A PARTIAL LEARNING PREDICTION OF RELIABILITY RANKING APPROACH FOR WSN SERVICES

Wei Xiong¹, Zhao Wu^{1*}, Bing Li² and Qiong Gu¹

School of Mathematics and Computer Science, HuBei University of Arts and Science, XiangYang, China¹

International School of Software, Wuhan University, Wuhan, China²

xwei9093@126.com, wuzhao73@163.com, bingli@whu.edu.cn, gujone@163.com

Abstract

Wireless Sensor Network Service Applications (WSA) are playing an important role in Wireless Sensor Network (WSN), which bridge the gap between WSN and existing widely deployed Service-Oriented Architecture (SOA) technologies. Function properties of WSN services are important, which assure correct functionality of WSA. Meanwhile, nonfunctional properties such as reliability might significantly influence the client-perceived quality of WSA. Thus, building high-reliability WSA is a critical research problem. Reliability rankings provide valuable information for making optimal WSN service selection from functionally equivalent service candidates. There existed several methods that can conduct reliability ranking prediction of WSN services. However, it is difficult to evaluate which one is better than another, because those acquire different rankings with different preference functions. This paper proposes a partial learning prediction of reliability ranking approach for WSN services on past service usage experiences of other WSAs, which can achieve higher accuracy and improve the performance by pruning candidate services. To validate our approach, large-scale experiments are conducted based on a real-world WSN service dataset. The results show that our proposed approach achieves higher prediction accuracy than other approaches.

Keywords: Quality-of-Service; WSN Service; Ranking Prediction; Reliability

1. INTRODUCTION

Wireless Sensor Network (WSN) is conducted to achieve interoperability between Internet and physical world (Gubbi, 2013; Bellavista, 2012), which has been imported to adapt business processes and underlying software infrastructure to changes on things quickly and flexibly. Meanwhile, determining how to build high-reliability WSN Service Applications (WSA) becomes a critical issue with wide-spread WSN services.

Software reliability is the probability of failure-free software operation for the specified period of time in a specified environment (Radatz, 1990). In traditional software reliability engineering, there are several approaches to building reliable software systems. Now, large-scale complex cloud applications lead to that building reliable system is exceedingly difficult. In case of service computing, user-observed reliability of a service-oriented system not only relies on the system itself, but also heavily depends on remote services and user characteristics (e.g., geographic locations, network conditions, operational profiles (Musa, 1993), etc.). Influenced by the unpredictable Internet connections and many other environmental and operational factors, different service users may experience different reliability performance on the same WSN service. We can improve the reliability of service-oriented systems using fault-tolerant WSN services, which employ functionally equivalent yet independent services. To optimally select reliable WSN services, we must be able to predict the reliability rankings of WSN services. The ability to predict reliability rankings of invoked WSN services early at

design phase can help to reduce maintenance cost and to produce more reliable service-oriented systems. Therefore, reliability ranking prediction of WSN services is crucial for the WSAs.

It is difficult for present reliability ranking prediction approaches to perform excellently, because we retrieve different rankings with different preference functions.

We propose a partial learning reliability ranking prediction approach for WSN services (PLRRP). The approach utilizes history logs of conducting real-world WSN service evaluation at client-side to predict reliability rankings of WSN services. Our approach helps to achieve higher prediction accuracy for reliability ranking prediction of WSN services.

The contributions of this paper can be summarized as follows:

1) We propose a partial learning prediction of reliability ranking approach for WSN services, so that it can achieve higher prediction accuracy and improve the performance by pruning candidate services.

2) We conduct comprehensive experiments on the real-world dataset, demonstrating the effectiveness of our approach.

The remainder of this paper is organized as follows: Section 2 describes a motivating Scenario. Section 3 presents our approach. Section 4 describes our experiments in detail. Section 5 discusses the related works, and Section 6 concludes the paper.

2. A MOTIVATING SCENARIO

An approach to reliability ranking prediction of WSN services needs to record failure probability of invoked services at client-side, where the data can be recorded with little overhead and without compromising the functionality and usefulness of WSA. When an invocation happens, the client-side should store records in logs. Then, the records is encoded and transferred to a proxy-server. This process can be made transparent to client and does not negatively influence system performance. Moreover, records logs convey only partial information of all services. Thus, user-service matrix of failure probability, which is constructed based on logs, is sparse.

Most straightforward approaches to reliability ranking prediction of WSN services evaluate all candidate services at client-side and rank the services based on observed values of failure probability. However, this approach is impractical in reality, since the invocations of WSN services may be charged in terms of a large number of service invocations, which are time and resource-consuming even if invocations are free, let alone service invocations may produce irreversible effects in real world. When the number of candidate services is large, it is difficult for the WSAs to evaluate all the WSN services efficiently, which means that user-service invocation matrix is data sparsity.

