Developing and Adopting Trust-aware Collaborative Prediction of QoS for Service-based Systems

Feng-Jian Wang, Chen-Yang Chen, Po-Han Chen
Dept. of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
Email: {fjwang@cs.nctu.edu.tw, admachen@gmail.com, sihalon@gmail.com}

***Abstract*—An application (service) can be composed of existing services. Generally, an appropriate service might be selected according to the predicted Quality of Service (QoS) values, and the most common approach for service selection is using collaborative filtering for prediction. In this paper, we present a trust-aware QoS prediction method for service selection, which is inspired by trust relationships in social networks. We use direct and indirect similarities of opinion among users, each of which are combinations of belief, disbelief, and uncertainty. Then, we can derive similarities among users, select similar users, and predict the QoS value of a service. The experimental results show that the accuracy of QoS values determined using our method is better than that using other methods.**

Keywords–Trust-aware; QoS Prediction; Service-Based Systems; Opinion.

I. INTRODUCTION

Web services are widely adopted to provide various functions in fields such as scientific research, e-commerce, healthcare, and aerospace. Developers can create applications with low costs and short development times by adopting existing services. However, individual services to be adopted for composition must meet not only functional but also non-functional requirements such as response time, reliability, and cost constraints. For better service selection, quality-of-service (QoS) values can be used. Several approaches have been presented to determine the QoS values of services [1]-[2]. Liang *et al*. [1] proposed a framework to predict QoS values by extracting service features. Zhen *et al*. [3] adopted matrix factorization to predict QoS. Zheng *et al*. [4] applied collaborative filtering (CF) to predict unknown QoS values. Their approach selects similar users based on the Pearson Correlation Coefficient (PCC). Moreover, two studies have been conducted for improving their CF-based approach by applying factors: neighborhood [2] and time-aware [5]. However, the predicted results are inaccurate when the number of invocation records is too low.

In general, similarity has propagative characteristics, so similarity exploitation of users that reside at longer distances is reasonable. To improve QoS prediction value accuracy, we propose an approach based on the concept "the more similar the users selected for prediction, the more accurate the predicted QoS value". For example, consider three users having the following properties: the first two users are not similar to each other, but both are similar to the third. With a two-hop distance [6], the first user may have some similarity to the second; in other words, there is an indirect similar relationship between them.

In our approach, prediction accuracy is improved based on a social trust network. For example, we might be able to know with certainty whether a proposition will be true or false [7]. Both *direct* and *indirect similarities* comprise the kernel adopted in our work. User can be trusted more with higher direct similarity. Also, inspired by Josang's paper [8], "Opinion" has been introduced to express the extent of belief in some events, and direct opinion can be defined to be composed of three factors: *uncertainty*, *belief*, and *disbelief*. An *indirect opinion* can be determined from the recommendation of other users based on their *direct opinion* [8]. Finally, the value for representing the indirect similarity of two users, can be estimated according to other users' indirect opinions.

We then present an algorithm to select users based on direct (i.e., PCC) and indirect similarity, and adopt their opinion to help improve QoS prediction by modifying QoS values using Resnick's formula correspondingly. The experimental results indicate that our approach is better than other approaches when the number of invocation records is small [4].

The rest of the paper is organized as follows. Section II describes the background and related work. Section III introduces our trust-aware approach. Section IV presents the experiments and makes comparisons. Finally, Section V concludes the whole paper and future work.

II. BACKGROUND

In this section, we describe service-based systems (SBS), QoS calculations, social trust, and related QoS evaluation and prediction approaches. Part A presents a review of QoS evaluation and prediction approaches for an application (service). Part B introduces trust and related work in recommendation systems.

A. QoS Evaluation and Prediction for SBS

SBSs are applications composed of individual services in more than one website. From the viewpoint of a composition workflow, a service can be treated as a process. During SBS development, it is better to select the most appropriate one from among web service candidates. For example, some consumers prefer cheap services, while others prefer highly reliable ones. It might be better to select a cheaper service if budget is a major consideration. Moreover, functionality might be quantified based on the values of necessary quality factors. With the rising popularity of SBSs, an increasing number of services with similar functions are supported by different service providers. Therefore, it is better to adopt an effective approach to evaluate Web services for selection, composition, and so on [9].

Collaborative filtering approaches [4][10] are widely adopted for predicting attribute values in recommendation systems. QoS values can be predicted using QoS records of different Web services from similar users, including failure probability, performance, and cost. Usually, similarities between users are defined in terms of PCC [11], which is a measure of the linear correlation between two variables X and Y, as expressed by (1), where $\overline{X}$ is the mean of X, $\overline{Y}$ is the mean of Y, and the coefficient value is between -1 and 1.

 ISBN: 978-1-61208-387-2

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \quad (1)$$

Data sparsity is one of the main challenges for current CF-based approaches [6]. When both numbers of users and services increase quickly, the *user-service matrix* (which will be explained later) commonly used in CF approaches could become very large and sparse. The cold-start problem is encountered because a new user invokes only a few services, which may lead to insufficiency of QoS values. To alleviate the cold-start problem, factors such as location [12] and time [7][11] have been adopted to improve prediction.

In conventional CF approaches, similar users are located within a distance of one hop. Similarity has propagative characteristics, so information exploitation of users located at longer distances would be reasonable. An example in Figure 1 shows that User 1 is similar to User 2, User 2 is similar to User 3, and both similarities are of one distance. It is reasonable to treat both User 1 and User 3 as indirectly similar, and there is a similarity of two-hop propagations between them [13].

