International Journal of Engineering Research ISSN:2319-6890 (online),2347-5013(print)
Volume No.6, Issue No.12, pp : 509-512 1 Dec. 2017

Optimal Web Service Selection Based On Network Distance And Web Service Performance

Miss Dipti Gumfawar[1], Ms. Manjushri Mahajan[2]
[1]Computer Network Department, [2]Computer Engineering Department
G.H. Raisoni College of Engineering and Management Wagholi, Pune, India
[1]dipti.polas@gmail.com, [2]manjushri.mahajan@raisoni.net

Abstract—The last decade has witnessed a tremendous growth of web services. web services are enabling organizations to use web as a market for selling their own web services. Nevertheless, it becomes more difficult to find the most appropriate web service that satisfies users' functional and non-functional requirements. Most of the former works in web service selection treat the QoS values as constants. However, QoS values of a service as perceived by a given user are intrinsically random. Proposed method mentioned is a novel approach to select the optimal web service considering response time and network latency using Hidden Markov Model.

Keywords—Web service, QoS (Quality of Service), Hidden Markov Model (HMM), response time, network latency, optimal web service.

I. Introduction

SERVICE reuse is often considered as a key aspect of Service-Oriented-Architecture (SOA). With tens of thousands of web services available on the Internet, and many of them are of equivalent functions, choosing the web services that meet a user's requirement becomes the core task of service-based software development.

SERVICE-ORIENTED computing and Web services are becoming more and more popular, enabling organizations to use the Web as a market for selling their own Web services and consuming existing Web services from others. Based on the latest statistics1, there are 28,593 Web services being provided by 7,728 different providers over the world and these numbers keep increasing in a fast speed. The explosive growth of Web services increases the difficulty for users to choose among many Web service candidates. Therefore, how to effectively select the Web services becomes a key challenge for the Web service community.

Recently, recommending qualified and preferred web services to users has attracted much attention in terms of the information overload problem. Web service recommendation is a process of pro-actively discovering and recommending suitable web services to end users. Several works have been done on service recommendation based on quality of service (QoS). Most of them employed collaborative filtering (CF) techniques some of them applied content-based approach, and a few of them combined CF approach with content-based techniques.

Collaborative filtering (CF) is a widely-used technique for Web service QoS prediction, which can be divided into memory-based CF, model-based CF, and other hybrid CF methods. Matrix factorization (MF) is a typical model-based approach in CF, which has been employed for QoS value prediction by academia and industry in recent years. In MF technology, the QoS values of Web services observed by different users can be represented as a user-service matrix. In the matrix, rows represent users, columns represent services, and each entry represents the QoS value of a service observed by a user. The main idea of MF-based QoS value prediction approaches is to train a model per the available QoS values in the user-item matrix (i.e., historical QoS values contributed by different users) to predict missing QoS values in the user-item matrix. Therefore, the reliability of user-contributed QoS values will highly influence the prediction accuracy of MF approaches. Unreliable users can cause negative impact on the prediction accuracy by providing unreliable QoS values.

In this paper, we were discussing about different approaches for selecting an optimal web service from available web services and various techniques for predicting the optimal web service. The paper is segmented into various sections in which section II is for Related work, section III consists of optimal web service selection methodology, section IV have conclusion of this paper.

II. Material and Methodology

Using HMM to measure and predict WS behaviour with respect to response time, our model consists of a two-step process. First step will require us to train the model to find optimal HMM parameters i.e., A, B & л, such that model best fits the training sequence. Training sequence in our model can be exploited by recording and labelling response time of a web service at regular intervals of time. Baum-Welch algorithm a case of expectation-maximization (EM) can be used to train the model. It iteratively improves the basic model which provides convergence to local optima, whereas second step, first requires us to compute current state of the system. Then based on current state, future behaviour of the system is predicted. This can be computed using VITERBI algorithm. Based on above two steps, for selecting an optimal WS and an optimal path for executing user requests our strategy can be further divided into following steps as depicted the system execution flow chart.

