QoS-Based Web Services Selection Using a Hidden Markov Model

Daniel G. Canton-Puerto*, Francisco Moo-Mena, Víctor Uc-Cetina
Facultad de Matemáticas, Universidad Autónoma de Yucatán, Anillo Periférico Norte, Tablaje Cat. 13615, Colonia Chuburná Hidalgo Inn, Mérida Yucatán, México.

* Corresponding author. Tel.: 52(999)1295037; email: daniel.gcanton@correo.uady.mx
Manuscript submitted August 29, 2015; accepted December 27, 2015.
doi: 10.17706/jcp.12.1.48-56

Abstract: Web services are a technology that has been growing rapidly since its inception earlier this century. They are so popular nowadays that they are being used for numerous purposes, spanning business, healthcare, education and government applications. They are used for accessing diverse types of data in domains such as the financial, climate forecasting and sports, among others. As a result of this increased popularity, it is very common to find a large number of applications providing exactly the same service. Having a large number of applications to choose from, one that integrates a selection mechanism using quality criteria for selecting the best choice, has an advantage over other applications that do not provide this feature. However, the selection of the best Web service, among various services with the same functionality, is not an easy task. Therefore, the design of a worthwhile technique to determine the best Web service by assessing its quality of service is important. Hidden Markov models are probabilistic methods that allow us to build behavior models of the Web services. Such models can be used to predict their behavior in the near future. In this paper, we propose a method for selecting Web services based on the response time QoS parameter and hidden Markov models.

Key words: Hidden Markov models, QoS, selection, web services.

1. Introduction

Web services are a technology that has been growing rapidly since its inception in the early twenty-first century. They are so popular nowadays that they are being used for numerous purposes, spanning business, healthcare, education and government applications. They are used for accessing diverse types of data in domains such as the financial, climate forecasting and sports, among others. Its popularity is due to their main characteristics: a variety of technologies are available to develop this kind of applications, the greater abstraction of the existing software allows less-technical users to use these applications and finally, the ability to add applications without modifying the original is very practical. As a result of this increased popularity, it is now possible to find a large number of applications that provide the same services, such as Yahoo.com, NOAA and Weather.com. For instance, all three of them provide a climate information service.

When we need to choose a Web service among several others with the same functionality, we need to apply certain pre-defined criteria, in order to make a reasonable decision. “Quality is the standard of something as measured against other things of a similar kind” [1], so according to this definition, it is a criterion to establish a difference between two or more Web services, which have the same functionality. In

Web services, the parameters used for measuring quality of service (QoS) are defined by Kang Chan *et al*. [2] in the World Wide Web Consortium (W3C), an international community that develops open standards to ensure the long-term growth of the Web. Among all these parameters, Raj *et al*. [3] highlight the response time, the number of successful responses in a period of time, availability, reliability and cost, as valid features to establish a selection process.

There are Web-based services designed to interoperate between applications. Being able to choose the best Web service for such interoperation is critical for the overall performance of the main Web service. An example of the importance of this approach can be seen in a recovery technique known as substitution, which is used in self-healing systems. This technique involves replacing an inoperative service with another one that is in normal state of operation. Obviously, the selection of a Web service, based on its quality, is essential to guarantee a successful substitution. Another clear example of the importance of Web service selection methods is the dynamical composition of Web services applications.

Selecting the best Web service between various services with similar functionality is not a simple task, so an application using some quality criteria for this selection would be in advantage over other applications that do not provide this feature.

In this paper we propose the use of hidden Markov models as a method that allows the selection of a Web service that is distinguished from others with similar functionality by having a higher quality.

The rest of the paper is organized as follows. Section 2 mentions some works related with the QoS selection of Web services; Section 3 presents the model we propose for the selection process. Section 4 discusses the training and tests made with the proposed the model. Finally in Section 5 we discuss the results obtained and future work.

2. Related Work

Sometimes we have to make decisions about which bank will provide us a service, or want to know which cable company is the best. In order to make such decisions, it is necessary to carry out a process where you have to choose an element from a pool, after comparing them carefully. This process is called selection, i.e. “the action or fact of carefully choosing someone or something as being the best or most suitable” [1].

