2014 IEEE 8th International Symposium on Service Oriented System Engineering

A Hierarchical Matrix Factorization Approach for Location-based Web Service QoS Prediction

Pinjia He, Jieming Zhu, Jianlong Xu and Michael R. Lyu
Shenzhen Research Institute, The Chinese University of Hong Kong, China
{pjhe, jmzhu, lyu}@cse.cuhk.edu.hk

***Abstract*—With the rapid growth of population of service-oriented architecture (SOA), services are playing an important role in software development process. One major issue we should consider about Web services is to dig out the one with the best QoS value among all functionally-equivalent candidates. However, since there are a great number of missing QoS values in real world invocation records, we can hardly do a detailed comparison among those selectable Web services. To address this problem, we propose a location-based hierarchical matrix factorization method to make efficient and accurate QoS prediction. In our method, we consider both global context and local information. We first apply matrix factorization (MF) on global user-service records and obtain a global prediction matrix. After that, we use MF to predict QoS values on some user-service groups, which are clustered by K-means algorithm. Then we combine global and local predicted QoS values to provide our final prediction. Extensive experiments show the effectiveness of our hierarchical approach which outperforms other popular methods.**

***Keywords*—Web service; QoS prediction; Location-aware; Clustering**

I. Introduction

Web services are reusable, self-describing and loosely-coupled software components designed to construct distributed systems over the Internet. Because of the booming of Web 2.0, more and more companies tend to develop their software systems by means of combining existing Web services. Using Web service is convenient and can greatly accelerate the system development process. But the accompanying problem is how to select the most suitable Web services that can not only provide the functions we need, but also provide non-functional good performance.

To select a Web service, we will pay attention to its functionality first. That is, we have to find out those Web services which provide the function we want. After that, we will have a set of functionally-equivalent Web services and what we need to do next is to select one from them for the purpose of giving us the best non-functional performance according to QoS values (e.g. response time, throughput, reliability, etc.).

One common precondition of the selection process is that we have already known QoS values of those candidate Web services. However, this may not be true due to the following reasons: (1) Service users (i.e. software system developers) usually requested very few Web services compared with the amount of all available ones. A lot of QoS values are still unobserved so that historical records may not be helpful. (2) It's impractical for service users to try out all the candidates when they want to select the best one. Because it's time-consuming and sometimes will cost a good deal of money. (3) From the service providers' perspective, imagine that they have to respond to many requests while most of which are sent for the purpose of testing or collecting QoS data. It will be quite annoying because those "fake" requests will cost a lot of computing resource as well as network communication resource. Therefore, owning all Web services' QoS values is almost impossible. There are a great number of missing values in historical records and we have to find a method to predict these values.

QoS prediction is such a technique that use existing QoS values in invocation records to calculate and predict those unobserved ones. Collaborative filtering (CF) is a popular and effective way to do QoS prediction. In real case, many user-service matrices are sparse, for which some collaborative filtering methods can not fit well. Matrix factorization is one of those state-of-the-art collaborative filtering approaches that can help us when the user-service matrix comes across sparsity attribute. This method assumes there are some latent factors that affect QoS values. Hence, in matrix factorization, the user-service matrix will be separated into two low rank matrices which can reveal the significance of each latent factor and prediction will be given based on them.

To improve the prediction accuracy, we are natural to explicitly take more factors into consideration. Geographical information is a factor that we can collect and make use of because user-perceived QoS is closely related to the location of users as well as services. It's easy to imagine that users located in the same region are more likely to enjoy similar service quality. Besides, services deployed in the same area often provide services with QoS values alike. Because users and services in the same region share identical network infrastructure so that they have similar network workload and bandwidth. Thus, it will be better to predict QoS values on matrix whose users and services both located in the same place. We propose a hierarchical QoS prediction approach which considers both local context and global information. Users and services are separated into several user-service groups according to their geographical information, which is longitude and latitude of the corresponding node in this paper. We will conduct QoS prediction globally first, utilizing the whole user-service matrix. After that, we predict those missing QoS values locally, making use of geographical information. At last, we combine globally and locally predicted values and derive our final prediction.