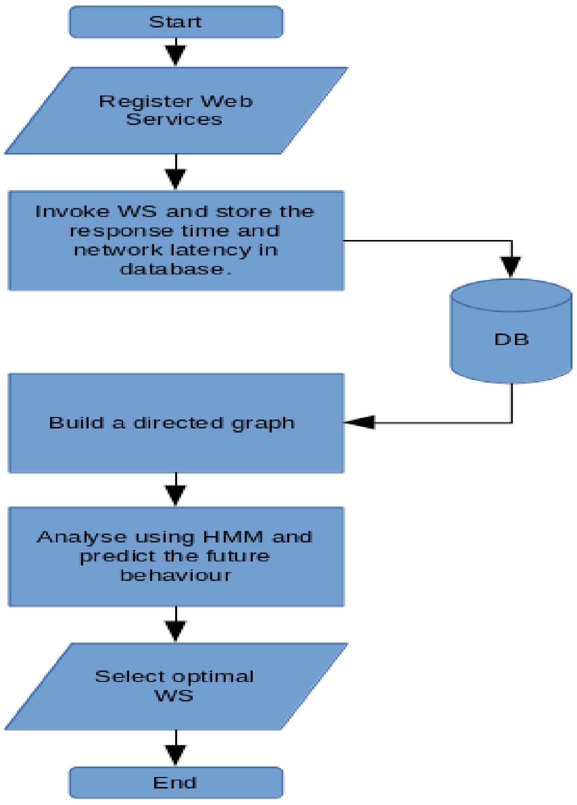
Start
Register Web Services
Invoke WS and store the response time and network latency in database.
DB
Build a directed graph
Analyse using HMM and predict the future behaviour
Select optimal WS
End

Figure 1 System execution flow chart

A. Exemplary Scenario

To analyse the behavioural pattern of hidden states, we have selected a weather forecast WS with best rank. More than 500 threads are used in parallel in a distributed environment. As in HMM, one does not know how many hidden-states to use, so we have supposed that target WS is running on a cluster of web server, containing 2 web servers. The analysis shows that at runtime behavioural patterns of the hidden states can be utilized for selecting optimal web services among the list of functionally equivalent web services. Hidden states with observation symbol A of different WSs can be connected at runtime to process user's request. In the next step, we will explain the process of building a directed graph among the hidden states of different web services used in composition.

TABLE 1

OBSERVATIONS UNITS

#	**Observations Symbols**	**Labels**
1	Normal Execution	A
2	Delay in Response	B
3	Error/Crash	C

TABLE 2

RESPONSE TIME PATTERN

Label	**Response Time (sec)**	**Remarks**
A	Response time <=5	Normal
B	Response time >= 5	Delay
C	No response for so long	Error

B. Directed Graph Among Hidden States

The key idea to select optimal WS for completing user requests is to exploit behavioural patterns of underlying hidden states. This can be done by recognizing hidden states' emission probabilities and combining them with integration patterns. As per our knowledge, previously different hidden states have never been considered with the integration patterns of WSC to estimate its behaviour in terms of RT. The basic structure of hidden states and corresponding observations in terms of RT is necessary to understand before building a directed graph.

C. Hidden States Analysis

Generally atomic web service is composed of different hidden states that are invisible to consumers as shown in Table 1. Each state is responsible to output certain results during time t. The probability of the response to certain requests depends on execution of these hidden states. When a consumer of a web service accesses a remote web service, sometimes he receives an unprecedented delay even under best operational conditions. We believe that this exceptional delay is because of unreliable hidden states responding to users' request at that time. Delay represents the state where system receives the results after long time, however, crash/error represents the state where WS crashes or user receives no response from WS. Predicting WS behaviour in our model is about evaluating hidden states' replies in terms of response time during the nth time interval. Generally, to build a highly available system, venders normally use different processing units to implement WS. In our model these processing units are termed as hidden states. For example, in case of web servers, network load balancing distributes incoming users' requests among multiple web servers to handle more traffic and faster response.

D. Response Time Analysis with HMM

Hidden states can entertain user's requests randomly and produce results any-time. As these hidden states, can also be accessed from other hidden states while service invocation, hence the model is of ergodic type. There may be/exist certain hidden states that may produce results in similar patterns of time intervals of having different configurations. For instance, hidden states 2 and 4 (defined in Table 1) can have similar response time patterns. Thus, for a given time interval we can define feature vectors including values defined in Table 1. Because of differences in implementation of hidden states (e.g., for hidden state 4 in Table 1, communication delay or latency for calling another service inside target web service will also be involved in overall response time) clear identification among feature values is required which in machine learning is referred to as Feature Normalization. These features may be categorized in terms of memory requirements, network requirements, software service requirements etc. This will help to define the initial values of the probability of success or failure of hidden states.

Table 1 shows general implementation of a WS used in exemplary scenario. User receives different observation symbols A, B, and C while he invokes a web service during a certain time interval t. Here observation symbols represent response time (as defined in Table 2) of output values. Depending on hidden states i.e., Web Server 1 and Web Server 2, the application users receive the results from a web service with either A, B, or C. In our approach when a user receives an unprecedented delay or error i.e., with response time pattern C, underlying hidden state is considered as an unreliable state S_{urel} as shown in Fig. 2. By initializing HMM parameters it can be ensured that the model transits to an unreliable state once the user receives exceptional delay or error. We have further explained this concept in the ‘‘Training the model’’ section below. For now, we can define some basic parameters of HMM in terms of component web service as:

1. States: Number of hidden states S within a component web service.
2. Observations: Distinct output observations i.e. V=(Normal (A), Delay (B), and Error/Crash (C)) such that output observation at time t is O_t where sequence of observation is $O = O_1, O_2, \ldots, O_n$ Here sequence of observation represents various response times generated by remote web service depending upon execution of relevant hidden state.
3. $A_{i,j}$ represents the transitional probability from hidden state Si following S_j.
4. B_{sj} represents the probability of hidden state generating output being produced from a hidden state S_j.
5. л represents the initial probability distribution values of underling hidden states.