In terms of Web services selection we must consider that a service can play whether a unique role (client or provider), or play both roles. When a Web service plays only a single role, it is a client for different service providers, or it is a provider answering client requests. When a Web service plays both roles, it is because it also needs information from other providers, in order to answer a client's request. The aim of the selection of Web services is to get the service that best suits the needs of a client. We focus on a non-functional properties selection approach.

This selection approach differentiates between Web services that have the same function or similar. To achieve this objective, non-functional properties are defined for the Web service, such as quality of service parameters. According to the W3C, the QoS parameters are [2]: performance, reliability, scalability, robustness, exception handling, accuracy, integrity, accessibility, availability, interoperability and security. D'Mello *et al.* [4] defined a taxonomy to classify selection techniques based on Web services QoS parameters.

Fig. 1 shows two major classifications: the choice to perform a single task and the selection of a service that is a part of a composition. In this paper we focus only in the works of the category “use of single QoS property”. In this category we can mention the work done by Kerrigan *et al.* [5] Moo-Mena *et al.* [6] and Ahmed *et al*. [7]. Kerrigan *et al*. [5] propose a selection process execution environment for Web services (WSMX [8]), which make a filtered and sorted list of providers based on the requirement of a single QoS parameter and then the user selects a provider from that list. Ahmed *et al*. [7] use a hidden Markov model

to select the best Web services to construct Web services compositions. Finally, Moo-Mena *et al*. [6] also uses a hidden Markov model to determine the best Web service provider based on the results of the provider transactions, where the better provider has a high rate of successful transactions.

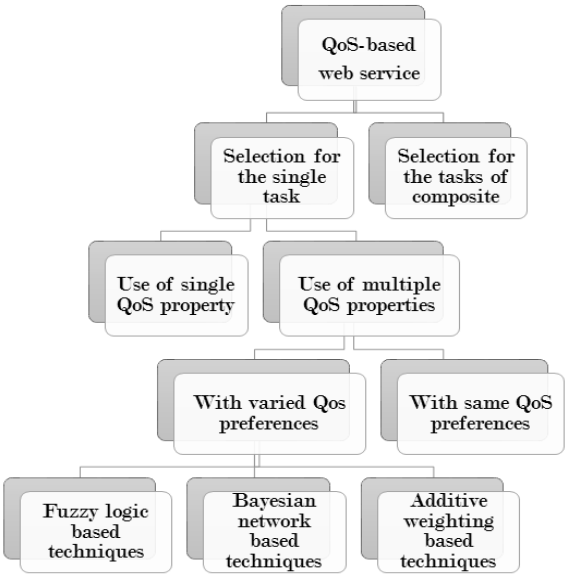


Fig. 1. Taxonomy of QoS aware web service selection techniques.

3. Methodology

A hidden Markov model is a stochastic process which has a set of hidden states, a set of observable states, a matrix representation of the transition probabilities between the hidden states and a matrix representation of the emission probabilities from the hidden states to the observable states. In a formal way a HMM is a stochastic process which has the following elements [9].

1) N: Number of states in the model

$$S = \{S_1, S_2, \dots, S_N\}$$

2) M: Number of distinct observations symbols in the alphabet

$$V = \{v_1, v_2, \dots, v_M\}$$

3) State transition probabilities

$$A = [a_{ij}] \text{ where } a_{ij} \equiv P(q_{t+1} = S_j \mid q_t = S_i)$$

4) Observation probabilities

$$B = [b_j(m)] \text{ where } b_j(m) \equiv P(O_t = v_m \mid q_t = S_j)$$

5) Initial state probabilities

$$\Pi = [\pi_i] \text{ where } \pi_i = P(q_1 = S_i)$$

In this paper we used a hidden Markov model to support the Web service selection process. This model use the QoS parameter response time (RT), which measures the time a provider takes to answer a client request. The model determines a good provider if it has optimum values of RT. In this new model, called HMMsel, three web services providers of climate data are considered, which form the set of hidden states. The set of observable states in this model consists of the time taken by providers to answer a client request. This last set has three elements, which represent the values of the QoS parameter, response time, divided into three levels of quality, obtained from the ITU-T G.1010 recommendation issued by the International

Telecommunication Union (ITU) [10]: optimum ("O"), normal ("N") and inadequate ("I"). Formally, the HMMsel is defined as follows:

1) $S = \{"Y", "W", "N"\}$
2) $V = \{"O", "N", "I"\}$
3) $A = [a_{ij}]_{3\times3}$
4) $B = [b_j(m)]_{3\times3}$
5) $\Pi = [\pi_i]_{1\times3}$

In the HMMsel model, the set S has the elements of Web services providers of climate data (Yahoo Weather, Weather.com and NOAA) [11]-[13]. The elements of the set of observable states V are the three levels of quality in the response time mentioned above.

