An Approach for Web Service QoS Dynamic Prediction

Hai Yan
[1,]North China University of Water Resources and Electric Power/ College of Information Engineering,Zhengzhou , China
[2,]Hohai University/ College of Computer and Information Engineering ,Nanjing , China
Email: haiyan@ncwu.edu.cn

Zhi-zhong Liu
Henan Polytechnic University/College of Computer sciences and technology,Jiaozuo, China

***Abstract*—In order to accurately predict the Web service Qos in a highly dynamic environment,we put forward a Qos dynamic prediction method based on Semi-Markov Processes(SMP) and Case-based Reasoning(CBR).This method firstly uses semi-Markov process to predict business state of web service in the future, and then applies the technology of CBR to predict Web service QoS, for example,when the service deals with a specific task. Experimental results show that this prediction method can improve the accuracy of Web service QoS greatly. The results provides a reliable basis for the objective evaluation and successful Web service composition.**

***Index Terms*—Web Service; QoS(Quality of Service); Dynamic Prediction; Semi-Markov Processes; Case-based Reasoning**

I. INTRODUCTION

Web service, as a new distributed computing model, has caused great concern of the industry and academia in recent years. With the rapid development of service-oriented computing technology, a large number of Web services with the same or similar functions but very different quality of service (QoS, Quality of Service) have appeared on the network, and QoS has become the main basis of the evaluation and selection of Web services.

Web service QoS refers to the non-functional attributes of the service. Currently, the QoS prediction of Web services is still in its initial stage, in which the average of the QoS arithmetic of Web service history is commonly used to approximate the QoS of the service [1,2]. In the open environment of the Internet, Web service QoS is highly dynamic [3,4]. First, the busy state of Web services processing tasks is an important factor of the dynamic changes of Web service QoS. When Web services are in a busy state, even dealing with the same task, its QoS is also different. Second, the task type and task quantity processed by Web services are also the main factors of the dynamic nature of QoS. Generally, Web services in the same state have different QoS when handling different types of tasks, and the data transfer rate is also a factor affecting the Web service QoS. However, the data transfer rate has greater impact to Web service QoS that needs to transfer large content, while the impact is relatively small for those Web services transferring less content. This paper mainly focuses on the prediction of Web service which is less influenced by the data transfer rate, setting the data transfer rate to a fixed value.

In order to accurately predict the QoS value of Web service in a period of time, this paper has proposed modeling on the transfer process of the state of Web service based on semi-Markov process to predict the state of the Web services in the future moment. On this basis, case-based reasoning technology is applied to predict the QoS of Web service in handling specific tasks. Experimental results show that the prediction method can significantly improve the accuracy of Web service QoS, which will provide a reliable basis for the objective evaluation of Web services.

II. QOS DYNAMIC PREDICTION MODEL

Dynamic prediction problems of Web service QoS can be described as follows: Given WS is a Web service, users submit task T to *WS* for processing in the next time t_f . The task type is TT_i and the task quantity is TQ_i . Solve the QoS value when *WS* service processes task *T*.

Due to the great impact of the state of Web services on QoS service and the dynamic nature of the state, so it is necessary to predict the state of the Web service at t_f . Based on the state prediction of Web service, the case-based reasoning technology is applied to predict the QoS when Web service processes task *T*. The

Manuscript received March 10, 2013; accepted September April 2013. This work was supported by the National Natural Science Foundation of China under Grant(No. 60805022) and the National High-Tech Research and Development Plan of China under Grant(Nos.2007AA01Z178)

doi:10.4304/jsw.8.10.2637-2643

dynamic prediction model of Web service QoS is shown in Figure 1.