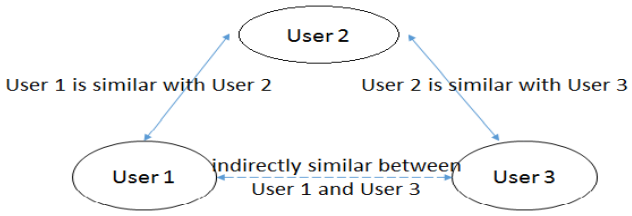

Figure 1. Example of similarity propagation.

B. Social Trust in Cloud Computing and Recommendation system

Social networks are used to reflect real-world relationships, allow users to share information, and form connections among users [14]. Chard *et al*. [14] designed a social cloud system and implemented it. In the social cloud system, users can discover storage services contributed by their friends based on existing trust relationships. In the study of recommendation systems, Singla *et al*. [15] applied data mining techniques to study the relationships among users, and observed that users are more likely to share interests with similar users. Pitsilis *et al*. [16] announced that the performance of a recommendation system can be enhanced based on potential trust among users.

Each of above trust models is adopted to solve one or more specific problems in social cloud systems or recommendation systems. In contrast, Subjective Logic [8] is a general trust model for representation and reasoning of trust. It operates on subjective beliefs and uses "*opinion*" to denote the representation of a subjective belief. Pitsilis *et al*. [13] and O'Donovan *et al*. [17] used subjective logic for reasoning the trustworthiness among users, and their experiments indicate that the accuracy of recommendation values can be improved with trustworthiness.

Because of the incompleteness and inconsistency of knowledge, it is impossible to know for sure whether a proposition would be true or false [7]. Thus, *opinion* [8] has been defined to express the extent of belief in some events, and the definition includes three factors: belief (b), disbelief (d), and uncertainty (u). The mathematical definition and computation of direct opinion are described below.

Let ω_e^i, $user_i$'s opinion about an event e, be a three tuple, where b_e^i is $user_i$'s belief in event e, d_e^i is $user_i$'s disbelief in event e, u_e^i is $user_i$'s uncertainty in event e and $\omega_e^i = \{b_e^i, d_e^i, u_e^i\}$. Figure 2 indicates the profile of (2).

$$b_e^i + d_e^i + u_e^i = 1, \quad \{b_e^i, d_e^i, u_e^i\} \in [0,1]^3 \quad (2)$$

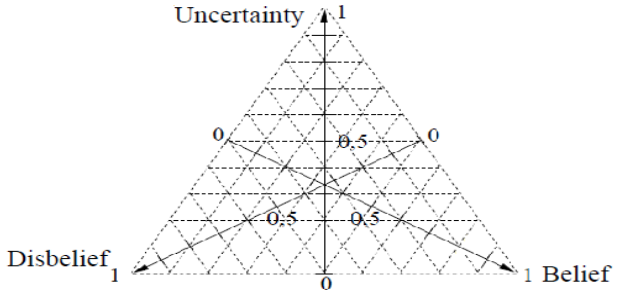

Figure 2. Opinion triangle.

Two operators are provided by *Subjective Logic: recommendation* $\otimes$ in (3) and *consensus* $\oplus$ in (4) [8]. Both can be used for deriving opinions regarding other users' opinions. Based on recommendation $\otimes$, $user_i$'s opinion about event e due to $user_j$ is derived from $user_j$'s recommendation. In (4), let the degree of trust of $user_i$ to $user_j$ be b_j^i. Then, b_e^{ij}, d_e^{ij} and u_e^{ij} are the belief, disbelief, and uncertainty values, respectively, about e of $user_i$ being persuaded by $user_j$ and can be determined as $b_j^i \times b_e^j$. d_e^{ij} is determined by multiplying b_j^i and d_e^j. $user_j$'s certainty can be defined as $1 - u_e^j$ (i.e., $b_e^j + d_e^j$). Let ω_e^{ij} be $user_i$'s opinion about e recommended by $user_j$, and let ω_j^i be $user_i$'s opinion about the degree of trust in $user_j$. Mathematically, ω_e^{ij} can be defined as follows:

$$\omega_e^{ij} = \omega_j^i \otimes \omega_e^j = \{b_e^{ij}, d_e^{ij}, u_e^{ij}\},$$
$$where \begin{cases} b_e^{ij} = b_j^i \times b_e^j \\ d_e^{ij} = b_j^i \times d_e^j \\ u_e^{ij} = 1 - b_j^i \times (1 - u_e^j) \end{cases} \quad (3)$$

The effect of the consensus operator $\oplus$ can help reduce uncertainty by applying opinions from both $user_i$ and $user_j$. The consensus opinion $\omega_e^{i,j}$ between $user_i$ and $user_j$ on event e can be defined as follows [8]:

$$\omega_e^{i,j} = \omega_e^i \oplus \omega_e^j = \{b_e^{i,j}, d_e^{i,j}, u_e^{i,j}\},$$
$$where \begin{cases} b_e^{i,j} = (b_e^i u_e^j + b_e^j u_e^i)/(u_e^i + u_e^j - u_e^i u_e^j) \\ d_e^{i,j} = (d_e^i u_e^j + d_e^j u_e^i)/(u_e^i + u_e^j - u_e^i u_e^j) \\ u_e^{i,j} = (u_e^i u_e^j)/(u_e^i + u_e^j - u_e^i u_e^j) \end{cases} \quad (4)$$

A consensus operator combines evidences for different users, and several approaches [13][16][18][19] provide different methods to combine opinions based on the consensus operator.