For the purpose of describing this problem intuitively, an example is given here. In Fig. 1, users and services in the same "cloud" means that they are in the same geographical region, while each arrow among them indicates a Web service

978-1-4799-3616-8/14 $31.00 © 2014 IEEE
DOI 10.1109/SOSE.2014.41

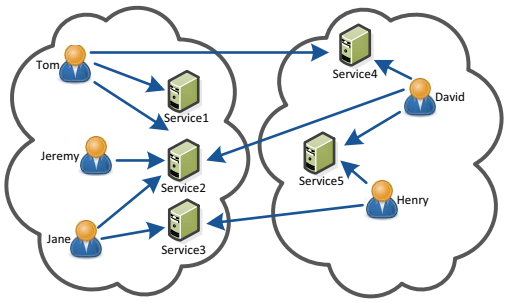


Fig. 1. A Simple Example of Web Service Invocations

call. Suppose that we want to predict the response time when Jeremy invokes Service2. From typical point of view, we will regard all the user-service invocation records as a big matrix and apply matrix factorization on it. But this method only considers global context. In our hierarchical method, we will also do matrix factorization on each "cloud", which is a small user-service matrix. After that, we will combine both global and local predicted QoS values to get a more accurate prediction.

The key contribution of this paper includes:

- We combine location information with matrix factorization approach, which makes fully use of location as well as historical request data.
- We propose a hierarchical way to perform matrix factorization, which can effectively improve the prediction accuracy.
- Extensive experiments are conducted on a real-world dataset which contains 1,974,675 invocation records. Experiment result shows that our prediction performance outperforms other state-of-the-art collaborative filtering methods.

The rest of this paper is organized as follows: Section II introduces the overall structure of hierarchical prediction method. Section III details steps in our model. Section IV presents experiment results. Related works are discussed in Section V. Finally Section VI concludes the paper.

II. Framework of Our System

Fig. 2 illustrates the framework of our location-based hierarchical recommendation system. Firstly we will collect and preprocess those existing service invocation records, keep a record of user list, service list, QoS values (e.g. response time, throughput) and other related information such as IP addresses. We implement matrix factorization on global historical invocation records first. Then our system will apply K-means algorithm on the dataset including all users as well as services according to their geographical information. (We will explain it in detail in Section IV) After clustering, we will get several groups, each one of which contains some users and services located in a close region. One single group itself forms a local user-service matrix. What our system do next is to apply matrix factorization approach on each one of them. The final work is to combine outcomes of global and local MF to produce the hierarchical prediction. We will introduce each step mentioned above in next Section.

III. Hierarchical Web Service Recommendation

In this section, we will explain our hierarchical recommendation approach in detail. We will first introduce the representation of geographical information in this paper and how we utilize it. After that, we will talk about how to form small user-service groups by K-means algorithm. Finally, hierarchical matrix factorization is explained.

A. Geographical Information

Longitude and latitude is the geographical information that we use. We design a tuple (log_i, lat_i) to represent the longitude and latitude value of user i or service i. Both log_i and lat_i are floating-point numbers ranging from -180 to 180. We map all users and services into a 2-dimensional space by using their longitude and latitude as their coordinates. Thus, we can apply clustering algorithms on this 2-dimensional space and form some user-service groups.

B. Finding User-service Groups

We apply K-means algorithm on both user nodes and service nodes to form clusters, which can reveal the geographical similarity between nodes. After clustering, we abandon clusters with only a handful of users or services because to get a reasonable and accurate prediction result, the number of users or services have to be bigger than the number of latent factors defined in matrix factorization. In K-means algorithm, we select initial center points first, and then we use an EM-type local search until the algorithm converges. We choose K-means algorithm because it is fast and has a good performance on low-dimension data (Our problem is 2-dimensional). The biggest drawback of K-means algorithm is that its performance is highly related to the given initial center points. But in the context of this paper, what we want to cluster are some nodes on a 2-dimensional geographical map. Thus we can easily choose the most suitable initial centers by drawing a 2-dimensional map of all users as well as services according to their longitude and latitude. When we finish the clustering step, we will get several clusters containing a number of users and services. We transform each cluster into a local user-service matrix by extracting corresponding QoS values from the big global matrix. An example is showed here to explain the transformation step in detail.