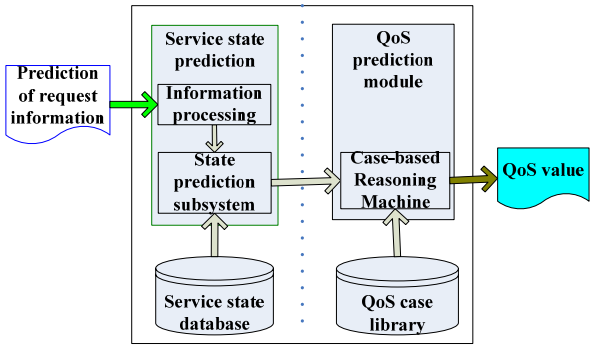

Figure 1. Dynamic prediction model of Web service QoS

The first module of the model is a Web service state prediction module. It consists of the Web service state database and the state prediction subsystem, which are responsible for predicting the state of Web services in the next moment t_f. Web service state database is in charge of the collection, sorting, and update of the Web service state transferring data, which provides data support for the state prediction subsystem. The other prediction module is composed of the Web service QoS case library and case-based reasoning machine,which is mainly used to predict the QoS when Web service handling a task. Web service QoS case library has responsibility for recording the state of Web services in processing tasks, characteristics in handling the task and the QoS of Web service in handling the task. The information will be compiled into specific cases of QoS to save and update QoS stories in accordance with certain rules, which will provide data support for case-based reasoning machine.

III. Web Services State Forecast Based On Semi-Markov Process

This section first defines the state of Web service according to how busy it is, and then models for the transfer process of Web services using the semi-Markov process, followed by the prediction for its state.

A. Definition of Web Service State

Busy state of Web services is essentially how busy it is in the system composed by Web services which should be measured from two levels. The first level is the Web server where Web services are, and its loading is a measurement level; another one is the concurrent access rates of the Web services, in general, its maximum amount is bound and could reflect its busy state. Here, the load of the web server and the concurrent access rates of the Web services are adopted to conduct comprehensive measurement of the busy state. The relevant formula is given below.

Definition 1 Load of Web server:

$$L(WSs) = w_{11}UR_{CPU} + w_{12}UR_{I/O} + w_{13}UR_{Memory} \quad (1)$$

$$, w_{11} + w_{12} + w_{13} = 1 .$$

UR_{CPU} indicates the utilization rate of the server CPU, $UR_{I/O}$ indicates the occupancy rate of server disk I/O, UR_{Memory} refers to the utilization rate of server memory occupancy. Where w_{11}, w_{12}, w_{13} are the weight of three kinds of hardware resource utilization rates when measuring the load of server, which could be determined by the business metric weight when the server is loaded. Its value can be determined based on the business types processed by the Web server.

Definition 2 Concurrent access rate of Web services:

$$L(WS) = \frac{\lambda_{current}}{\lambda_{\max}} \quad (2)$$

$\lambda_{current}$ represents the concurrent access amount of Web services, $\lambda_{\max}$ indicates the maximum concurrent access amount which the Web services can bear.

Definition 3 Busy state of Web services:

$$L = w_{21} \bullet L(WSs) + w_{22} \bullet L(WS) \quad (3) \quad ,$$

$$w_{21} + w_{22} = 1 .$$

w_{21} and w_{22} respectively indicates the significance of the server load and the Concurrent access rates of Web services for the measurement of Web services. On this basis, 5 busy states of Web services are defined, which are recorded by Level1,Level2,Level3,Level4and Level5, represented as the state set $S = \{L_1, L_2, L_3, L_4, L_5\}$. The follows are the definitions of these five states where $L(t)$ indicates the busy state of Web services at *t*, and $S(t)$ represents the state of service at t.

