This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

*Digital Object Identifier 10.1109/ACCESS.2017.Doi Number*

# A Trust-based Agent Learning Model for Service Composition in Mobile Cloud Computing Environments

**Wenjuan Li[1,2,3], Member, IEEE, Jian Cao[2], Member, IEEE, Keyong Hu[1], Member, IEEE, Jie Xu[2], Member IEEE, and Rajkumar Buyya[3], Fellow, IEEE**

[1]Qianjiang College, Hangzhou Normal University, Hangzhou, 310018, CHINA
[2]Computer Science and Technology, Shanghai Jiao Tong University, Shanghai, 200240, CHINA
[3]Cloud Computing and Distributed Systems (CLOUDS) Laboratory, the University of Melbourne, VIC 3052, AUS

Corresponding author: Jian Cao (e-mail: cao-jian@sjtu.edu.cn).

This work was supported by National Natural Science Foundation of China under Grant 61702151 and 61702320, National Key Research and Development Plan under grant 2018YFB1003800, the Research Project for Department of Education of Zhejiang Province under grant Y201635438.

**ABSTRACT** Mobile cloud computing has the features of resource constraints, openness and uncertainty which leads to the high uncertainty on its Quality of Service (QoS) provision and serious security risks. Therefore, when faced with the complex service requirements, an efficient and reliable service composition approach is extremely important. In addition, preference learning is also a key factor to improve user experiences. In order to address them, this paper introduces a three-layered trust-enabled service composition model for the mobile cloud computing systems. Based on fuzzy comprehensive evaluation method, we design a novel and integrated trust management model. Service brokers are equipped with a learning module enabling them to better analyze customers' service preferences especially in cases when the details of a service request are not totally disclosed. Because traditional methods cannot totally reflect the autonomous collaboration between the mobile cloud entities, a prototype system based on the multi-agent platform JADE is implemented to evaluate the efficiency of the proposed strategies. The experimental results show that our approach improves the transaction success rate and user satisfaction.

**INDEX TERMS** Mobile cloud computing, Service Composition, Trust management, User preference learning, Multi-agent technology

## I. INTRODUCTION

Mobile cloud computing is the application of cloud computing in mobile Internet. It refers to the delivery and use mode of IT resources or information services to provide/obtain infrastructure, platform, software (or applications) through mobile network in an on-demand and scalable manner. Mobile cloud computing builds up a hybrid application environment of cloud computing, Internet and mobile ends, improving the computational and storage capability of mobile terminals and providing users with a more rich and colorful functional experience.

However, mobile cloud computing inherits both the advantages and disadvantages of cloud computing and mobile internet. The features of resource constraints, openness and uncertainty lead to the high uncertainty and unstable Quality of Service (QoS) provision and serious security risks [1, 2]. Especially in the face of complex service requirements, how to achieve efficient service composition, and how to ensure the credibility of combined services has become hot issues in the mobile cloud computing researches [3].

Many valuable task scheduling and service composition strategies have been proposed for the traditional Internet environment. However, they can not cope well with the active collaboration of participants in the mobile cloud computing markets. For this reason, agent-based cloud computing models are introduced [4]. Mobile cloud systems based on a multi-agent architecture are much easier to reflect the autonomy, intelligence and initiative of cloud entities, and to realize the independent evolution of the cloud service market, which is closer to the essence of a commercial market [5-11].

To meet these requirements, we propose a Trust-based Agent Learning Model for Service Composition (TALMSC) in mobile cloud computing environments. We design a novel trust management model based on fuzzy comprehensive evaluation method and propose a trust-enabled service composition model. In order to obtain customers' service preferences and accelerate service classification, we equip

service brokers with a learning module based on a two-stage improved Fuzzy C-Means (FCM) learning mechanism which can also improve the transaction success rate and user satisfaction.

To make TALMSC more efficient, satisfactory and reliable, in the construction of our approach, the following key questions are addressed:

● What is the most suitable framework for the multi-agent based mobile cloud scheduling model? How can agents interact with each other?

● Since trust is fuzzy and context-aware, what is an efficient and integrated trust management model?

● What is the suitable learning algorithm to learn customers' service preferences?

In contrast to the existing research works, this paper mainly focuses on the impact of the external mechanisms on the service scheduling process. The main contributions of the paper are as follows:

(1) the design of a multi-agent based cloud service scheduling model under trust mechanism.
(2) a proposed novel trust management model based on fuzzy comprehensive evaluation method.
(3) the design of a two-stage improved FCM method based learning algorithm to obtain user service preferences.

Furthermore, we carry out a couple of experiments to test and evaluate the influence of trust and learning methods on the mobile cloud markets.

The rest of the paper is organized as follows. Section 2 briefly introduces the related work. Section 3 introduces system architecture along with the design details. A fuzzy comprehensive evaluation based trust model is proposed in Section 4. Section 5 donates a two-stage improved FCM learning mechanism in scheduling. And performance evaluation is presented in Section 6. The last section concludes the paper along with future work.

## II. RELATED WORK

We discuss the related efforts in the context of scheduling models, trust issues and user preference learning.

### *A. SERVICE SCHEDULING IN CLOUD COMPUTING ENVIRONMENTS*

Service scheduling is a key factor influencing the performance and user satisfaction of cloud computing systems [12]. Till now, researchers have put forward a couple of high-level research results.

The traditional task scheduling algorithms mainly focus on the optimization of time, including OLB MET, FCFS, RR, Min-Min, Max-Min, Sufferage and their improved algorithms. Since task scheduling is a NP-hard problem, the follow-up researches laid emphasis on the heuristic scheduling algorithm, including GA (Genetic Algorithm), SA (Simulated Annealing), ACO (Ant Colony Optimization), and PSO (Particle Swarm Optimization), etc.

Currently, the targets of task scheduling in cloud environments focus on two sides: (1) optimize the efficiency of the scheduling systems (throughput, load balancing, energy saving, etc.), and (2) optimize cloud users' QoS goals (deadline, budget, fairness, security, etc.). W. Dou et al. proposed a resource co-allocation method for the efficient and load balance scheduling [13]. S. Basu et al. designed a cognitive model of bio-inspired approach to find the optimal solution of task scheduling of IoT applications [14]. F. Damian et al. proposed energy-efficiency strategies for task scheduling under several security constraints [15]. W. Tian et al. developed a method to find the optimal solution to minimize the energy consumption in job migrations [16]. E. Alkhanak et al. proposed a completion time driven hyper-heuristic approach for cost optimization of cloud scheduling [17]. J. Ren et al. established a mathematical model of cloud task scheduling and proposed an improved simulated annealing algorithm to shorten the completion time of tasks under a given user satisfaction [18]. B. Lin et al. introduced a cost-driven strategy for the deadline-constrained workflow scheduling [19]. B. Keshanchi et al. proposed an improved genetic based cloud task scheduling algorithm and designed a series of approaches to analyze the correctness and efficient of their strategy [20]. L. Liu et al. designed an adaptive penalty function to accelerate the convergence and prevent the prematurity of GA based constrained scientific workflow scheduling algorithms in cloud computing environments [21].

In order to build and run service composition systems effectively and efficiently in mobile cloud computing environments, S. Deng et al. proposed novel service selection and scheduling methods, which could effectively get the optimal composition in terms of minimized energy consumption, lower running risks and optimal QoS, respectively [22-25]. Furthermore, in order to provide high quality of service (QoS) for service provisioning system, Deng et al. proposed a novel service cache scheduling method which efficiently takes advantage of the composability of services and indeed significantly improves the performance of service provision systems for real applications [26].

Aiming at better reflecting the autonomy and collaboration of cloud entities, some other scholars prefer the agent-based cloud systems. J. Gutierrez-Garcia and K. Sim [8] proposed fourteen scheduling heuristics for concurrently executing the bag of tasks in Cloud environments and also an elastic cloud resource allocation mechanism. They designed an agent-based Cloud BoT execution tool named CloudAgent to support concurrently BoTs execution in multiple Clouds [9]. As for the composition of cloud services, they proposed an agent-based approach to compose services in multi-Cloud environments for different types of Cloud services [10].

Different from the above solutions, this paper mainly focuses on using trust and leaning mechanisms to enhance the credibility, to accelerate service classification, and to improve the transaction success rate and user satisfaction of the mobile cloud markets.

### B. TRUST MANAGEMENT STRATEGIES

Trust has been proved to be an efficient mechanism to solve the reputation and reliability problems in the distributed open environments.

X. Li et al. proposed a dynamic trust model to accurately quantify and predict user's cognitive behavior [27]. In 2015, they presented a trust aware service brokering scheme named T-broker for the efficient service matching in cloud [28]. Y. Tan et al. proposed a combination-weighted approach based on relative entropy to evaluate user behavior [29]. Some researchers put forward evolutionary algorithms combined trust mechanisms [30, 31]. S. Wang proposed the trust assessment method based on the cloud model [32]. X. Xie designed the double excitation and deception detection based trust model [33]. K. Ahmadi proposed a trust-based decision making model for multi-agent societies [34]. Paper [35] and [36] presented the mechanisms of trust evaluation methods for the cloud-based applications. Y. Wang et al. introduced a trust-based probabilistic recommendation model for social networks [37]. To meet the trust requirements of multi-cloud communities, O. Wahab et al. proposed a three-fold solution including trust establishment, the bootstrapping of trust and trust-based hedonic coalitional game [38]. S. Deng et al. innovatively proposed a two-phase recommendation process to effectively utilize deep learning in initialization and to efficiently synthesize the users' interests with their trusted friends' interests together [39], which remarkably improved the recommendation accuracy and effectiveness.

Many literatures tries to solve the trustworthiness problems in the service selection. For example, M. Alhanahnah et al. presented a taxonomy of trust factors and their application in the real scenarios [40]. X. Li proposed a trust-based and multi-attribute service selection algorithm [41]. C. Hu et al. proposed a trust and spanning tree based cloud service organization approach to help cloud users eliminate malicious and spurious services [42]. Y. Wang et al. put forward the community trust-driven service selection model [43]. C. Hang et al proposed two distributed trust-aware service selection approaches for Service-Oriented Computing (SOC) environments [44]. Also, some researchers proposed service selection algorithms based on trust and QoS preferences, such as [45].

The premise of most of the above studies is that after each transaction, user evaluation is clear and quantifiable which is somewhat inaccurate. In most cases, user evaluation on services is subjective, fuzzy and jumping. Therefore, it's necessary to design a more suitable and integrated trust evaluation model.

### C. USER PREFERENCE LEARNING

User preferences leaning is intensively studied in social networks and recommendation systems. CF (Collaborative Filtering) is the most famous algorithm whose kernel idea is to use nearest neighbor to quantify and predict user preference.

J. Lim et al. developed a multi-agent reinforcement learning method to capture user specific preference in a smart environment [46]. X. Li et al. proposed an optimization algorithm based on a Gaussian model and a novel utility-based approach to estimate user preference [47]. Based on online graph regularized user preference learning (OGRPL), Z. Zhao et al. presented a new framework for a unified preference learning process [48]. Based on a deep user preference learning, Y. Yin et al. proposed a novel service recommendation method containing three prediction models for the cyber-physical systems [49]. They also proposed a service quality prediction mode which is able to incorporate network location and implicit associations among users and services [50].

The target of a CF systems is to predict and recommend the potential services to their users by understanding user preferences, which is essentially different from the main goal of this paper. The learning mechanism in this paper aims to help brokers understand user preferences, and thus adjust their resource introduction strategy to improve user satisfaction. However, the existing methods provide a very useful reference.

## III. DESIGN OVERVIEW

In this section, we discuss system architecture and agent interaction model of TALMSC.

### A. SYSTEM ARCHITECTURE

TALMSC mainly contains three components: JADE main container, transaction agents and cloud resource pool.

JADE main container or the front-end is created automatically when a JADE platform starts. It contains three parts: AMS (Agent Management System), DF (Directory Facilitator) and ACC (Agent Communication Channel). AMS provides white page and life cycle services, maintains AID (Directory of Agent Identifiers) and the states of all agents. DF provides yellow page service. ACC controls all the message exchange of the platform. AMS, DF and ACC are automatically loaded when a JADE platform starts. JADE main container provides a complete runtime environment for the execution of all the other agents.

