Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/100308

Improved Active Web Service Recommendation Based on Usage History

Guosheng Kang[1,*], *Jianxun Liu*[2], *Mingdong Tang*[2], *Buqing Cao*[2] *and Yu Xu*[3]

[1] School of Computer Science, Shanghai Key Laboratory of Data Science, Fudan University, Shanghai, China
[2] Department of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, China
[3] CNGL Centre for Global Intelligent Content, Trinity College Dublin, Ireland

Received: 8 Oct. 2015, Revised: 8 Jan. 2016, Accepted: 9 Jan. 2016
Published online: 1 May 2016

Abstract: With the increasing adoption and presence of Web services, how to recommend Web services to users that satisfy their potential functional and non-functional requirements effectively has become an important and challenging research issue. In this paper, we propose an enhanced Web service recommendation approach, named iAWSR (improved Active Web Service Recommendation), that explores service usage history of users to actively recommend Web services for them. In iAWSR, we propose new methods for computing functional similarity and non-functional similarity of Web service candidates, and a hybrid metric of similarity is developed by combining the two sources of similarity measurement. iAWSR then ranks publicly available Web services based on values of the hybrid metric of similarity, so that a top-k Web service recommendation list is generated for the user. We propose an effective overall evaluation metric to evaluate our improved approach. Large-scale experiments based on real-world Web service datasets are conducted. Experimental results show that iAWSR outperforms the existing approach AWSR on Web service recommendation performance.

Keywords: Web service, service recommendation, functional similarity, usage history, QoS utility

1 Introduction

Web services have been one of the standard technologies for sharing data and software, and the number of available Web services increases drastically on the Internet [1]. The explosive growth of Web services poses key challenges for discovering Web services that satisfy users' functional and non-functional requirements. Traditional Web service discovery approaches center around UDDI (Universal Description, Discovery and Integration) registries [2]. Unfortunately, UDDI is no longer the choice for publishing Web services, evidenced by the shutdown of the public UDDI registries by big players such as IBM, Microsoft, and SAP. Over the last few years, a considerable number of Web service search approaches have been proposed [3], and several Web service search engines, such as *Web Service List*[1], *Xmethods*[2], *seekda*[3], and *Web Service Supermarket*[4], have emerged. These search engines extensively exploit keyword-based search techniques. In a recent work by Zheng et al. [3], a Web service search engine is designed and developed that ranks Web services not only by functional similarities to a user's source query, but also by non-functional QoS characteristics of Web services. However, these approaches for Web service discovery are highly dependent on user queries.

Recently, Web service recommendation systems have received much attention, which recommend Web services to active users with high QoS (Quality of Service) in a proactive way from a larger number of Web service candidates. Most of the existing service recommendation approaches are based on Collaborative Filtering (CF) techniques [4,5]. These approaches first compute similarity of users or services, and then predict missing QoS values for users based on the QoS records of similar users or services. The Web services with top QoS values in a certain QoS attribute (i.e., response time, throughput,

[1] http://www.webservicelist.com
[2] http://www.xmethods.net
[3] http://webservices.seekda.com
[4] http://49.123.2.23:8080/WSSM/English/index

* Corresponding author e-mail: guoshengkang@gmail.com

etc.) are recommended to the active user. The rationale behind CF based Web service recommendation is that if two users perceived similar QoS values on their commonly invoked Web services, they are deemed as similar users, and it is likely that new Web services requested by a user will be similar to those of his similar users. Therefore, CF based approaches require there is enough similar users for prediction. In reality, a user probably only used a few services from a large pool of services, therefore the user-service matrix is likely sparse. As a result, CF based approaches becomes infeasible. Content-based recommendation can well address the above sparsity problem, since it recommends an item to a user based upon a description of the item and a profile of the user's interest. The other important limitation of CF based service recommendation approaches is that they solely focus on predicting missing QoS values but take little consideration on users' QoS preference. Without the QoS preference, QoS utility of Web service candidates cannot be identified for the target user. A Web service with high QoS values in some QoS attribute may have low QoS utility, because it may own low QoS values in other QoS attributes. Therefore, recommending services with high QoS value in some QoS attribute is insufficient. Based on the above observations, the active user's potential functional interest and QoS preference should be acquired in real Web service recommendation scenarios. With these information, recommendation system can recommend top-k Web services with user-desired functional and non-functional requirements effectively.

In our preliminary work, we presented AWSR (Active Web Service Recommendation), a Web service recommendation approach based on both users' interests and QoS preferences by exploring service usage history to recommend services actively [6]. In this paper, we propose iAWSR (improved Active Web Service Recommendation), an enhanced Web service recommendation approach based on Web service usage history. Improved methods of computing functional similarity and non-functional similarity of Web service candidates are proposed, both of which are then combined to develop a hybrid metric of similarity. The iAWSR ranks publicly available Web services based on the hybrid metric of similarity, so that a top-k Web service recommendation list is yielded for the active user. The contributions of the paper are listed as follows:

- –We present an improved active Web service recommendation approach based on Web service usage history with improved functional similarity and non-functional similarity computation methods of Web service candidates.
- –We propose an effective overall evaluation metric to evaluate our improved Web service recommendation approach.
- –We conduct large-scale experiments on real-world Web service datasets. And experimental results show that iAWSR outperforms the existing approach AWSR on Web service recommendation performance.

The rest of this paper is organized as follows: Section 2 introduces the related work. Section 3 presents the motivating example, and introduces the framework of our improved Web service recommendation approach. Section 4 discusses our improved active Web service recommendation approach in detail, including functional similarity, non-functional similarity, and hybrid similarity and Web service ranking. Section 5 describes the experimental results and analysis. At last, we draw conclusions and discuss our future work in Section 6.

2 Related Work

To discover high quality Web services, QoS models of Web service and QoS-driven Web service selection have attracted considerable attention [7,8,9,10], intending to identify optimal Web services from a set of Web service candidates according to users' requests considering both functional and non-functional requirements. In these study, it is assumed that a user explicitly specifies his functional interest (e.g., by using keywords) and QoS preference requirements, and submits them to the service discovery system [11]. Then the service discovery system matches the user's functional interest and QoS preference requirements, and returns Web services with the best matching degrees to the user [12].