Definition 4 State of Web services

If $0 \le L(t) \le 20\%$, $S(t) = S_1 = L_1$;

If $20\% < L(t) \le 40\%$, $S(t) = S_2 = L_2$;

If $40\% < L(t) \le 60\%$, $S(t) = S_3 = L_3$;

If $60\% < L(t) \le 80\%$, $S(t) = S_4 = L_4$;

If $80\% < L(t) \le 100\%$, $S(t) = S_5 = L_5$;

B. Semi-Markov process model of Web Service State transfer

Semi-Markov process [5] is based on the Markov process[6] and a random process of the combination of discrete-continuous of the two parameters. A semi-Markov process with finite state and discrete time has similar definition to the Markov process. The only difference is that in the semi-Markov process, the condition transition probability of the system state is not only related to the current state but also connected

to the lingering time of the system in this state. In the state transition process of Web services, the next state that Web service is going to reach is only relevant to the current state of Web services and its residence time on the current state, free from the influence of other states. This random process is in line with the semi-Markov process nature, which can be modelled by the semi-Markov process. The semi-Markov process of Web service state transition is shown in Figure 2, in which the circles represent the state of Web services and the text between the circles represents the transition probability between states.

The semi-Markov process model of Web service state transition can be defined as follows: $SM =< Z, P, F >$,

wherein:

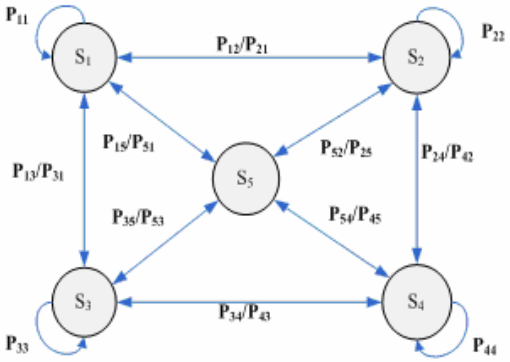

Figure 2. SMP model of Web service state transition

1) Z stands for the state set of Web services, $Z = \{1,2,3,4,5\}$. $Z = i, 1 \le i \le 5$ represents that Web services are in state S_i;
2) P refers to the transition probability matrix between the state; P_{ij} expresses the probability when services are transited from state *i* to state *j*, and $\sum_j p_{ij} = 1$.
3) Supposing the current state of Web services is *i*, Web services enter state *j* after staying time *d* in state*i*, and *d* obeys the probability distribution.

Given $SM =< Z, P, F >$ is the homogeneous semi-Markov process, its nuclear is $Q_{ij}(t) = P(Z_{n+1} = j, T_{n+1} - T_n \le t \mid Z_n = i) = p_{ij} F_{ij}(t)$. Among them, $p_{ij} = \lim_{t\to\infty} Q_{ij}(t) = P(Z_{n+1} = j \mid Z_n = i)$ and $F_{ij}(t) = \mathrm{P}(T_{n+1} - T_n \le t \mid Z_{n+1} = j, Z_n = i)$.

Define $H_i(t) = P(T_{n+1} - T_n \le t \mid Z_n = i)$, which expresses the probability distribution function of the residence time of Web services in state *i*, and $H_i(t) = \sum_{j=1}^{m} Q_{ij}(t)$. Literature [7] has provided the evolution equation Eq (4) of homogeneous semi-Markov process $Z = (Z_t, t \in R_0^+)$:

$$\begin{aligned}\phi_{ij}(t) &\doteq P(Z_t = j \mid Z_0 = i)\\ &= (1 - H_i(t))\delta_{ij} + \sum_{l=1}^{m} \int_0^t \phi_{lj}(t-\tau) dQ_{il}(\tau) \\ &= (1 - H_i(t))\delta_{ij} + \sum_{l=1}^{m} \int_0^t Q_{il}'(\tau)\phi_{lj}(t-\tau) d\tau\end{aligned} \quad (4)$$

,

$$\text{When } \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \ne j \end{cases};$$

We can calculate the transient distribution $\phi_{ij}(t)$ of homogeneous semi-Markov process by evolution equation (4). In order to quickly solve the equation (4), literature [8] gives the evolution equation Eq (5) of discrete-time homogeneous semi-Markov process (DTHSMP):

$$\phi_{ij}(kh) = d_{ij}(kh) + \sum_{l=1}^{m} \sum_{\tau}^{k} v_{il}(th)\phi_{lj}((k-t)h) \quad (5)$$