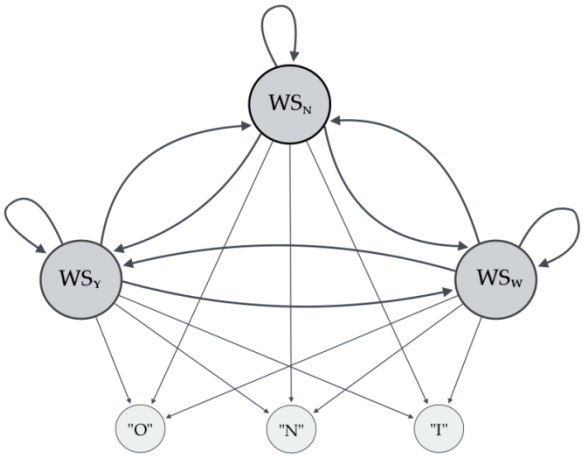


Fig. 2. HMMsel. Model to determine the behavior of climate data providers.

Fig. 2 shows how the HMMsel model is represented, in which the hidden states WS_N, WS_Y, WS_W represent the Web services providers of climate data, and the arrows connecting these states represent the transition probabilities, this is the probability of moving to another state given the current state. Moreover, observable states are represented by the levels of quality: optimum ("O"), Normal ("N") and inadequate ("I"). The connections between the hidden states and observable states represent emission probabilities, namely the probability of observing a given symbol from a given hidden state.

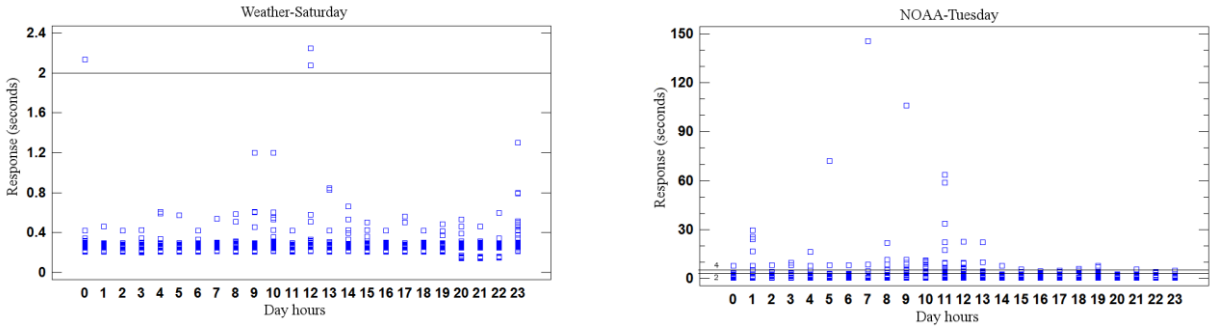
In HMMsel, the probabilities of the transition matrix A and emission matrix B are obtained with the Baum-Welch algorithm [9], using as input parameters, an observation sequence and a transition matrix $A^t$and an emission matrix B^t, both with initial values. The observation sequence was obtained after performing systematic requests to the web services providers: Yahoo (WS_Y) Weather (WS_W) and NOAA (WS_N), and the values of A^t and B^t matrices can be defined randomly. After obtaining the matrices A and B, another observation sequence is generated using the average of emission probabilities for each of the values of QoS parameter RT. Subsequently, the Viterbi algorithm [9] is used to decode the latter observation sequence and thus generate a new sequence called selection sequence (CS), containing web service providers that provide service to the following web services client requests.

4. Experimental Work

In this section, we describe the processing of the data when making requests to three real Web services and the process followed to train the Baum-Welch algorithm. We also explain the decoding of an observation sequence using the Viterbi algorithm, to establish an automatic selection mechanism. Finally, we present all the experimental results with both algorithms.

