Applied Mathematics & Information Sciences
An International Journal

Predictive Web Service Monitoring using Probabilistic Model Checking

Honghao Gao [1,2,3], *Huaikou Miao* [1,3], *Hongwei Zeng* [1,3]

[1] School of Computer Engineering and Science, Shanghai University, Shanghai 200072, P.R.China
[2] Computer Center, Shanghai University, Shanghai 200072, P.R.China
[3] Shanghai Key Laboratory of Computer Software Evaluating & Testing, Shanghai 201112, P.R.China

Received: 13 Jun. 2012, Revised: 27 Nov. 2012, Accepted: 12 Dec. 2012
Published online: 1 Feb. 2013

Abstract: Web service is vulnerable to stochastic failures due to the changeful Internet environment, which will seriously impact the reliability of business-critical applications. Web service must adapt to these changes. This paper studies the monitoring issue of Web service and proposes a predictive service monitoring approach to support service-oriented software dynamic evolutions. It aims to ensure that Web service is high-quality at runtime. Considering functional and non-functional requirements, we employ probabilistic model checking technique to quantitatively verify the interactive behaviour of Web service. However, limited by time and memory, frequently performing probabilistic model checking is not a high-efficiency solution to monitor Web service. Thus, forecasting the reliability becomes an important way to enhance the performance of Web service monitoring, by which we can put forward dynamic reconfigurations before the exception occurs. For this purpose, according to the historical probability values of Web service reliability recorded in monitoring phase, we adopt the mathematical statistics, unary linear regression analysis method, to predict the probability value of reliability based on invocation duration time. Finally, a case study is discussed and experiment results demonstrate the applicability and effectiveness of our approach.

Keywords: Web service monitoring, probabilistic model checking, reliability prediction, linear regression analysis.

1. Introduction

As the de facto standard, Web service has been an important solution to achieve resource sharing and application integration in the Internet era, which can develop the most promising e-commerce software featured with the on-demand changing computing paradigm, through service reuse and dynamic synthesis. Under the interoperability of Web service, complex business interactions fulfilled by the process of dynamic discovery, integration, coordination and execution of distributed service entities (atomic or composite service), via published and discoverable interfaces, can improve the productivity and work quality of enterprise. However, due to the heterogeneous, open and collaborative nature of Internet, any application failure may immediately result in service interrupt, which will seriously impact the correctness and reliability of core business process of service-oriented software. Although several languages and technologies have been proposed, e.g., BPEL4WS, WS-CDL, WSCL, and OWL-S, they only concern on Orchestration and Choreography composition behaviors for manufacturing a correct Web service at design phase. The functional requirement of business logics is their prior job in absent of providing related strategies or mechanisms to dynamically monitor both functional and nonfunctional attributes of Web service at runtime phase. But the nonfunctional property impacted by random behaviors commonly refers to the quality aspect of Web service, such as security, reliability and availability. Consequently, how to effectively monitor service behaviors remains to be a big challenge in the area of Web service research.

For example, when a customer asks an online agent for arranging a trip travel, services including the flight booking service, hotel reservation service, car rent service, express delivery service and credit-card payment service are required to be properly composited. Faults may be occurs during service execution. Since the loosely coupled nature of service integration and the inherently

* Corresponding author e-mail: gaohonghao@shu.edu.cn

open nature of Internet, the trip travel service will be vulnerable to stochastic failures, i.e, the instability network, the permanent invocation failure, or the processing timeout. To avoid these unexpected failures during service software running, it is practicable to monitor and predict the reliability of Web service increasing the flexibility of business interactions. Definitely, the better application service performs, the better business performance will be. One issue is to formally verify the quantitative property, such as "with probability 0.7 or less, a ticket message from express delivery service will be delivered". The other issue is to forecast the performance, such as "when will the credit card payment service be unavailable?".

To address these challenges, formal verification technique has been widely studied to verify Web service in literatures [5, 6, 8, 12, 14, 18, 22, 23], which can be categorized into qualitative verification and quantitative verification. In the context of qualitative verification, using model checking technique to verify the correctness of Web service can uncover bugs at the functionality level, such as model checker NuSMV and SPIN. Nakajima [18] applied a model checking technique to verify service flows based on SPIN. Diaz [5] proposed an approach to extract correct WS-BPEL skeleton documents from WS-CDL using timed automata as an intermediary model to check the correctness of generated Web service. Kova [14] used StateCharts to model composite services and verified the synchronization model of conversations among them using NuSMV. Foster [6] verified the mediated composite service specified by BPEL against the design model specified by Message Sequence Chart and Finite State Process notations. The tool [8] checked whether the composite service satisfies LTL properties. But these above works are hard to verify non-functional requirements. It demands that the quality of services (QoS) should meet consumers' expectations. Our research mainly focuses on the probability aspect of QoS. The stochastic behaviors, e.g., random component failures or message loss in communication between networked services, will seriously impact the reliability of Web service. For this propose, in the context of quantitative verification, applying probabilistic model checking technique can establish quantitative properties of a system model at the non-functionality level, such as model checker PRISM and MRMC. Hwang et al. [12] proposed a probability-based QoS model for atomic and composite Web service, namely viewed a QoS measure as a discrete random variable with a probability mass function (PMF), and described a framework to measure QoS of Web service workflows. Yang et al. [22] proposed a probability-based QoS model language Chorr to describe dynamic routing protocols. To perform stochastic analysis of QoS-related properties effectively, they translated the specification **Chorr** into the language PRISM. However, the historical data derived from verification results are not concerned.