,

$$\begin{cases} d_{ij}(kh) = (1 - H_i(kh))\delta_{ij}; \\ H_i(kh) = \sum_{j=1}^{m} Q_{ij}(kh); \\ v_{il}(kh) = \begin{cases} Q_{il}(h) & for\ k = 1; \\ Q_{il}(kh) - Q_{il}((k-1)h) & for\ k > 1; \end{cases} \\ Q_{ij}(kh) = P_{ij} F_{ij}(kh); \end{cases} \quad (6).$$

In the equation, *h* is the time discretization step size, and $\phi_{ij}(kh)$ represents the probability of Web service entering state *j* after elapsing *Kh*from the moment of entering the state *i* (To ignore time step in the following formulas). When Web services are in state*i* and have stayed a period of time *s* in state *i*, it needs to calculate the probability $\Pr(Z_{s+k'} = j \mid Z_0 = i, t_{soj} = s)$ in predicting the state of Web services after a period of time k'. When the length of residence time of Web services in each state violates exponential distribution, $\phi_{ij}(k', s) \ne \phi_{ij}(k')$. To address this issue, Lee [7] gives an evolution equation Eq (7) based on Eq (5) in calculating the length of residence time that can be randomly distributed:

$$\hat{\phi}_{ij}(k',s)=\frac{P(Z_{s+k'}=j,t_{soj}=s|Z_0=i)}{P(t_{soj}=s|Z_0=i)}$$
$$=\frac{d_{ij}(s+k')+\sum_{l=1}^{m}\sum_{\tau=s+1}^{s+k'}v_{il}(\tau)\phi_{lj}(s+k'-\tau)}{[1-H_i(s)]} \quad (7)$$

.

This paper uses EQ (7) to calculate the probability value of Web services in each state k' $\hat{\phi}_{ij}(k',s),\ j=1,2,3,4,5$ after an arbitrary time period. The largest state of probability value is the future state of Web services.

C. The predicted numerical solution of the Web Service state

Eq (6) shows that when calculating $\hat{\phi}_{ij}(k',s),\ j=1,2,3,4,5$, it is necessary to calculate the transition probability P_{ij} between the calculating states and the distribution function F_{ij} of their sojourn time staying in each state. In DTHSMP model, the distribution of a discrete-time variable y depends on the history data of the system. To obtain P_{ij} and F_{ij}, transferring data between various Web services states should be recorded in real-time and stored in the form of a triple $ST_l =< S_i, t_{soj}, S_j \mid S_i, S_j \in Z >,\ l=1,2,...$. $ST_l =< S_i, t_{soj}, S_j \mid S_i, S_j \in Z >$ indicates that Web services stay in state S_i for t_{soj} and then transfers to state S_j. Assuming that $STDB=\{ST_1, ST_2, ..., ST_n\}$ is the database transferred by Web service state during $[t_0, t_c]$. For all the states $S_i, S_j \in Z$, the definitions[9] are shown as follows:

(1) $N_i(n) \doteq \sum_{n=1}^{n} 1_{\{J_n=S_i\}}$ shows the number of times that Web services experience state S_i from t_0 to t_c;

(2) $N_{ij}(n) \doteq \sum_{n=1}^{n} 1_{\{J_{n-1}=S_i, J_n=S_j\}}$ describes the number of times that the service transfers from S_i to S_j from t_0 to t_c;

(3) $N_{ij}(k,n) \doteq \sum_{n=1}^{n} 1_{\{J_{n-1}=S_i, J_n=S_j, t_{soj}=k\}}$ represents the number of times that the service transfers to state S_j after staying time k in state S_i from t_0 to t_c, and $k \le t_c - t_0$;

