Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization

Zibin Zheng, Hao Ma, Michael R. Lyu, *Fellow*, *IEEE*, and Irwin King, *Senior Member*, *IEEE*

Abstract—With the increasing presence and adoption of web services on the World Wide Web, the demand of efficient web service quality evaluation approaches is becoming unprecedentedly strong. To avoid the expensive and time-consuming web service invocations, this paper proposes a collaborative quality-of-service (QoS) prediction approach for web services by taking advantages of the past web service usage experiences of service users. We first apply the concept of user-collaboration for the web service QoS information sharing. Then, based on the collected QoS data, a neighborhood-integrated approach is designed for personalized web service QoS value prediction. To validate our approach, large-scale real-world experiments are conducted, which include 1,974,675 web service invocations from 339 service users on 5,825 real-world web services. The comprehensive experimental studies show that our proposed approach achieves higher prediction accuracy than other approaches. The public release of our web service QoS data set provides valuable real-world data for future research.

Index Terms—Web service, QoS prediction, user-collaboration, matrix factorization

1 Introduction

Web services are self-described software applications designed to support interoperable machine-to-machine interaction over a network via standard interfaces and communication protocols [1]. Strongly promoted by the leading industrial companies, web services have been widely employed in a lot of domains. Quality of service (QoS) is usually employed to describe the nonfunctional characteristics of web services. With the growing presence and adoption of web services on the World Wide Web, QoS has become an important selling and differentiating point of the functionally equivalent web services.

In the recent literature, a number of QoS-based approaches have been proposed for web service composition [2], [3], [4], web service selection [5], [6], [7], [8], fault-tolerant web services [9], and so on. Accurate QoS values of web services are required for these QoS-based approaches to work well. To address the fundamental problem of how to obtain the web service QoS values, effective and efficient web service QoS value obtaining approaches are urgently needed. The QoS values of web services can be measured either at the server-side or at the client-side. QoS values measured at the server-side (e.g., price, popularity, etc.) are usually advertised by the service providers and identical for different users, while QoS values measured at the client-side (e.g., response-time, throughput, availability, etc.) can vary widely among users influenced by the unpredictable Internet connections and the heterogeneous user environments. To obtain accurate and personalized client-side web service QoS values for different service users, client-side web service evaluations [10], [11], [12], [13] are usually needed.

However, conducting real-world web service evaluation at the client-side is difficult and sometimes even impossible because: 1) web service invocations may be charged because the web services are usually provided and hosted by other organizations. Even if the web services are free, executing real-world web service invocations for evaluation purposes consumes resources of service providers and imposes costs of service users. 2) It is time-consuming and impractical for service users to evaluate all the web service candidates, since there are a lot of web services in the Internet. 3) Service users are usually not experts on web service evaluation and the common time-to-market constraints make in-depth evaluations of the target web services difficult.

Without sufficient client-side evaluations, accurate web service QoS values cannot be obtained. It is, thus, difficult for various QoS-based approaches, which employ these QoS values as input, to work well. To attack this critical challenge, we propose a neighborhood-integrated matrix factorization (NIMF) approach for collaborative and personalized web service QoS value prediction. The idea is that client-side web service QoS values of a service user can be predicted by taking advantage of the past web service usage experiences of other service users.

To encourage QoS value sharing among service users (usually developers of service-oriented systems), a framework is proposed based on a key concept of Web 2.0, i.e., user-collaboration. In this framework, the users are encouraged to contribute their individually observed web service QoS information to exchange for accurate and personalized web service QoS prediction. Employing the

- *Z. Zheng is with the Shenzhen Research Institute, The Chinese University of Hong Kong, and with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China. E-mail: zbzheng@cse.cuhk.edu.hk.*
- *H. Ma is with Microsoft Research, One Microsoft Way, Redmond, WA 98052. E-mail: haoma@microsoft.com.*
- *M.R. Lyu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, HSB 927, Hong Kong. E-mail: lyu@cse.cuhk.edu.hk.*
- *I. King is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, HSB 908, Hong Kong. E-mail: king@cse.cuhk.edu.hk.*

Manuscript received 19 Jan. 2011; revised 4 Aug. 2011; accepted 15 Nov. 2011; published online 15 Dec. 2011.
For information on obtaining reprints of this article, please send e-mail to: tsc@computer.org, and reference IEEECS Log Number TSC-2011-01-0002.
Digital Object Identifier no. 10.1109/TSC.2011.59.

web service QoS values from different users, our NIMF approach first finds out a set of similar users for the current users by calculating user similarities. Then, the NIMF approach employs both the local information of similar users and the global information of all available QoS values for fitting a factor model, and use this factor model to make personalized web service QoS prediction. By the collaboration of different service users, the QoS values of a web service can be effectively predicted in our approach even the current user did not conduct any evaluation on the web service and has no idea on its internal design and implementation details.

Our approach remedies the shortcomings of previous evaluation approaches [10], [11], [13] by avoiding the expensive and time-consuming real-world web service invocations. Complementary to various QoS-based approaches for web services, which mainly focus on using the QoS values, this paper focuses on providing accurate and personalized QoS values for the service users.

The contributions of this paper are twofold:

- First, we propose an NIMF approach for personalized web service QoS value prediction. Our approach explores the past web service usage experiences of service users by systematically fusing the neighborhood-based and the model-based collaborative filtering approaches to achieve higher prediction accuracy.
- Second, we conduct large-scale experiments and release a real-world web service QoS data set[1] for future research. To the best of our knowledge, the scale of our released web service QoS data set (including 339 distributed service users and 5,825 real-world web services as shown in Fig. 1) is the largest in the field of service computing. Based on this data set, extensive experimental investigations are conducted to study the QoS value prediction accuracy of our approach.

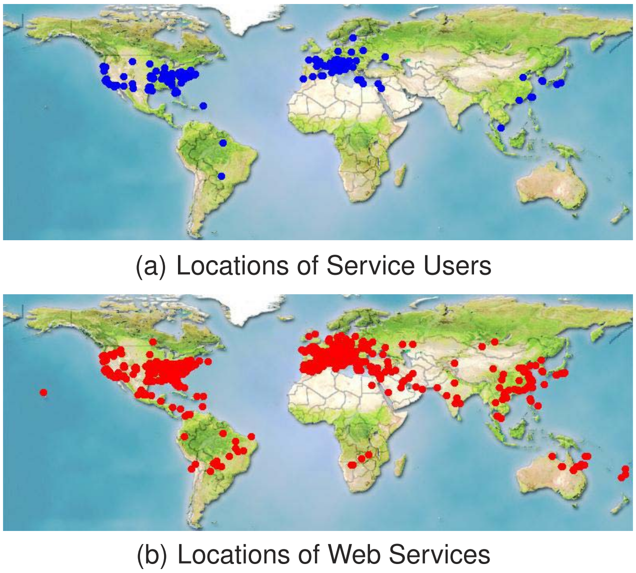
The remainder of this paper is organized as follows: Section 2 presents our collaborative QoS framework and personalized QoS value prediction approach. Section 3 describes our experiments. Section 4 introduces related work, and Section 5 concludes the paper.

2 Collaborative QoS Prediction

In this section, we first present a collaborative QoS framework for collecting QoS values from different users in Section 2.1. Then, based on the collected web service QoS values, we describe the web service QoS value prediction problem in Section 2.2, and propose a solution in Section 2.3 to Section 2.5.

2.1 Collaborative QoS Framework

Quality-of-service (QoS) is usually employed for describing nonfunctional characteristics of web services. While the server-side QoS values provide good indications of the server capacities, client-side QoS values provide more realistic measurements of the performance experienced by service users. Based on the previous investigations of web service QoS [2], [3], [4], the commonly used client-side web service QoS properties include:


(a) Locations of Service Users

(b) Locations of Web Services

Fig. 1. Location Information: (a) locations of service users, a total of 339 service users from 30 countries are plotted; (b) locations of web services, a total of 5,825 real-world web services from 73 countries are plotted. Each user in (a) invoked all the web services in (b). A total of 1,974,675 web service invocation results are collected.