Assume that we have 4 users and 5 services, then we can form a global QoS matrix such as Fig. 3(a). After K-means clustering, we find out that user u_2, u_3 and u_4 as well as service s_1, s_2 and s_5 are in the same cluster. Then those corresponding QoS values, which are cells with grey background color in Fig. 3(a), will be extracted to form a local user-service matrix such as Fig. 3(b). In real world problems, we will form several clusters and transform them to different local user-service matrices. These matrices as well as the global user-service matrix will be the input of our hierarchical matrix factorization approach.

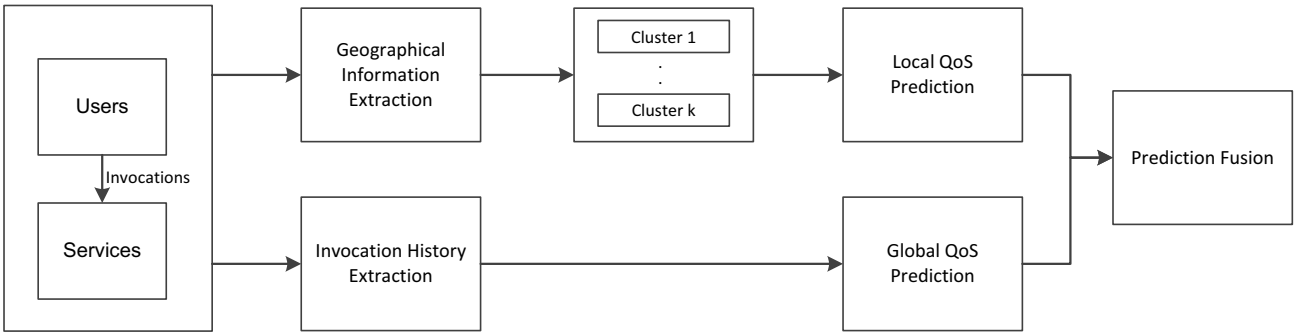
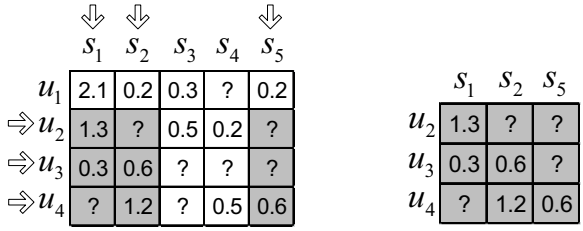


Fig. 2. Framework of Hierarchical Web Service Recommendation System



(a) Global matrix (b) Local matrix

Fig. 3. A Simple User-service Group Example

C. Hierarchical Matrix Factorization

To predict QoS values of both global matrix and local matrices, we use probabilistic matrix factorization model [1] (same as matrix factorization mentioned above). Matrix factorization is a state-of-the-art model-based collaborative filtering approach which can complete a matrix containing a lot of missing values. In user-service context discussed in this paper, matrix factorization model assumes there are several latent factors that affect both user side and service side of Web service invocation process. Thus, it approximates a matrix by the product of two low-rank matrices, each of which reveals the relationship between latent factors and users as well as services, respectively.

In this concrete problem, we suppose there are m users and n services. Hence we have an $m \times n$ matrix R. A cell in the matrix with index (i, j) is the QoS value generated by invocation from user i to service j. Since a user usually only called a few services before, the user-service matrix will be very sparse. To address this problem by matrix factorization, we will approximate matrix R by the product of two low rank matrices:

$$R \approx U^T S \tag{1}$$

where one of which is a $d \times m$ matrix U, while the size of the second low rank matrix S is $d \times n$. Each column of matrix U and S reveals the importance of latent factors for the corresponding user or service. For example, U_{di} reflects how much latent factor d affects the QoS values perceived by user i. Then we need to minimize the following term to calculate those missing values:

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i^T S_j)^2 \tag{2}$$

where I_{ij} indicates whether user i has invoked service j before. Its value is 1 when there's an invocation record between the corresponding user and service, while equals 0 otherwise. To avoid overfitting, two regularization terms are added to Eq. 2:

$$\frac{\lambda_1}{2}\|U\|_{Fro}^2 + \frac{\lambda_2}{2}\|S\|_{Fro}^2 \tag{3}$$

where $\|\cdot\|_{Fro}^2$ denotes the Frobenius norm. Then the following equation is used as objective function to perform prediction for those missing values:

$$\begin{aligned}\mathcal{L}(R, U, S) = &\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i^T S_j)^2 \\ &+ \frac{\lambda_1}{2}\|U\|_{Fro}^2 + \frac{\lambda_2}{2}\|S\|_{Fro}^2\end{aligned} \tag{4}$$

In [1], gradient descent is used on both matrix U and matrix S to get the local minimum of the objective function $\mathcal{L}$. We also use this method because it is fast and fits well in Web service context according to previous literatures [2]. The following equations are adopted to update values in matrix U and matrix S:

$$U' = U - \eta_1(\sum_{j=1}^{n} I_{ij}(U_i^T S_j - R_{ij})S_j + \lambda_1 U_i) \tag{5}$$

$$S' = S - \eta_2(\sum_{i=1}^{m} I_{ij}(U_i^T S_j - R_{ij})U_i^T + \lambda_2 S_j) \tag{6}$$

where η_1 and η_2 are learning rates of this algorithm. Then this matrix factorization method is applied to all the matrices we have, including global matrix and those local ones. We assume that after doing K-means clustering, we formed k clusters, which are then transformed to k local matrices. We say that an invocation record $(i, j) \in L_k$ when user i and service j are both in cluster k. Our hierarchical matrix factorization

approach is defined as:

$$P_{i,j} = \begin{cases} \hat{R}_{ij} & \text{if } (i,j) \notin L_k \\ \beta \hat{R}_{ij} + (1-\beta)\hat{R}_{k\_ij} & \text{if } (i,j) \in L_k \end{cases}$$

where $\hat{R}_{ij}$ is the QoS value between user i and service j predicted by matrix factorization on the global matrix, while $\hat{R}_{k\_ij}$ is the corresponding predicted QoS value given by using matrix factorization on local matrix k. β is the parameter to indicate how much global prediction affects our final outcome. This parameter can be tuned to find out the best balance between global context and local information.

IV. Experiments

In this section, we conduct experiments to evaluate prediction accuracy of our model. More specifically, we aim at answering these following questions:

- How does our method perform on real-world dataset compared with other methods?
- What is the influence of matrix density on our approach?
- How does β affect the prediction accuracy of our method?

A. Dataset

We use a public Web service dataset which contains information of 339 users, 5,825 services and 1,974,675 invocation records between them. There are two kinds of records: response time and throughput. It was collected by Zheng et al and detailly introduced in a related paper [3]. While including longitude and latitude of users, this dataset doesn't record services' geographical information. We collect longitude and latitude data of services according to their IP addresses by sending requests to IPLocation[1] and crawl the information we need in those responding web pages. In order to clearly clarify our idea, among all QoS values, we will only consider response time in this paper.

B. Metric

Mean Absolute Error (MAE) is used to evaluate the prediction accuracy of the hierarchical matrix factorization method. MAE is defined as:

$$MAE = \frac{1}{N} \sum_{i,j} |R_{ij} - \hat{R}_{ij}| \tag{7}$$

where R_{ij} is the QoS value of service j observed by user i, $\hat{R}_{ij}$ represents the corresponding predicted QoS value, N is the number of predicted QoS values. MAE reflects the overall difference between predicted values and real world invocation records. It's a negatively-oriented metric, which means lower values are better.