Recently, there has been an increasing interest in actively recommending qualified and preferred Web services to users without their initiating Web service requests. Currently, most existing Web service recommendation approaches are based on Collaborative Filtering (CF). CF algorithms can be divided into two categories: memory-based and model-based. However CF-based Web service recommendation approaches usually focus on memory-based methods [13]. Memory-based CF includes user-based and item-based approaches. User-based CF methods recommend the items liked by users who have similar interests with target users, while item-based CF methods recommend items for users which are similar to the ones they liked before.

Specifically, Shao et al. [14] propose a user-based CF algorithm using PCC (Pearson Correlation Coefficient) to compute similarity between users. For any active user, the missing QoS values of a Web service can be predicted by considering the corresponding QoS values of Web services used by his similar users. Zheng et al. [15] propose a novel hybrid collaborative filtering algorithm for QoS prediction of Web services by systematically combining both item-based PCC (IPCC) and user-based PCC (UPCC). Adapted from [15], Jiang et al. [4] present an improved similarity measurement for users and Web services, which takes the personal characteristics of users and Web services into account when calculating similarity using PCC. Chen et al. [16] recognize the influence of the

characteristics of Web services' QoS. According to their observation, user-perceived QoS attributes (such as responding time, reliability) are highly related to users' physical locations. They propose a scalable hybrid CF algorithm, which incorporates users' locations to help identify similar users. Further, Tang et al. [17] recognize the influence of services' locations in QoS prediction. A location-aware CF approach is proposed for Web service recommendation based on the fusion of QoS similarity and location closeness for both users and Web services. Yao et al. [18] propose a hybrid service recommendation approach by combining CF with content-based features of services. User preferences are statistically estimated using expectation maximization, while QoS preferences for service candidates are not considered. Most recently, some CF based Web service recommendation approaches employe the matrix factorization theory to improve the accuracy of QoS prediction [23,24,25,26].

However, previous QoS-based Web service recommendation approaches usually aim to predict the QoS values of Web services and consider little about Web services' usage history to extract users' QoS preferences. As a consequence, they cannot be directly employed in a real Web service recommendation system to recommend Web services with user-desired non-functional requirement. However, these approaches are very useful to filter and rank Web services on non-functional requirements in Web service selection scenarios. Zhang et al. [19] propose a Web service selection system which combines QoS-based matching score and the collaborative filtering based score. They improve the accuracy of the user similarity calculation based on the recorded user invocation and query history.

To recommend qualified and preferred Web services to end users proactively, our preliminary work proposed AWSR (Active Web Service Recommendation), a service recommendation approach based on both users' functional interests and QoS preferences by exploring service usage history [6]. However, the functional similarity and QoS preference in [6] still need to be improved. In this paper, we present iAWSR (improved Active Web Service Recommendation), an enhanced service recommendation approach based on service usage history as well. We propose improved methods for computing functional similarity and non-functional similarity for service candidates. These improvements are the motivation of this paper. A hybrid metric of similarity is developed to combine functional and non-functional similarity measurements. We also propose an effective overall evaluation metric to evaluate our improved Web service recommendation approach.

3 Preliminaries

3.1 Motivating Examples

Web service description documents are described by Web Service Description Language (WSDL). In [6], to compute the similarity of a Web service candidate with the service usage history, WSDL documents belonging to service usage history are merged into one document, so-called *BigWSDL*. This method is also used in [20]. Functional similarity is computed according to the similarity between *BigWSDL* and the description document of the Web service candidate using TF/IDF (Term Frequency/Inverse Document Frequency) [21] algorithm. However, some terms which actually are not important in single used Web service may become important in *BigWSDL* due to accumulation of the integration of all the used services. In contrast, some unique terms in single used service may become less important in *BigWSDL*. Therefore, the importance of some unimportant terms may be overestimated in *BigWSDL*, while the importance of the real important terms is underestimated.

For example, suppose there are three used Web services in usage history, their description documents are denoted by $WSDL_{h,1}$, $WSDL_{h,2}$ and $WSDL_{h,3}$ respectively. The contained terms are listed in Table 1. The term frequency of t_3 is 1/4 in $WSDL_{h,3}$, but it becomes larger in *BigWSDL* (i.e., 3/10). In contrast, as can be seen from Table 1 that t_1 is unique and important in $WSDL_{h,1}$ with a high frequency 2/3. However, the importance of t_1 in *BigWSDL* is decreased (i.e., 2/10). The above limitations partly affect the accuracy of functional similarity computation. Thus, we improve the functional similarity in this paper, which is introduced in Section 4.1 in detail.

Table 1: Terms of description documents in service usage history

WSDL	**terms**	**term frequency**				
		t_1	t_2	t_3	t_4	t_5
$WSDL_{h,1}$	(t_1,t_1,t_3)	$\frac{2}{3}$	0	$\frac{1}{3}$	0	0
$WSDL_{h,2}$	(t_2,t_3,t_4)	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	0
$WSDL_{h,3}$	(t_2,t_3,t_4,t_5)	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$
BigWSDL	$(t_1,t_1,t_2,t_2,t_3,t_3,t_3,t_4,t_4,t_5)$	$\frac{2}{10}$	$\frac{2}{10}$	$\frac{3}{10}$	$\frac{2}{10}$	$\frac{1}{10}$

In addition, in the preliminary work [6], the potential QoS preference is computed with the average historical QoS preference, which then is used to compute QoS utility for all the Web service candidates. However, in reality a user has different QoS preferences to different Web service candidates. Therefore, QoS preferences to different Web service candidates should be different, and computation of the potential QoS preference needs to be improved.

For example, consider two dimension QoS attributes. An user invoked three Web services, which are $WS_{h,1}$,

$WS_{h,2}$ and $WS_{h,3}$ respectively. The historical QoS preferences are listed in Table 2. As can be seen from Table 2, the user has opposite QoS preferences to $WS_{h,1}$ and $WS_{h,3}$. It is reasonable that users have similar QoS preferences to Web services with similar functionality. If we use the average QoS preference (0.5, 0.5) to compute the QoS utility for a Web service candidate which is similar to $WS_{h,1}$, it would be undesirable. This paper improves the accuracy of a user's potential QoS preferences for different Web service candidates, which is introduced in Section 4.2 in detail.

Table 2: QoS preferences for the Web services in usage history

WS	QoS preference
$WS_{h,1}$	(0.3, 0.7)
$WS_{h,2}$	(0.5, 0.5)
$WS_{h,3}$	(0.7, 0.3)
average QoS preference	(0.5, 0.5)

3.2 System Framework and Architecture

Now we describe the framework of our improved active Web service recommendation approach. As shown in Figure 1, iAWSR recommends a top-k Web service recommendation list for the active user according to the service usage history and Web service candidates which are collected from the Internet.