- *Response-time.* The time duration between a service user sending a request and receiving a response.
- *Throughput.* The average rate of successful message delivery over a communication channel. It is usually expressed as kilobits per second (kbps).
- *Failure-probability.* The probability that a web service invocation will fail.

To make accurate client-side QoS value predictions for a service user, our approach explores the past web service usage experiences of different service users. Inspired by the recent success of YouTube,[2] Wikipedia,[3] and BitTorrent,[4] we apply *user-collaboration*, the key concept of Web 2.0, to collect web service QoS values from different service users. As shown in Fig. 2, the idea is that, instead of contributing videos (*YouTube*) or knowledge (*Wikipedia*), the service users (usually developers of service-oriented systems) are encouraged to contribute/share their individually observed past web service QoS information. In our framework, if a service user would like to obtain the QoS value prediction service from our centralized server, he/she needs to contribute some QoS values. The service users can provide the QoS values to our server via: 1) input the values to a Web form directly or upload a file following our format; 2) by running a client-side web service evaluation application [14]; or 3) by employing a client-side middleware to automatically monitoring and contribute web service QoS values [15], [16]. If a service user contributes more web service QoS values, higher QoS value prediction accuracy can be achieved in our approach (technical details

[1] http://www.wsdream.net.

[2] http://www.youtube.com.

[3] http://www.wikipedia.org.

[4] http://www.bittorrent.com.

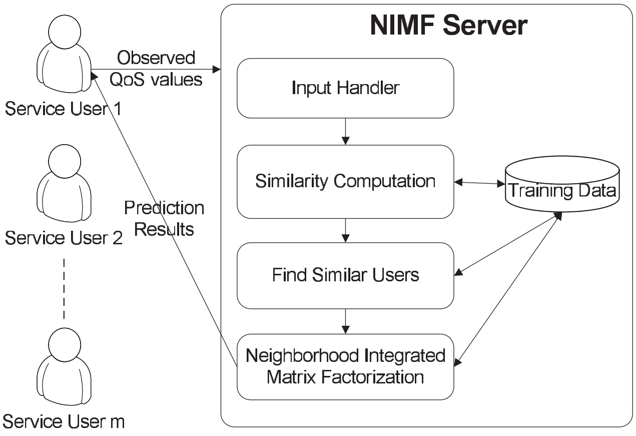


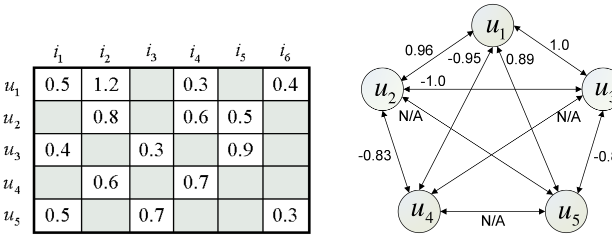
Fig. 2. Collaborative QoS prediction framework.

will be introduced in Section 2), since more user features can be mined from the contributed data. In this way, the service users are encouraged to contribute their observed web service QoS values. Beside the user-contributed QoS values, we also control a set of distributed computers to monitor the QoS performance of real-world web services (details will be introduced in Section 3.1). Based on the collected web service QoS values, collaborative web service QoS value prediction can be made.

2.2 Problem Description

The process of web service QoS value prediction usually includes a user-item matrix, as shown in Fig. 3a, where each entry in this matrix represents the value of a certain QoS property (e.g., *response-time* in this example) of a web service (e.g., i_1 to i_6) observed by a service user (e.g., u_1 to u_5). As shown in Fig. 3a, each service user has several response-time values of their invoked web services. Similarities between two different users in the matrix can be calculated by analyzing their QoS values on the same web services. Pearson correlation coefficient (PCC) [17] is usually employed for the similarity computation. As shown in the similarity graph in Fig. 3b, a total of five users (nodes u_1 to u_5) are connected with 10 edges. Each edge is associated with a PCC value in the range of $[-1, 1]$ to specify the similarity between user u_i and user u_j, where larger PCC value stands for higher similarity. The symbol *N/A* means that the similarity between user u_i and user u_j is nonavailable, since they do not have any commonly invoked web services. The problem we study in this paper is how to accurately predict the missing QoS values in the user-item matrix by employing the available QoS values. By predicting the web service QoS values in the user-item matrix, we can provide personalized QoS value prediction on the unused web services for the service users, who can employ these web service QoS values for making service selection, service ranking, automatic service composition, etc.

To obtain the missing values in the user-item matrix, we can employ the web service QoS values observed by other service users for predicting the web service performance for the current user. However, since service users are in different geographic locations and are under different network conditions, the current user may not be able to experience similar QoS performance as other service users.



	i_1	i_2	i_3	i_4	i_5	i_6
u_1	0.5	1.2		0.3		0.4
u_2		0.8		0.6	0.5	
u_3	0.4		0.3		0.9	
u_4		0.6		0.7		
u_5	0.5		0.7			0.3

(a) User-Item Matrix

(b) User Similarities

Fig. 3. A toy example.

To address this challenging web service QoS value prediction problem, we propose an NIMF approach, which makes the best utilization of both the local information of similar users and the global information of all the available QoS values in the user-item matrix to achieve better prediction accuracy. Our approach is designed as a two-phase process. In Phase 1, we calculate the user similarities using PCC and determine a set of Top-K similar users for the current user. Then, based on the neighborhood information, we propose an NIMF approach to predict the missing values in the user-item matrix in Phase 2. Details of these two phases are presented at Sections 3.2 and 3.3, respectively.

2.3 Phase 1: Neighborhood Similarity Computation

Given an $m \times n$ user-item matrix R consists of m service users and n web services, each entry in this matrix R_{ij} represents the value of a certain client-side QoS property of web service j observed by service user i. If user i did not invoke the web service j before, then $R_{ij} = null$. Employing the available web service QoS values in the user-item matrix, which are collected from different service users, the similarities between different service users can be computed by PCC or vector space similarity (VSS). PCC and VSS are two commonly used approaches for similarity computation. As mentioned in work [18], [19], PCC generally can achieve higher performance than VSS, since the former considers the differences in the user value style and can achieve high accuracy. Therefore, we employ PCC for the similarity computation in our approach.

Employing PCC, the similarity between two users i and k can be computed based on their observed QoS values on the commonly invoked web services with the following equation:

$$PCC(i,k) = \frac{\sum_{j \in J} \left(R_{ij} - \overline{R}_i\right)\left(R_{kj} - \overline{R}_k\right)}{\sqrt{\sum_{j \in J} \left(R_{ij} - \overline{R}_i\right)^2} \sqrt{\sum_{j \in J} \left(R_{kj} - \overline{R}_k\right)^2}}, \quad (1)$$

where J is the subset of web services that are invoked by both user i and user k, R_{ij} is the QoS value of web service j observed by service user i, and $\overline{R}_i$ and $\overline{R}_k$ are the average QoS values of different web services observed by service user i and k, respectively. From this definition, the similarity of two service users i and k, $PCC(i, k)$, is in the interval of [$-$1,1], where a larger PCC value indicates higher user similarity.

After calculating the similarities between the current user and other users, a set of Top-K similar users can be identified based on the PCC values. In practice, a service user may have limited number of similar users. Traditional Top-K algorithms ignore this problem and still include dissimilar users with negative PCC values, which will greatly influence the prediction accuracy. In our approach, we exclude the dissimilar service users who have negative correlations (negative PCC values). For a service user i, a set of similar users $\mathcal{T}(i)$ can, therefore, be identified by the following equation:

$$\mathcal{T}(i) = \{k | k \in \text{Top-K}(i), PCC(i,k) > 0, i \neq k\}, \tag{2}$$

where Top-K(i) is a set of the Top-K similar users to the current user i and $PCC(i,k)$ is the PCC similarity value between user i and user k, which can be calculated by (1). Note that the Top-K relations are not symmetrical. User k is in the Top-K neighbors of user i does not necessary indicate that user i is also in the Top-K neighbors of user k. With the neighborhood information, we can now design our NIMF model for the QoS value prediction.