Moreover, most straightforward approaches usually developed a preference function to combine all attributes, which is an aggregation procedure that produces aggregation rankings from a collection of individual rankings. The challenge is how to define preference function to combine all attributes, which can ensure monotone. To see how this condition happens, suppose that we have three individuals named i , h , and m , which includes two fields named X and Y , denoted by (X, Y) . Suppose further that

- individual i 's value is (2, 3),
- individual h 's value is (3, 4),
- individual m 's value is (2, 5.5).

Then, we have $h \succ i$ (since $3 > 2$ in X , $4 > 3$ in Y), $m \succ i$ (since $2 = 2$ in X , $5.5 > 3$ in Y). However, $h \prec, \succ m$ (since $3 > 2$ in X , $4 < 5.5$ in Y). As an example, given preference function $f(x, y) = \frac{x}{x+y}$, we retrieve $\langle i, .5 \rangle, \langle h, .7 \rangle, \langle m, .75 \rangle$, where the number with each individual indicates its score. Furthermore, given preference function $f(x, y) = \frac{2x}{2x+y}$, we retrieve $\langle i, .7 \rangle, \langle h, .10 \rangle, \langle m, .9.5 \rangle$. Thus, we retrieve different rankings with different preference function. This Scenario does show some of the complications that arise when different individual engages in various preferences in the presence of multiple criteria.

Next, there are some other issues that need to be addressed to improve the performance by pruning candidate services as well.

Thus, one of our tasks is to fulfill missing entry values in matrix due to data sparsity. we will extract Pareto set of WSN services from the matrix, achieve the preference

function by learning from them, which is applied to produce aggregation rankings for all services.

3. OUR APPROACH

3.1 Fulfilling Incomplete Failure Probability Matrix

3.1.1 Similarity Computation

Pearson Correlation Coefficient (PCC) (Benesty, 2008) is utilized in many recommendation systems to compute the similarity because it can be easily implemented and can achieve higher accuracy. User-based collaborative filtering adopts PCC (UPCC) (Shao, 2007) to compute similarity between service user u and v by the following equation:

$$Sim(u, v) = \frac{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}, \quad (1)$$

where $I_u \cap I_v$ is the subset of WSN services, which both user u and v have invoked previously, $r_{u,i}$ is a vector of failure probability values of WSN service i observed by service user u , $\bar{r}_u$ and $\bar{r}_v$ represent average failure probability values of different WSN services observed by service user u and v respectively. $Sim(u, v)$, the similarity of two service users, is in the interval of -1 to 1, where a larger PCC value indicates that service user u and v are more similar. When two service users have null WSN service intersection ($I_u \cap I_v = \emptyset$), $Sim(u, v)$ cannot be determined ($Sim(u, v) = null$), since we do not have information for the similarity computation.

Item-based collaborative filtering methods using PCC (IPCC) (Resnick, 1994) are similar to the user-based methods. The similarity computation of two WSN services i and j can be calculated by

$$Sim(i, j) = \frac{\sum_{u \in U_i \cap U_j} (r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U_i \cap U_j} (r_{u,i} - \bar{r}_i)^2} \sqrt{\sum_{u \in U_i \cap U_j} (r_{u,j} - \bar{r}_j)^2}}, \quad (2)$$

where $Sim(i, j)$ is the similarity between WSN service i and j , $U_i \cap U_j$ is the subset of service users who have invoked both service i and j previously, $\bar{r}_i$ and $\bar{r}_j$ represents the average failure probability values of the WSN service i and j observed by different service users. $Sim(i, j)$ is also in the interval of -1 to 1. When two WSN services have null service user intersection ($U_i \cap U_j = \emptyset$), the value of $Sim(i, j)$ cannot be determined ($Sim(i, j) = null$) as well.

3.1.2 Similar Neighbors Selection

To predict a missing value $r_{u,i}$ in the user-service matrix, a set of similar users $S(u)$ can be found by the following equation:

$$S(u) = \{v | v \in U, \text{sim}(u, v) \geq \tau_u\}, \quad (3)$$

and a set of similar WSN services $S(i)$ can be found by the following equation:

$$S(i) = \{j | j \in I, \text{sim}(i, j) \geq \tau_i\}, \quad (4)$$

where $T(u)$ is a set of top- k similar users to the user u and $T(i)$ is a set of top- k similar services to service i .

