Hybrid Model-Based Diagnosis of Web Service Compositions

Zhichun Jia, Rong Chen
School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China
zhichun.jia@dlmu.edu.cn, rchen@dl.cn

Abstract

Fault diagnosis of web services composition at run time is appealing in creating a consolidated distributed application. For this purpose, we propose a hybrid model-based diagnosis method which exploits service process description or historical execution information to enhance service composition model, and localize faults by comparing the exceptional execution and the correct execution with the maximum likelihood. Experiments are conducted to evaluate the effectiveness of our method in web service composition fault diagnosis.

Introduction

With the popularization of web service applications, web service raises great concerns especially how to diagnose faults to ensure reliable running of services. So far two categories of fault diagnosis techniques have been proposed, of which the Model-based Diagnosis methods (MBD) require the knowledge about structure, function and behavior of services to infer the cause of a failure whereas the recent historical data-based methods works on the statistical model of web services built upon historical execution information of web services (Han et al. 2010). The recent research into MBD models web service as synchronized automata and analyzes the structure and variable dependency of service to localize fault (Yan et al. 2009). However, MBD presumes the complete knowledge about the behaviors of web services while statistical methods are easy to set up but are vulnerable to semantic faults.

In this paper, we firstly present a diagnosis model which not only exploits process description but also exploits historical execution information to model. And then we localize service fault by finding the correct execution sequence with the maximum likelihood and comparing it with the exception execution sequence. Finally, we conduct the simulation experiments to evaluate our method. Experimental results show that our method is more effective than Yan's method in fault diagnosis for web service.

Copyright © 2013, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Diagnosis Model

Our diagnosis model incorporates three sub-models: behavior transition model, message transition model and behavior-message emission model. Behavior transition model describes the transitions among the service behaviors in a probabilistic way. Message transition model represents the transitions among the input and output messages in the same way. Behavior-message emission model depicts the associated relationships from behavior to message.

- Behavior transition model: $BTM = \{p(b_i, b_k)\}_{1 \le i,k \le n}$, where:

$$p(b_i, b_k) = \frac{|(b_i, b_k)|}{|(b_i, \bullet)|} \tag{1}$$

- Message transition model: $MTM = \{p(m_i, m_k)\}_{1 \le i,k \le w}$, where:

$$p(m_i, m_k) = \frac{|(m_i, m_k)|}{|(m_i, \bullet)|} \tag{2}$$

- Behavior-message model: $BMM = \{p(b_i, m_k)\}_{\substack{1 \le i \le n \\ 1 \le k \le w}}$, where:

$$p(b_i, m_k) = \frac{|(b_i, m_k)|}{|(b_i, \bullet)|} \tag{3}$$

For a web service to be diagnosed, n is the number of behaviors, w is the number of message types. $p(e_1, e_2)$ is the probability that (e_1, e_2) occurs. $|(e_1, \bullet)|$ is the number of the $(e_1, \bullet)$, where e_1 is an element of behavior or message type, and $\bullet$ is any of elements whose type are corresponding with e_2 in $p(e_1, e_2)$.

In our diagnosis model, we are able to not only convert Business Process Execution Language (BPEL) process description into three sub-models, but also convert historical service execution data into our diagnosis model. BPEL describes the normal behaviors of service and there is the same probability of occurrence for each behavior in the same position. For the historical service execution data, we only consider the successful executions into our diagnosis model. Hence, our model describes the probability of successful execution of a behavior. If the probability of a transition or emission is very low and it occurs in an exception execution, the behaviors in this transition or emission may be faulty behaviors. Moreover, if we obtain both BPEL diagnosis model and historical data diagnosis model, we use the weights wt_1 and wt_2 to combine these two diagnosis model, for example $BTM = wt_1 BTM_1 + wt_2 BTM_2$, and $wt_1 + wt_2 = 1$.

Diagnosis Method

Our diagnosis method is to find a correct activity transition which best matches the exception execution, and localize their differences as faults. We firstly assume that: either BPEL process description or historical execution data is available; we are able to convert it into our diagnosis model; the exception execution information is available, such as executed behaviors and messages sequence. And then we check every message transition in the exception execution. If the probability of a message transition is less than ε, we think it is a faulty message transition. For all faulty message transition, we find a correct message transition to substitute it, so that we obtain a correct message sequence. Next, we find a correct behavior sequence with the maximum likelihood by the correct message sequence. Finally, comparing two correct sequences with the exception execution, the discrepancies between them are faults. The detailed method is described at Algorithm 1.

Algorithm 1: Diagnosis Algorithm

Input: (*BTM*,*MTM*,*BMM*), exception execution *eet*

Output: diagnostic results *dr*

```
cm(1)=eet(1).mes;
for i=2:eet.length
  cm(i)=eet(i).mes;
  if MTM(cm(i-1),cm(i))<= ε
    cm(i)=max(MTM(cm(i-1),•));
    if MTM(cm(i-1),cm(i))=0
      cm(i-1)=max(BMM(eet(i-1).beh,•));
      cm(i)=max(BMM(eet(i).beh,•));
    end if
  end if
end for
cb=Viterbi(BTM,BMM,cm); n=1;
for i=1:eet.length
  if eet(i).beh!=cb(i)||eet(i).mes!=cm(i), dr(n)=eet(i); n++;
end for
return dr;
```

In Algorithm 1, we apply *Viterbi* algorithm to compute the correct behavior sequence with the maximum likelihood. ε is a minimum value for handling noise in diagnosis model. When there isn't noise in diagnosis model, ε is zero. max($\bullet$) returns a element with the maximum probability value in $\bullet$.

Experiments

To evaluate the effectiveness of our method, we set up a simulation environment. Given the numbers of service nodes, structure nodes and executions, it is able to generate web service and its execution information. Moreover, we randomly allotted a set of QoS attributes from QWS database (Al-Masri and Mahmoud 2008) to the generated service nodes. And then the service generates faults according to the QoS attributes.

In our experiments, we generate 5 experimental groups, and each group has 100 services with the same nodes. We compare the diagnostic accuracy of our method with *Yan*'s method in (Yan, Dague et al. 2009). From Figure 1, we can see that our method *DM* is more effective than *Yan*'s method without reference to service scale. Comparing with *Yan*'s method, our method considers not only the relationship between two contiguous behaviors and their messages but also the likelihood of whole execution trace. Hence, our method is able to diagnose the control fault and is more accuracy than *Yan*'s method in fault diagnosis of service.

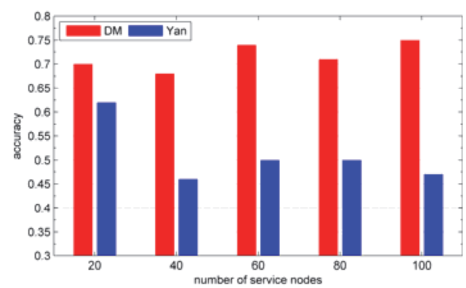

Figure 1 Comparison of Accuracy

References

Al-Masri, E., and Mahmoud, Q. H. 2008. Investigating Web Services on the World Wide Web. In *Proceedings of the 17th international conference on World Wide Web*, 795-804. Beijing, China: ACM.

Han, X., Shi Z., Niu W. et al. 2010. Similarity-Based Bayesian Learning from Semi-structured Log Files for Fault Diagnosis of Web Services. In *Proceedings of 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 589-596.

Yan, Y., Dague, P., Pencole Y. et al. 2009. A Model-based Approach for Diagnosing Faults in Web Service Processes. *The International Journal of Web Services Research (JWSR)* 6(1): 87-110.