According to the contents of Chapter 3.2, respectively, give the calculation formula of the transition probability function of Web service state p_{ij} and the distribution function of the residence time $F_{ij}(k)$:

$$\hat{p}_{ij}(n) \doteq \frac{N_{ij}(n)}{N_i(n)} \quad (8)$$

$$\hat{F}_{ij}(k,n) \doteq \sum_{r=1}^{k}\frac{N_{ij}(r,n)}{N_{ij}(n)} = \sum_{r=1}^{k}\frac{1}{N_{ij}(n)}\sum_{n=1}^{n} 1_{\{J_{n-1}=S_i, J_n=S_j, t_{soj}=r\}} \quad (9)$$

The description of Web service state prediction algorithm is given below:

Algorithm1. SMP-based prediction algorithm of Web service state:

Input: $< S_i, t_{soj}, t_f, STDB >$ // S_i refers to the current state of Web services; t_{soj} represents the residence time of Web services on the current state; t_f stands for the moment calling Web services in the future; *STDB* expresses the transition database of Web service state;

Output: the state of Web services in the future moment t_f;

1. Begin
2. Calculate the value of $N_i(n)$, $N_{ij}(n)$, and $N_{ij}(k,n)$ on the basis of *STDB*;
3. Calculate $\hat{P}_{ij}(n)$ and $\hat{F}_{ij}(k,n)$ according to Eq (8) and Eq (9);
4. Calculate the probability of each state $\hat{\phi}_{ij}(k',s),\ j=1,2,3,4,5$ based on Eq (6) and Eq (7);
5. Select the state with the highest probability as the state of the Web services in future moment t_f;

IV. Web Service QoS Prediction on Based on Case Reasoning

A. Presentation of QoS Cases

In CBR system, case is generally composed of the description part and the solution part of problems. In this paper, problem descriptions mainly include state, task type, and task quantity of Web services; Solution vector mainly consists of the QoS value when Web services process tasks. Here, QoS cases of Web services are defined as follows: $SC_i =< WSS_i, TT_i, TQ_i \| q_{i1}, q_{i2}, ..., q_{ik} .... >$. In the equation, WSS_i, TT_i, TQ_i respectively represents the state, task type, and task quantity of the i[th]QoS case. $q_{i1}, q_{i2}, ..., q_{ik} ....$ refers to the processing task type of Web services TT_i in state S_i and QoS value when task

quantity is TQ_i.

B. Organizations of QoS Cases

In order to improve the case retrieval speed, it is essential to organize historical cases in a reasonable way. Aiming at QoS dynamic prediction problems of Web services, and considering the fact that the state, task type and task quantity of Web services constitute important factors of the characteristics of the problem, Indexing these characteristics to organize QoS cases can improve case retrieval speed. As QoS of Web services between the different types of tasks are not comparable, so first classify QoS cases based on the task type; then continue to classify the QoS cases based on the state of Web services in cases; Finally, sequentially store QoS cases with the same task type and the same service state in accordance with the size of the task quantity.

C. QoS Case Retrieval

Before performing the QoS case retrieval, structure target case according to the prediction state of Web services and characteristics of tasks to be processed, and identify k historcialQoS cases which are most similar to the target case in the QoS case library according to the retrieval rule. The target case is defined as $TC = < WSS_0, TT_0, TQ_0 \| q_{01}, q_{02}, ..., q_{0k}, ... >$, in which WSS_0, TT_0, TQ_0 can be extracted from the prediction result of the service state and user requested information. $q_{01}, q_{02}, ..., q_{0k}, ...$ signifies the predictable QoS attribute values. Here, define that historical QoS cases are similar to the target cases if and only if the degree of similarity between the two is not smaller than a preset value r. In order to improve the case retrieval speed, the following QoS case retrieve rules are developed:

(1) First, retrieve historical QoS cases having the same task type as the target cases;

(2) Retrieve k historical QoS cases which are most similar to the target cases from that in (1); the similarity between the target cases with historical cases is divided into local similarity and global similarity. Local similarity is the similarity between the problem characteristics of target cases and historical cases, while global similarity is the weighted average of local similarity. Literature [10] verifies that Manhattan distance is the best method to calculate the local similarity. The global similarity formula of Manhattan distance is given below:

$$SIM(TC, SC_i) = \frac{w_{31} \bullet sim_1 + w_{32} \bullet sim_2}{w_{31} + w_{32}} \quad (10)$$

,In which:

(1) $sim_1 = 1 - \frac{| State_0 - State_i |}{N}$ is the similarity formula for Web service state, and N refers to the total number of the Web service state.