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

C. Comparison

In this section, we conduct experiments on state-of-the-art methods and compare them with the method proposed by this paper:

- **UPCC:** This is a memory-based collaborative filtering method, which calculates the similarity between each user pair and then use those most similar ones to predict for a certain user.
- **IPCC:** The main idea of this method is the same as UPCC. But instead of similar users, this approach focuses on discovering similar services. After finding those similar services, this approach does prediction based on them.
- **UIPCC:** This hybrid method is the linear combination of UPCC and IPCC, which can absorb the advantages from both UPCC method and IPCC method.
- **PMF:** This is a state-of-the-art model-based collaborative filtering method which performs well in completing matrix with a great deal of missing values. It assumes there are several latent factors affecting values in the matrix and uses the product of two low-rank matrices to do approximation.
- **HMF:** This is our method which uses hierarchical matrix factorization to predict missing QoS values.

As we mentioned above, the user-service matrix is quite sparse because each user may only invoke a few services. To make our experiments more convincing, we randomly remove some QoS values in the user-service matrix. After this preprocessing, we conduct experiments on user-service matrices with different density, which are $10\%, 15\%, 20\%, 25\%$ and 30%. A matrix with density 10% means that among all invocation values in that matrix, 90% of them are missing. In our prediction model, we set $\lambda_1 = \lambda_2 = 50$, $\eta_1 = \eta_2 = 0.0002$. The dimension is 10, which indicates 10 latent factors in our model. We set $\beta = 0.6$ because it gives us the best balance between global context and geographical information. (We will explain the impact of β in detail in Part IV-E.)

We run each approach 10 times and present the calculated average MAE values in Table I. Our model presents the lowest MAE values under all density settings, which indicates the hierarchical matrix factorization approach outperforms other methods. Thus, taking geographical information into consideration and using it in a hierarchical way really improve the prediction performance.

D. Impact of Density

In our model, the density of a user-service matrix means the ratio of the number of historical invocation records against the number of all the elements in that matrix. We conduct experiments on matrices with different density by randomly removing some QoS values. In our experiments, matrix density ranges from 10% to 50%, with interval of 5%. For other parameters, we set $\beta = 0.6$, $\lambda_1 = \lambda_2 = 50$ and $\eta_1 = \eta_2 = 0.0002$. For each density, we run our model 10 times and show the average MAE values in Fig. 4.

TABLE I. PERFORMANCE COMPARISON

Methods	Density = 10%	Density = 15%	Density = 20%	Density = 25%	Density = 30%
	MAE	MAE	MAE	MAE	MAE
UPCC	0.560	0.520	0.491	0.471	0.457
IPCC	0.599	0.524	0.463	0.439	0.421
UIPCC	0.552	0.500	0.453	0.431	0.415
PMF	0.515	0.471	0.446	0.428	0.416
HMF	**0.508**	**0.467**	**0.442**	**0.425**	**0.413**

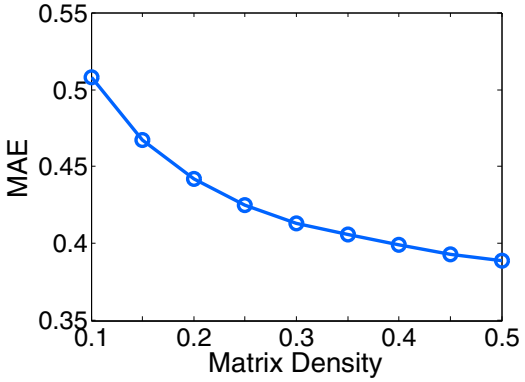


Fig. 4. Impact of Density on HMF

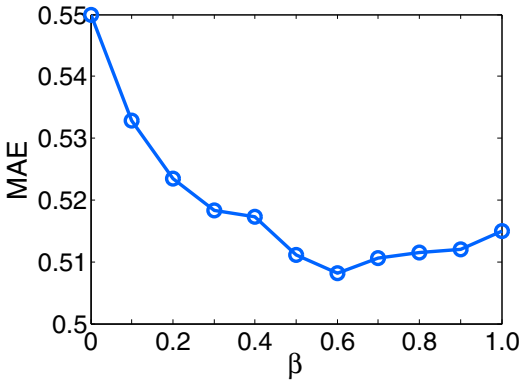


Fig. 5. Impact of β on HMF

Fig. 4 shows that when density of the matrix increases from 10% to 50%, the MAE of our model continuously goes down. That indicates more historical invocation records are helpful for our prediction accuracy, because it provides us with more information to analyze and make the performance of our approximation better. Besides, the curve goes down at a high speed at first but becomes smoothly as density increases. This phenomenon tells us that when the matrix is pretty sparse, knowing more historical invocation records is important for us to get an excellent prediction. But when the user-service matrix is not that sparse, bringing in more observed QoS values will only improve the performance of our model to little extent.