III. PROPOSED METHODOLOGY

This section presents an approach to provide a trust-aware QoS method for predicting the QoS values of web services. Figure 3 shows a global viewpoint of our approach, including both input and output of two functions: 1) to help service selection with QoS prediction 2) to help evaluate a new composite service. Our work in this paper is focused on function one (prediction part), which involves three stages: In Stage 1, indirect similarity between two users is determined by applying Subjective Logic [8]. In Stage 2, a set of similar users is selected according to their direct and indirect similarities to

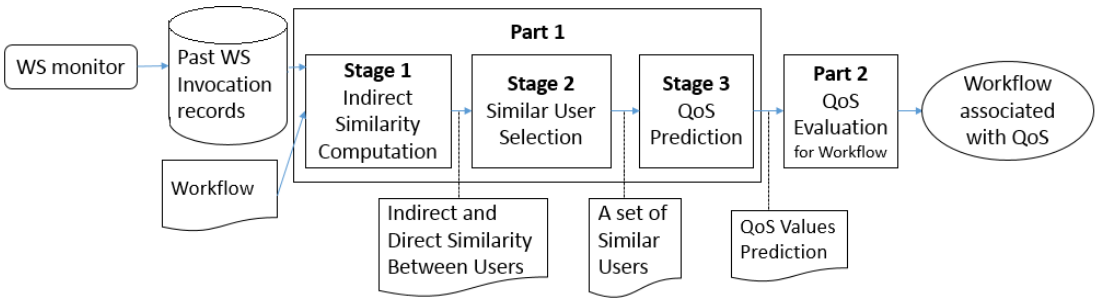

Figure 3. Workflow of trust-aware collaborative QoS prediction and evaluation.

the designated user. In Stage 3, QoS values are predicted by employing the QoS values of similar users. The three stages are described below.

A. Stage 1: Indirect Similarity Computation

The first stage introduces the *user-service* matrix, for calculating direct similarity and subjective logic [8] to determine indirect similarity between users in our approach.

1) Direct Similarity Calculation

A user-service matrix is a 2-dimensional matrix, where each row represents a distinct service to be invoked and each column represents a distinct user. Each entry in the matrix contains some recorded QoS values, which are usually called *invocation* records.

In our work, such entries in a user-service matrix are defined to contain three tuples, where the first tuple is reliability value, second is response time, and third is throughput for the corresponding service and invoker (user). The entry is called *null* when all values inside it are zero. For example, $user_1$ has never invoked $service_2$, and entry $E_{1,2}$ is null. $service_1$ has been invoked by $user_1$, and $E_{1,1}$ contains (90%, 100ms, 24), i.e., the reliability of $service_1$ is 90%, response time is 100 ms, and throughput is 24 kbps.

By applying PCC [11], direct similarity between $user_i$ and $user_j$ is computed by employing (5) according to QoS values of the services invoked by both of them.

$$Sim(i,j) = \frac{\sum_{s \in S_i \bigcap S_j} (E_{i,s} - \overline{E_i})^T (E_{j,s} - \overline{E_j})}{\sqrt{\sum_{s \in S_i \bigcap S_j} (|E_{i,s} - \overline{E_i}|)^2} \sqrt{\sum_{s \in S_i \bigcap S_j} (|E_{j,s} - \overline{E_j}|)^2}} \quad (5)$$

$$\overline{E_i} = \frac{1}{|s_i|} \sum_{s \in S_i} E_{i,s} \quad (6)$$

where i and j represent $user_i$ and $user_j$ separately, S_i and S_j are two sets of services invoked by $user_i$ and $user_j$ separately, $S_i \bigcap S_j$ is a set of services invoked by both of them and $\overline{E_i}$ is the average QoS value of all services invoked by $user_i$ shown in (6).

2) Indirect Similarity Calculation

In general, the greater the number of services invoked by both users, the greater is the number of common invocation records owned by them. Consider two users $user_i$ and $user_j$. To determine $user_i$'s uncertainty toward the notion that "applying $user_j$'s invocation records to predict QoS might be useful," the number of services invoked by both users may be adopted as evidence. When the number of common invocation records is larger, $user_i$ might be more sure that applying $user_j$'s invocation records to predict QoS is useful. Furthermore, the uncertainty can be determined using (7), a formula derived from [6]:

$$u(i,j) = (n_{i,j} + 1)^{-m} \quad (7)$$

where $n_{i,j}$ denotes the number of services invoked by $user_i$ and $user_j$. m is a positive number.

Practically, the number of services invoked is very small [4]. If m increases (e.g., to be greater than 2), the uncertainty determined using (7) decreases dramatically. Moreover, this indicates the lower uncertainty is achieved when fewer services are invoked by both users. Such an uncertainty value is impractical, and it is better and reasonable to assign m as 1 in our experience.