(2) $sim_2 = 1 - \frac{| TQ_0 - TQ_i |}{TQ_{\max}}$ represents the similarity formula for task quantity, and $TQ_{\max}$ expresses the largest task quantity that the service can handle.

w_{31}, w_{32} respectively represents the similarity of Web service state and weights of the similarity of task quantity. Supposing that the retrieved *k* historical QoS cases most similar to the target cases are $\{SC_1, SC_2, \ldots, SC_k\}$, the solutions to the target cases are generated based on the *k* similar cases in the following.

D. QoS Prediction

After the most similar QoS cases are retrieved, the QoS value of the similar cases is used to generate the QoS value of the target case. In the current case reuse technologies, the primary methods to generate solutions to target cases based on historical cases of solutions include the most similar case method [11], that is, using the solution of the case most similar to the target case as the solution to the target case, and the mean value of solutions of all similar cases [12], that is, the average solutions of *k* historical cases used as the solution to the target case. This article draws on the literature [13], that is, the solutions with a high degree of similarity with the target case have strong influence on the generation of solutions to the target case, in which the solution reuse formula with the global similarity as the weight is given in the follows:

$$QoS_0 = \sum_{i=1}^{k} \frac{\delta + Sim(TC, SC_i)}{\sum_{j=1}^{k} (\delta + Sim(TC, SC_j))} QoS_i \quad (11)$$

In the formula, $QoS_0 = < q_{01}, q_{02}, ..., q_{0k}, ... >$ represents the solution vector of the target case, that is, to predict the QoS value. $QoS_i = < q_{i1}, q_{i2}, ..., q_{ik}, ... >$ indicates that the solution vector of the historical QoS case SC_i, in which δ is an arbitrary number greater than 0. Do substitution of the QoS vectorsof *k* most similar cases into Eq (11), and the predicted QoS value can be obtained. The QoS prediction algorithm based on the case-based reasoning is given below.

Algorithm2. Prediction algorithm of CBR-based Web service QoS

Input: target cases

Output: the predicted QoS values

Step1. Structure target case based on task characteristics and the predicted Web service state;

Step2. Retrieve the historical QoS cases having the same task types with the target case;

Step3. Identify k historical QoS cases most similar to the problem cases based on global similarity;

Step4. According to equation (11), apply the QoS of similar cases to generate the QoS of target cases, and output the predicted QoS value;
Step5. Correct the new QoS cases and save them;

V. Simulation Experim Ent

A. Design of Experiment

In order to verify the effectiveness of dynamic QoS prediction methods proposed in this paper, we conduct simulation experiments on the campus network Web services application platform, developing the Web services used to calculate the rank and eigenvalues of large matrix. Here, the calculation of the rank of the matrix is called task 1, while the calculation of the eigenvalues of the matrix is called task 2, with the order of the matrix representing the task quantity. In order to simplify the experiment, only the server CPU utilization is used here to indicate the duty cycle of Web services. In the experiment, record server CPU utilization every 2min from 8:00 to 18:00 up to a total of 10 working days. Calculate the transition probability matrix of Web service state P_{ij} according to the recorded values and residence time distribution function $F_{ij}(kh), k = 1,2,...,K$. In order to obtain QoS cases, some tasks are randomly generated, which are randomly from Task1 and Task2 with task quantity a random number in [500, 2000]. These tasks are then randomly submitted to the service for processing. Record the following data in the course of task submission, the job submission time T_{sub} , the completion of the task processing time $T_{complete}$ and the processing time of the task $T_{sub} - T_{complete}$. During tasking process, record server CPU utilization every 5sec, and after the task is completed, use the average CPU utilization during task processing as the CPU load in dealing with the task and map it as the service state in accordance with the definition of service state in Chapter 3. When the service task is processed, structure QoS cases based on the service state of the record, the task type, the task quantity, and the task processing time and store them in accordance with the organization manner in Chapter 4. The experiment has collected 1000 QoS cases and predicts the reaction time of Web services, which is the QoS indicator. The prediction method of other QoS indicators is the same as the method of this indicator.