E. Impact of β

In our approach, the parameter β indicates how much will global context affects our final prediction. When β decreases, we will bring more geographical information into consideration. If $\beta = 1$, our method performs just the same as PMF since we don't gain any information from user-service clusters. When $\beta = 0$, we directly use locally-predicted QoS values if both the corresponding users and services are in user-service groups. To dig out the most suitable β, we conduct extensive experiments, varying the value of β from 0 to 1 at the step of 0.1. For other parameters, we set $density = 0.1$, $\lambda_1 = \lambda_2 = 50$ and $\eta_1 = \eta_2 = 0.0002$.

Fig. 5 shows the impact of β on our model. We can see that when β is small, which indicates we mainly take local prediction result into consideration, the MAE of our model is large. Because when we mainly predict corresponding QoS values by local prediction results, the number of available invocation records are quite small. Especially for a sparse matrix, information given by a user-service group is too little for location MF to dig out the relationship between latent factors and users as well as services. As β becomes bigger, the MAE of our model goes down. When $\beta = 0.6$, we get the smallest MAE, which indicates that 0.6 is the most suitable value for parameter β. When $\beta > 0.6$, the MAE goes up as β increases. That shows it really counts that we incorporate global context and local information. Because when we tend to use only global predicted values, the performance of our model degrades.

V. RELATED WORK

As the service-oriented architecture becomes pervasive, an increasing number of literature [4], [5], [6] raised discussion on Web services QoS field. Services composition [7], [8] and Web services selection [9], [10], [11] are two major problems that attract much attention from industry as well as academia. In terms of services composition, Alrifai et al. [12] proposed a hybrid method which allows us to tremendously reduce the overhead of finding a Web service combination strategy that satisfies a number of end-to-end QoS constraints. In service selection, Mohana el at. [13] classify existing service selection protocols into semantic or non-semantic class and design a fuzzy expert system to look for best service based on QoS values. For service composition and selection problems mentioned above, there is an premise that we have already known all the QoS values between users and services. However, this precondition is not always satisfied in real world web services circumstances. Because each user may only invoked a few services in the registry, there will be a lot of missing values and the user-service matrix might be quite sparse. To address this problem, we propose a model-based collaborative filtering method to predict those missing values.

Collaborative filtering is widely used in recommendation systems, whose main idea is to analyze the existing user-item records, dig out the relationship between users, items or both.

In recent years, a great deal of Web service related literature use collaborative filtering techniques to get accurate predicted Web services QoS values, which is necessary in most of service composition or service selection works. The most popular and common-used collaborative filtering methods fall in two categories: memory-based and model-based. In memory-based approach, the overall idea is to find out some neighbors first and use their QoS values to implement prediction. We have several ways to define the word "neighbor". User-based methods [14] look for similar users, who are defined as "neighbors" to the current user, and predict QoS value for current user according to neighbours' invocation records. Zheng et al. [15] proposed a hybrid method which take advantage of both related users and services. Tang et al. [16] designed a location-aware hybrid memory-based collaborative filtering method to further improve the prediction accuracy.

Model-based collaborative filtering methods use existing user-service invocation records to train a well designed model. After that, the trained model can predict those unobserved QoS values we want. In these model-based approaches, we separate user-service matrix into the product of two low rank matrix, which are user-latent factor matrix and service-latent factor matrix. We assume that only a few factors, which we may not know, determine user-perceived QoS values. User-latent factor matrix and service-latent factor matrix contain the relationship between user and latent factor as well as service and latent factor, respectively. Zhang et al. [17] proposed a real-time performance prediction method for cloud component based on tensor factorization. Lo et al. [2] added a location-related regularization term to original matrix factorization model which make use of users' altitude and latitude information.