Transaction agents in TALMSC are mainly three types: user agents, broker agents and provider agents. User agents implement market behavior on behalf of mobile cloud users including service requests submission, service selection, consumption and evaluation. Provider agents are on behalf of service providers. Their activities include choosing, managing and providing services. In order to improve the efficiency of service matching, broker agents are added whose role is similar to the service intermediary agency in real world. The behaviors of broker agents include service matching, credibility/trust evaluation, QoS management and service preference learning. Furthermore, since TALMSC deals with the composite service request of cloud users which may be beyond the capability of a single broker, brokers always need to cooperate and share with others their transaction information and resources.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



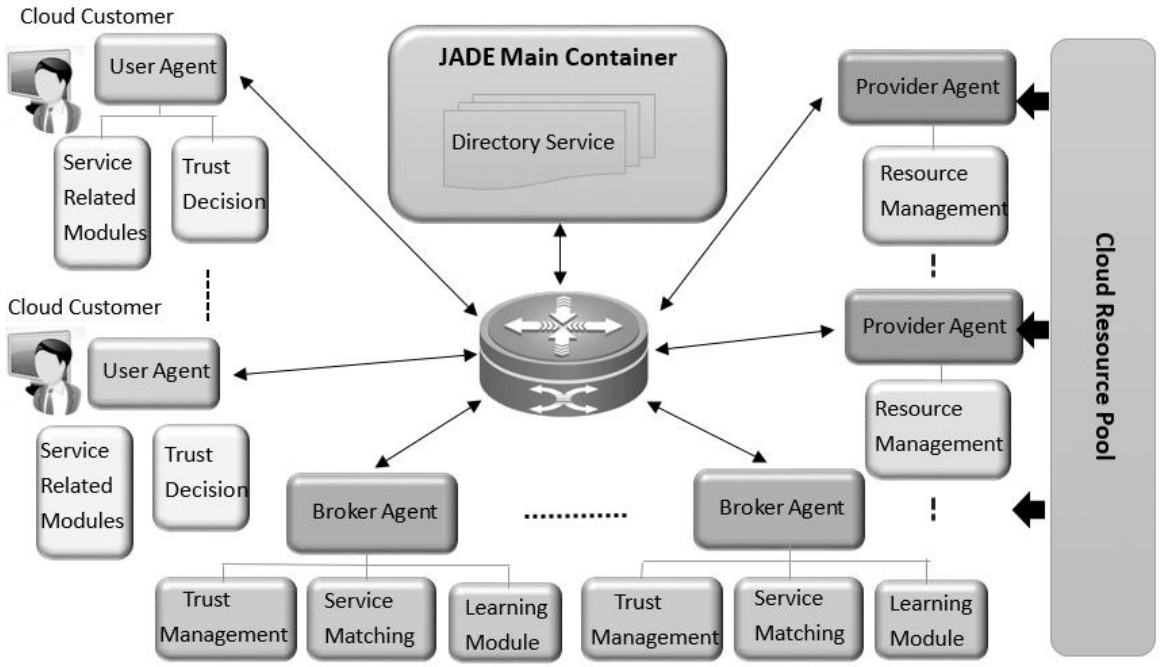


FIGURE 1. **The system architecture of TALMSC**

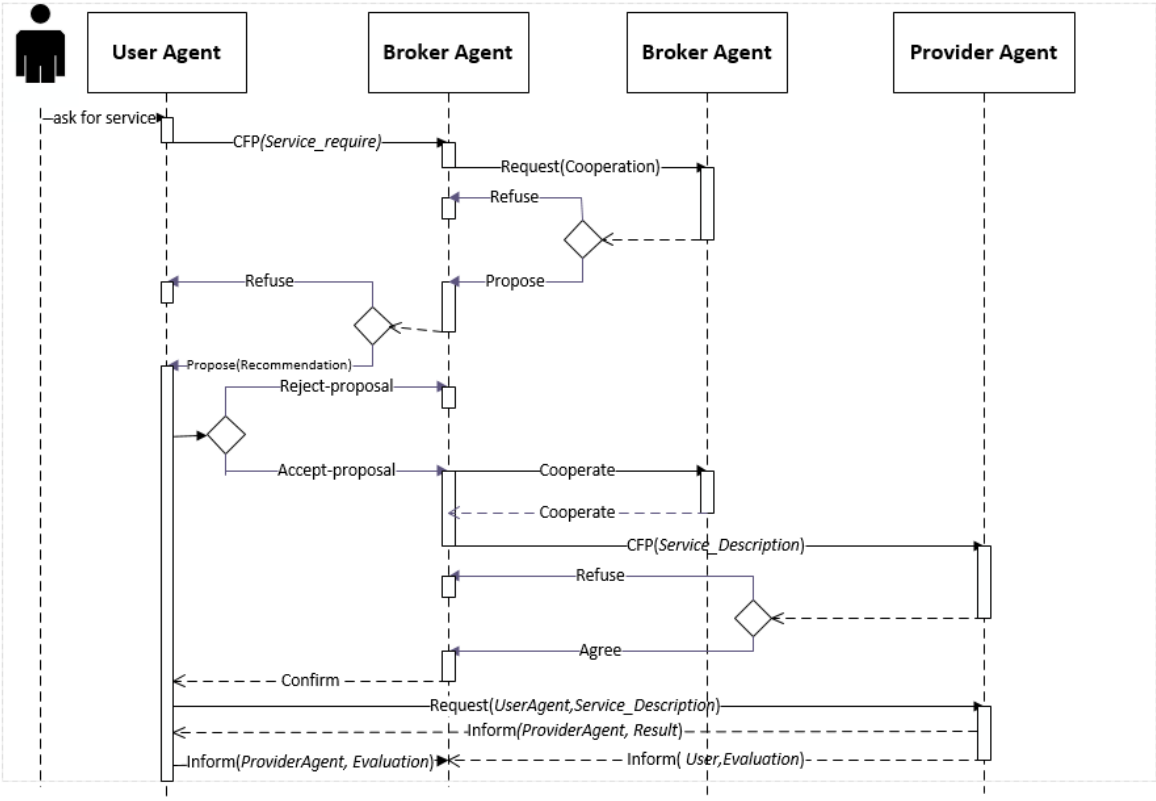


FIGURE 2. **The interaction model of mobile cloud entities**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access


Each intelligent agent in TALMSC has four necessary modules: (1) Self-maintaining module. It helps agents to manage their own affairs; (2) Message control and transportation module. It manages all the messages and communications between intelligent agents; (3) Market behavior module. It manages all the market transaction behaviors; (4) Monitoring and reporting module. It monitors the real-time dynamics of the market and provides analysis and reports. For user agents and broker agents, they are also equipped with a trust module which manages all trust related affairs including trust policy design, trust initialization, trust evaluation, and trust decision, etc. Furthermore, there is one special module for brokers: a learning module. This module is equipped with the learning algorithms to learn customers' service preferences and to accelerate the service classification.

### B. AGENT-BASED CLOUD INTERACTION MODEL

Fig.2 shows the interaction model between agents (user agent, provider agent and broker agent) in TALMSC. Following, a very simple example describes the details of the interaction.

When a cloud user requires a certain type of service, the corresponding user agent will broadcast the service request to its familiar brokers on behalf of the user. Then when a broker receives the request, it checks the service providers that it manages to see whether it can provide the recommendation singly. If so, it recommends the most suitable one according to the details of the request and sends recommendation to the user. If it can not provide service separately, it chooses some other brokers to cooperate and provides a combination recommendation. When the user receives all the recommendations or the deadline is up, it selects the best one and send an "accept" to the chosen broker and "refuse" to the others. The chosen broker then asks the service providers or collaborative brokers for confirmation and then sends feedback to the user. After that, a direct transaction channel between the user and the providers is set up. At last, the user and provider who are involved in the transaction send an evaluation to the broker.

## IV. TRUST MODEL BASED ON FUZZY COMPREHENSIVE EVALUATION METHOD

In this section, a novel Fuzzy Comprehensive Evaluation (FCE) based trust model is introduced to support the trust management in TALMSC.

### A. THE DEFINITION OF TRUST

Trust means the trust of the trustor in the trustee in the recognition of the trustee's identity and the trust of the trustee to complete some special task as expected over a specified period of time and in a particular context. Trust is a kind of decision made by the trustor, based on his own experience and other available knowledge. The indicators of trust include authenticity, honesty, reliability and stability.

A classical trust model usually contains three types of entities: trustee, trustor and recommender. Obviously, a trustee is the object needed to be evaluated before selection or transaction. A trustor is the subject who evaluate the trust of the other transactional entities. And a recommender is the one who provides his recommendation trust of the trustee to the trustor. The relationship between the above entities is shown in Fig. 3.

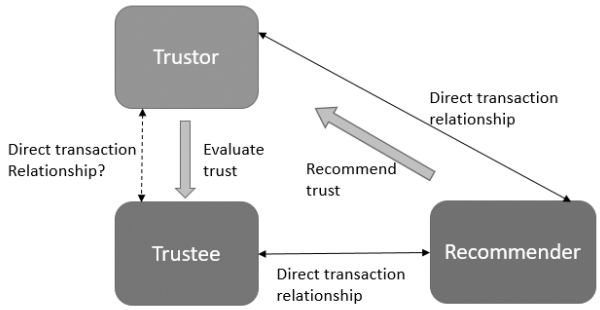


FIGURE 3. **The relationship between trust entities.**

If there exist the direct transactions between the trustor and the trustee, it's called that there exists the direct trust relationship between them. Otherwise, the trustor needs to ask the recommenders for recommendation who have the direct trust relationship with both the trustor and the trustee.

TALMSC assumes that the identity trust has been established at the stage of the market network initialization. Therefore, we only take into account the evaluation of the behavior trust and Equation (1) shows the general formula of it.

$$T_{tc}^{A\to B} = \alpha T_d^{A\to B,tc} + \beta T_r^{A\to B,tc} \tag{1}$$

Where $T_{tc}^{A\to B}$ refers to the integrated trust of cloud entity A to B in a special trading context (tc), $T_d^{A\to B,tc}$ refers to the direct trust of A to B, $T_r^{A\to B,tc}$ refers to the recommendation trust, and α and $\beta$ represent the weight of direct trust and recommendation trust respectively.

Trust is also context sensitive, since the QoS and credibility of a cloud entity may be quite different when he provides/faces with difference services. In order to accurately describe the trust of a cloud provider, trust multidimensional vector $PTrust$ is used. Due to the reason that this paper mainly takes into account three service types including computation, network and storage, $PTrust$ is represented by $PTrust = \{T_{cpu}, T_{bd}, T_{storage}\}$ where $T_{cpu}$ $T_{bd}$ $T_{storage}$ refer to the trust degree of the trustee when providing computation, network and storage services respectively.

### B. TRUST EVALUATION MODEL BASED ON FCE METHOD

FCE is a comprehensive decision making method for a certain object considering the influences of multi-factors and under the fuzzy circumstances. As is known, service trust has the features of subjectivity (depending on evaluation subject), objectivity (services have their objective attributes and value),

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access


uncertainty (changing with time and environment), multi-factor (multi-factors of the service QoS), and context sensitivity (the trust of the same provider becomes different when provides different services). It is suitable to use FCE method for the evaluation of service trust. Besides, within the field of fuzzy theory, FCE method has many advantages such as simple, good at deal with the multi-factor, multi-level complex problems. Therefore, this paper chooses FCE method to complete the evaluation of service trust.

(1) The general steps of trust evaluation based on FCE is as follows.

- Establish evaluation factors set $U$ ( $U = \{u_1, u_2, \dots, u_m\}$). In this paper, trust comprehensive evaluation involving three factors: computation, network and storage.

- Establish evaluation level $V$ ($V = \{v_1, v_2, \dots, v_n\}$). $V$ refers to the evaluation value/level of each factor. This paper chooses four levels for trust including "very trust", "normal trust", "normal untrusted", and "untrusted" to represent the trust degree of user in the service.

- Determine weight vector $A$ ( $A = \{a_1, a_2, \dots, a_m\}, a_i > 0, \sum a_i = 1$ ). Here, $a_i$ is the influence of factor $u_i$ to the final trust decision and the weight of different factors should be customizable according to the users' different service preferences and requirements.

- Construct fuzzy comprehensive judgement matrix R. In this step, we should firstly complete the single judgement of each factor $u_i$ of which the core is to determine the membership degree $r_j$ of $u_i$ belong to the evaluation degree $v_j$. After the evaluation of m influence factor, we get the judgement set $R_i$ ($R_i = \{r_{i1}, r_{i2}, \dots, r_{im}\}$ of factor $u_i$.