3.1.3 Missing Values Prediction

Denote the set of similar neighbors of user u by $S(u)$ and the set of similar neighbors of service i by $S(i)$.

User-based collaborative filtering method named as UPCC(Shao, 2007) applies similar users to predict the missing failure probability values by the following equation:

$$\hat{r}_u(u, i) = \frac{\sum_{v \in S(u)} \text{sim}(u, v) (r(v, i) - \bar{r}(v))}{\sum_{v \in S(u)} \text{sim}(u, v)}, \quad (5)$$

where $\bar{r}(u)$ denotes average failure probability values of user u , and $\bar{r}(v)$ denotes the average failure probability values of user v .

Similar to the user-based methods, item-based collaborative filtering method named as IPCC(Resnick, 1994) engages similar WSN services to predict the missing value by employing the following equation:

$$\hat{r}_u(u, i) = \frac{\sum_{j \in S(i)} \text{sim}(i, j) (r(u, j) - \bar{r}(j))}{\sum_{j \in S(i)} \text{sim}(i, j)}, \quad (6)$$

where $\bar{r}(i)$ denotes average failure probability values of service i , and $\bar{r}(j)$ denotes the average failure probability values of service j .

Since these two predicted values may have different prediction performance, we employ a tunable parameter α to balance these two predicted values. We integrate two methods by following equation:

$$\hat{r}(u, i) = \alpha \hat{r}_u(u, i) + (1 - \alpha) \hat{r}_i(u, i), \quad (7)$$

where parameter α determines how much the hybrid method relies on UPCC prediction or IPCC prediction. Parameter α provides the adaptability to the context.

3.2 Learning Preference Function

Now, we need to achieve preference function by learning based on Pareto set $\mathcal{L}$, which is applied to produce aggregation rankings for all WSN services.

3.2.1 A Framework of Learning Preference Function

We focus on modeling a user's preference function of the form $\mathcal{L}: I \times I \rightarrow R$, where $\mathcal{L}(i, j) \geq 0$ means that service i is more preferable to j for user u and vice versa where $\mathcal{L}(i, j)$ can be defined as the following equation:

$$\mathcal{L}(i, j) = \frac{1}{n} \sum_{u \in U} \hat{r}_u(u, i) - \hat{r}_u(u, j), \quad (8)$$

where $\mathbf{r}$ denotes weight vector, $\hat{r}$ is failure probability matrix. Then we can achieve aggregation strength between two WSN services by employing the following equation:

$$\mathcal{L}(i, j) = \frac{1}{n} \sum_{u \in U} \hat{r}_u(u, i) - \hat{r}_u(u, j). \quad (9)$$

The aggregation ranking r^* should be returned according to $\mathcal{L}(i, j)$

What is an appropriate measure of similarity between the rankings r_u observed by user u and the target aggregation ranking r^* ? Given an independently and identically distributed, the learner will select a ranking function $f(u, i)$ from a family of ranking functions $\mathcal{F}$ that maximizes the empirical $\mathcal{L}$:

$$\max_{f \in \mathcal{F}} \frac{1}{n} \sum_{i \in I} \mathcal{L}(f(i), r^*) \quad (10)$$

on training sample, where Kendall's $\mathcal{L}$ can be defined as:

$$\mathcal{L}(a, b) = \frac{P - Q}{P + Q} \quad (11)$$

P is the number of concordant pairs and Q is the number of discordant pairs under the finite domain of m services.

3.2.2 Partial Learning Mechanism

Definition 1. (Service Pareto Set) Given a service set S , each service's QoS values q_{ij} observed by users, service Pareto set is defined as

$\text{Par} = \{i \in S \mid \nexists j \in S, j \neq i, \text{ such that } q_{ij} < q_{ji} \text{ for all } i, j \in S\}$, where $i < j$ iff $\hat{r}(u, i) < \hat{r}(u, j)$, for $i, j \in S$, $u \in U$.

Next we study on a partial learning approach based on service Pareto set to obtain the optimal preference function.

Lemma 1. Any service that has the best score over services with respect to any monotone scoring function with respect to services must be in Pareto set.

Proof. Assume that this is not the case. Let $i, j \in \text{services}$, $\hat{r}(u, i)$ be a maximum, so $\hat{r}(u, i) \geq \hat{r}(u, j)$, but $\hat{r}(u, i) < \hat{r}(u, j)$. Then $\hat{r}(u, i) \geq \hat{r}(u, j)$, Contradiction $\square$.

Theorem 1. Any total ranking of services with respect to any monotone scoring function is a topological sort with respect to dominance relationship.

Proof. Assume otherwise. Thus, there exist $i, j \in \text{services}$ and a monotone scoring function W such that $i \prec j$ but $W(i) > W(j)$.