4.1. Data Collection and Analysis

This section describes how the data required by the Baum-Welch algorithm for the model parameters HMMsel is obtained. This data is obtained by making requests to three different weather Web services, using a script developed in Python, under the development environment for Eclipse PyDev. This script was developed using the API Python-weather-api [14], which allows us to obtain information related to the climate of three major web sites: Yahoo.com, Weather.com and NOAA.gov. These climate requests were made from the computer network of the Faculty of Mathematics at the University of Yucatan through the Internet. This implies that the recorded response times may be affected by communication errors in the network and it represents not only the performance of the Web service provider (Yahoo.com, Weather.com and NOAA.gov).



(a) Best response times (b) Worst response times

Fig. 3. Day and providers that present the best and worst response times.

Table 1. End-User Multimedia QoS Thresholds

QoS threshold		
Preferred	<	2 seconds
Acceptable	<	4 seconds
Inadequate	≥	4 seconds

Due to the nature of these Web services, we could not determine if their behavior would be uniform all the time, or if they could show a performance reduction during time intervals, and therefore take longer to answer the requests from other Web services; therefore, we thought in the possibility of training a model for each day of the week, i.e. a different model to make the provider selection depending on the weekday (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday or Sunday). We made requests to the three providers, recording the response time of each request, in order to determine the number of models needed to make the selection. These requests were made as follows: information was collected during 8 weeks, and every day requests were made at intervals of 1 hour. These requests were made for each involved provider. Each request was to ask for the ambient temperature of a city in the United States, which was randomly chosen from a list of ten different cities. We made ten requests per hour to each one of the providers, making a total of 240 registers by provider. Finally, we obtained 1 920 response time records for each provider. Each response time was recorded as follows: $p|c|t|tr$, where: p represents the Web service provider, c represents the consulted city, t is the measured temperature and tr is the response time (measured in seconds) of the request.

After obtaining the data, the following is to train the Baum-Welch algorithm, in order to obtain a model for each day of the week and with those models to decide whether a unique model or several models is used, depending on the behavior of providers. Discrete data is required to train the Baum-Welch algorithm,

however the response time is a continuous variable, hence to discretize these data we used the ITU-T G.1010 recommendation issued by the International Telecommunication Union (ITU) [10]. This document is intended to ensure a high quality in data communications by defining some thresholds that indicate how long a service provider is allowed to delay in responding to a request. It indicates three levels of quality, whether it is preferable (optimum), acceptable (normal) or inadequate. Table 1 shows the quality thresholds established by the ITU.

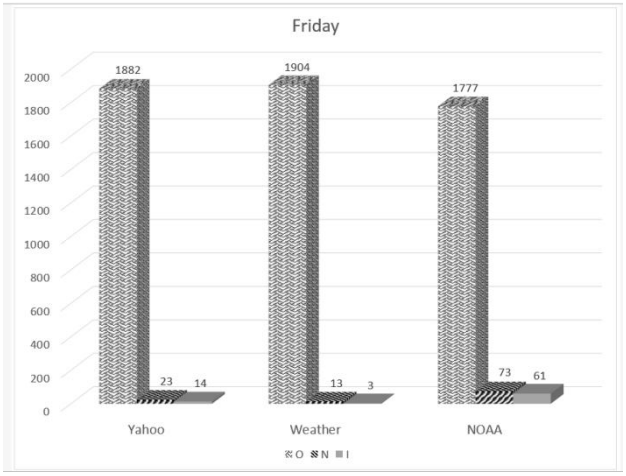


Fig. 4. Discrete data group by quality levels.

Before training the Baum-Welch algorithm, we made an analysis of the data, in order to have a guess about the behavior of the service providers. Fig. 3(a) shows the best response times and Fig. 3(b) shows the worst ones. We can note than even in the worst response times, most of the responses were under the 2 seconds threshold (optimum accordingly to Table 1). We also observed data after being discretized (see Fig. 4), and we confirm our perception that the response times are grouped into the optimum level of quality.