In the case that two users have high similarity values (e.g., high PCC value), the invocation records of one user might be useful for predicting the QoS value of the other user. To derive $user_i$'s belief value for "Applying $user_j$'s invocation records to predict QoS is useful," direct similarity (i.e., PCC value) between $user_i$ and $user_j$ may be adopted as evidence. Such a volume for $user_i$ can be defined according to (8), which is derived from [6]:

$$b(i,j) = \frac{1}{2}(1 - u(i,j))(1 + Sim(i,j)) \quad (8)$$

In (8), the certainty value is $1 - u(i,j)$. When $user_i$ and $user_j$ have a higher PCC value, they might have higher belief as well. Thus, $1 + Sim(i,j)$ is adopted to compute the weight for deciding the ratio of belief to disbelief. Moreover, equation (8) includes a one-half operation to restrict $b(i,j)$ within the interval [0,1]. Correspondingly, the disbelief of $user_i$ to such an application can be determined according to (2) (i.e., $b(i,j) + d(i,j) + u(i,j) = 1$).

An opinion can be direct or indirect. The value of a direct opinion can be derived by (7) and (8), which compute uncertainty and belief values [20]. The value of an indirect opinion can be derived from the recommendations of other users using the recommendation operator [8]. In (1), the PCC value between $user_i$ and $user_j$ is assigned as zero when neither of them invokes a service. In this case, $user_j$ is not put into the set of similar users for predicting the QoS value of $user_i$. However, when both users are similar to a third user, the invocation records of $user_j$ may be helpful for predicting the QoS value of $user_i$ because the similarity can be propagated via the third user [13]. To introduce the effect of indirect similarity, indirect opinion may be adopted.

An example in Figure 4 shows how to derive an indirect opinion. In the example, the belief values of trust degree for $User_i$ to ($User_k$, $User_l$ and $User_m$) and ($User_k$, $User_l$ and $User_m$) to $User_j$ are both larger than a threshold. Thus,

 ISBN: 978-1-61208-387-2

indirect opinion between $User_i$ and $User_j$, $\omega_j^{i(k,l,m)}$, can be determined using the recommendations of $User_k$, $User_l$, and $User_m$ [8].

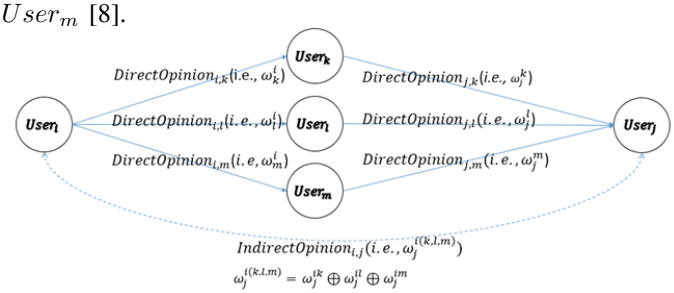

Figure 4. Example for Determining Indirect Opinion.

The indirect opinion of two users $user_i$ and $user_j$ can be defined as the consensus of direct opinions of users who have direct opinions about both aforementioned users. According to the example shown in Figure 4, $\omega_j^{i(k,l,m)}$ can thus represent the indirect opinion from $User_j$ to $User_j$. The indirect similarity between $user_i$ and $user_j$ can determined using indirect opinion by applying (9). Figure 5 shows the distribution of indirect similarity values and indicates that the growth rate of belief to indirect similarity value depends on the uncertainty value.

$$Sim'(i,j) = 2 \times (1 - u_j^i) \times b_j^i - 1 \quad (9)$$

1) where u_j^i is the uncertainty value and b_j^i is the belief value for $user_i$ to $user_j$. 2) direct opinions are obtained from users selected following the process described in the next paragraph. $1 - u_j^i$ is $user_i$' s certainty, employed to be a weighting value for adjusting the growth rate of belief to indirect similarity by multiplying b_j^i and $(1-u_j^i)$. Then, we define Sim' as multiplication by 2 and subtraction of 1 to restrict its value to [-1,1].

To select appropriate users to recommend the opinion, we define an opinion threshold (OThreshold) between 0 to 1 in order to determine the users whose opinions can affect the designated user effectively. OThreshold is helpful for predicting QoS.

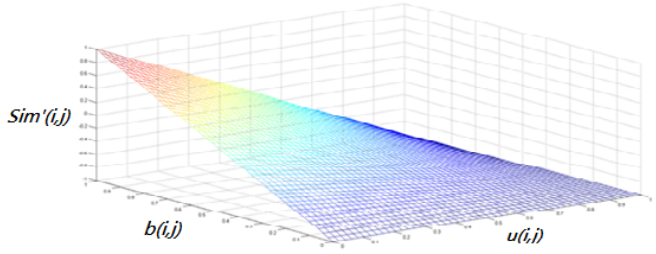

Figure 5. Distribution of (9).

Algorithm 1 details the procedure for deriving indirect similarity. Lines 3-7 select the users whose belief values from $User_i$ to them and from them to $User_j$ are larger than $OThreshold$ and place them into $TrustUserSet$. Indirect opinion between $User_i$ and $User_j$ is initialized by their direct opinion. Lines 8-11 represent a loop that considers all recommendations from the users in the trust user set and determines indirect opinion of each user for $User_i$ to $User_j$. At each turn of the loop, the indirect opinion value due to or via recommendation of $User_l$ is determined using the recommendation operator at line 9. Each indirect opinion is combined with the consensus operator at line 10. Because u_j^i and b_j^i are two attributes in ω_j^i, $Sim'(i,j)$ is computed after ω_j^i is computed.