VI. Conclusion and Future Work

This paper presents a location based hierarchical matrix factorization approach to predict missing QoS values. Instead of doing matrix factorization on the whole large user-service matrix, such as most existing model-based collaborative filtering methods do, we make use of both global context and geographical information of users as well as services. We apply matrix factorization on user-service groups, which are smaller matrices, clustered by longitude and latitude information of each user and service node. Finally, we combine the prediction result given by global matrix factorization and locally predicted QoS values. Experiments show that our hierarchical approach obtain higher accuracy than other similar methods.

In the future, to continuously improve our prediction performance, we have three aspects of things to do. First, we can try out other location information such as AS (Autonomous System), IP prefix and continent information of nodes. Besides, from the clustering algorithm's perspective, we may consider K-means Parallel and BFR since they are scalable clustering algorithm, which can arm us the ability to deal with web service prediction problem in big data circumstance. In terms of matrix factorization itself, we will find out some other factors except location to improve the prediction outcome.

Acknowledgment

The work described in this paper was 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 Nos. JCYJ20120619153834216), and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 415311).

References

[1] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in neural information processing systems*, 2007, pp. 1257–1264.

[2] W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "An extended matrix factorization approach for qos prediction in service selection," in *Proceedings of the 9th International Conference on Services Computing (SCC'12)*. IEEE, 2012, pp. 162–169.

[3] Z. Zheng, Y. Zhang, and M. R. Lyu, "Distributed qos evaluation for real-world web services," in *Proceedings of the 17th International Conference on Web Services (ICWS'10)*. IEEE, 2010, pp. 83–90.

[4] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient qos-aware service composition," in *Proceedings of the 18th international conference on World Wide Web (WWW'09)*. ACM, 2009, pp. 881–890.

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

[6] Z. Zheng and M. R. Lyu, "Personalized reliability prediction of web services," *ACM Trans. Softw. Eng. Methodol.*, vol. 22, no. 2, pp. 12:1–12:25, Mar. 2013.

[7] J. E. Haddad, M. Manouvrier, G. Ramirez, and M. Rukoz, "Qos-driven selection of web services for transactional composition," in *Proceedings of the 6th International Conference of Web Services (ICWS'08)*, 2008, pp. 653–660.

[8] D. Hamlet, "Tools and experiments supporting a testing-based theory of component composition," *ACM Trans. Softw. Eng. Methodol.*, vol. 18, pp. 12:1–12:41, June 2009.

[9] L. Mei, W. K. Chan, and T. H. Tse, "An adaptive service selection approach to service composition," in *Proceedings of the 6th International Conference of Web Services (ICWS'08)*, 2008, pp. 70–77.

[10] P. Bonatti and P. Festa, "On optimal service selection," in *Proceedings of the 14th international conference on World Wide Web (WWW'05)*, 2005, pp. 530–538.

[11] A. Goscinski and M. Brock, "Toward dynamic and attribute based publication, discovery and selection for cloud computing," *Future Generation Comp. Syst.*, vol. 26, no. 7, pp. 947–970, 2010.

[12] M. Alrifai, T. Risse, and W. Nejdl, "A hybrid approach for efficient web service composition with end-to-end qos constraints," *ACM Transactions on the Web (TWEB'12)*, vol. 6, no. 2, p. 7, 2012.

[13] R. Mohana and D. Dahiya, "Approach and impact of a protocol for selection of service in web service platform," *SIGSOFT Softw. Eng. Notes*, vol. 37, no. 1, pp. 1–6, Jan. 2012. [Online]. Available: http://doi.acm.org/10.1145/2088883.2088896

[14] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the 14th conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1998, pp. 43–52.

[15] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Wsrec: A collaborative filtering based web service recommender system," in *Proceedings of the 16th International Conference on Web Services (ICWS'09)*. IEEE, 2009, pp. 437–444.

[16] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for qos-based service recommendation," in *Proceedings of the 19th International Conference on Web Services (ICWS'12)*. IEEE, 2012, pp. 202–209.

[17] Y. Zhang, Z. Zheng, and M. R. Lyu, "Real-time performance prediction for cloud components," in *ISORC Workshops*. IEEE, 2012, pp. 106–111.