In a similar way, after performing the above operations for all the objects which need to be evaluated, the evaluation matrix R is obtained. So far, we obtain a fuzzy relationship from U to V of the evaluated object, as follows.

$$R = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1m} \\ r_{21} & r_{22} & \dots & r_{2m} \\ \dots & \dots & \ddots & \vdots \\ r_{n1} & r_{n2} & \dots & r_{rm} \end{bmatrix}$$

The frequency method is used here to determine R. First of all, the continuous trust value is classified into different evaluation level by dividing the intervals, and then the frequency of the historical data of the index value in the change interval is used as the membership degree of the fuzzy subset. This method is not computationally intensive, and will not increase too much overhead to trust decision.

- Obtain the final comprehensive evaluation results by weighted average method $B = A \cdot R$.

(2) The evaluation of the direct trust

The direct trust is the trust relationship that obtained during the direct interaction between the trustor and the trustee. $T_d^{A\to B,tc}$ ( $T_d^{A\to B,tc} = \{t_{d1}, t_{d2}, \dots, t_{dm}\}$ ) is used to represent the direct trust degree belong to each evaluation factor.

Equation (2) shows how to calculate the direct trust $t_{di}$.

$$t_{di}^{A\to B,tc} = \frac{TNum_{t\in v_i}^{A\to B,tc}}{TNum_{total}^{A\to B,tc}} \tag{2}$$

Here, $TNum_{t\in v_i}^{A\to B,tc}$ means the number of transactions whose trust evaluations are belong to the level $v_i$ between A and B in the service context tc, and $TNum_{total}^{A\to B,tc}$ refers to the total transaction times.

(3) The evaluation of the recommendation trust

The recommendation trust vector $T_r^{C\to B,tc}$ ( $T_r^{C\to B,tc} = \{t_{r1}, t_{r2}, \dots, t_{rm}\}$ ) is the trust evaluation degree of the recommender in the trustee in the context tc.

TALMSC uses two steps to get the recommendation trust: 1) the trustor select several reliable recommenders; 2) calculate the recommendation trust of the trustee by combining the recommenders' recommendation weight and recommendation value.

Equation (3) shows the calculation method of the recommendation trust $T_r^{A\to B,tc}$.

$$T_r^{A\to B,tc} = \frac{\sum_{P\epsilon\Omega}(T_d^{A\to C,recommend} * T_d^{C\to B,tc})}{Length(\Omega)} \tag{3}$$

Where $T_d^{C\to B,tc}$ is the direct trust of recommender C in the trustee B in context tc, $T_d^{A\to C,recommend}$ refers to the trust of $A$ in the recommender $C$ in the context of recommendation. $\Omega$ represents the set of the cloud entities that trustor adopts their recommended values. And $Length(\Omega)$is the size of the recommendation set.

The method of calculate $T_d^{A\to C,recommend}$ is similar with the calculation of the common direct transaction trust. Equation (4) shows how to calculate the recommendation weight $t_{di}$.

$$t_{di}^{A\to C,recommendation} = \frac{TNum_{t\in v_i}^{A\to B,recommendation}}{TNum_{total}^{A\to C,rcommendation}} \tag{4}$$

### C. A CASE STUDY

In the following part, a simple example shows how the above comprehensive method works. Assume that one cloud user requires a computation service and he has to make a choice within the following six providers.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



First of all, we take into account three factors including computational, network and storage capability which influence the QoS of a computation service. So $U = \{Computation, Bandwidth, Storage\}$ . Secondly, we establish the evaluation level, four trust levels are considered. $V = \{"L1 - very\ trust", "L2 - normal\ trust", "L3 - normal\ untrusted", "L4 - untrusted"\}$. Thirdly, the user evaluates each provider from the above perspective and asks for the recommendation from the credible recommenders.

Tab. 1 shows the number of direct transactions that belong to each evaluation level. Tab. 2 shows the recommendation trust of the providers and Tab.3 shows the recommendation weight.

TABLE I
THE HISTORY TRUST OF THE PROVIDERS

| Provider ID | $T_{cpu}$ | | | | $T_{bd}$ | | | | $T_{storage}$ | | | | $TNUM_t$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | L-1 | L-2 | L-3 | L-4 | L-1 | L-2 | L-3 | L-4 | L-1 | L-2 | L-3 | L-4 | |
| *P1* | 132 | 9 | 6 | 3 | 30 | 30 | 30 | 60 | 75 | 15 | 15 | 45 | 150 |
| *P2* | 40 | 20 | 30 | 10 | 70 | 20 | 5 | 5 | 90 | 5 | 3 | 2 | 100 |
| *P3* | 144 | 18 | 9 | 9 | 108 | 45 | 18 | 9 | 126 | 36 | 18 | 0 | 180 |

TABLE II
THE RECOMMENDATION TRUST OF THE PROVIDERS

| Recommender ID | Provider ID | $T_r^{Recommender \to Pi}$ | | | |
|---|---|---|---|---|---|
| | | L-1 | L-2 | L-3 | L-4 |
| *P4* | *P1* | 0.7 | 0.15 | 0.1 | 0.05 |
| | *P2* | 0.5 | 0.2 | 0.2 | 0.1 |
| | *P3* | 0.8 | 0.2 | 0.1 | 0 |
| *P5* | *P1* | 0.9 | 0.1 | 0 | 0 |
| | *P2* | 0.4 | 0.2 | 0.2 | 0.2 |
| | *P3* | 0.7 | 0.1 | 0.1 | 0.1 |

TABLE III
THE RECOMMENDATION WEIGHT OF THE RECOMMENDERS

| Provider ID | |
|---|---|
| *P4* | 0.8 |
| *P5* | 0.6 |

According to Equation (2) and Tab.1, the trust judgement matrix of provider *P1* $T_d^{user \to P1}$ is obtained.

$$R_{T_d^{user \to P1}} = \begin{bmatrix} 0.88 & 0.06 & 0.04 & 0.02 \\ 0.2 & 0.2 & 0.2 & 0.4 \\ 0.5 & 0.1 & 0.1 & 0.3 \end{bmatrix}$$

Assume the weight vector of computational service is$A = \{0.6, 0.2, 0.2\}$. According to the weighted average method $B = A \cdot R$ which is discussed in the above section, we obtain the direct trust evaluation of *P1*.

$$\begin{pmatrix} 0.6 & 0.2 & 0.2 \end{pmatrix} * \begin{bmatrix} 0.88 & 0.06 & 0.04 & 0.02 \\ 0.2 & 0.2 & 0.2 & 0.4 \\ 0.5 & 0.1 & 0.1 & 0.3 \end{bmatrix} =$$

$$\begin{pmatrix} 0.668 & 0.096 & 0.084 & 0.152 \end{pmatrix}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



Therefore $T_d^{user\to P1} = (0.668 \quad 0.096 \quad 0.084 \quad 0.152)$.

Similarly, we get trust judgement matrix of P2 and P3.

$$R_{T_d^{user\to P2}} = \begin{bmatrix} 0.4 & 0.2 & 0.3 & 0.1 \\ 0.7 & 0.2 & 0.05 & 0.05 \\ 0.9 & 0.05 & 0.03 & 0.02 \end{bmatrix}$$

$$R_{T_d^{user\to P3}} = \begin{bmatrix} 0.8 & 0.1 & 0.05 & 0.05 \\ 0.6 & 0.25 & 0.1 & 0.05 \\ 0.7 & 0.2 & 0.1 & 0 \end{bmatrix}$$

The $T_d^{user\to P2} = (0.56 \quad 0.17 \quad 0.196 \quad 0.074)$ and $T_d^{user\to P3} = (0.74 \quad 0.15 \quad 0.07 \quad 0.04)$.

Combined with the recommendation weight of the recommender P4 and P5, we can obtain each provider's recommendation trust shown as follows.

$$T_r^{Recommender\to P1} = (0.55 \quad 0.09 \quad 0.04 \quad 0.02)$$

After normalization, we can get the recommendation trust of the user in P1, that is $T_r^{user\to P1} = (0.786 \quad 0.129 \quad 0.057 \quad 0.028)$. Similarly, $T_r^{user\to P2} = (0.457 \quad 0.2 \quad 0.2 \quad 0.143)$ and $T_r^{user\to P3} = (0.716 \quad 0.149 \quad 0.095 \quad 0.04)$.

Finally, combining the direct and recommendation trust, the integration trust of each provider is obtained. Here, the weight of the direct trust is 0.7, and the weight of the recommendation trust is 0.3.

$$T^{user\to P1} = (0.703 \quad 0.106 \quad 0.076 \quad 0.115)$$

$$T^{user\to P2} = (0.529 \quad 0.179 \quad 0.197 \quad 0.095)$$

$$T^{user\to P3} = (0.733 \quad 0.149 \quad 0.078 \quad 0.04)$$

According to the principle of max membership degree, the user can come to the conclusion that P3 is the most credible in this situation.

## V. TRUST AND LEARNING ENABLED SERVICE COMPOSITION MODEL

### A. TRUST-ENABLED LEARNING AGENT MODEL

Agent is a special software architecture used to simulate mutual cooperation and communication between different individuals or groups. Each Agent is located in the appropriate environment, or as part of its own environment. They can sense the changes in the surrounding environment thus to change their own decisions.

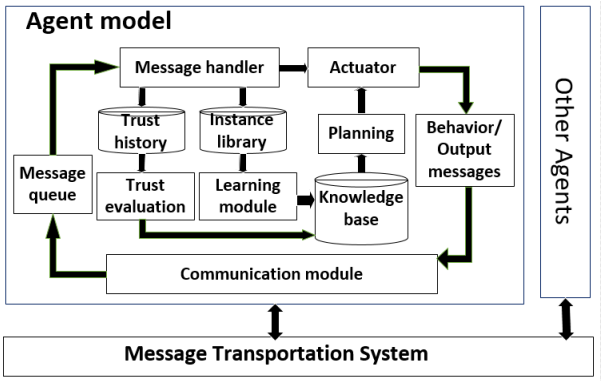


FIGURE 4. **The trust-enabled learning agent model.**

Fig.4 shows the inner structure of the intelligent agents in TALMSC. It contains the necessary components of a normal intelligent agent including the sensors, the message processing and the communication modules which enables it to perceive the external environment, deal with messages and communicate with other agents. In order to help agents adapt to the continuously changing cloud markets and learn their transaction partners' preferences, a learning module is added. Learning is the fundamental part for agents to correct and generalize their behaviors based on the feedback from the mission environment. Constructing a learning module within an agent makes him no longer be confined to those predefined behaviors. In addition, a trust module is equipped which manages all the trust mechanisms in TALMSC. Trust can help agents distinguish good and bad nodes, therefore improving the transaction success rate and maintaining the orderliness and stability of the cloud markets.

Fig.5 shows the main structure of the active learning module in TALMSC. The brokers constantly study and learn something new from the trading history. They collect transaction data, choose to use some special learning algorithms to do the data processing, and then obtain the analysis result which may be the profit, cost or their customers' service preferences. Learning has many benefits for brokers. First of all, it helps to update the learning methods and adjust the parameters to obtain a more precise evaluation later. It can also help the brokers to revise their resource strategies to become more competitive in the cloud market. In this paper, learning ability is mainly used by the brokers to adjust their provider/resource introduction strategy.