Algorithm 1 Deriving Indirect Similarity

Input: $User_i$, $User_j$ and $OThreshold$
Output: Indirect similarity $Sim'(i,j)$ between users

1: **function** CALC_INDIRECTSIMILARITY()
2: $TrustUserSet \leftarrow \emptyset$,$w_l^i \leftarrow \emptyset$, $w_j^i \leftarrow \emptyset$;
3: **for** each user $User_k$ **do**
4: **if** both $b(i,k)$, $b(k,j) > OThreshold$ **then**
5: $TrustUserSet \leftarrow TrustUserSet \bigcup \{User_k\}$
6: **end if**
7: **end for**
8: **for** each user IN $TrustUserSet$ named as $User_l$ **do**
9: $\omega_l^i \leftarrow \omega_l^i \otimes \omega_j^l$ ▷ recommended by $User_l$ with recommendation operator
10: $\omega_j^i \leftarrow \omega_j^i \oplus \omega_l^i$ ▷ ω_l^i is added to ω_j^i by consensus operator
11: **end for**
12: **return** $Sim'(i,j)$ ▷ defined in (9)
13: **end function**

Figure 6. Deriving Indirect Similarity

B. Stage 2: Similar Users Selection

To select similar users who may help predict QoS values for the designated user, the first step is to divide users into two groups according to a given number k, where the PCC values between the first-group users (i.e., similar users) and the designated user are larger than those between the second-group users and the designated user. In general, QoS value predictions are inaccurate when k is unsuitable. For example, if the value of k is too large, users not similar to the designated user might be selected. Because there is no effective method to determine a suitable k value for division currently, an eigenvalue k can be adopted to be the number of first-group users instead. We study the influence of k and find that QoS values are more accurate and suitable when k is set to be within a specific range (set as 15 here). Besides, in a cold-start situation, PCC may not be able to find an adequate number of similar users. Our approach adopts indirect similarity values to improve user selection in the case there are less than k users selected owing to direct similarities. The selection approach associated with indirect similarity is the same as the first one except 1) the users are those who were not selected before, 2) selection data is indirect similarity, and 3) amount being selected is equal to k–the number of users selected in the first stage.

Algorithm 2 is designed for this selection. Each user in $AllUsers$ contains PCC values and indirect similarity values, determined using (5) and Algorithm 1, respectively, and the calculations are set in lines 2-5 of said algorithm. Lines 7-14 constitute a while loop for selecting similar users according to their PCC values to $User_i$. Lines 16-23 perform selection based on indirect similarity if the first while loop cannot get k users.

C. Stage 3: QoS Prediction

The QoS value predictions are determined according to (10) based on Resnicks's formula [11][21]. Let $\widehat{E_{i,s}}$ be $user_i$'s QoS value prediction for $service_s$, $\overline{E_i}$ and $\overline{E_j}$ represent the mean QoS values of all services invoked by $User_i$ and $User_j$

Algorithm 2 Selecting Similar Users

Input: $User_i$, an eigenvalue k, and $AllUsers$
Output: a set of similar users to $User_i$

```
1: function SIMILAR_USER_SELECTION()
2:     for each user named User_j in AllUsers do
3:         User_j.Similarity ← Sim(User_i, User_j)
4:         User_j.IndirectSimilarity ← Calc_IndirectSimilarity(User_i, User_j, OThreshold)
5:     end for
6:     select_count = 0
7:     while select_count < k do
8:         Fetch a user from AllUsers who has the greatest PCC value and is not in Similar_Users, and name it as User_j
9:         if User_j.Similarity ≤ 0 then
10:            break
11:        end if
12:        Similar_Users ← Similar_Users ⋃ {User_j}
13:        select_count = select_count + 1
14:    end while
15:    if select_count < k then
16:        while select_count < k do
17:            Fetch a user from AllUsers who has the greatest indirect similarity value and is not in Similar_Users, and name it as User_j
18:            if User_j.IndirectSimilarity ≤ 0 then
19:                break
20:            end if
21:            Similar_Users ← Similar_Users ⋃ {User_j}
22:            select_count = select_count + 1
23:        end while
24:    end if
25:    return Similar_Users
26: end function
```

Figure 7. Selecting Similar Users

respectively, and $S(i)$ be a set of users similar to $user_i$ generated by Algorithm 2 and $W_{i,j}$ be a weight value indicating the similarity between $user_i$ and $user_j$. Here $Similarity_{i,j}$ refers to the similarity between $user_i$ and $user_j$.

$$\widehat{E_{i,s}} = \overline{E_i} + \sum_{j \in S(i), E_{j,s} \neq null} W_{i,j} \times (E_{j,s} - \overline{E_j}) \quad (10)$$

$$W_{i,j} = \frac{Similarity_{i,j}}{\sum_{k \in S(i)} Similarity_{i,k}} \quad (11)$$

Algorithm 3 is designed to predict the QoS value of $user_i$. In the algorithm, a set of similar users is selected at line 2 by applying Algorithm 2. The predictive QoS value is initialized with the average QoS value of all services invoked by $User_i$ in line 3. Lines 4-9 constitute a loop for updating the predictive QoS value according to the QoS values of all similar users for $Service_s$. Finally, line 10 returns the predicted QoS values of $Service_s$ invoked by $user_i$.

IV. Experiments for QoS Prediction

In this section, we introduce our experiments and results, and compare the results with those from existing works. Part A describes the technique for comparing the evaluation results of QoS prediction. Then, we discuss the influence of parameters such as the number of invoked services, eigenvalue k, and opinion threshold in Part B.