As noted in the proof of Lemma 1, because $i \prec j$, $f(u, i) \leq f(u, j)$, for $u \in \text{users}$. However, then $W(i) > W(j)$, by scoring function W 's definition. Contradiction. $\square$

Thus, if $W(j) > W(i)$, it is possible that $j \prec i$, but we are certain that $i \prec j$ does not exist. If $W(j) > W(i)$, then either $j \prec i$ and $i \prec j$, or i and j are incomparable with respect to dominance relationship. Service j of Pareto Set is a service such that there is no $i \in \text{services}$ such that $i \prec j$. Then, all services in Pareto Set ranks higher than others outside Pareto Set. Furthermore, as noted in Theorem 1, after sorting all services with respect to the monotone function, no services in the front cannot be dominated by services in the back, and services outside Pareto Set ranks statically. Thus, if we can rank services in Pareto Set, the final top- k ranking will be included in Pareto Set. Next we focus on how to extract Pareto set.

3.2.3 Pareto Set Extraction

The Pareto set is important for several applications involving multi-criteria decision making. We develop a Pareto set extraction algorithm based on dominance relationship. Details of our algorithm are described in Algorithm 1. The algorithm has a time complexity $O(n^2)$, where n denotes the number of services.

Algorithm 1 Pareto set extraction

Input $\hat{r}$

Output $\hat{L}, \hat{L}^H$

Notation

$\hat{r}$ is failure probability records.

W is an window.

$\hat{L}$ is Pareto set.

$\hat{L}^H$ is a preference set of the other individual items that are dominated by any other item on all dimensions.

T is a heap.

tmp is a temp file.

Body

unfinish=true

$T = \text{heap}(\hat{r})$

while (unfinish) **do**

unfinish=false

while ($t_i \in \text{next}(T)$) **do**

while ($d \in W$)

If not-dominated(t_i, d) **then**

If full(W) **then**

unfinish=true

break

else

$W = W \cup t_i$

end

else
 $\text{next}(T) = \text{next}(T) \cup t_i$

end

end

If (unfinish) **then**

$S = \text{openfile}(\text{tmp})$

$\text{write}(S, t_i)$

while ($t_i \in \text{next}(T)$) **do**

If not-dominated(t_i) **then**

$\text{write}(S, t_i)$

end

end

end

$\text{free}(T)$

$\text{close}(S)$

$\hat{L} = \text{merge}(W)$

$W = \emptyset$

$T = \text{heap}(\text{tmp})$

end

return $\hat{L}, \hat{L}^H$

3.2.4 Ranking SVM Method

We could design an algorithm and a scoring function W by maximizing (10) based on Pareto set. Consider the class of linear ranking functions

$$(S_i, S_j) \cdot \hat{r} * \hat{r}(i, j) > 0 \quad (12)$$

where $\hat{r} = (\hat{r}_1, \hat{r}_2, \dots, \hat{r}_k)$ which is a weight vector that is adjusted by learning from Pareto set, $\hat{r}(i) = (\hat{r}_1(i), \hat{r}_2(i), \dots, \hat{r}_k(i))$.

This is equivalent to

$$(S_i, S_j) \cdot \hat{r}_{lin} * \hat{r}(u, i) > \hat{r}(u, j) \quad (13)$$

...

$$(S_i, S_j) \in \mathcal{P}_{u_i} : \frac{1}{\|u_i\|} \langle u_i, i \rangle > \frac{1}{\|u_j\|} \langle u_j, j \rangle \quad (14)$$

However, a direct generalization of this problem shows that is NP-hard. It is possible to approximate the solution by introducing non-negative slack variables $\frac{1}{\|u_i\|} \langle u_i, i \rangle$ and minimizing the upper bound $\frac{1}{\|u_i\|} \langle u_i, i \rangle$, just like in classification SVMs (Cortes, 1995). Adding SVM regularization for margin maximization to the objective leads to the following optimization problem, which is similar to the ordinal regression approach.

$$\text{minimize: } L(\frac{1}{\|u_i\|} \langle u_i, i \rangle) \quad (15)$$

subject to:

$$(S_i, S_j) \in \mathcal{P}_{u_i} : \frac{1}{\|u_i\|} \langle u_i, i \rangle - \frac{1}{\|u_j\|} \langle u_j, j \rangle \leq \frac{1}{C} \quad (16)$$

...