### B. LEARNING-DRIVEN SERVICE COMPOSITION MODEL

TALMSC deals with cloud users' composite service requests which requires the brokers, accordingly, have the composite service ability. Fig.6 shows the procedure of composite service scheduling.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



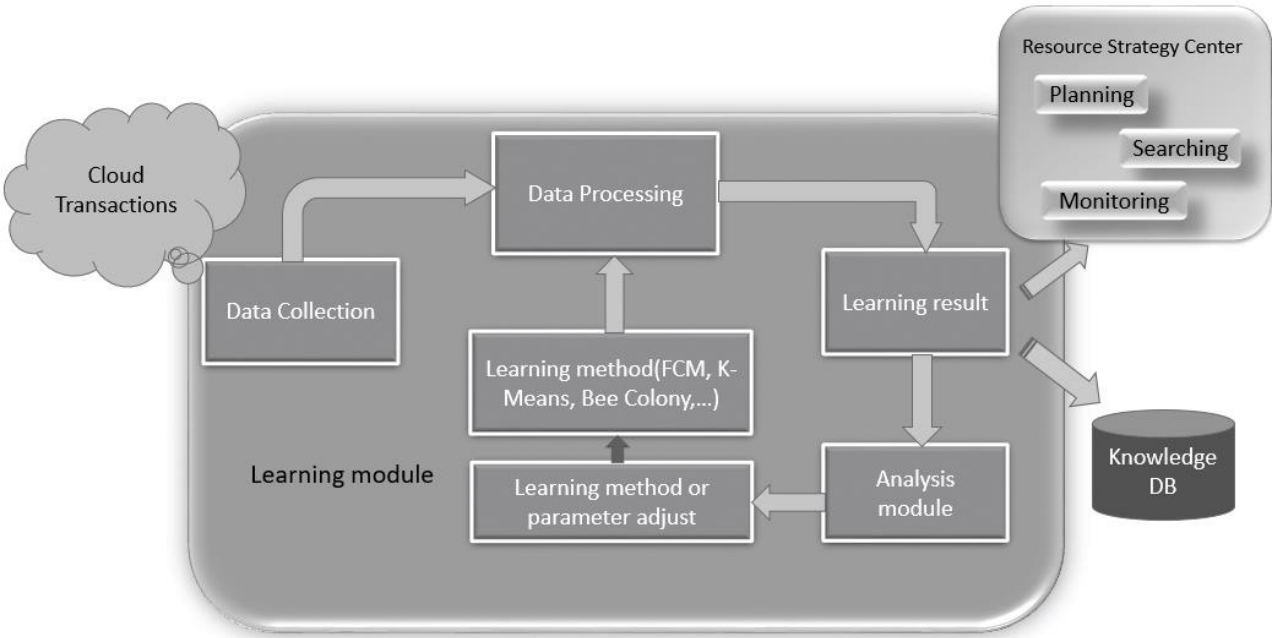


FIGURE 5. **The active learning module in TALMSC.**

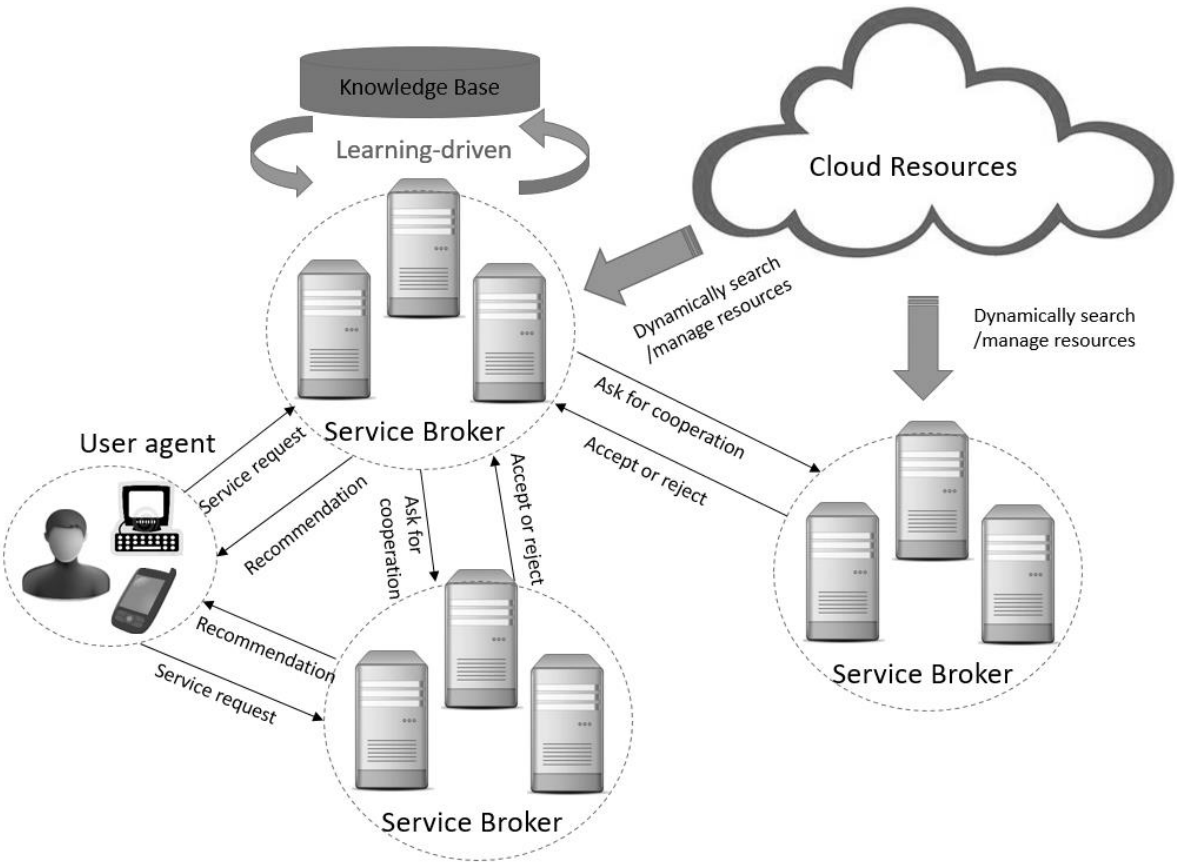


FIGURE 6. **Learning-driven service composition model**

The kernel part of the service scheduling process can be described as follows. Initially, a user agent sends a service request to the familiar brokers. If a broker receives the request and it can deal with the request by itself, it recommends services directly to the user. However, if it can not, it asks other brokers to cooperate and then sends reply to the user. After all the replies arrives, the user chooses the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access


best broker and sends an "accept" message to the chosen one and a "reject" to the others. The chosen broker then asks the service providers or the collaborative brokers for confirmation and helps the user and the providers set up the transaction. After each transaction, the market entities perform evaluations and feedbacks in order to make a better service selection next time. The service brokers with the learning abilities can gradually learn their users' service preferences and modify their plan of cloud providers or resources.

The key role of service scheduling is broker agents. Algorithm 1 shows the operation mechanism of brokers.

**Algorithm 1**: broker agents' operation model

```
Procedure Recommend_Service (my_service_type, provider_list, broker_list)
    While service request queue != NULL do
        Get the first service request q_first from the queue;
        evaluate the request_type of q_first;
        if (request_type==my_service_type) then
            find provider that best fit q_first;
            recommend choose_provider to the user
        else
            request other familiar brokers for cooperation according to request_type;
            wait for responses from other brokers;
            if all answers received or deadline then
                find some brokers to cooperate according to price or other factors;
            end if
            if (find_cooperation_broker is success) then
                send recommendation to the user
            else
                send refuse to the user
            end if
        end if
    end while
end procedure

Procedure Deliver_service (user, chosen_Provider)
    if receive user agent cooperation request then
        if it is a simple service then
            Send_cooperation(chosen_Provider, request_content);
            Waiting responding from chosen_Provider;
            if receive responding then
                Deliver_cooperation(user, responding_message);
            end if
        else
            Send_cooperation(Collaborate_broker, request_content);
            Send_cooperation(Chosen_Provider, request_content);
            Waiting responding from broker;
            Waiting responding from provide;r
            if receive responding then
                Deliver_cooperation(user, responding_message);
            end if
        end if
    end if
end procedure

Procedure Handle_evaluation (decision_threshold)
    if receive satisfaction then
        if satisfaction > decision_threshold then
            Coopration(Consumer,Provider) → Positive sample
        else
            Coopration(Consumer,Provider) → Negative sample
        end if
    end if
end procedure
```

There are three parts in the service scheduling model: service recommendation, service delivery and evaluation handling. The recommendation helps users find the most suitable service. However the service may be provided by one broker or by several brokers. Service delivery deals with the service confirmation and the transaction construction between the corresponding user, the collaborative brokers and the providers. Evaluation handling processes the feedback from users and uses the data to build the training set for a further analysis.

### C. LEARNING ALGORITHM BASED ON TWO-STAGE FCM METHOD

(1) The Fuzzy C-Means (FCM) clustering method

The FCM method is a partition-based clustering algorithm. The kernel idea of FCM is to make the similarity between objects divided into the same cluster the largest, and the similarity between different clusters the smallest.

Assume $X = \{x_1, x_2, \dots, x_n\}$ is a finite data set in the space $R^n$, the FCM method can divide $X$ into c ($2 \le c \le n$) clusters while minimizing the value function of non-similarity index. If we use $V = \{v_1, v_2, \dots, v_c\}$ to represent the center of the clusters, the general form of the objective function is as follows.

$$J(U,V) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m d_{ij}^2 = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m \|x_i - v_j\|^2$$
$$\forall j = 1,..n, u_{ij} \epsilon [0,1], \sum_{i=1}^{c} u_{ij} = 1, \ \ m \ge 0 \qquad (5)$$

Where $U$ is a fuzzy classification of $X$, m is the fuzzy weighted index which is used to control the fuzzy degree of membership matrix. Generally speaking, the larger the value of m, the higher fuzzy degree it is. Although m can be assigned a random value larger than 1, It is generally

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access


considered that m = 2 is most suitable. Therefore, the value of m in the subsequent experiments is set 2.

Constructing the following new objective function, we can find the necessary condition to make equation (5) reach its minimum value.

$$J'(U,V,\lambda) = J(U,V) + \sum_{j=1}^{n} \lambda_j (\sum_{i=1}^{c} u_{ij} - 1)$$
$$= \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m d_{ij}^2 + \sum_{j=1}^{n} \lambda_j (\sum_{i=1}^{c} u_{ij} - 1) \quad (6)$$

Where $\lambda_j, j = 1,..,n$ is the n constrained Lagrangian multiplier. Deriving all the input parameters, the necessary conditions to minimize the equation (5) can be obtained.

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \quad (7)$$

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} (\frac{d_{ij}}{d_{kj}})^{2/(m-1)}} \quad (8)$$

Therefore, the general steps of FCM algorithm is as follows.

- Step 1: Initialize the fuzzy classification matrix $U$ using the random numbers between 0 and 1 and satisfying the constraints in equation (5).
- Step 2: Calculate $V$ (the centers of c clusters) using equation (7).
- Step 3: Calculate the value function according to equation (6). The algorithm stops if it is less than a certain threshold, or if its change is less than a specific threshold from the last time.
- Step 4: Calculate the new fuzzy classification matrix $U$ using equation (8). Go back to step 2.

(2) Learning algorithm based on two-stage FCM method

In cloud computing environments, the service requirements of users are always diverse, fuzzy, subjective and changing. Therefore, we use the FCM method to help brokers better analyze their customers' service preferences and thus adjust their service strategies including resources introduction and management strategies to improve user satisfaction. However, FCM is insufficient. For example, the objective function may converge to a local extreme points. The main problem exists in the improper choice of the initial cluster centers. In order to solve this problem, we improve the traditional FCM algorithm and propose a two-stage FCM method.

The first stage is the initialization of cluster centers. We use improved Maxmin [51] algorithm to achieve this goal. The concept of data density is introduced here in order to avoid the use of noise points as the initial clustering center.

Data density refers to the radius of the specified m data objects centered on the specific data. Obviously, the larger the radius, the more sparse of data in the area where $x_i$ is located; on the contrary, the more dense. Points with higher density are usually considered to be in the data center. Therefore, instead of randomly selecting the first data center like the traditional Maxmin algorithm, we use the point with the smallest data density as the first cluster center.

The main steps of initialization based on improved Maxmin algorithm is as follows.

- Step 1: Calculate the density separately for the data points in data set $X = \{x_1, x_2, \dots, x_n\}$, choose the one $x_i$with the smallest density as the first cluster center $v_1$.
- Step 2: Find the data point $x_j$ farthest from $x_i$ as the second cluster center $v_2$.
- Step 3: Calculate the distance from the rest of the data points in $X$ to the cluster centers. Use $d_{min}^z$ to mark the minimum value to the z$^{th}$ center.
- Step 4: Compare data in $\{d_{min}^z\}$ and find the max one max$\{d_{min}^z\}$, set the data point to be the new cluster center. Judge if c cluster centers has been found, if so go to step 5, otherwise repeat step 3~4.
- Step 5: Divide other data into each cluster by the principle of minimum distance to the center.

The second stage is to use the FCM algorithm to obtain c clusters and get the value of each cluster center.

- Step 1: Set the current number of iterations ($l = 0$), the maximum number of iterations max_iterations, the threshold ξ, and the initial class centers $V^0$ obtained by stage-1.
- Step 2: Obtain the fuzzy membership matrix $U^l$ according to $V^l$.
- Step 3: Calculate the new cluster center set $V^{l+1}$.
- Step 4: Judge if $l >$ max_iterations or $\|V^l - V^{l+1}\| < \xi$. If so, go to the next step, otherwise, repeat step 2-3.
- Step 5: Identify the category of the customers' service preference according the principle of maximum membership degree and output.