2.4 Phase 2: Neighborhood-Integrated Matrix Factorization

A popular approach to predict missing values is to fit a factor model to the user-item matrix, and use this factor model to make further predictions. The premise behind a low-dimensional factor model is that there is a small number of factors influencing the QoS usage experiences, and that a user's QoS usage experience on a web service is determined by how each factor applies to the user and the web service.

Consider an $m \times n$ user-item matrix R, the matrix factorization method employs a rank-l matrix $X = U^T V$ to fit it, where $U \in \mathbb{R}^{l \times m}$ and $V \in \mathbb{R}^{l \times n}$. From the above definition, we can see that the low-dimensional matrices U and V are unknown, and need to be estimated. Moreover, this feature representations have clear physical meanings. In this linear factor model, a user's web service QoS values correspond to a linear combination of the factor vectors, with user-specific coefficients. More specifically, each column of U performs as a "factor vector" for a user, and each column of V is a linear predictor for a web service, predicting the entries in the corresponding column of the user-item matrix R based on the "factors" in U. The number of factors (in other word, the length of the "factor vector") is called *dimensionality*. By adding the constraints of the norms of U and V to penalize large values of U and V, we have the following optimization problem [20]:

$$\begin{aligned}\min_{U,V} \mathcal{L}(R,U,V) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}^R \left(R_{ij} - U_i^T V_j\right)^2 \\ + \frac{\lambda_U}{2}\|U\|_F^2 + \frac{\lambda_V}{2}\|V\|_F^2,\end{aligned} \tag{3}$$

where I_{ij}^R is the indicator function that is equal to 1 if user u_i invoked web service v_j and is equal to 0 otherwise, $\|\cdot\|_F^2$ denotes the Frobenius norm, and λ_U and λ_V are two parameters. The optimization problem in (3) minimizes the sum-of-squared-errors objective function with quadratic regularization terms. It also has a probabilistic interpretation with Gaussian observation noise, which is detailed in [20].

The above approach utilizes the global information of all the available QoS values in the user-item matrix for predicting missing values. This approach is generally effective at estimating overall structure (global information) that relates simultaneously to all users or items. However, this model are poor at detecting strong associations among a small set of closely related users or items (local information), precisely where the neighborhood models would perform better. Normally, the available web service QoS values in the user-item matrix are very sparse; hence, neither of the matrix factorization or neighborhood-based approaches can generate optimal QoS values. To preserve both global information and local information mentioned above, we employ a balance parameter to fuse these two types of information. The idea is that every time when factorizing a QoS value, we treat it as the ensemble of a user's information and the user's neighbors' information. The neighbors of the current user can be obtained by employing (2). Hence, we can minimize the following sum-of-squared-errors objective functions with quadratic regularization terms:

$$\begin{aligned}&\mathcal{L}(R,S,U,V) \\ &\quad = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}^R \left(R_{ij} - \left(\alpha U_i^T V_j + (1-\alpha)\sum_{k\in\mathcal{T}(i)} S_{ik} U_k^T V_j\right)\right)^2 \\ &\qquad + \frac{\lambda_U}{2}\|U\|_F^2 + \frac{\lambda_V}{2}\|V\|_F^2,\end{aligned} \tag{4}$$

where $\mathcal{T}(i)$ is a set of Top-K similar users of user u_i and S_{ik} is the normalized similarity score between user u_i and user u_k, which can be calculated by

$$S_{ik} = \frac{PCC(i,k)}{\sum_{k\in\mathcal{T}(i)} PCC(i,k)}. \tag{5}$$

A local minimum of the objective function given by (4) can be found by performing gradient descent in U_i, V_j:

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial U_i} &= \alpha\sum_{j=1}^{n} I_{ij}^R V_j \left(\left(\alpha U_i^T V_j + (1-\alpha)\sum_{k\in\mathcal{T}(i)} S_{ik} U_k^T V_j\right) - R_{ij}\right) \\ &\quad + (1-\alpha)\sum_{p\in\mathcal{B}(i)}\sum_{j=1}^{n} I_{pj}^R S_{pi} V_j \Bigg(\Bigg(\alpha U_p^T V_j \\ &\quad + (1-\alpha)\sum_{k\in\mathcal{T}(p)} S_{pk} U_k^T V_j\Bigg) - R_{pj}\Bigg) + \lambda_U U_i,\end{aligned} \tag{6}$$

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial V_j} &= \sum_{i=1}^{m} I_{ij}^R \left(\left(\alpha U_i^T V_j + (1-\alpha)\sum_{k\in\mathcal{T}(i)} S_{ik} U_k^T V_j\right) - R_{ij}\right) \\ &\quad \times \left(\alpha U_i + (1-\alpha)\sum_{k\in\mathcal{T}(i)} S_{ik} U_k^T\right) + \lambda_V V_j,\end{aligned} \tag{7}$$

where $\mathcal{B}(i)$ is the set that includes all the users who are the neighbors of user u_i. To reduce the model complexity, in all of the experiments we conduct, we set $\lambda_U = \lambda_V$.

2.5 Complexity Analysis

The main computation of the gradient methods is to evaluate the object function $\mathcal{L}$ and its gradients against

TABLE 1
Statistics of the WS QoS Data Set

Statistics	Values
Num. of Service Users	339
Num. of Web Services	5,825
Num. of Web Service Invocations	1,974,675
Range of Response-time	1-20 s
Range of Throughput	1-1000 kbps

the variables. Because of the sparsity of matrices R and S, the computational complexity of evaluating the object function $\mathcal{L}$ is $O(\rho_R l + \rho_R K l)$, where ρ_R is the number of nonzero entries in the matrix R, and K is the number of similar neighbors. K is normally a small number since a large number of K will introduce noise, which will potentially hurt the prediction accuracy. The computational complexities for the gradients $\frac{\partial \mathcal{L}}{\partial U}$ and $\frac{\partial \mathcal{L}}{\partial V}$ in (7) are $O(\rho_R K l + \rho_R K^2 l)$ and $O(\rho_R l + \rho_R K l)$, respectively. Therefore, the total computational complexity in one iteration is $O(\rho_R K l + \rho_R K^2 l)$, which indicates that theoretically, the computational time of our method is linear with respect to the number of observations in the user-item matrix R. This complexity analysis shows that our proposed approach is very efficient and can scale to very large data sets.

3 Experiments

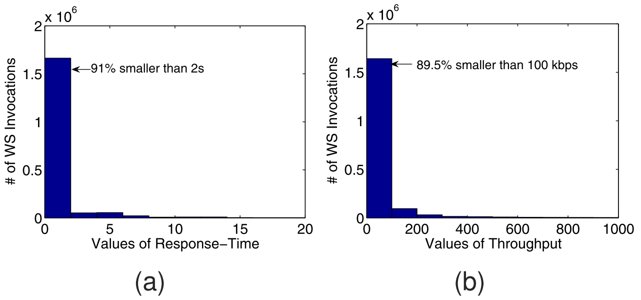
In this section, we conduct experiments to compare the prediction accuracy of our NIMF approach with other state-of-the-art collaborative filtering methods. Our experiments are intended to address the following questions: 1) How does our approach compare with the published state-of-the-art collaborative filtering algorithms? 2) How does the model parameter α affect the prediction accuracy? 3) What is the impact of the *matrix density*, *Top-K values*, and *dimensionality* on the prediction accuracy?

3.1 Data Set Description

We implement a *WSCrawler* and a *WSEvaluator* employing JDK 6.0, Eclipse 3.3, and Axis 2.[5] Employing our *WSCrawler*, addresses of 5,825 openly accessible web services are obtained by crawling web service information from www.seekda.com, a well-known web service search engine. Axis2 is employed to generate client-side web service invocation codes and test cases automatically. In total, 78,635 Java Classes and 13,644,507 lines of Java codes are generated in our experiments.

To evaluate the QoS performance of real-world web services from distributed locations, we deploy our *WSEvaluator* to 339 distributed computers of PlanetLab,[6] which is a distributed testbed made up of computers all over the world. In our experiment, each PlanetLab computer invokes all the web services. As shown in Fig. 1, a total of 1,974,675 real-world web service invocation results are collected from these 339 service users on 5,825 real-world web services.