$$(S_i, S_j) \in \mathcal{P}_{u_i} : \frac{1}{\|u_i\|} \langle u_i, i \rangle - \frac{1}{\|u_j\|} \langle u_j, j \rangle \leq \frac{1}{C} \quad (17)$$

$$\frac{1}{\|u_i\|} \langle u_i, i \rangle \leq \frac{1}{C} \quad (18)$$

where C is a parameter that allows trading-off margin size against training error. Geometrically, the margin $\frac{1}{C}$ is the distance between the closest two projections within all target rankings, which is set to 1 here. The above optimization problem is convex and has no local optima. By rearranging the constraints (16) as

$$\frac{1}{\|u_i\|} \langle u_i, i \rangle - \frac{1}{\|u_j\|} \langle u_j, j \rangle \leq \frac{1}{C} \quad (19)$$

it becomes apparent that the optimization problem is equivalent to that of a classification SVM on pairwise difference vectors $\frac{1}{\|u_i\|} \langle u_i, i \rangle - \frac{1}{\|u_j\|} \langle u_j, j \rangle$. Due to this similarity, it can be solved using decomposition algorithms similar to those used for SVM classification. Then, the algorithm has a time complexity $O(n^3)$, where n denotes the number of Pareto set $\mathcal{P}$.

Finally, we can achieve the aggregation rankings r^* for all services due to equation (12) with weight vector $\frac{1}{\|u_i\|}$ by learning.

3.2.5 Complexity Analysis

This section discusses the upper bound on the worst-case computational complexity of the ranking prediction algorithm of WSN services reliability. We assume there are m service users and n WSN services in the training matrix.

As shown in fulfilling matrix, algorithms 1 and the aggregation ranking approach, Time complexity of fulfilling matrix is $O(m^2 n^2)$, where n denotes the number of services and m denotes the number of users; Time complexity of algorithms 1 is $O(n^2)$, where n denotes the number of services. Time complexity of aggregation ranking approach is $O(k^3)$, where k denotes

the number of Pareto set $\mathcal{P}$ and is less than the number of services. Thus, the total computational complexity of whole procedure is $O(k^3 m^2 n^2)$, which indicates that the computational complexity is scalable to the size of datasets, so that the PLRRP approach can be employed on very large datasets.

4. EXPERIMENTS

4.1 Data Set Description

We adopt a real-world WSN service dataset, which was collected in our current WSN service system, which includes 23 million WSN service invocation results from 102 distributed service users on 3474 available real-world WSN services. We observe, collect, and contribute evaluation results of target WSN services to our centralized server, which is implemented by CentOS7.0, Dock1.3.2, Kubernetes0.6.2, and etcd0.4.6. Each WSA conducts a number of invocations on each selected WSN service and the invocation results are recorded. The dataset contains QoS records of service invocations on 3474 WSN services from 102 service users, which are transformed into a user-service matrix. Each entry of user-service matrix is a value of failure probability. We utilize failure probability matrix to achieve the reliability ranking prediction of WSN services in experiments. Although we only study failure probability in the experiments, PLRRP approach can be applied to other reliability properties easily.

4.2 Metrics

We utilize the normalized discounted cumulative gain (NDCG) (Karvelin, 2002) metric, which can be calculated by

$$NDCG_k = \frac{DCG_k}{IDCG_k} \quad (20)$$

where DCG_k and $IDCG_k$ denote the discounted cumulative gain (DCG) values of predicted rankings and ideal rankings of Top- k services, respectively.

The value of DCG_k can be calculated by

$$DCG_k = \sum_{i=1}^k \frac{rel_i}{\log_2 i} \quad (21)$$

where rel_i is the reliability value of the service at position i of rankings, the value of k is in the interval of 1 to n , where n is the total number of WSN services.

4.3 Comparison

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

- 1) Greedy. This method is proposed for ranking a set of items, which treats explicitly rated items and unrated items equally (Liu, 2008). It does not guarantee that

- the explicitly rated items will be ranked correctly.
- 2) CloudRank1. This method calculates reference values between items, and employs these values for making QoS ranking prediction (Zheng, 2013).
 - 3) CloudRank2. This method considers confidence levels of different preference values, which serves to achieve better ranking accuracy (Zheng, 2013).
 - 4) Rank Aggregation. This method performs adequate information fusion for WSN service selection by inferring global QoS (Hofreiter, 2013).

Both CloudRank1 and CloudRank2 employ KRCC (the Kendall Rank Correlation Coefficient) (Zheng, 2013) for user similarity computation. However, our method employs PCC for user similarity computation.