(3) Learning based resource adjustment strategies of the brokers

Algorithm 2 describes how learning module works.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access


Algorithm 2: broker agents adjust resource strategy

```
Procedure Eliminate_provider (transaction_num, transaction_threshold, n)
    if transaction_num > transaction_threshold then
        Sort provider by transaction number;
    end if
    Eliminate least used n providers;
  end procedure

 Procedure GetUserPreferece (transaction_num, transaction_threshold, max_iteration, threshold)
      Training sample ← positive and negative sample;
      Build training set;
      Normalized(samples);
      if transaction_num > transaction_threshold then
          use Maxmin method to initialize the fuzzy classification matrix U;
          iteration=0;
          diff=0;
          While (iteration<=max_iteration and diff>=threshold)
         do
             Calculate the cluster center V;
             Update U;
             Diff=difference between the new and old center;
          end While
          Compute the frequency of samples belong to each cluster according to the cluster center;
          Output the user_preference of the samples;
      end if
  end Procedure

  Procedure Select _providers(user_preference)
      for unsaturation provider Pi do
          if provider suffice user_preference then
              if broker unsaturation then
                  provider_list ←Pi;
              else
                  one literation finished and Break;
              end if
          end if
      end for
  end procedure
```

From Algorithm 2, we can see that the learning model mainly consists of three parts: elimination, learning and selection. The elimination deletes the non-popular providers from list. The learning is the core part. It utilizes the transaction history to obtain user preferences. The selection introduces some new providers. The learning model is an iterative process which must go through the above three steps. In the continuous iterative process, brokers gradually identify the type of services their customers are pursuing: cost-effective, big storage, high speed computation, and so on.

## VI. PERFORMANCE EVALUATION

Two kinds of experiments are performed to evaluate the efficiency of TALMSC. Trust mechanisms are tested on NetLogo [52]. The service scheduling and learning ability are tested on JADE [53], a multi-agent framework.

### A. THE EFFICIENCY OF TRUST MECHANISMS

(1) The design of the experiments

The aim of this set of experiments is to test the performance of the proposed trust model. Since in cloud computing environments, the details of services are always transparent to users, therefore, for users, brokers can be seen as their service providers. On the other hand, for brokers, other collaborative brokers or providers can be treated as their providers. From this perspective, trust simulation system does not need to fully cover all the entities in the mobile cloud systems. Therefore, we only take into account two trust parties: the mobile cloud users and the service providers.

In NetLogo, turtles are the subjective entities that represent users to take actions and the links are the social relations between the turtles. In our simulation experiments, the blue color person-shaped turtles are used to represent users while the red color star-shaped turtles represent providers. The blue edges represent the ordinary transaction relationship between users and providers, the yellow edges represent the recommendation relationship between users and providers, and the red edges is the cooperation relations between service providers.

Below are the main steps of the trust simulation experiments.

- Generate a specified number of users and providers whose performances are randomly deployed. In this experiment, performance mainly refers to the trust.
- Initialize the relationship network randomly. The relations include the transaction or recommendation relationships between the market entities (users and service providers) and the cooperation relationships between the providers.
- Each user is required to complete 100 times of transactions. In each transaction, a user first chooses one familiar provider. However, if no one is found, the user asks for recommendation. Unluckily, if still no one is available, it chooses a random one to complete the transaction.
- If a transaction is completed successfully, a green colored edge (transaction edge) between the user and the provider is created. Conversely, the transaction edge between them disappears and accordingly a recommendation edge also disappears if the partner is recommended.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



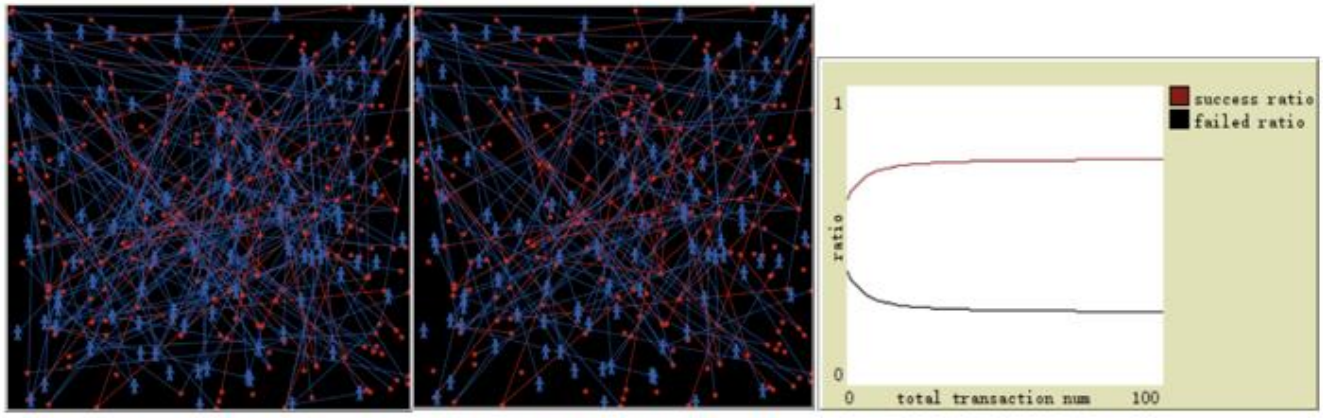


FIGURE 7. **The simulation scene of the random transactions with 30% malicious provider nodes (leftmost: the initial state, middle: the final state after 100 transactions per node, rightmost: the transaction success rate)**

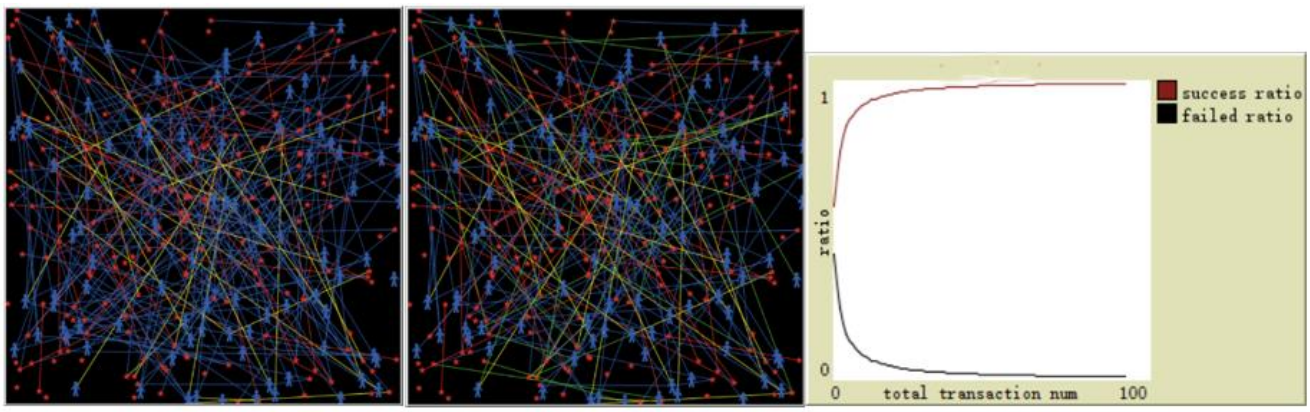


FIGURE 8. **The simulation scene of the proposed trust-based transactions with 30% malicious provider nodes (leftmost: the initial state, middle: the final state after 100 transactions per node, rightmost: the transaction success rate)**

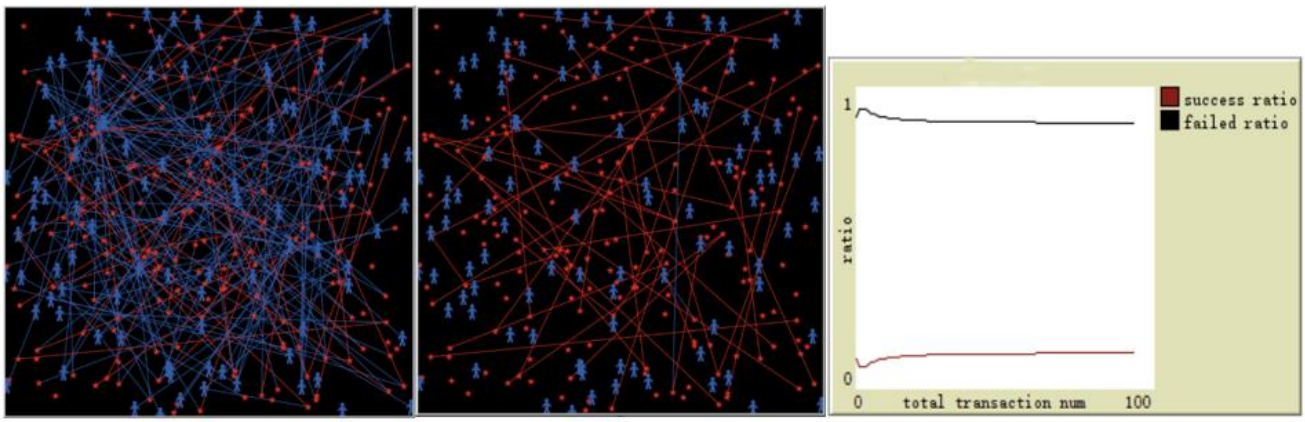


FIGURE 9. **The simulation scene of the random transactions with 95% malicious provider nodes (leftmost: the initial state, middle: the final state after 100 transactions per node, rightmost: the transaction success rate)**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



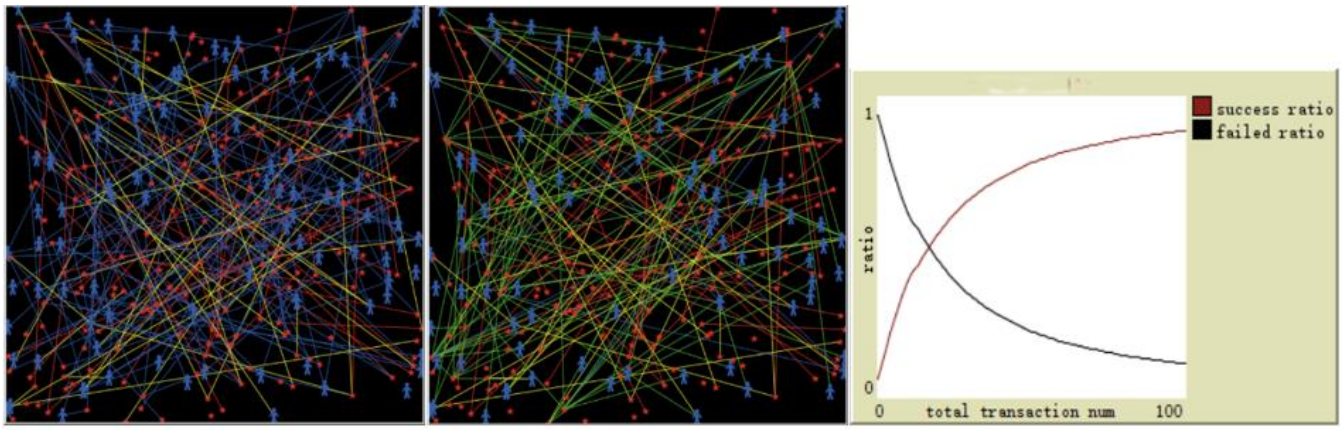


FIGURE 10. **The simulation scene of the proposed trust-based transactions with 95% malicious provider nodes (leftmost: the initial state, middle: the final state after 100 transactions per node, rightmost: the transaction success rate)**

The parameters in the experiments are shown in Tab.4.

TABLE Ⅳ
THE PARAMETERS OF TRUST SIMULATION

| Provider number | User number | Link probability | Malicious ratio |
|---|---|---|---|
| 0~2000 | 0~1000 | 20% | 0~100% |

(2) Result analysis

We focus on trust impact on the transaction success rate and the relationship maintenance of the cloud market. Since it becomes very difficult to distinguish if there are too many entities in the market, therefore, we display the scenes where there are 200 providers and 100 users. Fig.7 ~Fig.10 show the results. For each figure, there are three sub-figures: the leftmost one shows the initial state (the randomly generated trading network), the middle one shows the end state (the post-transaction network after 100 times of transactions per user), and the rightmost one shows the transaction success rate reported by NetLogo reporter. we let the range of malicious node from 0 to 100%, although in the real market, there are probably not so many malicious entities, this is just for the performance and effectiveness test of our solution. Fig. 7 and Fig.8 show the simulation result under the circumstance that 30% of the provider nodes are malicious, while Fig.9 and Fig.10 are the result of 95% malicious provider nodes.