Algorithm 3 Predicting QoS values of a service

Input: $User_i$, and $Service_s$
Output: predicted QoS values of $Service_s$ for $User_i$

```
1: function QOS_PREDICTION()
2:     similar_users ← Similar_User_Selection(User_i, k)
3:     Ê_{i,s} ← Ē_i
4:     for User_j in similar_users do
5:         if E_{j,s} ≠ null then
6:             W_{i,j} ← Similarity_{i,j} / Σ_{k∈S(i)} Similarity_{i,k}        ▷ by (11)
7:             Ê_{i,s} ← Ê_{i,s} + W_{i,j} × (E_{j,s} − Ē_j)
8:         end if
9:     end for
10:    return Ê_{i,s}
11: end function
```

Figure 8. Predicting QoS values of a service

A. Comparison of Prediction Accuracy

To evaluate the accuracy of a QoS prediction technique, Mean Absolute Error (MAE) [4] is adopted for calculating the difference between real and predicted QoS values. In general, the smaller the calculated MAE value, the more accurate is the QoS prediction for a service. MAE is defined as follows [22]:

$$MAE = \frac{\sum_{S \in PS_i} |\hat{E}_{i,s} - E_{i,s}|}{|PS_i|} \quad (12)$$

where PS_i is a set of services whose QoS values are derived by the prediction for $user_i$, and $E_{i,s}$ and $\hat{E}_{i,s}$ is a pair of real and predicted QoS values of $Service_s$ for $user_i$ respectively.

In our evaluation work, the calculation of real QoS values is based on the invocation records collected in [23], and widely adopted as real QoS values for evaluation of QoS prediction methods [4][12][20]. The data in [23] were obtained from www.wsdream.net, a service broker website, and they represent 100 real-world Web services. To monitor the QoS values of these services, the system deployed 150 service consumers on Planet-Lab [24] distributed across 20 countries to invoke the services, and recorded 1,500,000 Web service invocations executed a hundred times by 150 distributed users on 100 Web services. Each invocation record indicates the QoS values of a service invoked by a user, and includes three items: response time, throughput, and execution state (i.e., failure or success).

The computation methods for QoS values are quality dependent. For example, reliability can be determined based on execution state. To minimize errors in our experiments, the response time of a service for a user was defined as the average of the response time from all invocation records of the service for the user. So does throughput, which is defined as the average of all throughput value. The reliability value of $service_j$ for $user_i$ is calculated according to (13):

$$R_{i,j} = \frac{S_{i,j}}{N_{i,j}} \quad (13)$$

where $S_{i,j}$ is the successful invoking times of $service_j$, and $N_{i,j}$ is the invoking times of $service_j$. That is, a smaller $R_{i,j}$ means higher reliability.

Given that each user invokes some services only in the real world, the user-service matrix is sparse. However, it is difficult

TABLE I. PREDICTION ACCURACY COMPARISON

Metric	Methods	Reliability			Response time			Throughput		
		Density 5%	Density 10%	Density 15%	Density 5%	Density 10%	Density 15%	Density 5%	Density 10%	Density 15%
MAE	UMEAN	0.0701	0.0755	0.067	1504.90	1545.41	1419.11	2557.50	2561.75	2548.39
	IMEAN	0.0877	0.0863	0.091	1665.00	1705.83	1758.32	2690.27	2637.06	2732.41
	UPCC	0.0442	0.0398	0.037	2076.14	974.48	859.48	1505.07	1361.50	821.41
	IPCC	0.0426	0.0435	0.0390	1371.27	1072.75	955.25	1635.31	1676.25	1608.47
	HPCC	0.0711	0.0538	0.0456	1663.59	1279.14	1278.79	713.04	532.45	308.76
	MF	0.4899	0.4293	0.3886	1721.41	1487.87	1426.77	1813.59	333.91	222.95
	Our Approach	0.0371	0.0362	0.0350	1363.01	879.75	794.67	1306.74	1153.64	636.70

to derive appropriate sparse matrixes (e.g., in different density) to carry out the experiments from real world databases. In our experiment, the QoS values of 100 Web services invoked by 150 users calculated according to the invocation records are stored in a 150×100 user-service matrix called *answer* matrix and we define an *experimental sparse* matrix, a 150×100 user-service matrix. The QoS values in the experimental sparse matrix are constructed based on the answer matrix. For example, if the density of the sparse matrix is 5%, 5% of the entries in the experimental sparse matrix are filled with real QoS values from the answer matrix and the others are filled with NULL. The experimental sparse matrix is built in steps:

1) A new user-service matrix is built and all entries are set to NULL.
2) Let NE be the number of entries (e.g., 15,000 in our experimental sparse matrix), $density$ be the density of the experimental sparse matrix, and NS be the number of selected entries given as multiplication of NE and $density$. A set of numbers SN is derived randomly by selecting NS non-repetitive numbers within the interval [1,NE].
3) The indexes of selected entries according to the number RN in SN can be derived by (14) and (15)
4) The selected entries the of experimental sparse matrix are filled with the QoS values contained in the corresponding entries of the answer matrix.