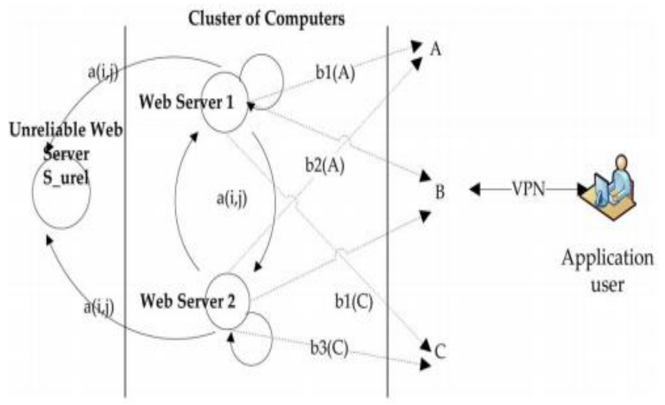

Figure 2 Hidden states and corresponding observation symbol

E. Basic Concept of Directed Graph

Fig. 1 indicates that A, B and C are the discrete emissions representing Normal, Delay and Crash respectively. These emissions are based on the execution of different hidden states as per the defined range (mentioned in Table 1). Nonetheless, emissions (A, B, and C) for different web services having similar functionalities consist of vector of probabilities during time interval t as shown in Fig. 3, e.g., For hidden state1, HS1P1 represents hidden state1 with probability-1 and HSnPn represents hidden state-n with probability-n. To build a directed graph among hidden states, following is the basic definition used in our model.

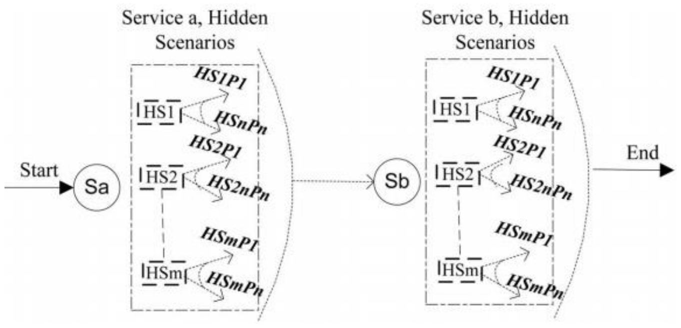

Figure 3 Example of various hidden states scenario

III. Mathematical Model

```
Input:   DIRECTED_GRAPH(V,E)
// V = Set of hidden states   group by web services and E
represents edges from each Hidden State group by obser-
vation symbol i.e. (Good /Normal /Bad)
Output: OPTIMAL_PATH

1.  for each $v_s$ in V
2.      for each $v_{hs}$ in $V_s$
3.          for each e (start, $v_{hs}$) in E
4.              CALCULATE_MIN
5.              CREATE_PATH $P_i$

6.          If e ($v_{hs}$, End) true
7               FIND_MIN_PATH $P_i$
8               ADD MIN_PATH $P_i$ to Graph G

9           else
10                  Initialize_Parameter
11  For each $P_i$ in Graph G
12      For each $P_i$ in $Group_i$
13          SUM_QoS
14          FIND_MIN_PATH
15          OP=Build_OPTIMAL_PATH
16  IF ( $P_i$ in Graph G ends) true;
17  Return OP;
```

In our model, we exploited pool of WSs having similar edge from hidden state of web service S_i to hidden states of web service S_k with the observation symbol U having probability Pi. Therefore, it can be observed that during anytime interval t the edge from hidden state HS_i can be categorized into various feasible solutions when invoked by service users. The process of selecting the best feasible solution produced by underlying hidden state among several feasible solutions is shown in Algorithm 1. Algorithm 1 indicates that application at client side exploits HMM parameters for each WS to build a directed graph among hidden states. Later, it returns the optimal path by calculating MIN of QoS values i.e. response time. Finally, system uses this optimal path to execute user's requests efficiently and reliably.

IV. Result analysis

When user connects with the system, then based on prediction results system finds the WS, probabilistic behaviour of whose hidden states are at their best level during that interval and then connect with it. Later algorithm1 is exploited to select an optimal path with minimum probabilistic value for executing user requests in the most efficient and reliable way.