By processing the invocation results, we obtain two $339 \times 5{,}825$ user-item matrices. One matrix contains response-time values, while the other one contains throughput values. The statistics of our web service QoS data set is summarized in Table 1, the distributions of response-time and throughput values are shown in Fig. 4, and more experimental details (e.g., detailed list of service users and web services, the user-item matrix, the detailed web service invocation results, etc.) are released online[7] for future research. As shown in Table 1, the ranges of response-time and throughput are 0-20 s and 0-1,000 kbps, respectively. Fig. 4a shows that 91 percent of the response-time values are smaller than 2 s, and Fig. 4b shows that 89.5 percent of the throughput values are smaller than 100 kbps.



(a) (b)

Fig. 4. Value distributions.

Although we only study the response-time and throughput in the experiments, the proposed NIMF approach can be applied to other QoS properties easily. When predicting value of a certain QoS property, the value of the entry in the user-item matrix is the corresponding QoS value (e.g., response-time, throughput, failure probability) observed by a user on a certain web service. Our NIMF approach can be employed on different QoS properties directly without any modifications.

3.2 Metrics

We use mean absolute error (MAE) and root-mean-squared error (RMSE) metrics to measure the prediction quality of our method in comparison with other collaborative filtering methods. MAE is defined as

$$MAE = \frac{\sum_{i,j} |R_{ij} - \widehat{R}_{ij}|}{N}, \tag{8}$$

where R_{ij} denotes the observed QoS value of web service j observed by user i, $\widehat{R}_{ij}$ is the predicted QoS value, and N is the number of predicted values. The MAE is the average over the verification sample of the absolute values of the differences between a prediction result and the corresponding observation. The MAE is a linear score, which means that all the individual differences are weighted equally in the average.

RMSE is defined as

$$RMSE = \sqrt{\frac{\sum_{i,j} (R_{ij} - \widehat{R}_{ij})^2}{N}}. \tag{9}$$

In RMSE, the difference between a prediction result and the corresponding observed values are each squared and then averaged over the sample. Finally, the square root of the average is taken. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to

[5] http://ws.apache.org/axis2.

[6] http://www.planet-lab.org.

[7] http://www.wsdream.net.

TABLE 2
Performance Comparison (a Smaller MAE or RMSE Value Means a Better Performance)

QoS Properties	Methods	Matrix Density = 5%		Matrix Density = 10%		Matrix Density = 15%		Matrix Density = 20%	
		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Response-time (0-20 s)	UMEAN	0.8785	1.8591	0.8783	1.8555	0.8768	1.8548	0.8747	1.8557
	IMEAN	0.7015	1.5813	0.6918	1.5440	0.6867	1.5342	0.6818	1.5311
	UPCC	0.6261	1.4078	0.5517	1.3151	0.5159	1.2680	0.4884	1.2334
	IPCC	0.6897	1.4296	0.5917	1.3268	0.5037	1.2552	0.4459	1.2095
	UIPCC	0.6234	1.4078	0.5365	1.3043	0.4965	1.2467	0.4407	1.2012
	NMF	0.6182	1.5746	0.6040	1.5494	0.5990	1.5345	0.5982	1.5331
	PMF	0.5678	1.4735	0.4996	1.2866	0.4720	1.2163	0.4492	1.1828
	NIMF	**0.5514**	**1.4075**	**0.4854**	**1.2745**	**0.4534**	**1.1980**	**0.4357**	**1.1678**
Throughput (0-1000 kbps)	UMEAN	54.0084	110.2821	53.6700	110.2977	53.8792	110.1751	53.7114	110.1708
	IMEAN	27.3558	66.6344	26.8318	64.7674	26.6239	64.3986	26.6364	64.1082
	UPCC	26.1230	61.6108	21.2695	54.3701	18.7455	50.7768	17.5546	48.2621
	IPCC	29.2651	64.2285	27.3993	60.0825	26.4319	57.8593	25.0273	55.4970
	UIPCC	25.8755	60.8685	19.9754	54.8761	17.5543	47.8235	16.0762	47.8749
	NMF	25.7529	65.8517	17.8411	53.9896	15.8939	51.7322	15.2516	48.6330
	PMF	19.9034	54.0508	16.1755	46.4439	15.0956	43.7957	14.6694	42.4855
	NIMF	**17.9297**	**51.6573**	**16.0542**	**45.9409**	**14.4363**	**43.1596**	**13.7099**	**41.1689**

large errors. This means the RMSE is most useful when large errors are particularly undesirable.

3.3 Comparison

In this section, to show the prediction accuracy of our NIMF approach, we compare our method with the following approaches:

1. *UMEAN (user mean).* This method employs a service user's average QoS value on the used web services to predict the QoS values of the unused web services.
2. *IMEAN (item mean).* This method employs the average QoS value of the web service observed by other service users to predict the QoS value for a service user who never invoke this web service previously.
3. *UPCC (user-based collaborative filtering method using PCC).* This method is a very classical method. In this paper, it employs similar users for the QoS value prediction [21], [22].
4. *IPCC (item-based collaborative filtering method using PCC).* This method is widely used in industry company like Amazon. In this paper, it employs similar web services (items) for the QoS value prediction [17].
5. *UIPCC.* This method combines the user-based and item-based collaborative filtering approaches and employs both the similar users and similar web services for the QoS value prediction [23].
6. *NMF (nonnegative matrix factorization).* This method is proposed by Lee and Seung [24], [25]. It differs from other matrix factorization methods in that it enforces the constraint that the factorized factors must be nonnegative. NMF is also widely used in collaborative filtering community.
7. *PMF (probabilistic matrix factorization).* This method is proposed by Salakhutdinov and Minh [20]. It uses user-item matrix for the recommendations, and it is based on probabilistic matrix factorization.

In the real world, the user-item matrices are usually very sparse since a service user usually only invokes a small number of web services. In this paper, to conduct our experiments realistically, we randomly remove entries from the user-item matrix to make the matrix sparser with different density (i.e., 5, 10, 15, and 20 percent). Matrix density 5 percent, for example, means that we randomly select 5 percent of the QoS entries to predict the remaining 95 percent of QoS entries. The original QoS values of the removed entries are used as the expected values to study the prediction accuracy. The above seven methods together with our NIMF method are employed for predicting the QoS values of the removed entries. The parameter settings of our NIMF method are $\alpha = 0.4$, Top-K $= 10$, $\lambda_U = \lambda_V = 0.001$, and dimensionality $= 10$ in the experiments. The experimental results are shown in Table 2, and the detailed investigations of parameter settings will be provided in Sections 3.4 through 3.7.

From Table 2, we can observe that our NIMF approach obtains smaller MAE and RMSE values (indicating better prediction accuracy) consistently for both response-time and throughput with different matrix densities. The MAE and RMSE values of throughput in Table 2 are much larger than those of response-time, since the range of throughput is 0-1,000 kbps, while the range of response-time is only 0-20 s. With the increase of matrix density from 5 to 20 percent, the MAE and RMSE values of our NIMF method become smaller, since denser matrix provides more information for the missing value prediction. Among all the prediction methods, our NIMF method generally achieves better performance on both MAE and RMSE, indicating that integrating the neighborhood information into matrix factorization model can achieve higher value prediction accuracy. These experimental results demonstrate that our interpretation on the formation of QoS values is realistic and reasonable.

3.4 Impact of Parameter α

In our NIMF method, parameter α controls how much our method relies on the users themselves and their similar users. If $\alpha = 1$, we only employ the users' own characteristics for making prediction. If $\alpha = 0$, we predict the users' QoS values purely by their similar users' characteristics. In other cases, we fuse the users' own characteristics with the neighborhood information for missing QoS value prediction.