4.2. Training the Baum-Welch Algorithm

We trained seven different models, one per weekday, using as input parameters seven different sequences of observations, and the same initial transition and emission matrices. The initial probabilities of transition were established as the same for each transition, i.e. the probability of going from a state i to a state j is 0.33. The initial emission probabilities were established as follows: 0.6 for observing an optimum response time, 0.3 for normal and 0.1 for inadequate. After running the Baum-Welch algorithm we obtained as a result seven models with no substantial differences between them, therefore we used a unique model for the provider selection.

Table 2. Final Selection Model for Climate Web Services

$$A^F = \begin{bmatrix} 0.332725883 & 0.332725883 & 0.332732039 \\ 0.332725883 & 0.332725883 & 0.332732039 \\ 0.332720755 & 0.332720755 & 0.33272691 \end{bmatrix}$$ Transition matrix

$$B^F = \begin{bmatrix} 0.971049967 & 0.017788028 & 0.011162005 \\ 0.971049967 & 0.017788028 & 0.011162005 \\ 0.971054401 & 0.017785689 & 0.01115991 \end{bmatrix}$$ Emission matrix

The final model (see Table 2) used for the selection of a Web service provider, was obtained by combining information from the other seven models obtained for each day of the week using (1) and (2) to obtain resulting transition matrices $A^F = \left[a_{ij}^F\right]_{3\times3}$ and $B^F = \left[b_{ij}^F\right]_{3\times3}$ respectively. To construct the equations (1)

and (2) two three-dimensional arrays were created, called $D^A = [d^A_{kij}]_{7\times3\times3}$ and $D^B = [d^B_{kij}]_{7\times3\times3}$ containing transition and emission matrices obtained for every weekday.

$$a^F_{ij} = \frac{\sum_{k=1}^{7} d^A_{kij}}{7} \quad (1)$$

$$b^F_{ij} = \frac{\sum_{k=1}^{7} d^B_{kij}}{7} \quad (2)$$

4.3. Selection Using the Viterbi Algorithm

In this section, the matrices A^F and B^F in Table 2 are used to decode a sequence constructed using the vector $V = [0.971051445 \quad 0.017787248 \quad 0.011161307]$, where its elements represent the probability of observing a determined response time. These elements were calculated using the average of each column of the B^F matrix.

The selection was tested using a sequence of 100 elements. We executed the Viterbi algorithm using as input parameters the matrices A^F and B^F, and the sequence described above. We obtained as a result a sequence of hidden states, corresponding to the climate Web service providers. This sequence had the same elements, i.e. all elements were the same Web service provider. The previous result is because of the results obtained for the emission matrix B^F, in which it is observed that the three climate Web service providers are equally good, reporting the same probability for every level of quality in the response time. Because of this situation, the Viterbi algorithm considers that using only a unique provider is better, regardless of the values that have the transition matrix A^F.

As a consequence of this situation, another alternative to the selection using the Viterbi algorithm was used. This alternative only considers the transition probabilities. This strategy consists in generating a sequence of providers, considering the same probability of choosing each provider. Table 3 shows the result according to the quality levels of 5000 requests made to different providers are consistent with the results obtained after training Baum-Welch algorithm.

Table 3. Global Results of the Selection Test

Quality	Percentage
Optimum	96%
Normal	3%
Inadequate	1%

Table 4. Results Grouped by Web Service of the Selection Test

Web service	Quality	Percentage
NOAA	"O"	93%
NOAA	"N"	5%
NOAA	"I"	2%
Yahoo	"O"	97%
Yahoo	"N"	2%
Yahoo	"I"	1%
Weather	"O"	97%
Weather	"N"	3%
Weather	"I"	0%

In Table 4 the results of quality levels are presented, grouped with respect to climate Web service provider. In this table we can see that the results are consistent with the data used to train the Baum-Welch algorithm in the same way and they are also consistent with the results of the training.

With regard to these results it can be noted that it is better to use the second selection strategy when the providers are equally good. However, if the behavior changes, the Viterbi algorithm is used to make the selection.