Due to the process of quantitative verification not only involves functional verification but also refers to performance analysis, the objective of our paper is to employ probabilistic model checking technique to monitor Web service which exhibits random behavior. From the qualitative result featured with "yes" or "no" and its probability value, we can implement correctness verification and reliability estimation of service-oriented software. However, frequently executing probabilistic model checking may easily become a bottleneck because the time and memory needed to handle the whole state space of the possible behaviors of Web service may be huge and infinite. While historical data can be used as the basis of estimating future outcomes [23]. To this end, Web service monitoring requires the forecast mechanism supporting service reliability dynamic prediction. Thus, we are motivated to predict the reliability of Web service for revealing the trend of probability via statistical analysis.

In this paper, we propose a two-phase service monitoring method, that is,

1. Web Service Monitoring Phase. Due to the uncertainty of Internet environment, each Web service presents different stochastic behaviors inherently. To formally verify the service functionality and estimate service performance, probabilistic model checking is used to the quantitative verification of Web service for fault detection and vulnerability analysis.
2. Web Service Prediction Phase. To achieve the reliability prediction from the variety of historical probability values, the regression analysis method is employed after obtaining a set of statistics data of Web service reliability.

In contrast to conventional monitoring approaches, the probability value of our method generated in monitoring phase helps to identify the failure service and its traces. And the linear equation constructed in prediction phase contributes to forecast when the service will be failure, indicating potential service failure.

The remainder of this paper is organized as follows: Section 2 introduces main techniques. Section 3 gives out a prototype. Section 4 shows a case study and experiments. Section 5 summarizes the related works. Section 6 draws a conclusion and future research directions.

2. The Predictive Monitoring Methodology

2.1. Quantitative Monitoring Using Probabilistic Model Checking

A Web service may become malfunctioned or unavailable at run time with a certain probability. The reliability refers to the probability that the business function will successfully execute. Thus, quantitative verification used to monitor the quality of Web service provides a chance of recovery once a problem has been detected.

We adopt one of famous probabilistic models, Discrete-Time Markov Chains (DTMC), to formalize behaviors of Web service annotated with probability. The model then is analyzed by verifying properties specified in form of temporal logic PCTL, and evaluated through probabilistic model checker PRISM. It is defined as follow:

Definition 1(WS-DTMC). *Web Service DTMC, called WS-DTMC, is a labeled transition system where each state corresponds to a Web service entity and each transition is annotated with a probability value indicating the likelihood of its occurrence during possible Web service invocations. It is a tuple* $(S, s_0, \delta, P, AP, L)$*, where,*

–S is a finite set of states;
–s_0 is an initial state;
–$\delta : S \to S$ is a finite set of edge relations;
–$P : \delta \to [0, 1]$ is a transition probability matrix, with $\Sigma_{s' \in S} P(s, s') = 1$ for all $s \in S$;
–AP is a finite set of atomic propositions;
–$L : S \to 2^{AP}$ is labeling function that assigns, to each state $s \in S$, a set $L(s)$ of atomic propositions.

Since WS-DTMC is a performance model, the quantitative verification of WS-DTMC focused on probability reachability property can verify both functional and non-functional behaviors. For any state s, $s' \in S$, the transition probability matrix gives the probability $P(s, s')$ of making a transition from s to s' in one discrete step. The probability of finite path $\omega = s_0, s_1, , s_{n-1}, s_n \in \mathrm{Path}_{\mathrm{WS-DTMC}}(\mathrm{s_0})$ belonged to the set of path fragments with starting state s_0 is:

$$\begin{aligned} Pr_s(\omega) &= \prod_{0 \le i \le n} P(s_i, s_{i+1}) \\ &= P(s_0, s_1)P(s_1, s_2) \dots P(s_{n-1}, s_n) \end{aligned} \quad (1)$$

Generally, the measure to compute the value $Pr_s\{\omega \mid \omega \models \phi\} \sim p$ relates to reachability probability, where ϕ is an temporal logic satisfied in path ω, p is a threshold and $\sim \in [\le, <, \ge, >]$ is the restrictive symbol. We mainly uses two key PCTL formulae, Probabilistic Computation Tree Logic [10, 15], to evaluate the performance, namely $P_{\sim P}(\phi)$ and $P_{=?}(\phi)$.