$$index_{service} = (RN \div columns\ of\ the\ matrix) + 1 \quad (14)$$

$$index_{user} = RN\ \%\ rows\ of\ the\ matrix \quad (15)$$

To compare the accuracy of our approach with other approaches, we implemented a sequence of methods based on 1) user-based CF approach using PCC (UPCC) [22], 2) item-based CF approach using PCC (IPCC) [25], 3) hybrid PCC (HPCC) [26], 4) user-mean (UMEAN), 5) item-mean (IMEAN), 6) matrix factorization (MF) [3] and 7) our trust-aware QoS prediction approach to predict the QoS values of services. MAE values were calculated based on the predicted QoS values and QoS values from the answer matrix. To study the accuracy under different densities, experimental sparse matrixes of 5%, 10%, and 15% were adopted as the training matrices. To minimize errors, each approach was looped 50 times for 10 randomly selected users, and the average MAE was calculated. Table I summarizes the contributions of our work:

1) Our QoS prediction approach obtains better prediction accuracy in terms of reliability values and response time for densities of 5%-15%. However, MAE values of reliability are accurate and so close in these applied methods because the calculation of service reliability are really high in our dataset.
2) The prediction accuracy in terms of throughput was also better than that of other methods under 5% density. However, when the number of invocation records is large (i.e., density is 10% or 15%), the matrix factorization [3] and hybrid PCC [26] approaches obtained better prediction accuracy for QoS attributes with values close to each other for different users (e.g., throughput).
3) A few previous studies [3][26] dealt with the prediction of one or two distinct QoS attributes. However, our approach is suitable for three QoS attributes.

B. Impacts Observed for Interval Factors

Table I also shows that the MAE values of each QoS attribute are in a distinct range in our approach (e.g., the MAE values of reliability and response time are 0.035-0.0371 and 794-1363, respectively). Observation on line charts with the MAE values in a graph is difficult because the range differences are large. We adopted Normalization Mean Absolute Error (NMAE) [2] to depict the line chars for studying the influence of number of invoked services, k, and opinion threshold, as follows [2]:

$$NMAE = \frac{MAE}{\sum_{S \in PS_i} \frac{\hat{E}_{i,s}}{|PS_i|}} \quad (16)$$

The number of services invoked by a user may change the prediction accuracy in our approach. To study this effect, the number of services was varied from 5 to 50, and the data were incremented by 5. Figure 9 shows the NMAE values of response time, throughput, and reliability in our approach. Their values declined from 0.449 to 0.288, 0.24 to 0.01, and 0.020 to 0.015, respectively, thus indicating that our approach can be improved when a (designated) user invokes a greater number of services.

To study the influence of eigenvalue k on the prediction accuracy, k was varied from 2 to 20. Figure 10 shows that the NMAE value of our approach is the smallest when k was 12 or 14. This indicates that the predicted QoS values were more accurate when eigenvalue k was within a specific range.

To study the influence of opinion threshold ($OThreshold$) on the prediction accuracy, the data were detected in the opinion threshold at increments of 0.1 from 0.1 to 0.9. Figure 11 shows that the NMAE value obtained using our approach is the smallest when the opinion threshold is 0.2-0.4. This indicates that the predicted QoS values were more accurate when the opinion threshold was within a specific range.

 ISBN: 978-1-61208-387-2

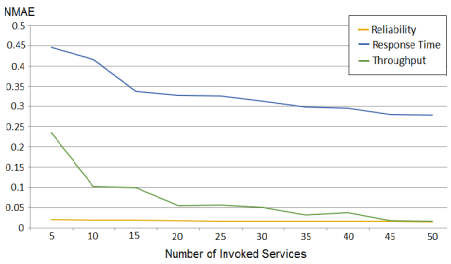

Figure 9. Influence of invoked services.

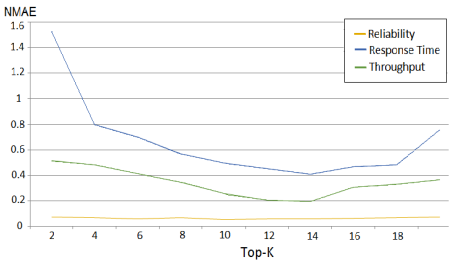

Figure 10. Influence of eigenvalue k.

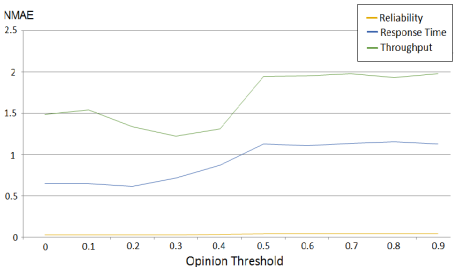

Figure 11. Influence of opinion threshold.

V. Conclusion and Future Work

In this paper, we present a trust-aware approach to predict QoS values of services more accurately. In our approach, the opinions of selected users are used to improve PCC values. These opinions are selected based on three factors, namely, belief, disbelief, and uncertainty, of the designated users who share a greater number of common services. Moreover, we introduce the indirect similar property to help select users for cold-start services. The experiments indicate that our approach provides better prediction for service selection.

There are at least two issues that warrant further study: 1) Only three QoS attributes were adopted in our approach. The accuracy of QoS values might be improved by applying a greater number of QoS attributes. 2) Indirect similarity might improve with the use of additional features such as user location, reputation of provider, and user preferences.

References

[1] Z. Liang, H. Zou, J. Guo, F. Yang, and R. Lin, "Selecting web service for multi-user based on multi-qos prediction," in Services Computing, 2013 IEEE International Conference on, June 2013, pp. 551–558.

[2] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. C. Zhou, and Z. Wu, "Predicting quality of service for selection by neighborhood-based collaborative filtering," Systems, Man, and Cybernetics: Systems, IEEE Transactions on, vol. 43, no. 2, March 2013, pp. 428–439.