5. Conclusion

The Baum-Welch algorithm is very useful because it can model a system behavior by collecting the results of one variable. The choice of the initial matrices of transition and emission is a critical fact, because this can affect the convergence of the algorithm. Given that the choice of the initial probabilities of transition and emission as inputs to the Baum-Welch algorithm does not follow any specific procedure, any information about how this choice should be made would be very useful. In case we do not have any information that suggests how to choose this initial data, the next step is to choose them randomly. On the other hand, the proposed selection method using Viterbi algorithm can be used in cases where the service providers have heterogeneous behavior. In case the service provider behavior is homogeneous, we recommend another selection method. It would be desirable that such selection techniques use other QoS parameters (cost, performance, capacity, etc.) and/or add user opinion evaluation. By using the alternative selection technique, we observe that results obtained were consistent with the behavior observed during the data analysis phase. Therefore, we can conclude that the results of training Baum-Welch algorithm were adequate. Finally, as future work in this research area we can highlight: the improvement of the selection model through the addition of more QoS parameters, as recommended by Raj *et al*. [3], such as availability, cost, throughput and reliability. A way to add these QoS parameters in the model would consist in carrying out a ranking function that combines the QoS information.

References

[1] Oxford dictionaries. Retrieved August 2015, from http://www.oxforddictionaries.com/

[2] Park, S.-W., *et al*. (2003). QoS for web Services: requirements and possible approaches.

[3] Raj, R., & Sasipraba, T. (2010). Web service selection based on qos constraints. *Trendz in Information Sciences Computing*, 156-162.

[4] D'Mello, D., & Ananthanarayana, V. S. (2010). A review of quality of service (qos) driven dynamic web service selection techniques. *Proceedings of International Conference on Industrial and Information Systems* (pp. 201–206).

[5] Kerrigan, M. (2006). Web service selection mechanisms in the web service execution environment (wsmx). *Proceedings of the 2006 ACM Symposium on Applied Computing* (pp 1664–1668).

[6] Canton-Puerto, D. G., Moo-Mena, F., & Uc-Cetina, V. (2012). Dynamic web services selection using a hidden Markov model. *Proceedings of 9th International Conference on Electrical Engineering, Computing Science and Automatic Control* (pp. 1–6).

[7] Ahmed, W., Wu, Y., & Zheng, W. (2015). Response time based optimal web service selection. *IEEE Transactions on Parallel and Distributed* Systems, *26(2-2),* 551-561.

[8] Web Service Execution Environment. Retrieved April 2014, from http://www.wsmx.org

[9] Jurafsky, D., & Martin, J. (2009). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. *Prentice Hall Series in Artificial Intelligence.* Pearson Prentice Hall.

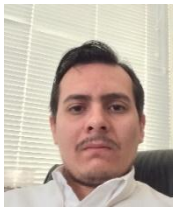
[10] International Telecommunication Union. *G.1010: End-user Multimedia Qos Categories*.

[11] Yahoo! Weather. Retrieved September 2014, from https://weather.yahoo.com/

[12] The Weather Channel. Retrieved September 2014, from https://www.weather.com/

[13] National Oceanic and Atmospheric Administration. Retrieved September 2014, from

http://www.noaa.gov/wx.html
[14] Python Weather-API. Retrieved September 2014, from https://code.google.com/p/python-weather-api/



Daniel G. Canton-Puerto is a Professor in computer sciences at Universidad Autónoma de Yucatán, in Mérida, México. He received a master degree in computer sciences and his BS in computer sciences from the Universidad Autónoma de Yucatán in 2014 and 2003 respectively. His research interest include web services, bayesian reasoning and machine learning.

Francisco Moo-Mena is a Professor in computer sciences at Universidad Autónoma de Yucatán, in Mérida, Mexico. From the Institute National Polytéchnique de Toulouse, in France, he received a master degree in computer science and a PhD, in 2003 and 2007, respectively. He also received another master degree in distributed systems from the Instituto Tecnológico y de Estudios Superiores de Monterrey, Mexico, in 1997. He received a BS in computer systems engineering from the Instituto Tecnológico de Mérida, Mexico, in 1995. His research interests include self-healing systems, web service architectures, and semantic web services.

Víctor Uc-Cetina received the PhD degree (Dr. rer. nat.) in computer science from Humboldt-Universität zu Berlin, Germany, in 2009; his M. Sc in intelligent systems from itesm in 2002 and his BS in computer systems engineering from ITM in 1999. He is currently working as an assistant professor in the Facultad de Matemáticas at Universidad Autónoma de Yucatán, México. His research interests include artificial intelligence, machine learning, Bayesian reasoning and vision learning.