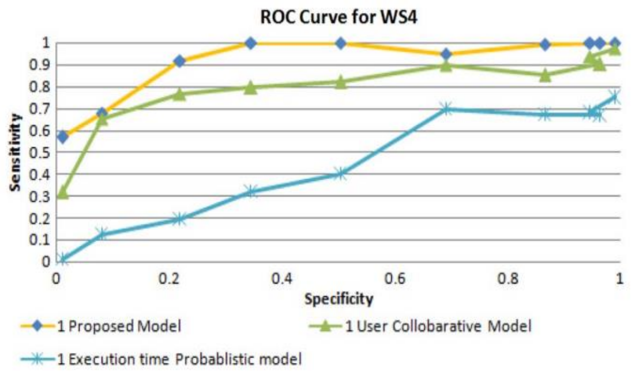

Figure 4 ROC curve for best WS among given pool of WS

V. Conclusions

In this paper, we at first propose a probabilistic model for predicting response time of web service and then selected an optimal web service at runtime from the list of functionally equivalent web services. To know the probabilistic insight of WSs we have used HMM. In our model, we have assumed that WS is deployed on a cluster of web servers and sometime the delay or crash during WS invocation is because the bad node in sever clustering responds to users' requests. With the help of HMM we have predicted the probabilistic behaviour of these web servers and then selected the WS based on their probabilistic value. Experiment shows that the proposed model is more general and detailed in comparison to existing models. This not only predicts the overall behaviour of composite web service but it further provides the solution to complete user requests in the most efficient and reliable way.

Acknowledgement

I like to thanks people for helping me for giving importance guidance about project work. I am very thankful to those who help me during my paper publishing as well as project dissertation stage. I am thankful to journal for giving me best opportunity to publish my paper. I am sincerely thankful to my head of department, my project guide and other staff members from department for supporting me.

References

i *A. Birukou et al., "Improving web service discovery with usage data," IEEE Softw., vol. 24, no. 6, pp. 47–54, Nov./Dec. 2007. http://ieeexplore.ieee.org/abstract/document/4375242/*

ii *Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," IEEE Trans. Services Comput., vol. 4, pp. 140–152, 2011. http://ieeexplore.ieee.org/document/5674010*

iii *Y. Jiang, J. Liu, M. Tang, and X. Liu, "An effective Web service recommendation method based on personalized collaborative filtering," in Proc. 9th Int. Conf. Web Services (ICWS 2011), 2011, pp. 211–218. http://ieeexplore.ieee.org/document/6009391/*

iv *Y. Xu, J. Yin, and W. Lo, "A unified framework of QoS-based web service recommendation with neighborhood-extended matrix factorization," in Proc. 6th IEEE Int. Conf. Service Oriented Computing and Applications (SOCA 2013), 2013, pp. 198–205. http://ieeexplore.ieee.org/document/6717306/*

v *Z. Li, Z. Bin, L. Ying, G. Yan, and Z. Zhi-Liang, "A web service QoS prediction approach based on collaborative filtering," in Proc. IEEE Asia-Pacific Services Comput. Conf., 2010, pp. 725–731. http://ieeexplore.ieee.org/document/5708681/*

vi *M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for QoS-based service recommendation," in Proc. Int. Conf. Web Services, 2012, pp. 202–209. http://ieeexplore.ieee.org/document/6257808/*

vii *W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "Collaborative web service QoS prediction with location-based regularization," in Proc. IEEE 19th Int. Conf. Web Services, 2012, pp. 464–471. http://ieeexplore.ieee.org/document/6257841/*

viii *J. Wu, L. Chen, Y. Feng, Z. Zheng, M. C. Zhou, and Z. Wu, "Predicting quality of service for selection by neighborhood-based collaborative filtering," IEEE Trans. Syst., Man, Cybern.: Syst., vol. 43, no. 2, pp. 428–439, Mar. 2013. http://ieeexplore.ieee.org/document/6301755/*

ix *G. Kang, J. Liu, M. Tang, X. Liu, B. Cao, and Y. Xu, "AWSR: Active web service recommendation based on usage history," in Proc. Int. Conf. Web Services, 2012, pp. 186–193. http://ieeexplore.ieee.org/document/6257806/*

x *L. Yao, Q. Z. Sheng, A. Segev, and J. Yu, "Recommending web services via combining collaborative filtering with content-based features," in Proc. Int. Conf. Web Services, 2013, pp. 42–49.*

xi *http://ieeexplore.ieee.org/document/6649560/*

xii *M. Tang, Z. Zheng, G. Kang, J. Liu, Y. Yang, T. Zhang "Collaborative Web Service Quality Prediction via Exploiting Matrix Factorization and Network Map" IEEE, 2016. http://ieeexplore.ieee.org/document/7378981/*

xiii *W. Ahmed, Y. Wu, and W. Zheng "Response Time Based Optimal Web Service Selection" IEEE 2015. http://ieeexplore.ieee.org/document/6684156/*