In the real world, the user-service matrix are usually very sparse since a service user usually invokes a small number of WSN services. In this study, to conduct our experiments realistically, we randomly remove entries from the user-service matrix to make the matrix sparser with different densities (i.e., 10, 20, and 30 percent). Matrix density 10 percent, for example, means that we randomly select 10 percent of the failure probability entries to predict the reliability rankings. The rankings based on the original full matrix are employed as ideal rankings to study the ranking prediction accuracy. The above three methods together with our PLRRP method are

employed for making ranking prediction for each user in the user-service matrix employing the available failure probability information in the matrix. We set the training user number to 80 or 102, top- k to 50 and $\frac{1}{L}$ to 0.3 in the prediction methods in this experiment. NDCG10 indicates that the ranking accuracy of the top 10 items is investigated. The value of NDCG10 can be calculated by equation (20). So do NDCG80 as well. The results of Table 1 show that:

- 1) Under all experimental settings, our method obtains the largest NDCG value mostly, which indicates better prediction accuracy. This observation indicates that achieving the preference function by learning, which is applied to produce aggregation rankings for all services may provide more accurate.
- 2) The NDCG value of PLRRP has enhanced with the increase of training matrix density from 10 to 30 percent, indicating that denser user-service matrix provides more information, which will impact the prediction accuracy.
- 3) With the increase of the training user number from 80 to 102, the prediction accuracy achieves some enhancement, since larger training matrix may provide more information for the prediction.

Table I. Performance Comparison.

Methods	Training Users=80						Training Users=102					
	Matrix Density=10%		Matrix Density=20%		Matrix Density=30%		Matrix Density=10%		Matrix Density=20%		Matrix Density=30%	
	NDCG10	NDCG80	NDCG10	NDCG80	NDCG10	NDCG80	NDCG10	NDCG80	NDCG10	NDCG80	NDCG10	NDCG80
Greedy	0.4202	0.6858	0.7068	0.8118	0.7863	0.8750	0.4422	0.7021	0.8002	0.8660	0.8880	0.9105
CloudRank1	0.4436	0.6943	0.7227	0.8232	0.8049	0.8841	0.4675	0.7115	0.8454	0.8866	0.8956	0.9263
CloudRank2	0.4453	0.7022	0.7268	0.8372	0.8137	0.8853	0.4732	0.7228	0.8693	0.8902	0.9036	0.9272
Rank Aggregation	0.4477	0.7089	0.7286	0.8472	0.8244	0.8903	0.4882	0.7363	0.8703	0.8911	0.9087	0.9299
PLRRP	0.4566	0.7237	0.7599	0.8533	0.8441	0.8921	0.4766	0.7361	0.8795	0.8956	0.9121	0.9328

4.4 Impact of Matrix Density

As shown in Table 1, the prediction accuracy of our PLRRP approach is influenced by matrix density. To study the impact of the matrix density on the prediction results, we change the matrix density from 10 to 100 percent with a step value of 10 percent. We set the training user number to 80 or 102, top- k to 50 and $\frac{1}{L}$ to 0.4 in this experiment.

Figure 1 shows the results, where Figure 1a and 1b are the results when the training user number is 80, and Figure 1c and 1d are the results when the training user number is 102. Figure 1 show that the prediction accuracy of the PLRRP approach increases when the matrix density is increased from 10 to 100 percent.

Figure 1. Impact of parameter matrix density

4.5 Impact of Parameter λ

In our PLRRP method, parameter λ controls how much our method relies on IPCC and UPCC. If $\lambda=0$, we only employ IPCC for making prediction. If $\lambda=1$, we predict the failure probability purely by UPCC. In other cases, we integrate IPCC and UPCC for missing failure probability value prediction. we set the training user number to 80 or 102, top-k to 50 and the matrix density to 0.3 in this experiment.

Figure 2 shows the impacts of parameter λ on the results. We observe that optimal λ value settings can achieve better prediction accuracy. For failure probability, as λ increases, the NDCG value reaches a certain summit, and then decreased. This phenomenon confirms the intuition that purely using UPCC or purely employing IPCC cannot generate better ranking prediction performance than integrating these two favors together.

Figure 2. Impact of parameter λ

4.6 Impact of Parameter Top-k

Parameter top-k determines the number of similar users to the user u or similar services to service i employed in our PLRRP approach. To study the impact of parameter top-k, we change the values of parameter top-k from 10 to 80 with a step value of 10. We set the training user number to 80 or

102, λ to 0.4 and the matrix density to 0.3 in this experiment.

Figure 3 shows that optimal λ value settings Can achieve better prediction accuracy. When the parameter top-k is increased from 10 to 80, the NDCG value reaches a certain summit, and then decreased.