The results of the random transactions show that due to the existence of the low-reputation service providers, the success rate of transactions is very low and the trading network gradually shrinks which indicates that the mobile cloud users gradually lose confidence in the use of cloud services. On the contrary, the results of the trust-enabled transactions show that by loading the proposed trust mechanisms, the transaction success rate is guaranteed and the provider-user network is robust even if there exists a great number of malicious nodes. The main reason is that trust helps the users make decisions before trading, which greatly reduces the chances of the failed or invalid transactions. At the same time, the trust recommendation mechanism helps the users to find those reliable dealers who provide their required services, thereby continuously expanding the trading network and making the cloud market more stable and robust.

### *B. THE SERVICE COMPOSITION PLATFORM BASED ON JADE*

(1) Multi-agent platform JADE

Java Agent Development Framework (JADE) is a software development framework designed to develop the multi-agent systems and the smart agent applications that follow the FIPA standard. It consists of two main parts: a FIPA-compliant agent platform and a software package for developing Java agents. Fig.11 shows the architecture of a standard JADE platform.

Based on JADE framework, TALMSC implements the main parties and their interactions in the mobile cloud markets, realizes the composite service scheduling, provides the trust and learning module for some specific agents, sets a reporter to investigate the evolvement of the market and the influence of the newly introduced mechanisms.

(2) Multi-agent messaging mechanisms in TALMSC

In the simulation platform, there are mainly three types of cloud agents: the user agents, the broker agents and the provider agents. TALMSC designs the messaging mechanism between the different agents. There are mainly four types of messages shown in Tab.5. CFP messages are used for service/cooperation requests, PROPOSE messages are used for recommendations, CONFIRM messages are used for confirmations, and INFORM messages are used for sending the result, either the service or the evaluation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



TABLE Ⅴ
MESSAGE DEFINED IN THE COMPOSITE SERVICE SCHEDULING PLATFORM

| Message | Representation Forms | Sender | Receiver | Function |
|---|---|---|---|---|
| CFP | CFP(*UserAgent*,*Service_Description*) | *UserAgent* | *BrokerAgent* | User asks broker for service recommendation |
| | CFP(*BrokerAgent*,*Service_Description*) | *BrokerAgent* | *ProviderAgent* | Broker asks provider to deal with a new service request |
| | REQUEST(*UserAgent*,*Service_Description*) | *UserAgent* | *ProviderAgent* | User asks provider to provide service |
| | REQUEST(*BrokerAgent*,*Service_Description*) | *BrokerAgent* | *BrokerAgent* | Broker asks another broker for cooperation |
| PROPOSE | PROPOSE(*BrokerAgent*, *Recommendation*) | *BrokerAgent* | *UserAgent* | Broker sends service recommendation to user |
| | PROPOSE(*BrokerAgent*, Co*Recommendation*) | *BrokerAgent* | *BrokerAgent* | Broker sends service recommendation to the collaborative broker |
| CONFIRM | ACCEPT-PROPOSAL/FALURE(*BrokerAgent*) | *UserAgent* | *BrokerAgent* | User agrees with the recommendation or not |
| | ACCEPT/REJECT(*ProviderAgent* ) | *ProviderAgent* | *BrokerAgent* | Provider agrees to provide service or not |
| | ACCEPT/REJECT(*BrokerAgent*) | *BrokerAgent* | *UserAgent* | Broker agrees to provide service or not |
| | AGREE/REFUSE(*BrokerAgent*) | *BrokerAgent* | *BrokerAgent* | Broker agrees to cooperate and provide joint service or not |
| | PROPAGATE(*BrokerAgent*) | *BrokerAgent* | *UserAgent* | Broker tells user he has received the request and is trying to find service for the user |
| | QUERY-IF(*BrokerAgent*,*Service_Description*) | *BrokerAgent* | *BrokerAgent* | Broker asks collaborative broker for confirmation |
| INFORM | INFORM(*ProviderAgent*, *Result*) | *ProviderAgent* | *UserAgent* | Provider tells user the result |
| | INFORM(*UserAgent*, *ProviderAgent*, *Evaluation*) | *UserAgent* | *BrokerAgent* | User tells broker the service evaluation result |

TABLE Ⅵ
THE MAIN PARAMETERS OF THE EXPERIMENT

| User number | Broker number | Provider number | Service frequency | Max number of brokers | Max number of providers | Service type |
|---|---|---|---|---|---|---|
| 1000 | 200 | 5000 | 1/minute/user | 20/user | 100/broker | computation/network /storage |

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



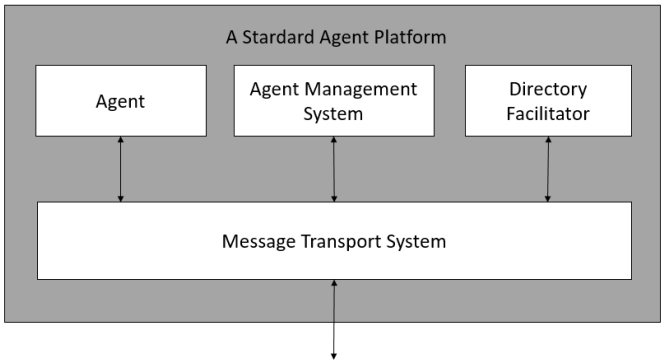


FIGURE 11. **The architecture of a standard JADE platform**

(3) The Interaction of Agents

Based on the JADE framework, the service composition system is designed. Fig.1 shows the infrastructure of the simulation platform, and also the main entities and relations. Tab. 6 shows the main parameters of this set of experiments. In order to investigate the learning ability of the brokers, the max number of providers that each broker manages is restricted and the max number of brokers that one user can be connected to is also limited. Here, we take into account three types of services: computation, network and storage.

Two representative images are captured by JADE sniffer tool to intercept the interaction process between the agents. Due to the huge number of agents and the main target in this part is to investigate the collaborative mechanism between the broker agents, here, the messages of provider agents are hidden.

Fig.12 shows a relative simple circumstance when a service is recommended by a single broker. First of all, User_0 sent a service request message (CFP) to its familiar broker agents (Broker_3, Broker_4, Broker_7, and Broker_13). The brokers then replied with "I've got the request" messages (PROPAGATE). At the same time they tried to find the suitable resources for User_0. When it is done, they sent the recommendations to useragent0 (PROPOSE). After all the recommendation messages arrived, User_0 found a most satisfied one, and sent "agree" (ACCEPT-PROPOSAL) to the chosen one (Broker_7) and "disagree" (FAILURE) to the others. Broker_7 confirmed (CFP), and after the transaction, User_0 sent the evaluation feedback (INFORM) to Broker_7.

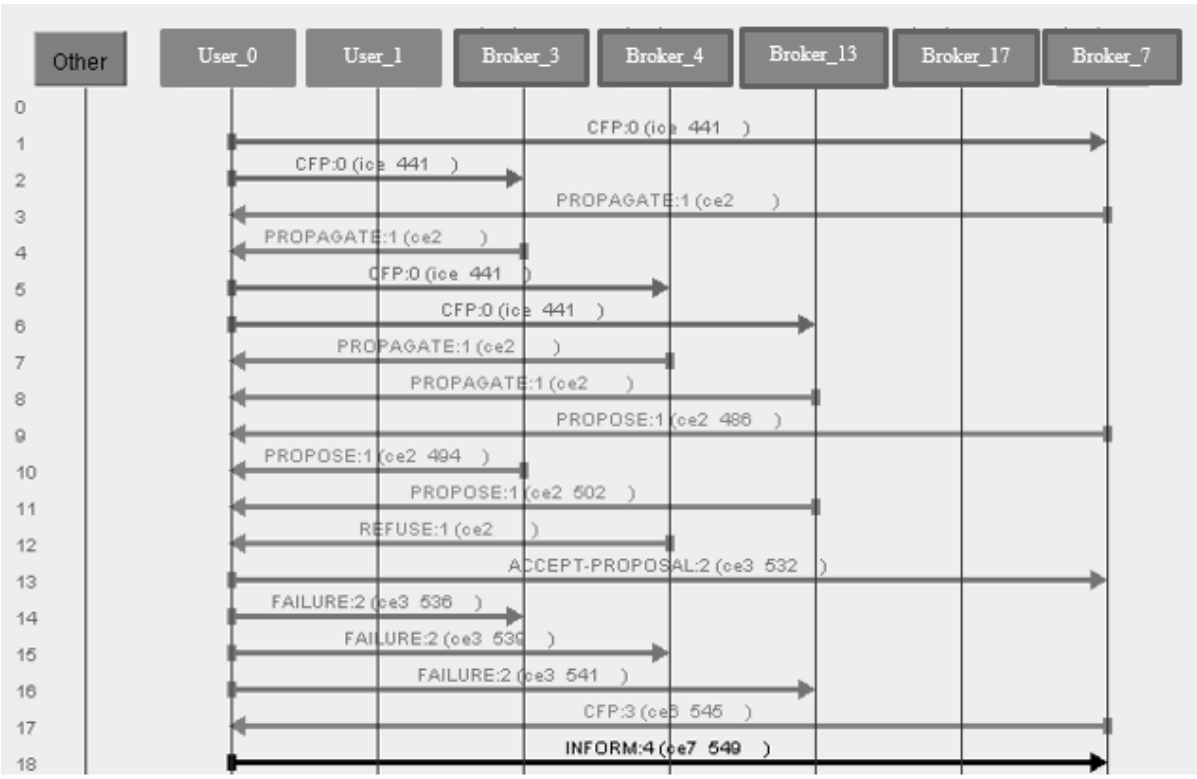


FIGURE 12. **A single service provision process**

Fig.13 shows a more complex situation when a service recommendation is provided by more than one broker, mainly displaying the service cooperation process. User_2 sent a service request to Broker_7 (the only broker it knew). However Broker_7 was not able to provide the service by itself. Obviously, Broker_7 did not want to miss the transaction. So after sending a PROPAGATE message to the user, it sent cooperation requests (REQUEST) to its familiar broker agents (Broker_1, Broker_2, Broker_4, and Broker_6). The cooperation brokers answered with the recommendation messages (PROPOSE). Broker_7 chose to cooperate with Broker_1 and Broker_4 and it sent

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



recommendation to User_2. Since User_2 had no other choices at this moment, it sent ACCEPT-PROPOSAL to Broker_7. When Broker_7 received the ACCEPT-PROPOSAL message, it asked Broker_1 and Broker_4 for confirmations (QUERY-IF). After receiving AGREE from them, Broker_7 sent confirmation message (CFP) to User_2 and at the end of the transaction, User_2 sent a feedback message (INFORM) to Broker_7.

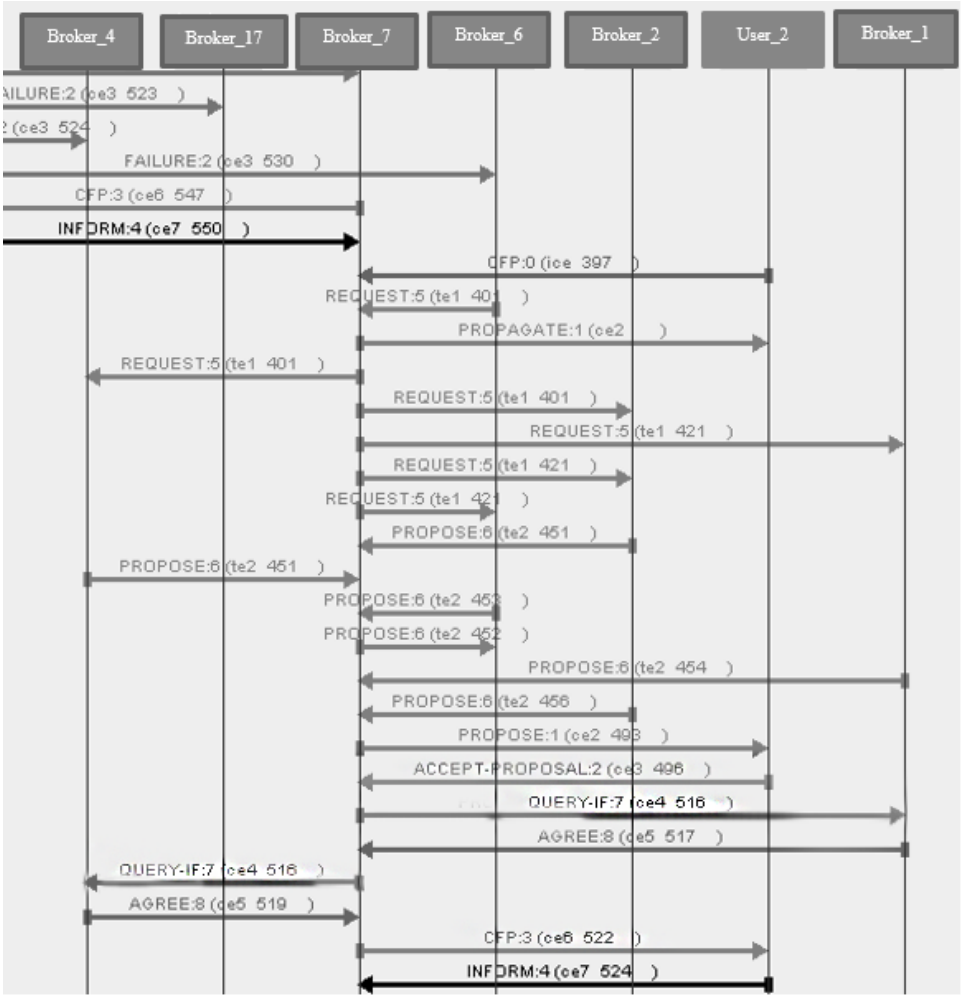


FIGURE 13. **A combination service recommendation process**

TABLE Ⅶ
SLA PARAMETERS OF THE ALI-CLOUD ECS SERVICE

| Type | CPU core | Ram | Storage | Bandwidth | Price/month |
|---|---|---|---|---|---|
| 1 | 1 | 1GB | 40GB | 1TB | $4.5 |
| 2 | 1 | 1GB | 40GB | 2TB | $10 |
| 3 | 1 | 2GB | 40GB | 3TB | $19 |
| 4 | 2 | 4GB | 60GB | 4TB | $39 |
| 5 | 2 | 8GB | 80GB | 5TB | $79 |