[3] Z. Zheng, H. Ma, M. Lyu, and I. King, "Collaborative web service qos prediction via neighborhood integrated matrix factorization," Services Computing, IEEE Transactions on, July 2013, pp. 289–299.

[4] Z. Zheng and M. Lyu, "Collaborative reliability prediction of service-oriented systems," in Software Engineering, 2010 ACM/IEEE 32nd International Conference on, vol. 1, May 2010, pp. 35–44.

[5] Z. Liu, Z. Liu, and T. Lu, "A location and time related web service distributed selection approach for composition," in Grid and Cooperative Computing, 9th International Conference on, Nov 2010, pp. 296–301.

[6] G. Pitsilis and S. J. Knapskog, "Social trust as a solution to address sparsity-inherent problems of recommender systems," Recommender Systems and the Social Web, 2009, pp. 33–40.

[7] A. JÃÿsang, "Reliability analysis with uncertain probabilities," in Proceedings of the 4th International Conference on Probabilistic Safety Assessment and Management (PSAM4). Springer, Heidelberg, 1998.

[8] A. Jøsang, "A logic for uncertain probabilities," Int. J. Uncertain. Fuzziness Knowl.-Based Syst., vol. 9, no. 3, Jun. 2001, pp. 279–311. [Online]. Available: http://dl.acm.org/citation.cfm?id=565980.565981

[9] Z. Zheng, Y. Zhang, and M. Lyu, "Investigating qos of real-world web services," Services Computing, IEEE Transactions on, vol. 7, no. 1, Jan 2014, pp. 32–39.

[10] R. Burke, "Hybrid recommender systems: Survey and experiments," User Modeling and User-Adapted Interaction, vol. 12, no. 4, Nov. 2002, pp. 331–370.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, ser. CSCW '94. ACM, 1994, pp. 175–186.

[12] J. Zhu, Y. Kang, Z. Zheng, and M. Lyu, "Wsp: A network coordinate based web service positioning framework for response time prediction," in Web Services (ICWS), 2012 IEEE 19th International Conference on, June 2012, pp. 90–97.

[13] G. Pitsilis and S. J. Knapskog, "Social trust as a solution to address sparsity-inherent problems of recommender systems," in Proceedings of 2009 ACM Conference on Recommender Systems, 2009, pp. 33–40.

[14] K. Chard, S. Caton, O. Rana, and K. Bubendorfer, "Social cloud: Cloud computing in social networks," in Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, July 2010, pp. 99–106.

[15] P. Singla and M. Richardson, "Yes, there is a correlation: - from social networks to personal behavior on the web," in Proceedings of the 17th International Conference on World Wide Web, ser. WWW '08. New York, NY, USA: ACM, 2008, pp. 655–664. [Online]. Available: http://doi.acm.org/10.1145/1367497.1367586

[16] G. Pitsilis and L. Marshall, "Modeling trust for recommender systems using similarity metrics," in Trust Management II, ser. The International Federation for Information Processing, Y. Karabulut, J. Mitchell, P. Herrmann, and C. Jensen, Eds. Springer, 2008, vol. 263, pp. 103–118.

[17] J. O'Donovan and B. Smyth, "Trust in recommender systems," in Proceedings of the 10th International Conference on Intelligent User Interfaces, ser. IUI '05. New York, NY, USA: ACM, 2005, pp. 167–174. [Online]. Available: http://doi.acm.org/10.1145/1040830.1040870

[18] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in Proceedings of the 29th Australasian Computer Science Conference - Volume 48, ser. ACSC '06. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2006, pp. 85–94. [Online]. Available: http://dl.acm.org/citation.cfm?id=1151699.1151710

[19] A. Josang and T. Bhuiyan, "Optimal trust network analysis with subjective logic," in Emerging Security Information, Systems and Technologies, 2008. SECURWARE '08. Second International Conference on, Aug 2008, pp. 179–184.

[20] Y. Zhang, Z. Zheng, and M. Lyu, "Wspred: A time-aware personalized qos prediction framework for web services," in Software Reliability Engineering, IEEE International Symposium, Nov 2011, pp. 210–219.

[21] P. Resnick and H. R. Varian, "Recommender systems," Commun. ACM, vol. 40, no. 3, Mar. 1997, pp. 56–58. [Online]. Available: http://doi.acm.org/10.1145/245108.245121

[22] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, ser. UAI'98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 43–52. [Online]. Available: http://dl.acm.org/citation.cfm?id=2074094.2074100

[23] Z. Zheng, Y. Zhang, and M. Lyu, "Distributed qos evaluation for real-world web services," in Web Services (ICWS), 2010 IEEE International Conference on, July 2010, pp. 83–90.

[24] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: An overlay testbed for broad-coverage services," SIGCOMM Comput. Commun. Rev., vol. 33, no. 3, Jul. 2003, pp. 3–12. [Online]. Available: http://doi.acm.org/10.1145/956993.956995

[25] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in Proceedings of the 10th International Conference on World Wide Web, ser. WWW '01. New York, NY, USA: ACM, 2001, pp. 285–295. [Online]. Available: http://doi.acm.org/10.1145/371920.372071

[26] Z. Zheng, H. Ma, M. Lyu, and I. King, "Wsrec: A collaborative filtering based web service recommender system," in Web Services, 2009. ICWS 2009. IEEE International Conference on, July 2009, pp. 437–444.

 ISBN: 978-1-61208-387-2