Figure 3. Impact of parameter top- k

5. RELATED WORK AND DISCUSSION

The approaches to reliable software systems include fault prevention (e.g., rigorous development process), fault removal (e.g., testing and debugging techniques) (Tsai, 2008), fault tolerance (designing diversity) (Lyu, 2015), and fault forecasting (Gokhale, 2002) in traditional software reliability engineering. Moreover, distributed systems employ the methods of software fault tolerance to enable reliability (Gorender, 2007), which include recovery block (Randell, 1995), N-Version Programming (NVP) (Avizienis, 1995), N self-checking programming (Laprie, 1990), distributed recovery block (Kim, 1989), and so on.

Security, reliability and performance in WSA is an important issue, which has attracted wide attention from researchers (Torres, 2011; Kifayat, 2013; Seo, 2013). Great progress has been made in this field.

Services have been widely employed in a lot of domains in WSN (e.g., service composition, service selection, fault-tolerant services, and etc). Based on QoS, various approaches have been proposed for service selection (Yu, 2007), which enables optimal service to be identified from a set of functionally similar or equivalent candidates.

QoS ranking prediction using collaborative filtering technique was conducted firstly by Liu et al. (Liu, 2008), named Greedy. CloudRank is the first personalized QoS ranking prediction framework for Web services by Zheng et al. (Zheng, 2013), where two approaches are studied: CloudRank1 and CloudRank2, and a series of large-scale experiments are carried out based on a real Web services dataset.

Moreover, Hofreiter B et al. (Hofreiter, 2013) propose a rank aggregation for QoS-Aware WSN service Selection. However, it is a bit rough even it is based on the Condorcet criterion implemented in Condorcet measure.

6. CONCLUSION AND FUTURE WORK

In this study, we propose a partial learning prediction of reliability ranking approach for WSN services on the past service usage experiences, which requires no additional service invocations when making reliability rankings. By taking advantage of past usage experiences of other WSAs, our ranking approach identifies and aggregates the preferences between pairs of services to produce the rankings of services. Our approach extracts the Pareto set of WSN services from the matrix based on dominance relationship, and achieves the preference function by learning from Pareto set, so that it can achieve higher prediction accuracy for ranking prediction and improve the performance by pruning candidate services. The results of extensive experiments show the effectiveness of our approach.

The PLRRP approach in this study can be utilized to predict rankings of WSN services reliability in different contexts. We plan to conduct more studies by employing time factor. Client-side reliability values usually manifest

trend, seasonality, periodicity and random on the time dimension due to resource-consuming of the host where service is located. Moreover, if network traffic factor is considered, the issue will become more complicated.

7. ACKNOWLEDGMENT

The authors appreciate the reviewers for their informative comments. The work described in this study was fully supported by the National Key Research and Development Program of China (No. 2016YFB0800400), the National Natural Science Foundation of China under grant (Nos. 61572371) and the Natural science foundation of Hubei province in China (Nos. 2016CFB406).

8. REFERENCES

- Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Futur. Gener. Comput. Syst. Int. J. Grid Comput. Science*, v(29), pp. 1645–1660.
- Bellavista, P., Cardone, G., Corradi, A., Foschini, L. (2012). The Future Internet convergence of IMS and ubiquitous smart environments: An IMS-based solution for energy efficiency. *J. New. Comput. Appl.*, v(35), pp. 1203–1209.
- Radatz J., Geraci A., Katki F. IEEE standard glossary of software engineering terminology. *IEEE Std. 1990*, 610121- 990: 121990.
- Musa J. D. (1993). Operational profiles in software-reliability engineering. *Software, IEEE.*, v(10), pp. 14-32.
- Benesty J., Chen J., Huang Y. et al. (2008). Pearson correlation coefficient, Noise reduction in speech processing, Springer Berlin Heidelberg.
- Shao L., Zhang J., Wei Y., Zhao J., Xie B., Mei H. (2007). Personalized QoS Prediction for WSN services via Collaborative Filtering. *Proceedings of the Fifth WSN services (ICWS '07)*, pp. 439-446.
- Resnick P., Iacovou N., Suchak M., Bergstrom P., Riedl J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of Computer Supported Cooperative Work*; pp.175-186.
- Cortes C., Vapnik V. N.. (1995). Support vector networks, *Machine Learning Journal.*, v(20), pp. 273-297
- Karvelin K. J., Kekalainen J. (2002). Cumulated Gain-Based Evaluation of IR Techniques, *ACM Trans. Information Systems*, v(20), pp. 422-446.
- Liu N. N., Yang Q. (2008). Eigenrank: a ranking-oriented approach to collaborative filtering, *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp.83-90.
- Zheng Z., Wu X., Zhang Y., et al. (2013). QoS ranking prediction for WSN services, *IEEE Transactions on Parallel and Distributed Systems*, v(24), pp. 1213-1222.
- Hofreiter B., Marchand-Maillet S. (2013). Rank Aggregation for QoS-Aware WSN service Selection and Composition, *Proceedings of the 6th International Conference on Service-Oriented Computing and Applications (SOCA2013)*, pp. 252-259.
- Tsai W. T., Zhou X., Chen Y., Bai X. (2008). On Testing and Evaluating Service-Oriented Software, *Computer*, v(41), pp. 40-46.
- Lyu M. R. (2005). *Software Fault Tolerance*, Wiley.