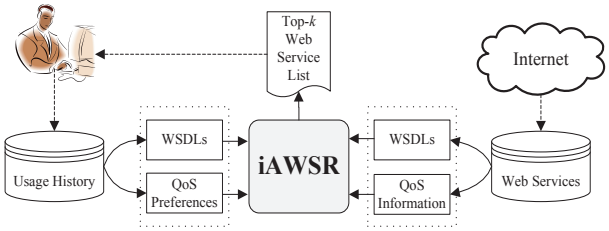

Fig. 1: Framework of improved active Web service recommendation

The iAWSR is the core of the framework. It can be seen from Figure 1 that iAWSR must be provided with service usage history and available Web services on the Internet. More detailedly, WSDL description documents and QoS preferences of used Web services from usage history have to be provided, and WSDL description documents and QoS information of Web service candidates have to be provided.

Figure 2 illustrates the architecture of iAWSR. iAWSR first computes the similarity between used Web services and Web service candidates based on their WSDL documents using TF/IDF algorithm. Then the potential functional similarity to a Web service candidate can be acquired with the average similarity between the Web service candidate and all the used Web services. While please note that any other service similarity calculation methods could extend our approach, like the method in [27,28]. The potential QoS preference to a Web service candidate can be acquired based on the Web service similarities and historical QoS preferences, where the similarity value to a used Web service determines the contribution of its QoS preference to the potential QoS preference for the Web service candidate. With the potential QoS preference, the QoS utility (i.e., non-functional similarity) can be naturally calculated with weighted summation of QoS. iAWSR combines both functional and non-functional features of Web services for ranking in the Web Service Ranking component. Finally, a top-k Web service recommendation list is then generated based on potential functional similarity and non-functional similarity.

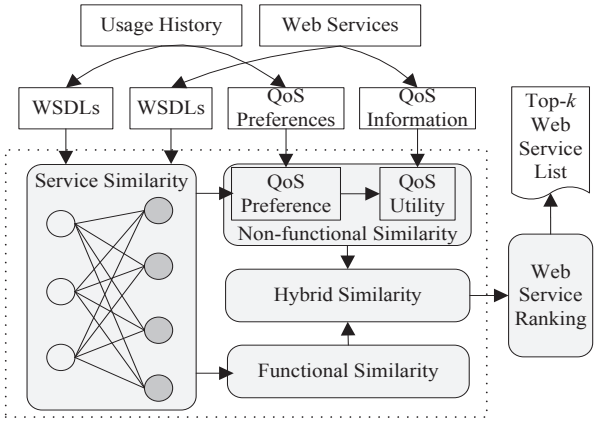

Fig. 2: Architecture of iAWSR

4 Improved Active Web Service Recommendation Approach

Suppose there are M used Web services in recent usage history of the active user, which are $WS_{h,1}$, $WS_{h,2}$, $\cdots$, $WS_{h,M}$ with their corresponding WSDL description documents $WSDL_{h,1}$, $\cdots$, $WSDL_{h,M}$, and $P_{h,i}$ is the QoS preference vector when using $WS_{h,i}$. There are N Web service candidates $WS_{c,1}$, $WS_{c,2}$, $\cdots$, $WS_{c,N}$ for Web service recommendation, whose WSDL description documents are $WSDL_{c,1}$, $\cdots$, $WSDL_{c,N}$ and functional similarity are $S_{c,1}$, $S_{c,2}$, $\cdots$, $S_{c,N}$, QoS vectors are $QS_{c,1}$, $QS_{c,2}$, $\cdots$, $QS_{c,N}$, and their QoS utilities are $U_{c,1}$, $U_{c,2}$, $\cdots$, $U_{c,N}$. With these notations, next we discuss our improved active Web service recommendation approach in detail.

4.1 Functional Similarity

We describe a model of functional similarity with the similarity between a user's usage history and a Web service candidate, since the usage history implies the user interests. All the terms in WSDL documents discovered by the search engine are looked upon as the corpus. We use TF/IDF algorithm to weight the importance of terms in the corpus. TF/IDF is a statistical measure used to evaluate how important a word is in a document across the whole corpus. The importance increases proportionally to the number of times a word appearing in the document but is offset by frequency of the word in the corpus. Variations of the TF/IDF weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance to a given user query [21]. In particular, it analyzes the most common terms appearing in each Web service description document but appearing less frequently in other description documents.

We extract the meaningful words from the WSDL documents to form the corpus. WSDL is the Web service description document which conforms to the rules of XML document. There may be many words or tokens which need to be preprocessed. There are mainly three ways of *WSDL Preprocessing* as follows:

- –*Normalization*: As for the misspelled and abbreviated words in real-word WSDL documents, they need to be replaced by normalized forms.
- –*Word Stemming*: A stem is the basic part of a word that never changes even morphologically infected. As for these words with prefix (e.g., -in, -un, -dis, -non, etc.) or suffixes (e.g., -s, -es, -ed, -er, or,-ing, -ion, etc.), word stemming must be performed for these words to eliminate the difference among inflectional morphemes.
- –*Tokens and Stop-words Removing*: As for tokens and stop-words with little substantive meaning in real-word WSDL documents, they are removed in preprocessing.

A term count in a document is simply the number of times the given term appears in the document. This count is then normalized to prevent a bias towards longer documents (which may have a higher term count regardless of the actual importance of that term in the document) in order to give a measure of the importance of the term. Thus the term frequency $tf(t_j, WSDL_{c,i})$ of the j^{th} term in corpus within the WSDL document $WSDL_{c,i}$ of $WS_{c,i}$ is defined in the simplest case as the occurrence count of the term in the document. After WSDL documents are preprocessed, valid terms are returned. Their term frequencies are then calculated as follows:

$$tf(t_j, WSDL_{c,i}) = \frac{freq(t_j, WSDL_{c,i})}{|WSDL_{c,i}|} \quad (1)$$

where t_j is j^{th} term in corpus; $WSDL_{c,i}$ is the WSDL document of i^{th} Web service candidate $WS_{c,i}$; $freq(t_j, WSDL_{c,i})$ is the occurrence number of t_j in $WSDL_{c,i}$; $|WSDL_{c,i}|$is the number of terms in $WSDL_{c,i}$.