Fig. 5 shows the impacts of parameter α on the prediction results. We observe that optimal α value settings can achieve

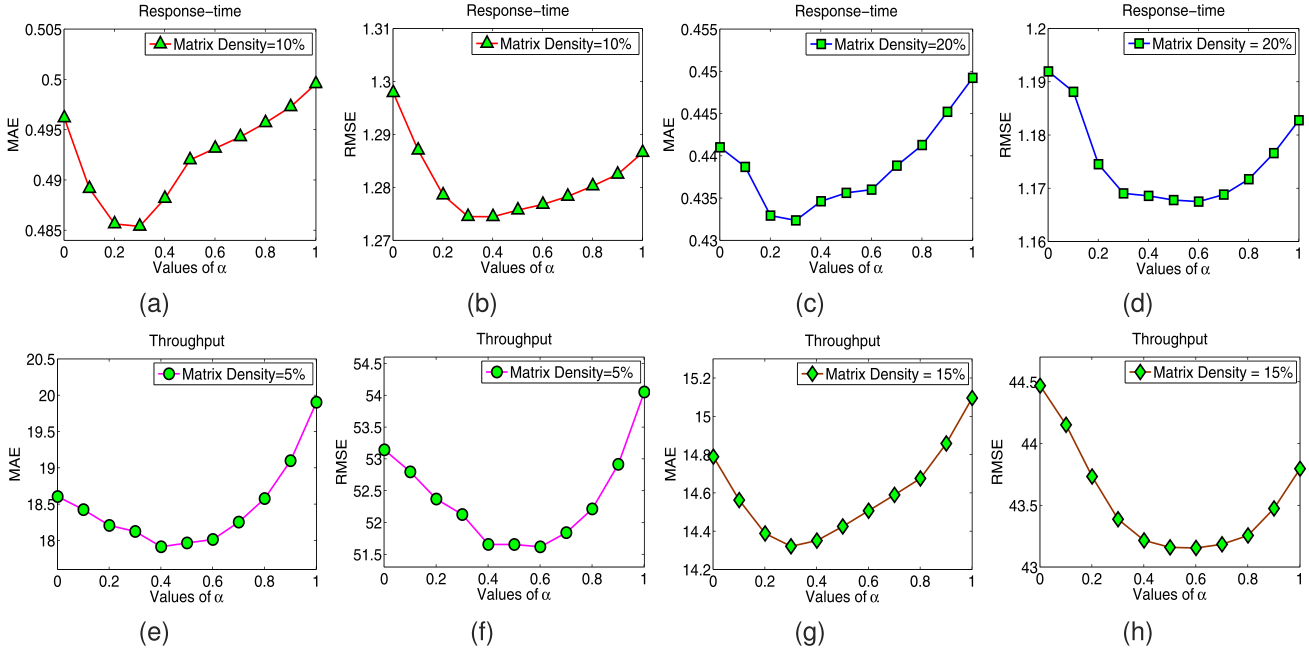


Fig. 5. Impact of parameter α (dimensionality $= 10$).

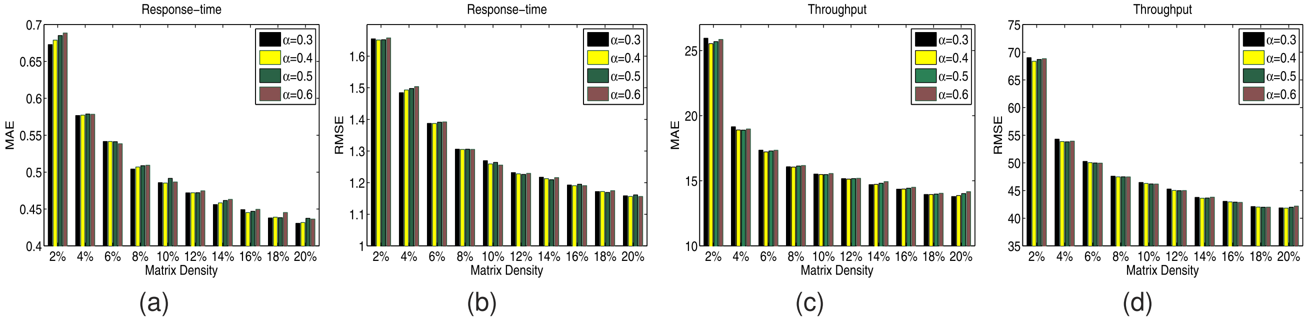


Fig. 6. Impact of matrix density (dimensionality $= 10$, $\alpha = 0.4$).

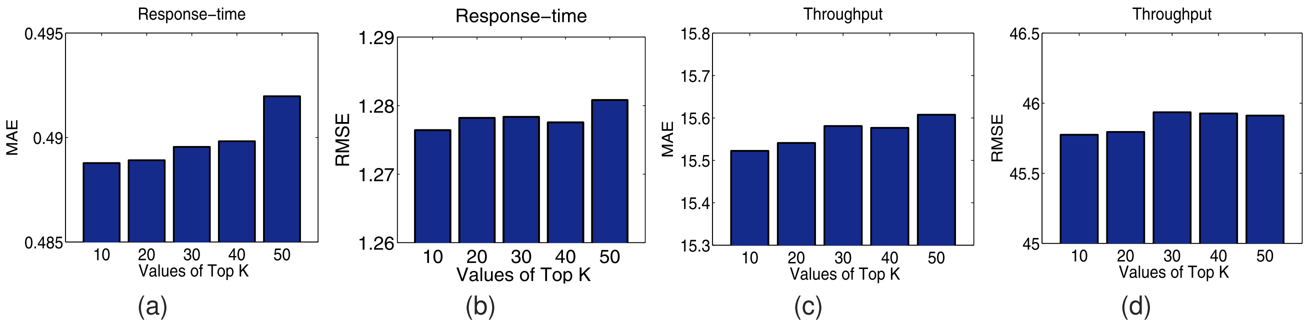
better prediction accuracy, which demonstrates that fusing the matrix factorization methods with neighborhood-based methods will improve the prediction accuracy. No matter for response-time or throughput, as α increases, the MAE and RMSE values decrease (prediction accuracy increases) at first, but when α surpasses a certain threshold, the MAE and RMSE values increase (prediction accuracy decreases) with further increase of the value of α. This phenomenon confirms the intuition that purely using the matrix factorization method or purely employing the neighborhood-based method cannot generate better QoS value prediction performance than fusing these two favors together.

From Figs. 5a and 5b, when using user-item matrix with 10 percent density, we observe that our NIMF method achieves the best performance when α is around 0.3, while smaller values like $\alpha = 0.1$ or larger values like $\alpha = 0.7$ can potentially degrade the model performance. In Figs. 5c and 5d, when using user-item matrix with 20 percent density, the optimal value of α is also around 0.3 for MAE and around 0.6 for RMSE. The optimal values of MAE and RMSE are different because MAR and RMSE are different metrics following different evaluation criteria. As the same with Figs. 5a, 5b, 5c, and 5d the optimal α values of Figs. 5e, 5f, 5g, and 5h are all between 0.3 and 0.6. This observation indicates that optimally combining the two methods can achieve better prediction accuracy than purely or heavily relying one kind of method, and this is why we use $\alpha = 0.4$ as the default settings in other experiments. The same as Table 2, another observation from Fig. 5 is that denser matrix provides better prediction accuracy.

3.5 Impact of Matrix Density

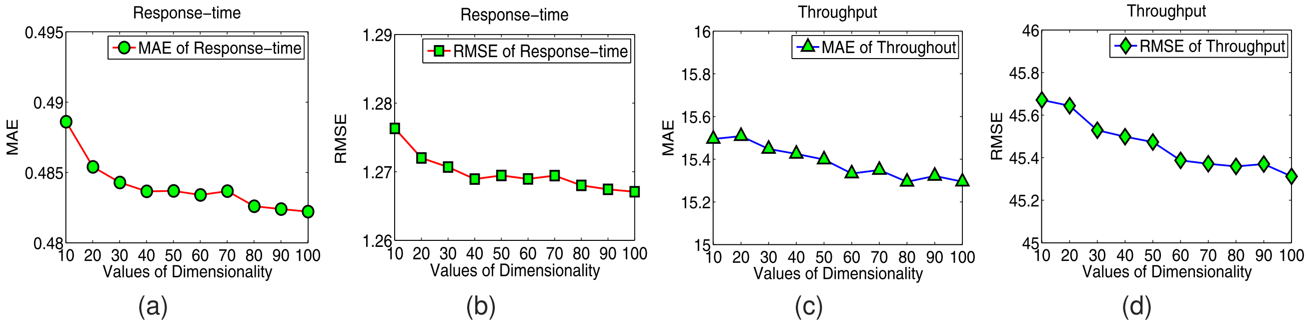
As shown in Table 2 and Fig. 5, the prediction accuracy of our NIMF method is influenced by the matrix density. To study the impact of the matrix density on the prediction results, we change the matrix density from 2 to 20 percent with a step value of 2 percent. We set Top-K $= 10$, dimensionality $= 10$, and $\alpha = 0.4$ in this experiment.