B. Determination of Key Parameters

In DTHSMP model, h and K are the key parameters. Their values will affect the accuracy of the prediction of the service state. The test method used here is the best values of h and K. The specific experimental step is to randomly select a point in time, determine the state of the service in this moment and the residence time in this state, and then to predict the state of the service after the elapsed time kh. To this end, we provide respectively different combinations of parameters and do 30 times of predictions, respectively, in each of the parameter combinations. Besides, we calculate the accuracy rate of the prediction according to the formula $R_1 = 1 - \frac{|\text{State}_{Predicted} - \text{State}_{Real}|}{N}$, in which N is number of the service state. Table 1 shows the prediction accuracy of the service state prediction method under different parameter settings. As can be seen from the table, when the discrete time step is short, the prediction accuracy of service state is relatively high. This is because the short discrete time steps are better able to characterize the distribution function. Therefore, under the premise of meeting the actual prediction demand, the value of h should be as small as possible. As the value of K has great impact to the calculation time of the state prediction, the value of K should be also as small as possible. For the prediction in this paper, h = 60sec and K = 5

C. Comparisons of prediction results

In order to verify the accuracy of the QoS prediction of Web services, first randomly generate target cases with the task type as Task1 or Task2 and the task quantity randomly generated in the [500, 2000]. Then generate the predictable period of time in [1h, 5h] and randomly select the time to start prediction, from 8:00 to 18:00. Next, detect the state of the service as well as the residence time of the service in the current time in the predicted time, and then predict the state that the service will reach after a period of time kh. After the service state is predicted, use Algorithm 2 to predict the time of the service processing tasks. At the same time, starting from the predicted time, submit the target case to the service for processing after kh period of time, and record the actual time of completing tasks. Finally, compare the prediction time of task with the actual value, and apply the formula $R_2 = 1 - \frac{|T_{Predicted} - T_{Real}|}{T_{Real}}$ to calculate the accuracy of prediction. This experiment has produced 20 target cases. In the course of calculating the global similarity of target case and the historical cases, set $w_{31} = 0.55, w_{32} = 0.45$a, and the number of similar cases is set to 5; experimental results are shown in Figure 3. As can be seen from Figure 3, the accuracy of task processing time calculated by applying the prediction method in this paper is 73.43%, which is much more accurate than the traditional mean metric.

TABLE 1

PREDICTION ACCURACY TABLE UNDER DIFFERENT PARAMETER SETTINGS

	h=60sec	h=120sec	h=180sec
K=5	0.893	0.820	0.735
K=10	0.817	0.762	0.69
K=15	0.781	0.674	0.615

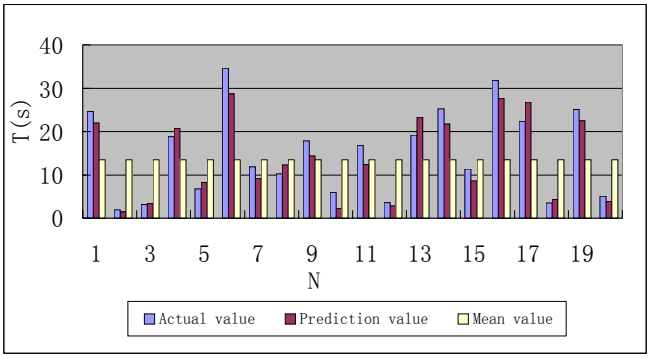

Figure 3. Comparison of the actual value, prediction value and mean value of task processing time