The inverse document frequency $idf(t_j, WSDL_{c,i})$ is a measure of the general importance of the term t_j (obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient), calculated as follows:

$$idf(t_j, WSDL_{c,i}) = \log_2 \frac{|WSDL|}{|\{WSDL_{c,i} : t_j \in WSDL_{c,i}\}|} \quad (2)$$

where $|WSDL|$ is the cardinality of WSDL documents. Hence, $|WSDL|$ is the number of Web service candidates, i.e., $|WSDL| = N$; $|\{WSDL_{c,i} : t_j \in WSDL_{c,i}\}|$ equals to the number of documents where the term t_j appears. If the term is not in the corpus, this will lead to a division-by-zero. We adopt a common way adjusting the denominator to $1 + |\{WSDL_{c,i} : t_j \in WSDL_{c,i}\}|$ for this problem.

The common implementation of TF/IDF gives equal weights to term frequency and inverse document frequency (i.e., $\omega = tf * idf$). However, WSDL documents are generally short. Hence, we choose to give higher weight to the IDF value to normalize the inherent bias with Formula (3). The reason behind this modification is to normalize the inherent bias of TF measure in short documents [22].

$$\omega_{i,j} = tf(t_j, WSDL_{c,i}) * idf^2(t_j, WSDL_{c,i}) \quad (3)$$

A high weight in TF/IDF is reached by a high term frequency and a low inverse document frequency of the term in the whole collection of documents. The weights hence tend to filter out common terms.

A user's interests may change over time. The recently used Web services imply the user's recent interests. To acquire the user's latest interests accurately, we consider the recently used Web services as the usage history. Suppose there are n terms in the corpus. Then with TF/IDF algorithm, each $WSDL_{h,i}$ is transformed into a term vector $Webservice_{h,i}$. Similarly, each $WSDL_{c,i}$ is transformed into a term vector $Webservice_{c,i}$ as well, which are defined respectively as follows:

$$WebService_{h,i} = (\omega_{h,1}, \omega_{h,2}, \cdots, \omega_{h,k}, \cdots, \omega_{h,n})$$

$$WebService_{c,i} = (\omega_{c,1}, \omega_{c,2}, \cdots, \omega_{c,k}, \cdots, \omega_{c,n})$$

where $\omega_{h,j}$ and $\omega_{c,j}$ are the weights of t_j in $WSDL_{h,i}$ and $WSDL_{c,i}$ respectively. Here, if t_j does not appear in $WSDL_{h,i}$ or $WSDL_{c,i}$, then $\omega_{h,j} = 0$ or $\omega_{c,j} = 0$. We measure the text similarity *texSim* between $Webservice_{c,i}$ and $Webservice_{h,j}$, which is calculated as follows:

$$texSim = \frac{\sum_{j=1}^{m} \omega_{c,j} \times \omega_{h,j}}{\sqrt{(\sum_{j=1}^{m} \omega_{c,j}^2) \times (\sum_{j=1}^{m} \omega_{h,j}^2)}} \quad (4)$$

To measure the functional similarity of two Web services, text similarity is not enough. Two Web services with a low text similarity may also contain similar or common service operation. Thus, the similarity of service operations from two Web services should be considered to evaluate their similarity. A Web service operation consists of three elements $OP_f = (\boldsymbol{K}, \boldsymbol{In}, \boldsymbol{Out})$ [3]. The *keywords* (***K***) element of service operation i is a vector of words $\boldsymbol{K}^i = (k_1^i, k_2^i, \cdots, k_{l'}^i)$. The *input* (***In***) and the *output* (***Out***) elements are vectors $\boldsymbol{In}^i = (in_1^i, in_2^i, \cdots, in_{m'}^i)$ and $\boldsymbol{Out}^i = (out_1^i, out_2^i, \cdots, out_{n'}^i)$, where in_k^i and out_k^i are terms appeared in the *input* and *output* respectively. Thus, service operations are described as sets of terms. By applying the TF/IDF measure into these sets, we can also measure the service operation similarity *opSim* of two service operations by using the Cosine Similarity. Considering that a Web service may contain multiple service operations, we measure the operation similarity of two Web services with the maximal service operation similarity of two service operations from the two Web services respectively. Therefore, if two Web services have high text similarity and high service operation similarity, we can say that they are similar. Based on the above observations, we define the similarity $S_{i,j}$ of two Web services as follows, where φ and ϕ are adjustment factors, satisfying $\varphi + \phi = 1$.

$$Sim(WS_{c,i}, WS_{c,j}) = \varphi textSim + \phi opSim \tag{5}$$

Then potential functional similarity $S_{c,i}$ of $WS_{c,i}$ with the Web service usage history can be calculated with the average similarity between $WS_{c,i}$ and all the used Web services, which is calculated as follows:

$$S_{c,i} = \frac{\sum_{j=1}^{M} S_{i,j}}{M} \tag{6}$$

Based on the above discussion, we present the algorithm for functional similarity of Web service candidates, shown in Algorithm 1.

Algorithm 1 Functional Similarity

Input: $WSDL_{h,1}, \cdots, WSDL_{h,M}, WSDL_{c,1}, \cdots, WSDL_{c,N}$
Output: $S_{c,1}, S_{c,2}, \cdots, S_{c,N}$

```
for j=1 to M do
    Transform WSDL_{h,j} into Webservice_{h,j} with TF/IDF;
end for
for i=1 to N do
    S_{c,i} = 0;
    Transform WSDL_{c,i} into Webservice_{c,i} with TF/IDF;
    for j=1 to M do
        S_{i,j} = Sim(WS_{c,i}, WS_{h,j});
        S_{c,i}=S_{c,i}+S_{i,j};
    end for
    S_{c,i}=S_{c,i}/M;
end for
return S_{c,1}, S_{c,2}, ..., S_{c,N};
```