Fig. 6 shows the experimental results, where Figs. 6a and 6b are the experimental results of response-time, and Figs. 6c and 6d are the experimental results of throughput. Fig. 6 shows that when the matrix density is increased from 2 to 4 percent, the prediction accuracy of the NIMF method is significantly enhanced. With the further increase of matrix density, the speed of prediction accuracy enhancement slows down. This observation indicates that when the matrix is very sparse, the prediction accuracy can be greatly enhanced by collecting more QoS values to make the matrix denser.



(a) (b) (c) (d)

Fig. 7. Impact of parameter Top-K (dimensionality $= 10$, $\alpha = 0.4$).



(a) (b) (c) (d)

Fig. 8. Impact of dimensionality ($\alpha = 0.4$, matrix density $= 10$ percent).

3.6 Impact of Top-K

The Top-K value determines the number of similar users employed in our NIMF method. To study the impact of the Top-K values on the prediction results, we vary the values of Top-K from 10 to 50 with a step value of 10. We set dimensionality $= 10$, $\alpha = 0.4$, and matrix density $= 10$ percent in this experiment.

Figs. 7a and 7b show the MAE and RMSE results of response-time, while Figs. 7c and 7d show the MAE and RMSE results of throughput. Fig. 7 shows that the MAE and RMSE values slightly increase (prediction accuracy decrease) when the Top-K value is increased from 10 to 50. This is because too large Top-K value will introduce noise (dissimilar users), which will potentially hurt the prediction accuracy. In all the four figures from Figs. 7a, 7b, 7c, and 7d, the Top-K value of 10 obtains the best prediction accuracy, and this is why we use Top-K $= 10$ as the default experimental settings in other experiments.

3.7 Impact of Dimensionality

Dimensionality determines how many factors are used to factorize the user-item matrix. To study the impact of the dimensionality, we vary the values of dimensionality from 10 to 100 with a step value of 10. We set Top-K $= 10$, $\alpha = 0.4$, and matrix density $= 10$ percent in this experiment.

Figs. 8a and 8b show the experimental results of response-time, while Figs. 8c and 8d show the experimental results of throughput. As shown in Fig. 8, the values of MAE and RMSE decrease when the dimensionality is increased from 10 to 100. These observed results coincide with the intuition that relative larger values of dimensions generate better recommendation results. However, the computational time of our NIMF approach is linear with respect to the value of dimensionality. Larger dimensionality value will require longer computation time. Moreover, the dimensionality cannot be set to a very high value because it will cause the overfitting problem, which will potentially hurt the recommendation quality.

4 Related Work and Discussion

Web services QoS has been widely discussed in a number of research investigations [26], [27], [28], [29], [30]. Zeng et al. [4] employ five generic QoS properties (i.e., execution price, execution duration, reliability, availability, and reputation) for dynamic web service composition. Ardagna and Pernici [3] use five QoS properties (i.e., execution time, availability, price, reputation, and data quality) when making adaptive service composition in flexible processes. Alrifai and Risse [31] propose an efficient service composition approach by considering both generic QoS properties and domain-specific QoS properties. In this paper, we focus on predicting the client-side QoS values for the service users.

The previous QoS-based web service approaches (e.g., web service composition [31], [3], [32], [4], web service selection [5], [6], [7], [8], etc.) usually assume that web service QoS values are already known or can be easily obtained from the service providers or third-party registries. This paper complements these QoS-based approaches by providing a collaborative QoS value prediction approach. The predicted QoS values provided by our approach can be employed by other QoS-based approaches in the field of service computing.

Usage experience of different users is a necessity to address the complex problems faced by the software engineering society. Meneely et al. [33] investigate the relationship between developer collaboration structure and software product reliability. Bird et al. [34] study latent subcommunities from the e-mail social network of several open source projects. In this paper, we apply the concept of user-collaboration to enable web service usage experience sharing between service users (usually developers of the service-oriented systems). Our NIMF approach takes advantage of the past web service usage experience of service users to make web service QoS value prediction for the current user.

Collaborative filtering methods are widely adopted in commercial recommender systems [35], [36], [17]. Two types of collaborative filtering approaches are widely studied: neighborhood-based (memory-based) and model-based. The most analyzed examples of neighborhood-based collaborative filtering include user-based approaches [21], [37], item-based approaches [38], [39], and their fusion [40], [23]. User-based approaches predict the missing values of a user based on the values of similar users. Item-based approaches predict the missing values of a current user based on the computed information of items similar to those chosen by the current user. Neighborhood-based approaches often use the PCC algorithm [17] and the VSS algorithm [21] as the similarity computation methods. PCC-based collaborative filtering approaches generally can achieve higher prediction accuracy than the VSS-based algorithms, since PCC considers the differences of the user value characteristics.

In the model-based approaches, on the other hand, training data sets are used to train a predefined model. Examples of model-based approaches include the clustering model [41], aspect models [42], and so on. Recently, several matrix factorization methods [43], [44], [20], [45] have been proposed for collaborative filtering. These methods focus on fitting the user-item matrix with low-rank approximations, which is engaged to make further predictions. The premise behind a low-dimensional factor model is that there is only a small number of factors influencing the values in the user-item matrix, and that a user's factor vector is determined by how each factor applies to that user. The neighborhood-based methods utilize the values of similar users or items (local information) for making value prediction, while model-based methods, like matrix factorization models, employ all the value information of the matrix (global information) for making value prediction. Different from these previous works, our approach takes advantages of both the local information of similar users and global information of the whole matrix to achieve better QoS value prediction accuracy.

There is limited work in the literature employing collaborative filtering methods for web service QoS value prediction. One of the most important reasons that obstruct the research is the lack of real-world web service QoS data sets for experimental studies. Without convincing and sufficient real-world web service QoS data, the characteristics of web service QoS values cannot be fully mined and the prediction accuracy of the proposed prediction algorithms cannot be justified. A few approaches [46], [47] mention the idea of applying neighborhood-based collaborative filtering methods for web service QoS value prediction. However, these approaches simply employ a movie rating data set, i.e., MovieLens [17], for experimental studies, which is not convincing enough. Shao et al. [22] propose a user-based PCC method for the web service QoS value prediction. However, only 20 web services are involved in the experiments. Zheng et al. [23] propose a neighborhood-based approach (i.e., UIPCC) for QoS value prediction by combining the UPCC and IPCC approaches. However, only 100 web services are studied. Compared with these previous work, we conduct large-scale experimental studies on 5,825 real-world web services in this paper. Moreover, as shown in Section 3.3, our NIMF approach provides much better prediction accuracy than the UPCC approach [22] and the UIPCC approach [23].

Al-Masri and Mahmoud [48] release a web service QoS data set that is observed by one service user on 2,507 web services. The fact that client-side web service QoS values can vary widely among users limits the applicability of this data set. Our released data set, on the other hand, includes QoS information observed by 339 service users in heterogenous environments on 5,825 web services. To the best of our knowledge, our data set is the largest scale web service QoS data set in the published work of service computing.

5 Conclusion and Future Work

Based on the intuition that a user's web service QoS usage experiences can be predicted by both the user's own characteristics and the past usage experiences of other similar users, we propose an NIMF approach for making personalized QoS value prediction. Our NIMF approach systematically fuses the neighborhood-based and model-based collaborative filtering approaches to achieve higher prediction accuracy. The extensive experimental analysis shows the effectiveness of our approach.

After obtaining the predicted QoS values on the unused web services, most service users will make invocations to the selected web services. The QoS values of these web service invocations contain valuable information for improving the QoS prediction accuracy. We plan to design better incentive mechanisms and automatic approaches to enable the real-time sharing of these web service usage experiences among service users. Moreover, we plan to apply our approach to the cloud computing environments, where the web service QoS value collection becomes easier, since the user applications which invoke the web services are usually deployed and running on the cloud.

