities. Proceedings of the 2009 International Conference on High Performance Computing & Simulation, Leipzig, Germany. (HPCS 2009), June, pp. 1–11 (2009)
25. Jiang, W., Wang, G., Wu, J.: Generating trusted graphs for trust evaluation in online social networks. Future Gener. Comput. Syst. (FGCS) **31**, 1–11 (2014)

Ali Mohsenzadeh received his B.S. degree in Computer Engineering and M.S. degree in Information Technology from Mazandaran University of science and Technology in 2012 and 2014, respectively. His research interests and papers are mostly in the areas of Evolution Algorithms, parallel processing, Grid and cloud Computing, and Task scheduling in Dynamic Environments, Petri Net, and Information Technology.

Homayun Motameni received his B.S. degree in Computer Engineering from the Shahid Beheshti University, Tehran in 1995. He received his M.S. and Ph.D. degrees in Computer Engineering from Islamic Azad University-Science and Research Branch in 1998 and 2007, respectively. His current research interests include Petri Net, software systems modeling and evaluation using Petri Net, and machine learning.

Meng Joo Er is currently a Professor with the Division of Control and Instrumentation, School of Electrical and Electronic Engineering (EEE), NTU. His research interests include control theory and applications, fuzzy logic and neural networks, computational intelligence, cognitive systems, robotics and automation, sensor networks, and biomedical engineering. He has authored 5 books, 16 book chapters, and more than 400 refereed journal and conference papers in his research areas of interest.