4.2 Non-Functional Similarity

Suppose m criteria are used for assessing the quality of $WS_{c,i}$, i.e., $QS_{c,i} = (q_{i,1}, q_{i,2}, \ldots, q_{i,m})$, where $q_{i,j}$ represents the value of the j^{th} quality attribute. There are two types of QoS attributes. If the higher the value, the lower the quality, this QoS attribute is considered as a negative criterion (e.g., response time and cost). On the other hand, if the higher the value, the higher the quality, this QoS attribute is considered as a positive criterion (e.g., availability and reliability). Each QoS criterion value should be normalized to achieve uniform measurement. In this section, we transform each criterion value to a real value between 0 and 1 by comparing it with the maximum and minimum values of that particular criterion among all available Web service candidates. Concretely, for a negative criterion, the normalized value of $q_{i,j}$ would be scaled by $q'_{i,j}$ according to Formula (7), and for a positive criterion, $q_{i,j}$ would be scaled by $q'_{i,j}$ according to Formula (8) which are defined as follows:

$$q'_{i,j} = \begin{cases} \frac{Q_{\max}(j) - q_{i,j}}{Q_{\max}(j) - Q_{\min}(j)}, & if Q_{\max}(j) \neq Q_{\min}(j) \\ 1, & if Q_{\max}(j) = Q_{\min}(j) \end{cases} \tag{7}$$

$$q'_{i,j} = \begin{cases} \frac{q_{i,j} - Q_{\min}(j)}{Q_{\max}(j) - Q_{\min}(j)}, & if Q_{\max}(j) \neq Q_{\min}(j) \\ 1, & if Q_{\max}(j) = Q_{\min}(j) \end{cases} \tag{8}$$

where the maximum value $Q_{\max}(j)$ and minimum value $Q_{\min}(j)$ of the j^{th} criterion are defined as Formula (9) and (10). We denote $QS'_{c,i}$ as the QoS vector of $WS_{c,i}$ after normalization.

$$Q_{\max}(j) = \max_{\forall i \in [1,n]} q_{i,j} \tag{9}$$

$$Q_{\min}(j) = \min_{\forall i \in [1,n]} q_{i,j} \tag{10}$$

Suppose the potential QoS preference to $WS_{c,i}$ is $P_{c,i}$, which is a weight vector $P_{c,i} = (w_{c,1}, w_{c,2}, \cdots, w_{c,m})$ used to represent the user's preferences given to different QoS criteria with $w_{c,j} \in R_0^+$ and $\sum_{j=1}^{m} w_{c,j} = 1$. Then the QoS utility $U_{c,i}$ of $WS_{c,i}$ is calculated as follows:

$$U_{c,i} = QS'_{c,i} \times P_{c,i}^T = \sum_{j=1}^{m} w_{c,j} \times q'_{i,j} \tag{11}$$

In iAWSR, we derive potential QoS preferences to Web service candidates from historical QoS preferences of the user. It is reasonable to assume that the user has similar QoS preferences to Web services with similar functionality. More similar two Web services are, more similar QoS preferences the user has to them. Based on this rational, the potential QoS preference to $WS_{c,i}$ is calculated as follows:

$$P_{c,i} = \sum_{j=1}^{M} \frac{S_{i,j}}{\sum_{j=1}^{M} S_{i,j}} P_{h,j} \tag{12}$$

where $\sum_{j=1}^{M} S_{i,j} = M \times S_{c,i}$, since $S_{c,i}$ is the average similarity with used Web services; Based on the above discussion, we outline the algorithm for non-functional similarity of service candidates, shown in Algorithm 2.

Algorithm 2 Non-Functional Similarity

Input: $WSDL_{h,1}, \cdots, WSDL_{h,M}, P_{h,1}, P_{h,2}, \cdots, P_{h,M}$, $S_{c,1}, \cdots, S_{c,N}, WSDL_{c,1}, \cdots, WSDL_{c,N}, QS_{c,1}, \cdots, QS_{c,N}$
Output: $U_{c,1}, U_{c,2}, \cdots, U_{c,N}$

for j=1 to M **do**
  Transform $WSDL_{h,j}$ into $Webservice_{h,j}$ with TF/IDF;
end for
for i=1 to N **do**
  Transform $WSDL_{c,i}$ into $Webservice_{c,i}$ with TF/IDF;
  $QS'_{c,i} = normalize\left(QS_{c,i}\right)$;
  $P_{c,i}$=**0**;
  for j=1 to M **do**
    $S_{i,j} = Sim(WS_{c,i}, WS_{h,j})$;
    $P_{c,i} = P_{c,i} + \frac{S_{i,j}}{MS_{c,i}} P_{h,i}$;
  end for
  $U_{c,i} = QS'_{c,i} \times P^T_{c,i}$;
end for
return $U_{c,1}, U_{c,2}, \cdots, U_{c,N}$;

4.3 Hybrid Similarity and Web Service Ranking

After acquiring functional similarity and non-functional similarity of Web service candidates, next we combine the two factors to a hybrid similarity. A final rating score R_i of $WS_{c,i}$ is used to evaluate the hybrid similarity for achieving the recommendation goal, which is calculated as follows:

$$R_i = \lambda \frac{1}{\log_2\left(P_{S_i}+1\right)} + (1-\lambda)\frac{1}{\log_2\left(P_{U_i}+1\right)} \qquad (13)$$

where P_{S_i} is the functional rank position, and P_{U_i} is the non-functional rank position of $WS_{c,i}$ among all Web service candidates. Since the absolute values of functional similarity and QoS utility indicate different features of Web services and include different units and ranges, rank positions rather than absolute values are a better choice to indicate the appropriateness of all Web service candidates [3]. $1/\log_2(p+1)$ calculates the appropriateness value of a candidate in position p for a user's potential requirements. $\lambda \in [0,1]$ defines how much the functionality factor is more important than the non-functionality factor in the final recommendation results. Here, λ can be a constant to allocate a fixed percentage of two parts' contributions to the final rating score R_i. However, it is more reasonable if λ is expressed as a monotone decreasing function with P_{S_i}, shown as follows:

$$\lambda = f\left(P_{S_i}\right)$$

λ is small when the position in the functional similarity rank is lower. This means a Web service is inappropriate if it cannot provide the required functionality to the user no matter how high the QoS utility is. The relationship between recommendation accuracy and the formula of λ could be identified to extend the iAWSR approach further.

5 Performance Evaluation

As mentioned in related work, CF-based approaches only focus on QoS prediction and they do not specify how to recommend Web services for users based on user interests and QoS preferences. In contrast, we explore the active users potential QoS preferences and potential functional interests from the service usage history in our work. Thus, CF based Web service recommendation approaches are not comparable to our proposed approach. Therefore, in this section, we discuss experiments to study the performance of our approach compared with the existing state-of-the-art approach AWSR. We first evaluate the functional and non-functional performance respectively; then we evaluate the overall performance with an effective metric.