The NIMF approach in this paper can only be employed to predict client-side QoS properties which have different values among users. We plan to conduct more studies to prediction QoS values of server-side QoS properties. In this paper, due to the lack of real-world data sets for conducting experiments, we only conduct experimental studies on response-time and throughput. We are currently collecting data on failure-probabilities of the real-world web services. The effort requires long observation duration and sufficient web service invocations for accurate measurement. More experimental studies on the failure-probability and other web service QoS properties will be conducted in our future work.

Acknowledgments

The authors appreciate the reviewers for their informative comments. The work described in this paper was fully supported by the National Basic Research Program of China (973 Project No. 2011CB302603), the National Natural Science Foundation of China (Project No. 61100078), the Shenzhen Basic Research Program (Project

No. JCYJ20120619153834216, JC201104220300A), and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 415410 and CUHK 413210).

References

[1] L.-J. Zhang, J. Zhang, and H. Cai, *Services Computing.* Springer, 2007.

[2] M. Alrifai, D. Skoutas, and T. Risse, "Selecting Skyline Services for QoS-Based Web Service Composition," *Proc. 19th Int'l Conf. World Wide Web (WWW '10),* pp. 11-20, 2010.

[3] D. Ardagna and B. Pernici, "Adaptive Service Composition in Flexible Processes," *IEEE Trans. Software Engeering,* vol. 33, no. 6, pp. 369-384, June 2007.

[4] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q.Z. Sheng, "Quality Driven Web Services Composition," *Proc. 12th Int'l Conf. World Wide Web (WWW '03),* pp. 411-421, 2003.

[5] P.A. Bonatti and P. Festa, "On Optimal Service Selection," *Proc. 14th Int'l Conf. World Wide Web (WWW '05),* pp. 530-538, 2005.

[6] V. Cardellini, E. Casalicchio, V. Grassi, and F.L. Presti, "Flow-Based Service Selection for Web Service Composition Supporting Multiple QoS Classes," *Proc. Fifth Int'l Conf. Web Services (ICWS '07),* pp. 743-750, 2007.

[7] L. Mei, W.K. Chan, and T.H. Tse, "An Adaptive Service Selection Approach to Service Composition," *Proc. Sixth Int'l Conf. Web Services (ICWS '08),* pp. 70-77, 2008.

[8] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints," *ACM Trans. Web,* vol. 1, no. 1, pp. 1-26, 2007.

[9] N. Salatge and J.-C. Fabre, "Fault Tolerance Connectors for Unreliable Web Services," *Proc. 37th Int'l Conf. Dependable Systems and Networks (DSN '07),* pp. 51-60, 2007.

[10] V. Deora, J. Shao, W. Gray, and N. Fiddian, "A Quality of Service Management Framework Based on User Expectations," *Proc. First Int'l Conf. Service-Oriented Computing (ICSOC '03),* pp. 104-114, 2003.

[11] E. Maximilien and M. Singh, "Conceptual Model of Web Service Reputation," *ACM SIGMOD Record,* vol. 31, no. 4, pp. 36-41, 2002.

[12] W.-T. Tsai, X. Zhou, Y. Chen, and X. Bai, "On Testing and Evaluating Service-Oriented Software," *IEEE Computer,* vol. 41, no. 8, pp. 40-46, Aug. 2008.

[13] G. Wu, J. Wei, X. Qiao, and L. Li, "A Bayesian Network Based QoS Assessment Model for Web Services," *Proc. Int'l Conf. Services Computing (SCC '07),* pp. 498-505, 2007.

[14] Z. Zheng and M.R. Lyu, "Optimal Fault Tolerance Strategy Selection for Web Services," *Int'l J. Web Service Research,* vol. 7, no. 4, pp. 21-40, 2010.

[15] D. Bianculli, W. Binder, L. Drago, and C. Ghezzi, "Transparent Reputation Management for Composite Web Services," *Proc. Sixth Int'l Conf. Web Services (ICWS '08),* pp. 621-628, 2008.

[16] Z. Zheng and M.R. Lyu, "An Adaptive QoS-Aware Fault Tolerance Strategy for Web Services," *J. Empirical Software Eng.,* vol. 15, no. 5, pp. 323-345, 2010.

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," *Proc. ACM Conf. Computer Supported Cooperative Work,* pp. 175-186, 1994.

[18] J. Breese et al., "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. 14th Conf. Uncertainty in Artificial Intelligence (UAI '98),* pp. 43-52, 1998.

[19] H. Ma, I. King, and M.R. Lyu, "Effective Missing Data Prediction for Collaborative Filtering," *Proc. 30th Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '07),* pp. 39-46, 2007.

[20] R. Salakhutdinov and A. Mnih, "Probabilistic Matrix Factorization," *Proc. Advances in Neural Information Processing Systems,* pp. 1257-1264, 2007.

[21] J.S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. 14th Ann. Conf. Uncertainty in Artificial Intelligence (UAI '98),* pp. 43-52, 1998.

[22] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized QoS Prediction for Web Services via Collaborative Filtering," *Proc. Fifth Int'l Conf. Web Services (ICWS '07),* pp. 439-446, 2007.

[23] Z. Zheng, H. Ma, M.R. Lyu, and I. King, "QoS-Aware Web Service Recommendation by Collaborative Filtering," *IEEE Trans. Services Computing,* vol. 4, no. 2, pp. 140-152, Apr.-June 2011.

[24] D.D. Lee and H.S. Seung, "Learning the Parts of Objects by Non-Negative Matrix Factorization," *Nature,* vol. 401, no. 6755, pp. 788-791, Oct. 1999.

[25] D.D. Lee and H.S. Seung, "Algorithms for Non-Negative Matrix Factorization," *Proc. Advances in Neural Information Processing Systems,* pp. 556-562, 2000.

[26] M.C. Jaeger, G. Rojec-Goldmann, and G. Muhl, "QoS Aggregation for Web Service Composition Using Workflow Patterns," *Proc. IEEE Eighth Int'l Conf. Enterprise Computing,* pp. 149-159, 2004.

[27] D.A. Menasce, "QoS Issues in Web Services," *IEEE Internet Computing,* vol. 6, no. 6, pp. 72-75, Nov./Dec. 2002.

[28] M. Ouzzani and A. Bouguettaya, "Efficient Access to Web Services," *IEEE Internet Computing,* vol. 8, no. 2, pp. 34-44, Mar./Apr. 2004.

[29] S. Rosario, A. Benveniste, S. Haar, and C. Jard, "Probabilistic QoS and Soft Contracts for Transaction-Based Web Services Orchestrations," *IEEE Trans. Services Computing,* vol. 1, no. 4, pp. 187-200, Oct.-Dec. 2008.

[30] Z. Zheng, Y. Zhang, and M.R. Lyu, "Distributed QoS Evaluation for Real-World Web Services," *Proc. Eighth Int'l Conf. Web Services (ICWS '10),* pp. 83-90, 2010.

[31] M. Alrifai and T. Risse, "Combining Global Optimization with Local Selection for Efficient QoS-Aware Service Composition," *Proc. 18th Int'l Conf. World Wide Web (WWW '09),* pp. 881-890, 2009.

[32] V. Cardellini, E. Casalicchio, V. Grassi, F.Lo Presti, and R. Mirandola, "QoS-Driven Runtime Adaptation of Service Oriented Architectures," *Proc. Seventh Joint Meeting European Software Eng. Conf. and ACM SIGSOFT Symp. Foundations of Software Eng. (ESEC/FSE '09),* pp. 131-140, 2009.

[33] A. Meneely, L. Williams, W. Snipes, and J. Osborne, "Predicting Failures with Developer Networks and Social Network Analysis," *Proc. 16th ACM SIGSOFT Symp. Foundations of Software Eng. (FSE '08),* pp. 13-23, 2008.

[34] C. Bird, D.S. Pattison, R.M. D'Souza, V. Filkov, and P.T. Devanbu, "Latent Social Structure in Open Source Projects," *Proc. 16th ACM SIGSOFT Symp. Foundations of Software Eng. (FSE '08),* pp. 24-35, 2008.