Gokhale S. S. , Trivedi K. S. (2002). Reliability Prediction and Sensitivity Analysis Based on Software Architecture, *Proceedings of Symp. Software Reliability Eng.* (ISSRE '02), pp.64-78.

Gorender S., de Araujo Macedo R.J., Raynal M. (2007). An Adaptive Programming Model for Fault-Tolerant Distributed Computing, *IEEE Trans. Dependable and Secure Computing*, v(4), pp. 18-31.

Randell B. , Xu J. (1995). The Evolution of the Recovery Block Concept, *Software Fault Tolerance*, v(3), pp. 1-21.

Avizienis A. (1995). The Methodology of N-Version Programming, *Software Fault Tolerance*, v(3), pp. 23-46.

Laprie J., Arlat J., Beoune C., Kanoun K. (1990). Definition and Analysis of Hardware- and Software-Fault-Tolerant Architectures, *Computer*, v(23), pp. 39-51.

Kim K. , Welch H. (1989). Distributed Execution of Recovery Blocks: An Approach for Uniform Treatment of Hardware and Software Faults in Real-Time Applications, *IEEE Trans. Computers*, v(38), pp. 626-636.

Torres, C., Glosekotter, P. (2011). Reliable and energy optimized WSN design for a train application, *J. Syst. Arch.*, v(57), pp. 896-904.

Kifayat, K., Merabti, M., Shi, Q., Abbas, S. (2013). Component-based security system (COMSEC) with QoS for wireless sensor networks, *Secur. Commun. Netw.*, v(6), pp. 461-472.

Seo, H., Shim, K.A., Kim, H. (2013). Performance enhancement of TinyECC based on multiplication optimizations. *Secur. Commun. Netw.*, v(6), pp. 151-160.

Yu T., Zhang Y., Lin K. J. (2007). Efficient Algorithms for WSN services Selection with End-to-End QoS Constraints, *ACM Trans. Web*, v(1), pp. 1-26.

Authors

Wei Xiong is the lecturer at Hubei University of Arts and Science, China. He received his B.S. in computer science and technology from Central China Normal University in 1995, M.S. degrees in software engineering from Northwestern Poly-technical University in 2007 and his Ph.D. degree in computer science from Wuhan University, China in 2015. His research interest includes the development of distributed system and service computing, as well as software engineering. He participates in the IEEE Computer Society Sister Society Associate Program.

Zhao Wu is the professor at Hubei University of Arts and Science, China. He received his B.S. in computer application from China University of Geoscience in 1999, M.S. degrees in computer application from Wuhan University of Technology in 2003 and his Ph.D. degree in computer science

from Wuhan University in 2007. He is a member of the China Computer Federation. His research interests include cloud computing, service computing and internet of things.

Bing Li is the professor at Wuhan University. He is a expert in software engineering and service computing. Bing Li received his Bachelor's degree from the Department of Computer Science at Huazhong University of Science and Technology (HUST) in 1990 and his Master's degree from HUST in 1997.

He has received his PhD at the Computer Science School at HUST in 2003. He has worked as a post doctor researcher in The State key laboratory of software engineering (SKLSE) & School of Computer from 2003 to 2005. He plays very active role in scientific and professional public appearance: he is a senior member of the China computer Federation(CCF). He serves as member in three CCF technical Committee: open systems, software engineering and service computing. In recognition of his contributions to scientific and technological progress Bing Li received "Second Class Award of the Chinese National Scientific and Technological Progress Award (2009)", "First Class Award of the HuBei Province Scientific and Technological Progress Award (2008, 2011)", "First Class Award of the Natural Sciences Award of Chinese Ministry of Education (2007)".

Qiong Gu is the associate professor at Hubei University of Arts and Science, China. She received the M.S. degree and the Ph.D. degree in 2006 and 2009, respectively, in China University of Geosciences. She is a member of the China Computer Federation. Her research interests include data mining, machine learning, net-mediated public sentiment, and internet of things.