5.1 Functional Evaluation

In this experiment, we study the relevance of the recommended Web services to the Web service usage history without considering the non-functional performance of Web services. By comparing our approach with AWSR, we observe that the top-k Web services in our recommended list are highly relevant to the Web service usage history even without any available QoS values.

The benchmark adopted for evaluating the functional performance of our approach is OWL-S service retrieval test collection OWLS-TC v2 [11]. This collection consists of 578 Web services covering 7 application domains (i.e., education, medical care, food, travel, communication, economy, and weaponry), where there are more than 100 Web services in economy, education and travel domains. The dataset statistics is illustrated in Table 3. The benchmark includes WSDL files of the Web services. Since the QoS feature is not considered in this experiment, we set the QoS utility value of each Web service as 1.

Table 3: Dataset statistics

Domain	Communication	Economy	Education	Food
#services	29	206	135	25
Domain	Medical	Travel	Weapon	**Total**
#services	52	106	25	578

We randomly select 10 Web services as the usage history, 5 from communication domain and 5 from

economy domain. If a recommended Web service is also from the same domains that Web service usage history belongs, we say it is relevant in our experiments. Then we measure how many Web services belong to communication and economy domains in the recommendation list with the metrics: recall and precision, which are calculated by:

$$Recall_k = \frac{|Rel \cap Rec_k|}{|Rel|} \tag{14}$$

$$Precision_k = \frac{|Rel \cap Rec_k|}{|Rec_k|} \tag{15}$$

where Rel is the relevant set of Web services for Web service usage history, and Rec_k is the set of top-k Web services in the recommendation results.

Since users tend to check only top few Web services in common recommendation scenarios, an approach with high top-k precision values is practical in reality. Figure 3 shows the experimental results of iAWSR and AWSR. In Figure 3(a), the top-k recall values of iAWSR are higher than that of AWSR. In Figure 3(b), the top-k precision values of iAWSR are also constantly higher than that of AWSR. Similar effects can be observed from Figure 4 and Figure 5, if we select service usage history from education and food domains, or medical and travel domains. Hence, the above experimental results indicate that more relevant Web services are recommended in high positions by our improved approach.

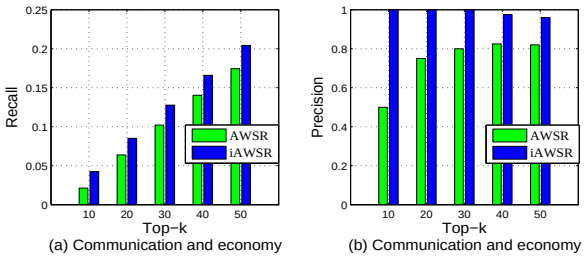

Fig. 3: Recall and precision in communication and economy domains

5.2 Non-Functional Evaluation

To evaluate the non-functional performance, we measure the computation accuracy of potential QoS preference, because the accuracy of QoS preference determines the non-functional performance. Suppose there are 10 Web services in service usage history from 3 domains, and their historical QoS preferences are listed in Table 4.

We compute the average square error between the potential QoS preference and the QoS preferences of the

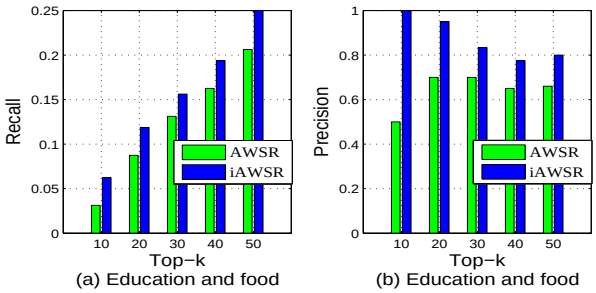

Fig. 4: Recall and precision in education and food domains

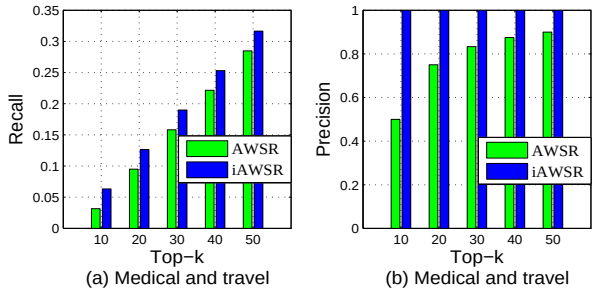

Fig. 5: Recall and precision in medical and travel domains

used Web services which are from the same domain as the Web service candidate, shown in Table 5. For example, in iAWSR approach, the average square error of $WS_{c,3}$ is calculated as follows:

$$\frac{1}{2} \times \left(\sqrt{(0.3123-0.32)^2 + (0.6877-0.68)^2} + \sqrt{(0.3123-0.30)^2 + (0.6877-0.70)^2} \right) = 0.0141$$

Table 4: QoS preferences in usage history

Web service	Preference	Domain
$WS_{h,1}$	(0.42,0.58)	economy
$WS_{h,2}$	(0.37,0.63)	economy
$WS_{h,3}$	(0.38,0.62)	economy
$WS_{h,4}$	(0.39,0.61)	economy
$WS_{h,5}$	(0.40,0.60)	economy
$WS_{h,6}$	(0.41,0.59)	economy
$WS_{h,7}$	(0.80,0.20)	education
$WS_{h,8}$	(0.78,0.22)	education
$WS_{h,9}$	(0.32,0.68)	travel
$WS_{h,10}$	(0.30,0.70)	travel

As can be seen from Table 5 that iAWSR approach shows much lower average square error than AWSR approach, indicating that the potential QoS preferences acquired by iAWSR are more accurate than AWSR.

Therefore, iAWSR achieves better recommendation performance than AWSR under the non-functional evaluation metric.