### *C. THE EFFECT OF LEARNING MECHANISM*

In TALMSC, users' service requirements are semi-open to the brokers. We choose five attributes to describe the function of a service: 1) CPU, 2) ram, 3) disk capacity, 4) bandwidth, and 5) price. The value of each attribute in the simulation system was derived from the SLA of the Ali-Cloud ECS service [54].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access


Tab.7 shows the parameters of the ECS service. In order to remove the influence of dimension on service evaluation, the value was normalized to the range within 1~100. Doubtlessly, a complete service request includes the above five factors. In order to test the performance of the learning ability, however, users only tell three of them to their potential providers. Therefore, learning is necessary to obtain the real service preferences of the customers.

Here, we perform four groups of experiments in almost the same market situations to test the efficiency of the improved FCM, the FCM, the K-Means and the random transactions. Fig. 14 shows the comparison of the effect to the convergence ratio. Convergence in this paper mainly refers to the stable state of a broker in which its service type and capability are stable including the list of providers it manages, his QoS, user satisfaction, etc. And the convergence ratio is the proportion of brokers in the stable state. From the figure, we can see that the convergence speeds of those situations when the brokers are equipped with learning ability are much faster than that of the random transactions. The reason is that learning mechanism can help brokers to obtain their customers' service preferences, thus help them quickly clarify their market positioning. When the service classification in a market is completed, the brokers are saturated which means they no longer need to introduce new providers/resources. Among them, the improved-FCM converges fastest, which indicates that a better learning method can accelerate the classification of a market. Fig. 15 shows the curve of user satisfaction. Again, the improved-FCM method gains the highest score, indicating that once a broker can quickly grasp the service preferences of its customers, it can maintain user satisfaction at a high level.

Fig.16-19 show the influences of the different decision factors. The first factor is the decision threshold of satisfaction which determines whether a transaction sample is a positive example or the opposite. Fig.16 and Fig. 17 show the effect of the different thresholds on the success ratio of service matching and user satisfaction from which we can see that in the early period, the lower threshold obtains the higher success matching rate and satisfaction. The reason is that a relatively low threshold allows brokers to introduce resources and recommend services more aggressively and more actively, resulting in a higher initial success rate and satisfaction. However, over time, the higher decision threshold which requires more stringent resource classification conditions ensure that user preferences are better studied, and the success rate and satisfaction are gradually improved. However, a threshold is also inappropriate to be too high because it limits the number of samples for learning. Therefore, the final satisfaction and success rate of the three thresholds are very close.

The second factor is the transaction thresholds which determines when a broker starts a new round of learning. Fig. 18 and Fig.19 show the results. Obviously, the longer each iteration, the more transactions and more samples a broker can learn, thereby ensuring the accuracy of learning and maintaining a higher success matching rate and user satisfaction. However, given enough time, when user preferences gradually become clear, the effects of the different transaction thresholds are gradually approaching.

### *D. THE EFFECT OF TRUST*

In the service composition simulation platform, we also examine the effect of trust to the transaction success rate of the composite service scheduling system. Fig. 20 shows the result.

From Fig. 20, we can see that trust maintains the transaction success rate at a relatively high level even if there exist a great number of malicious providers. The reason is that trust decision is performed before each transaction which ensures cloud users always trade with the credible partners, thereby avoiding the invalid or false transactions.

## VII. CONCLUSION AND FUTURE WORK

This paper proposed a novel trust-enabled service composition model (TALMSC) for mobile cloud environments. TALMSC is a three-tier mobile cloud market model which includes the mobile cloud users (customers), the service providers, and the service intermediator (broker). Brokers are the key entities who manage the providers and help the users to find the most suitable providers/resources. In order to improve the efficiency, reliability and satisfaction of service scheduling, trust and learning mechanisms are introduced in the service matching process.

A novel integrated trust model based on FCE method is proposed. The new trust mechanism is comprehensive, context-aware, and able to combine the direct trust with the recommendation trust. In addition, a two-stage improved FCM algorithm is designed to improve the learning ability of brokers. We tested the efficiency of the trust mechanism on NetLogo and based on JADE, we developed a multi-agent based service composition system by which the performances of four related methods (the two-stage improved FCM, the FCM, the K-Means and the random transaction) were evaluated. The simulation results prove that learning ability is a very important factor in improving user satisfaction when the providers are not clear about their customers' service preferences and the improved FCM learning method is efficient.

As part of the future work, we plan to explore the following issues: 1) how can trust integrate well with the other modules like service matching, learning, forecasting, etc. 2) with the learning ability, brokers are easily to obtain users' service preferences and adjust their market strategies which will speed up the market differentiation. Thus, how the brokers will classify or differentiate, as well as how the cloud market evolves.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2904081, IEEE Access



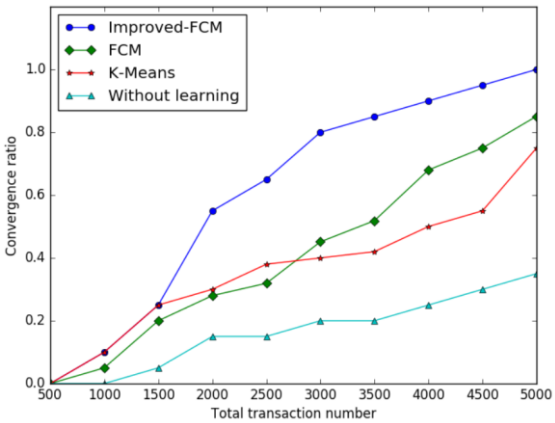


FIGURE 14. **The comparison of convergence ratio**

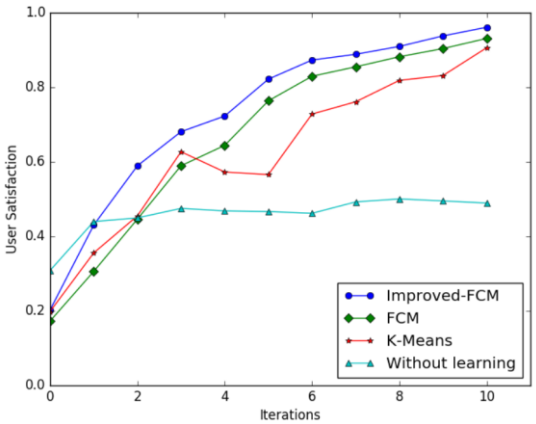


FIGURE 15. **The comparison of user satisfaction**

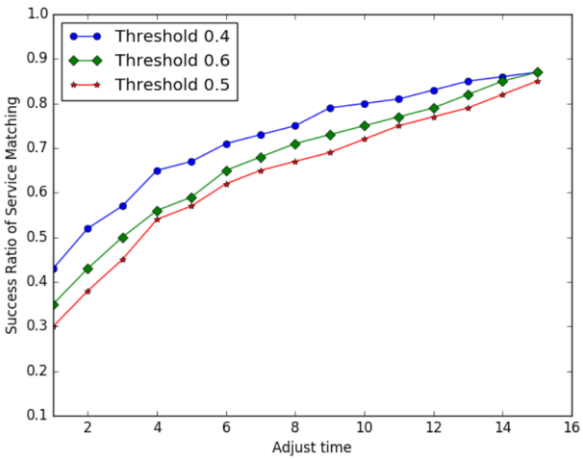


FIGURE 16. **The effect of different decision thresholds on success ratio of service matching**

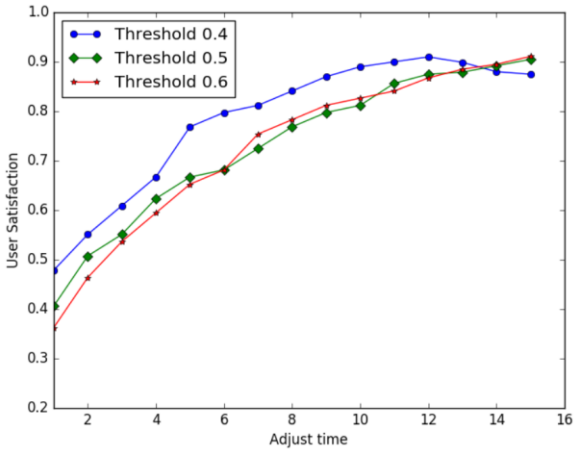


FIGURE 17. **The effect of different decision thresholds on user satisfaction**

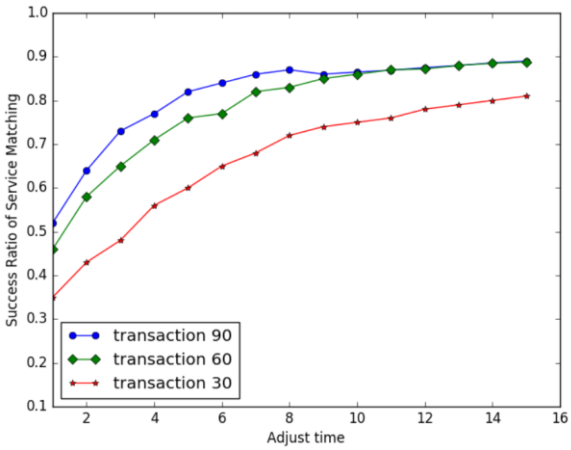


FIGURE 18. **The effect of different transaction thresholds on success ratio of service matching**

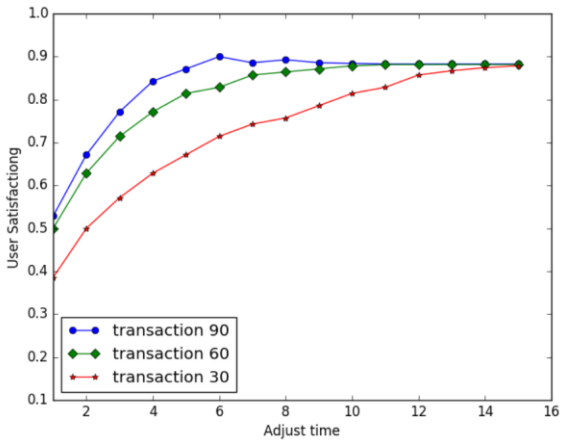


FIGURE 19. **The effect of different transaction thresholds on user satisfaction**

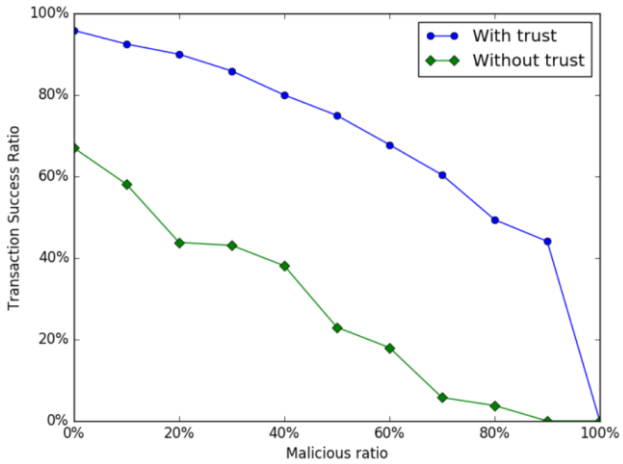


FIGURE 20. **The effect of trust on transaction success rate**

## ACKNOWLEDGMENT

The authors would like to thank the editor and the anonymous reviewers for their insightful and constructive comments and suggestions on improving this paper.