[35] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *User Modeling and User-Adapted Interaction,* vol. 12, no. 4, pp. 331-370, 2002.

[36] X. Su, T.M. Khoshgoftaar, X. Zhu, and R. Greiner, "Imputation-Boosted Collaborative Filtering Using Machine Learning Classifiers," *Proc. ACM Symp. Applied Computing (SAC '08),* pp. 949-950, 2008.

[37] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering," *Proc. 22nd Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '99),* pp. 230-237, 1999.

[38] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," *IEEE Internet Computing,* vol. 7, no. 1, pp. 76-80, Jan./Feb. 2003.

[39] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," *Proc. 10th Int'l Conf. World Wide Web (WWW '01),* pp. 285-295, 2001.

[40] J. Wang, A.P. de Vries, and M.J. Reinders, "Unifying User-Based and Item-Based Collaborative Filtering Approaches by Similarity Fusion," *Proc. 29th Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '06),* pp. 501-508, 2006.

[41] G. Xue, C. Lin, Q. Yang, W. Xi, H. Zeng, Y. Yu, and Z. Chen, "Scalable Collaborative Filtering Using Cluster-Based Smoothing," *Proc. 28th Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '05),* pp. 114-121, 2005.

[42] T. Hofmann, "Latent Semantic Models for Collaborative Filtering," *ACM Trans. Information System,* vol. 22, no. 1, pp. 89-115, 2004.

[43] H. Ma, I. King, and M.R. Lyu, "Learning to Recommend with Social Trust Ensemble," *Proc. 32nd Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '09),* pp. 203-210, 2009.

[44] J.D.M. Rennie and N. Srebro, "Fast Maximum Margin Matrix Factorization for Collaborative Prediction," *Proc. 22nd Int'l Conf. Machine Learning (ICML '05),* pp. 713-719, 2005.

[45] R. Salakhutdinov and A. Mnih, "Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo," *Proc. 25th Int'l Conf. Machine Learning (ICML '08),* pp. 880-887, 2008.

[46] K. Karta, "An Investigation on Personalized Collaborative Filtering for Web Service Selection," honours programme thesis, Univ. of Western Australia, 2005.

[47] R.M. Sreenath and M.P. Singh, "Agent-Based Service Selection," *J. Web Semantics,* vol. 1, no. 3, pp. 261-279, 2003.

[48] E. Al-Masri and Q.H. Mahmoud, "Investigating Web Services on the World Wide Web," *Proc. 17th Int'l Conf. World Wide Web (WWW '08),* pp. 795-804, 2008.

Zibin Zheng received the PhD degree from the Chinese University of Hong Kong in 2011. He is an associate research fellow at Shenzhen Research Institute, the Chinese University of Hong Kong. He is also a guest research member at the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, China. He served as a program committee member of IEEE CLOUD (2009), CLOUD COMPUTING (2010 and 2011), and IEEE SCC (2011). His research interests include service computing, cloud computing, and software reliability engineering. He received the ACM SIGSOFT Distinguished Paper Award at ICSE '10 and the Best Student Paper Award at ICWS '10. First Runner-Up Award at the 2010 IEEE Hong Kong Postgraduate Research Paper Competition, and the IBM PhD Fellowship Award 2010-2011.

Hao Ma received the BEng and MEng degrees from the School of Information Science and Engineering from Central South University, in 2002 and 2005, respectively, and the PhD degree from the Chinese University of Hong Kong (CHUK) in 2010. He worked as a system engineer at Intel Shanghai before he joined CUHK. He is currently a researcher at Microsoft Research Redmond. His research interests include information retrieval, data mining, machine learning, social network analysis, and recommender system.

Michael R. Lyu received the BS degree in electrical engineering from the National Taiwan University, Taipei, in 1981, the MS degree in computer engineering from the University of California, Santa Barbara, in 1985, and the PhD degree in computer science from the University of California, Los Angeles, in 1988. He is currently a professor in the Department of Computer Science and Engineering, Chinese University of Hong Kong, China. He is also the director of the Video over Internet and Wireless Technologies Laboratory. His research interests include software reliability engineering, distributed systems, fault-tolerant computing, mobile networks, web technologies, multimedia information processing, and e-commerce systems. He has published more than 400 refereed journal and conference papers in these areas. He has participated in more than 30 industrial projects and helped to develop many commercial systems and software tools. He was the editor of two book volumes: *Software Fault Tolerance* (Wiley, 1995) and *The Handbook of Software Reliability Engineering* (McGraw-Hill, 1996). He initiated the First International Symposium on Software Reliability Engineering in 1990. He was the program chair for ISSRE '96 and general chair for ISSRE '01. He was also the PRDC '99 program cochair, WWW '10 program cochair, SRDS '05 program cochair, PRDC 2005 general cochair, ICEBE '07 program cochair, SCC '10 program cochair, and DSN '11 general chair, and has served on program committees for many other conferences including HASE, ICECCS, ISIT, FTCS, DSN, ICDSN, EUROMICRO, APSEC, PRDC, PSAM, ICCCN, ISESE, and WI. He has been frequently invited as a keynote or tutorial speaker to conferences and workshops in the US, Europe, and Asia. He has been on the editorial board of the *IEEE Transactions on Knowledge and Data Engineering*, the *IEEE Transactions on Reliability*, the *Journal of Information Science and Engineering*, and *Software Testing, Verification and Reliability Journal*. He received Best Paper Awards at ISSRE '98 and ISSRE '03. He is a fellow of both the IEEE and the AAAS for his contributions to software reliability engineering and software fault tolerance. He is also a Croucher senior research fellow.

Irwin King received the BSc degree in engineering and applied science from the California Institute of Technology, Pasadena, in 1984, and the MSc and PhD degrees in computer science from the University of Southern California, Los Angeles, in 1988 and 1993, respectively. He joined the Chinese University of Hong Kong in 1993. His research interests include machine learning, information retrieval, web intelligence and social computing, and multimedia processing. In these research areas, he has published more than 140 refereed journals (*JMLR*, *ACM TOIS*, *IEEE TNN*, *IEEE BME*, *PR*, *IEEE SMC*, *JAMC*, *JASIST*, *IJPRAI*, and *NN*) and conference manuscripts (NIPS, CIKM, SIGIR, IJCAI, ICML, IJCNN, ICONIP, ICDAR, and WWW). In addition, he has contributed more than 20 book chapters and edited volumes. Moreover, he has more than 30 research and applied grants. One notable system he has developed is the Chinese University Plagiarism IDentification Engine system, which detects similar sentences and performs readability analysis of text-based documents in both English and in Chinese to promote academic integrity and honesty. He is an associate editor of the *IEEE Transactions on Neural Networks*. He is a member of the editorial boards of the *Open Information Systems Journal*, *Journal of Nonlinear Analysis and Applied Mathematics*, and *Neural Information Processing Letters and Reviews Journal*. He has also served as special issue guest editor for *Neurocomputing*, *International Journal of Intelligent Computing and Cybernetics*, *Journal of Intelligent Information Systems*, and *International Journal of Computational Intelligent Research*. Currently, he is serving on the Neural Network Technical Committee and the Data Mining Technical Committee under the IEEE Computational Intelligence Society (formerly the IEEE Neural Network Society). He is also a vice-president and governing board member of the Asian Pacific Neural Network Assembly. He is serving or has served as a program and/or organizing member in numerous top international conferences and workshops (e.g., WWW, ACM MM, CIKM, ICME, ICASSP, IJCNN, ICONIP, and ICPR). He has also served as a reviewer for international conferences as well as journals (e.g., *Information Fusion*, *IEEE TCAS*, *SIGMOD*, *IEEE Transactions on Neural Networks*, *IEEE Pattern Analysis and Machine Intelligence*, *IEEE Transactions on Multimedia*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on System, Man, and Cybernetics, Machine Vision and Applications*, *International Journal of Computer Vision*, *Real-Time Imaging*, *SPIE Journal of Electronic Imaging*, and *International Journal of Pattern Recognition and Artifcial Intelligence*). He is a senior member of the IEEE and a member of the ACM, the IEEE Computer Society, the International Neural Network Society, and the Asian Pacific Neural Network Assembly.