Table 5: Square error of potential QoS preferences

Web Service	Method	Preference	Square Error	Domain
$WS_{c,1}$	AWSR	(0.4570, 0.5430)	0.0877	economy
	iAWSR	(0.4063, 0.5937)	0.0214	
$WS_{c,2}$	AWSR	(0.4570, 0.5430)	0.4709	education
	iAWSR	(0.8034, 0.1966)	0.0190	
$WS_{c,3}$	AWSR	(0.4570, 0.5430)	0.2079	travel
	iAWSR	(0.3123, 0.6877)	**0.0141**	

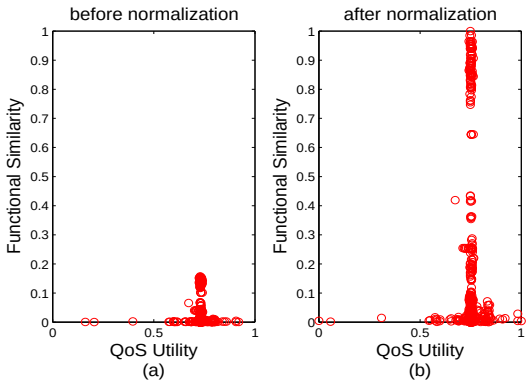

Fig. 6: Distribution of functional similarity and QoS utility values

5.3 Overall Evaluation

To evaluate the overall performance of our approach, both functional and non-functional similarity should be incorporated in the overall metric. [6] introduced DCG (Discounted Cumulative Gain) values as the performance evaluation metrics for functional similarity and QoS utility respectively, which are defined as follows:

$$DCG_k = \sum_{i=1}^{k} \frac{(2^{S_i} - 1)}{\log_2(1 + p_i)} \tag{16}$$

$$DCG_k = \sum_{i=1}^{k} \frac{(2^{U_i} - 1)}{\log_2(1 + p_i)} \tag{17}$$

where S_i and U_i are the functional similarity and QoS utility respectively. p_i is the rank position of i^{th} Web service in the top-k Web service recommendation results. Usually, to evaluate the overall performance, we combine the two factors directly with weighted summation, which may be effective when the two factors are both evenly distributed variables. However, here regarding our research, the distributions of functional similarity and QoS utility values are not evenly distributed. If we combine these two factors with weighted summation directly, the final value of combination would mainly be decided by the relative larger variable. Even though we can normalize values of functional similarity and QoS utility with the maximum difference normalization method similar with Formula (8), their distributions do not change, which is shown in Figure 6.

Similar to Formula (13), functional and non-functional rank positions are used instead of functional similarity and QoS utility values. The large functional similarity and QoS utility values should have large DCG values, so we change P_{S_i} and P_{U_i} to their inverses, shown in Formula (18). Here λ shares the same value as Formula (13).

$$DCG_k = \sum_{i=1}^{k} (\lambda \frac{2^{\frac{1}{P_{S_i}}} - 1}{\log_2(1 + p_i)} + (1 - \lambda) \frac{2^{\frac{1}{P_{U_i}}} - 1}{\log_2(1 + p_i)}) \tag{18}$$

Next, we introduce experiments to evaluate the overall performance against AWSR under the proposed metric. We use the same Web service dataset provided by [6]. As shown in Figure 7, the DCG values of iAWSR are constantly much higher than that of AWSR, and this tendency does not change with the vary of λ. Therefore, our approach outperforms the existing approach AWSR on the overall recommendation performance metric, which proves the effectiveness of our proposed approach.

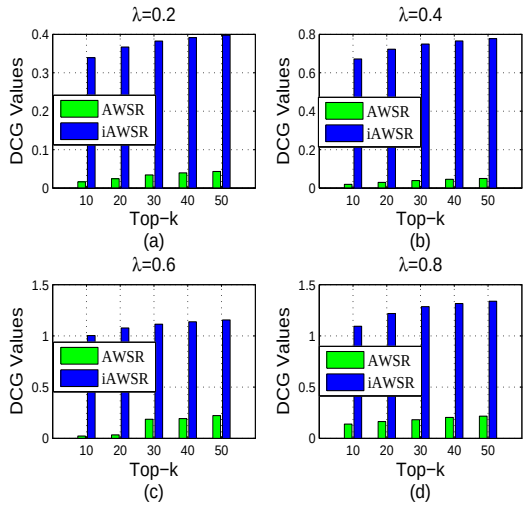

Fig. 7: DCG of top-k Web services

6 Conclusion

In this paper, we propose iAWSR (improved Active Web Service Recommendation), an enhanced Web service

recommendation approach based on Web service usage history. This new approach improves computation accuracy of both functional similarity and QoS preference to Web service candidates, and proposes a hybrid metric of similarity computation by combining functional similarity with non-functional similarity. An effective overall evaluation metric is proposed to evaluate our improved active Web service recommendation approach. Experimental results show that iAWSR outperforms the existing approach AWSR on Web service recommendation performance.

In future work, we will collect real-world data for Web service usage history to evaluate our approach further, and conduct more experiments to study the performance of the new proposed approach. Clustering algorithms will be considered to improve the computation of potential functional similarity and QoS preference.

Acknowledgement

The work was supported by the National Natural Science Foundation of China (No. 61572186, 61572187 and 61402168), the Scientific Research Fund of Hunan Provincial Education Department of China (No.15K043), State Key Laboratory of Software Engineering (SKLSE) of China (Wuhan University) (No. SKLSE2014-10-10), and the Science Foundation Ireland (Grant 12/CE/I2267) as part of CNGL (www.cngl.ie) at Trinity College Dublin.

References

[1] Mike P Papazoglou, Paolo Traverso, Schahram Dustdar, et al. “Service-oriented computing: State of the art and research challenges”. Computer, Vol. 40, No. 11, pp. 38-45, 2007.

[2] Dimitrios Kourtesis, Iraklis Paraskakis. “Combining SAWSDL, OWL-DL and UDDI for semantically enhanced web service discovery”. The Semantic Web: Research and Applications Springer, pp. 614-628, 2008.

[3] Y. Zhang, Z. Zheng, M.R. Lyu. “WSExpress: a QoS-aware search engine for web services”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 91-98, 2010.

[4] Y. Jiang, J. Liu, M. Tang, et al. “An effective Web service recommendation based on personalized collaborative filtering”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 211-218, 2011.

[5] Z. Zheng, M.R. Lyu. “Collaborative reliability prediction of service-oriented systems”. Proceedings of 32nd ACM/IEEE International Conference on Software Engineering, ACM, pp. 35-44, 2010.

[6] G. Kang, J. Liu, M. Tang, et al. “AWSR: Active Web Service Recommendation Based on Usage History”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 186-193, 2012.

[7] S.S. Yau, Y. Yin. “QoS-Based Service Ranking and Selection for Service-Based Systems”. Proceedings of International Conference on Service Computing, IEEE Computer Society, pp. 56-63, 2011.

[8] G. Kang, J. Liu, M. Tang, et al. “Web Service Selection for Resolving Conflicting Service Requests”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 387-394, 2011.