D. Experimental Analysis

The experimental results indicate that the QoS values of Web services obtained by the QoS prediction methods are more accurate than those obtained by the traditional QoS calculation method. The experimental results further illustrate that the factors influencing the QoS fluctuations should be taken into consideration. As this experiment only applies to the measurement of the busy state of Web services using CPU utilization of the server rather than using the server's memory and I / O utilization, which decreases the prediction accuracy. In subsequent studies, the setting of experimental environment will be further improved to refine the quantization of the busy state and take into account the factors influencing the QoS fluctuations such as the type of network and specific location of users, etc.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China with the Serial Number of 60805022 and the National High-Tech Research and Development Plan of China with the Serial Number of 2007AA01Z178 .

REFERENCES

[1] Zeng L Z, Benatallah B, Ngu AHH, et al. QoS-aware middleware for web services composition [A]. IEEE Trans. on Software Engineering[C]. Washington DC, USA: IEEE Computer Society.2004,30(5):311−327.

[2] Bai Jianbo*, Hao Yuzhe, Miao Guochang .Integrating Building Automation Systems based on Web Services[J].Journal of Software.2011,6(11):2209-2216

[3] Hwang SanYi, Wang Haojun, Tang Jian, et al.A probabilistic approach to modeling and estimating the QoS of web service-based workflows[J]. Information Sciences 2007,177(23): 5484-5503.

[4] Fan XQ,Jiang CJ,Wang JL,Pang SC. Random-Qos-Aware reliable Web service composition [J].Journal of software 2009. 20(3): 546-556.

[5] Altinok Y, Kolcak D. An application of the semi-Markov model for earthquake occurrences in North Anatolia, Turkey[J]. Journal of the Balkan Geophysical Society, 1999, 2(4): 90–99.

[6] Qian Zhang, Ming Li , Xuesong Wang , Yong Zhang Reinforcement Learning in Robot Path Optimization[J].Journal of Software.2012,3(3):2209-2216

[7] Corradi G, Janssen J, Manca R. Numerical treatment of homogeneous semi markov processes in transient case-a straight forward approach[J]. Methodology and Computing in Applied Probability,2004.6(2):233–246.

[8] Lee J K, Hou J C, Modeling steady-state and transient behaviors of user mobility: formulation, analysis, and application[A]. Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing[C]. Florence,Italy:ACM,2006:85–96.

[9] Barbu V, Limnios N. Nonparametric estimation for failure rate functions of discrete time semi-markov processes[A]. Probability, Statistics and Modelling in Public Health [C]. New York, USA:Springer press,2006,53-72.

[10] Li Y F, Xie M, Goh T N. A study of mutual information based feture selection for case based reasoning in software cost estimation[J]. Expert System with Applications 2009,36(3):5921-5931.

[11] Shiu S , Pal S K . Foundations of soft case based reasoning [M]. New York: John Wiley & Sons, 2004.

[12] Walkerden F, Jeffery R. An empirical study of analogy-based software effort estimation[J]. Empirical Software Engineering, 1999.4(2):135–158.

[13] Shepperd M, Schofield, C. Estimating software project effort using analogies[A]. IEEE Transactions on Software Engineering[C]. Washington DC, USA:IEEE Computer Society.1997,23:736–743.

Hai Yan: was born in Zhengzhou, is a professor of North China University of Water Resources and Electric Power of College of Information Engineering, is Ph.doctoral student of Hohai University of College of Computer and Information Engineering. She has published over 20 refereed research articles in academic conferences and journals in the areas of service compute and data mining .The major field focuses on the research of services science and data mining.

Zhi-zhong Liu received Ph.D. degrees from Hohai University of College of Computer and Information Engineering. He is currently an associate professor with the CS Department, Henan Polytechnic University. His research focuses on the Evolutionary algorithm and web services selection.