## REFERENCES

[1] T. Noora, S. Zeadallyb, A. Alfazic, and Q. Sheng, "Mobile cloud computing: Challenges and future research directions," Journal of Network and Computer Applications, vol. 115, pp.70–85, Aug. 2018.

[2] C. Huang, X. Mei, G. Zhao, and J. Wu et al, "Transaction modelling and execution analysis of uncertainty composition service in mobility computing environments," Science China: Information Science, vol. 45, no. 1,pp.70-96, Jan. 2015.

[3] S. Deng, L. Huang, H. Wu, W. Tan, J. Taheri, A. Zomaya, and Z. Wu. "Toward Mobile Service Computing: Opportunities and Challenges," IEEE Cloud Computing, vol. 3, no.4, pp32-41, 2016.

[4] K. Sim, "Agent-based Approaches for Intelligent Intercloud Resource Allocation," IEEE Trans. on Cloud Computing, 2018, early access, DOI 10.1109/TCC.2016.2628375

[5] K. Sim, "Agent-based cloud commerce," in *Proc.* of the IEEE International *Conf.* on Industrial Engineering and Engineering Management, Hong Kong, China ,2006, pp. 717-721.

[6] K. Sim, "Towards complex negotiation for cloud economy," in *Proc.* of the International Conference on Grid and Pervasive Computing: Advances in Grid and Pervasive Computing. Springer Berlin Heidelberg, 2010, pp. 395-406.

[7] K. Sim, "Complex and concurrent negotiations for multiple interrelated e-markets," IEEE Trans. On Cybernetics, vol. 43, pp. 230-245, Feb. 2013.

[8] J. Gutierrez-Garcia, and K. Sim, "A family of heuristics for agent-based elastic Cloud bag-of-tasks concurrent scheduling," Future Generation Computer Systems, vol. 29, no.7, pp.1682-1699, Sept. 2013.

[9] J. Gutierrez-Garcia, and K. Sim, "Agent-based Cloud bag-of-tasks execution," The Journal of Systems and Software, vol.104, pp.17–31, Jun. 2015.

[10] J. Gutierrez-Garcia, and K. Sim, "Agent-based Cloud service composition," Applied Intelligence, vol. 38, no.3, pp 436–464, Apr. 2013.

[11] H. Li and F. Ye, "Subscription invoking model of web services based on mobile agent in mobile computing," Computer Engineering and Design, vol. 31, no. 6, pp.1339-1342, Aug. 2010.

[12] L. Bittencourt, A. Goldman, E. Madeira, and N. Fonseca et al, "Scheduling in distributed systems: A cloud computing perspective," Computer Science Review, vol. 30, pp. 31-54, Nov. 2018.

[13] W. Dou, X. Xu, X. Liu, and L. Yang et al, "A Resource Co-Allocation method for load-balance scheduling over big data platforms," Future Generation Computer Systems, vol. 86, pp.1064-1075, Sept. 2018.

[14] S. Basu, M. Karuppiah, K. Selvakumar, and K. Li et al, "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment," Future Generation Computer Systems, vol.88, pp.254-261, Nov. 2018.

[15] F. Damián, A. Jakóbik, D. Grzonka, and K. Joanna et al, "Security supportive energy-aware scheduling and energy policies for cloud environments," Journal of Parallel and Distributed Computing, vol. 119, pp.191-202, Sept. 2018.

[16] W.Tian, M.He, W.Guo, and W.Huang et al, "On minimizing total energy consumption in the scheduling of virtual machine reservations," Journal of Network and Computer Applications, Vol. 113, pp. 64-74, Jul. 2018.

[17] E. Alkhanak, and S. Lee, "A hyper-heuristic cost optimization approach for Scientific Workflow Scheduling in cloud computing," Future Generation Computer Systems, vol.86, pp.480-506, Sept. 2018.

[18] J. Ren, and X. Zhong, "Cloud Task Scheduling research with multidimensional QoS Constraints," Microelectronics& computer, vol.35, no.7, pp.97-100,105, Jul. 2018

[19] B. Lin, W.Guo, and G. Chen, "Scheduling strategy for science workflow with deadline constraint on multi-cloud," Journal on Communications, vol.39, no.1, pp.56-69, Jan. 2018.

[20] B. Keshanchi, A. Souri, and N. Navimipour, "An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: formal verification, simulation, and statistical testing," Journal of Systems and Software, vol. 124, pp.1-21, Feb. 2017.

[21] L. Liu, M. Zhang, R. Buyya, and Q. Fan, "Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing," Concurrency and Computation: Practice and Experience, vol. 29, no. 5, e3942, Sept. 2016.

[22] S. Deng, H. Wu, W. Tan, Z. Xiang, Z. Wu, "Mobile Service Selection for Composition: An Energy Consumption Perspective," IEEE Trans. Automation Science and Engineering, vol. 14, no. 3, pp1478-1490, 2017.

[23] S. Deng, L. Huang, J. Taheri, J. Yin, M. Zhou, A. Zomaya, "Mobility-Aware Service Composition in Mobile Communities," IEEE Trans. Systems, Man, and Cybernetics: Systems, vol.47, no.3, pp.555-568 , Mar. 2017.

[24] S. Deng, L. Huang, Y. Li, H. Zhou, Z. Wu, X. Cao, M. Kataev, L. Li, "Toward Risk Reduction for Mobile Service Composition," IEEE Trans. Cybernetics, vol. 46, no. 8, pp1807-1816, 2016.

[25] S. Deng, L. Huang, D. Hu, J. Zhao, Z. Wu, "Mobility-Enabled Service Selection for Composite Services," IEEE Trans. Services Computing, vol.9,no.3, pp. 394-407 , May/Jun. 2016.

[26] S. Deng, Z. Xiang, J. Yin, J. Taheri, A. Zomaya, "Composition-Driven IoT Service Provisioning in Distributed Edges," IEEE Access,vol.6, pp. 54258-54269 , Sept. 2018.

[27] X. Li, and X. Gui, "Cognitive model of dynamic trust forecasting," Journal of Software, vol.21, no.1, pp. 163-176, Jan. 2010.

[28] X. Li, and H. Ma, "T-Broker: A Trust-Aware Service Brokering Scheme for Multiple Cloud Collaborative Services," IEEE

Transactions on Information Forensics and Security, vol. 10, no.7,pp. 1402-1415,Jul. 2015.

[29] Y. Tan, and C. Wang, "Trust Evaluation Based on User Behavior in Cloud Computing," Microelectronics & Computer, vol. 32, no. 11, pp. 147-151, Nov. 2015.

[30] S. Sanadhya, and S. Singh, "Trust Calculation with Ant Colony Optimization in Online Social Networks," Procedia Computer Science, vol. 54, pp. 186-195, 2015.

[31] E. Ugur, S. Sen and A. Burak, "GenTrust: A genetic trust management model for peer-to-peer systems," Applied Soft Computing, vol. 34, pp.693-704, Sept. 2015.

[32] S. Wang, L. Zhang, and H. Li, "Evaluation approach of subjective trust based on cloud model," Journal of Software, vol. 21, no. 6, pp. 1341-1352, Jun. 2010.

[33] X. Xie, L. Liu, and P. Zhao, "Trust model based on double incentive and deception detection for cloud computing," Journal of Electronics & Information Technology, vol. 34, no. 4, pp. 812-817, Apr. 2012.

[34] K. Ahmadi, and V. Allan, "Trust-based decision making in a self-adaptive agent organization," ACM Trans. Auton. Adapt. Syst. vol. 11, no. 2, pp. 1-25, Jun. 2016.

[35] M. Sule, and M. Li, "Trust modeling in cloud computing," in Proc. of the 2016 IEEE Symposium on Service-Oriented System Engineering. Oxford, UK ,2016, pp. 78-83.

[36] E. AbdAllah, M. Zulkernine, Y. Gu, and C. Liem, "TRUST-CAP: A Trust Model for Cloud-based Applications," in Proc. of the 2017 IEEE 41st Annual Computer Software and Applications Conference, Turin, Italy, 2017, pp. 584-589.

[37] Y. Wang, G. Yin, Z. Cai, and Y. Dong et al, "A trust-based probabilistic recommendation model for social networks," Journal of Network and Computer Applications, vol. 55, pp. 59-67, Sept. 2015.

[38] O. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Towards trustworthy multi-cloud services communities: A trust-based hedonic coalitional game," IEEE Transactions on Services Computing, vol. 11, no. 1, pp.184-201, Jan. 2018.

[39] S. Deng, L. Huang, G. Xu, X. Wu, Z. Wu, "On Deep Learning for Trust-Aware Recommendations in Social Networks," IEEE Trans. Neural Netw. Learning Syst., vol.28,no.5, pp. 1164-1177 , May. 2017.

[40] M. Alhanahnah, P. Bertok, and Z. Tari, "Trusting cloud service providers: Trust phases and a taxonomy of trust factors," IEEE Cloud Computing, vol. 4,no. 1, pp. 44-54, Mar. 2017.

[41] X. Li, J. He, and Y. Du, "Trust Based Service Optimization Selection for Cloud Computing," International Journal of Multimedia and Ubiquitous Engineering, vol. 10, no. 5, pp. 221-230, May. 2015.

[42] C. Hu, J. Liu, and J. Lium, "Services selection based on trust evolution and union for cloud computing," Journal of Communications, vol. 32, no. 7, pp. 71-79, Jul. 2011.

[43] Y. Wang, J. Zhou, and H. Tan, "CC-PSM: A Preference-Aware Selection Model for Cloud Service Based on Consumer Community," Mathematical Problems in Engineering, Hindawi Publishing Corporation, vol. 2015, DOI:10.1155/2015/170656.

[44] C. Hang, and M. Singh, "Trustworthy service selection and composition," ACM Trans. Auton. Adapt. Syst. vol. 6, no. 1, Feb. 2011.

[45] H. Wang, C. Yu, L. Wang, and Q. Yu, "Effective BigData-Space Service Selection over Trust and Heterogeneous QoS Preferences," IEEE Transactions on Services Computing, vol. 11, no. 4, pp. 644-657, Jul. 2017.

[46] J. Lim, H. Son, D. Lee, and D. Lee, "An MARL-Based Distributed Learning Scheme for Capturing User Preferences in a Smart Environment," in Proc. of the 14th International Conference on Services Computing, Honolulu, HI, USA, 2017, pp. 132-139.

[47] X. Li, G. Xu, E. Chen, and L. Li, "Learning User Preferences across Multiple Aspects for Merchant Recommendation," in Proc. of the 2015 IEEE International Conference on Data Mining, Atlantic City, NJ, USA, 2015, pp.865-870.

[48] Z. Zhao, H. Lu, D. Cai, and X. He et al, "User Preference Learning for Online Social Recommendation," IEEE Transactions on Knowledge and Data Engineering, vol. 28, no.9,pp.2522-2534, Sept. 2016.

[49] Y. Yin, F. Yu, Y. Xu, L. Yu, and J. Mu, "Network Location-Aware Service Recommendation with Random Walk in Cyber-Physical Systems," Sensors, vol. 17, no. 9, pp. 2059, Sept. 2017.

[50] Y. Yin, L. Chen, Y. Xu, and J. Wan, "Location-Aware Service Recommendation With Enhanced Probabilistic Matrix Factorization," IEEE Access, vol. 6, pp. 62815-62825, Sept. 2018.

[51] M. Halkidi and M. Vazirgiannis, "On clustering validation techniques," Journal of Intelligent Information Systems, vol.17, no.2-3, pp. 107-145, Dec. 2001.

[52] NetLogo, NetLogo User Manual version 6.0.3, [Online]. Available: https://ccl.northwestern.edu/netlogo/docs/.

[53] Jade company, JADE document, [Online]. Available: http://jade.tilab.com/.

[54] Ali-Cloud. The SLA of Ali-Cloud ECS service, [Online]. Available: https://www.alibabacloud.com/zh/product/ecs?spm=a2796.7960336.224002.1.45325179B9MORx#pricing