[9] L. Qi, Y. Tang, W. Dou, et al. “Combining local optimization and enumeration for QoS-aware Web service composition”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 34-41, 2010.

[10] M. Alrifai, D. Skoutas, T. Risse. “Selecting skyline services for QoS-based web service composition”. Proceedings of International World Wide Web Conference, ACM, pp. 11-20, 2010.

[11] M. Klusch, B. Fries, K. Sycara. “Automated semantic web service discovery with OWLS-MX”. Proceedings of Autonomous Agents and Multiagent Systems, ACM, pp. 915-922, 2006.

[12] D. Ardagna, R. Mirandola. “Per-flow optimal service selection for Web services based processes”. Journal of Systems and Software, Vol. 83, No. 8, pp. 1512-1523, 2010.

[13] David Goldberg, David Nichols, Brian M Oki, et al. “Using collaborative filtering to weave an information tapestry”. Communications of the ACM, Vol. 35, No. 12, pp. 61-70, 1992.

[14] L. Shao, J. Zhang, Y. Wei, et al. “Personalized qos prediction forweb services via collaborative filtering”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 439-446, 2007.

[15] Z. Zheng, H. Ma, M.R. Lyu, et al. “Wsrec: a collaborative filtering based web service recommender system”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 437-444, 2009.

[16] X. Chen, X. Liu, Z. Huang, et al. “RegionKNN: a scalable hybrid collaborative filtering algorithm for personalized Web servicer ecommendation”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 9-16, 2010.

[17] M. Tang, Y. Jiang, J. Liu, et al. “Location-Aware Collaborative Filtering for QoS-Based Service Recommendation”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 202-209, 2012.

[18] L. Yao, Q.Z. Sheng, A. Segev, et al. “Recommending Web Services via Combining Collaborative Filtering with Content-based Features”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 42-49, 2013.

[19] Q. Zhang, C. Ding, C.H. Chi. “Collaborative Filtering Based Service Ranking Using Invocation Histories”. Proceedings of International Conference on Web Services, IEEE Coputer Society, pp. 195-202, 2011.

[20] B. Cao, J. Liu, M. Tang, et al. “Mashup Service Recommendation based on User Interest and Social Network”. Proceedings of International Conference on Web Services, IEEE Computer Society, pp. 388-394, 2013.

[21] K.S. Jones. “A statistical interpretation of term specificity and its application in retrieval”. Journal of documentation, Vol. 28, No. 1, pp. 11-21, 1972.

[22] S. Robertson. “Understanding inverse document frequency: on theoretical arguments for IDF”. Journal of documentation, Vol. 60, No. 5, pp. 503-520, 2004.

[23] W. Lo, J. Yin, S. Deng, et al. “Collaborative web service qos prediction with location-based regularization”. Proceedings

of 2012 IEEE 19th International Conference on Web Services. IEEE, pp. 464-471, 2012.
[24] Z. Zheng, H. Ma, M. Lyu, et al. "Collaborative Web service QoS prediction via neighborhood integrated matrix factorization". IEEE Transactions on Service Computing, Vol. 6, No. 3, pp. 289-299, 2013.
[25] Y. Xu, J. Yin, W. Lo, et al. "Personalized Location-Aware QoS Prediction for Web Services Using Probabilistic Matrix Factorization". Proceedings of Web Information Systems Engineering. Springer, pp. 229-242, 2013.
[26] P. He, J. Zhu, Z. Zheng, et al. "Location-based Hierarchical Matrix Factorization for Web Service Recommendation". Proceedings of 2014 IEEE International Conference on Web Services. IEEE, pp. 297-304, 2014.
[27] J. Wu, L. Chen, Z. Zheng, et al. "Clustering web services to facilitate service discovery". Knowledge and information systems, Vol. 38, No. 1, pp. 207-229, 2014.
[28] L. Chen, Y. Wang, Q. Yu, et al. "WT-LDA: User Tagging Augmented LDA for Web Service Clustering". International Conference on Service Oriented Computing Springer, pp. 162-176, 2013.

Guosheng Kang received his M.S. degree in computer science from Hunan University of Science and Technology. He in is currently a Ph.D. candidate in School of Computer Science, Fudan University, Shanghai, China. He was a research assistant at Hunan University of Science and Technology from 2011 to 2012. His publications have appeared in some well-known conference proceedings and international journals, such as ICWS, ICSOC, APSCC, IEEE BigData, TSC etc. His research interests include service computing and cloud computing, data-centric business process management.

Jianxun Liu received his M.S. and Ph.D. degree in computer science from the Central South University of Technology in 1997 and Shanghai Jiao Tong University in 2003, respectively. He is now a professor in School of Computer Science and Engineering, Hunan University of Science and Technology. His current interests include workflow management, service computing and cloud computing, semantics and knowledge grid. He has published more than 60 papers in well-known conferences and journals, such as ICWS, SCC, Information and Management, Future Generation Computer Systems, Concurrency and Computation: Practice and Experience, etc. He served as a program committee member or program chair of the major conferences in his area (ICWS, SCC, SCA, CGC, APBPM, TrustCom, DASC, SKG, etc.) and a reviewer for TKDE, TSC, TPDS, etc.

Mingdong Tang received his Ph.D. degree in computer science from the Institute of Computing Technology Chinese Academy of Sciences, China, in 2010. He is now an associate professor in School of Computer Science and Engineering, Hunan University of Science and Technology. His current research interests include distributed computing, service computing and cloud computing. He has published more than 50 papers in well-known conferences and journals, such as TWEB, JWSR, ICWS, SCC, IEEE BigData, SCIENCE CHINA Information Sciences, etc. He is also on the editorial boards of several journals and has served as a program committee member for many international conferences.

Buqing Cao received his M.S. degree in computer science from the Central South University of Technology in 2007, and Ph.D. degree from Wuhan University in 2011. He is now a associate professor in School of Computer Science and Engineering, Hunan University of Science and Technology. His current interests include service computing, social network, software engineering. He has published more than 40 papers in well-known conferences and journals, such as ICWS, SCC, JWSR, IEEE BigData, China Communications, GCC, etc.

Yu Xu received his M.S. degree in computer science from Hunan University of Science and Technology. He is currently a PhD candidate in Trinity College Dublin, Ireland. He has been an IEEE student member since 2011. His publications have appeared in some well-known conference proceedings and international journals, such as ICWS, SCC, SKG, etc. His research interests include service computing and cloud computing, social network.