The operator $P_{\sim P}(\phi)$ yields a *true* value when the probability of a path formula ϕ being true in state satisfies the bound $\sim p$. Otherwise, it outputs false. For example, $P_{<0.5}(\mathbf{F}\, state = bad)$ means that "with probability 0.5 or less, a bad state will be visited eventually". It refers to compute the maximal or minimal reachability probability.

The operator $P_{=?}(\phi)$ returns the probability value for given path formula. For example, $P_{=?}(\mathbf{F}\, state = bad)$ means that "what is the probability that a bad state will be traversed?". It refers to iteration computing of matrix-vector multiplication representing the probability value of n-step reachability.

2.2. Reliability Prediction using Unary Linear Regression Analysis Method

As the reliability of Web service may change randomly over time, predicting the reliability of Web service can accelerate their response to the changeful Internet environment. We use regression analysis approach [11, 16] to do reliability prediction for Web service. The result is to assign an interdependence of the quantitative relationship between variables of the reliability value and the invocation duration time. The procedure of prediction includes following steps:

1.Calculate the regression coefficient for showing the single changing influence of each independent variable based on the dependent variable.
2.Estimate the goodness of fit for indicating how smooth the regression equation model is.
3.Compute the standard error for controlling the accuracy of reliability prediction.

Definition 2(WS-LREM). *Web Service Linear Regression Equation Model, called WS-LREM, is introduced for Web service reliability prediction* $\hat{y} = a + bt$*, where the independent variable* t *is the invocation duration time, the dependent variable* $\hat{y}$ *is the reliability probability. As explained in coordinate system* (x, y)*, constant* b *is the determination coefficient and constant* a *is the vertical intercept.*

Let $u = y - \hat{y}$ be a stochastic disturbance item between the actual probability value and prediction value. To minimize the stochastic disturbance item u, the least square method is adopted to figure out the point of inflection. Then, the symbols a and b are considered as derivation factors and the coefficient is calculated by the partial derivatives of Σu^2. After that, we set the partial derivative equal to 0 and calculate the unknown numbers a and b.

$$\begin{aligned} &\sum u^2 = \sum (y - \hat{y})^2 = \sum (y - a - bt)^2 \Rightarrow \\ &\begin{cases} \frac{\partial \sum u^2}{\partial a} = -2 \sum (\mathrm{y} - a + bt) = 0 \\ \frac{\partial \sum u^2}{\partial b} = -2 \sum (\mathrm{y} - a - bt)\, t = 0 \end{cases} \end{aligned} \quad (2)$$

The linear equation system with two unknowns a, b makes $\sum u^2$ be maximal, where the result of value a and b is the best solution for the given regression equation.

$$\begin{aligned} &\begin{cases} na + b \sum t = \sum c \\ a \sum t + b \sum t^2 = \sum ty \end{cases} \Rightarrow \\ &\qquad \begin{cases} a = \frac{\sum y}{n} - b\frac{\sum t}{n} = \bar{y} - b\bar{t} \\ b = \frac{n \sum ty - (\sum t \sum y)}{n \sum t^2 - (\sum t)^2} \end{cases} \end{aligned} \quad (3)$$

The symbol $\sum t$ is the sum of invocation duration times for independent variable,$\sum y$ is the sum of

probability values for dependent variable, n is the observed sample of monitoring numbers, and $\bar{t}$, $\bar{y}$ represent the average of independent variable t and dependent variable y, respectively.

However, the prediction accuracy is subject to the impact of random sampling errors. Therefore, the decision coefficient r^2 is used to represent the depend relation between the changed value y and the varied value t, where $(\hat{y} - \bar{y})$ is the regression deviation, and $(\bar{y} - \hat{y})$ is the surplus deviation.

$$r^2 = \frac{\sum(\hat{y} - \bar{y})^2}{\sum(y - \hat{y})^2} \Rightarrow r = \sqrt{1 - \frac{\sum(y - \hat{y})^2}{\sum(y - \bar{y})^2}} \quad (4)$$

The correlation coefficient $r \in [0, 1]$ states the smooth of regression equation model that the greater the decision coefficient is the better the goodness of fit can be obtained. Otherwise, the predicted value is influenced by random factor so that we should enlarge the observed sample n.

Since it may overestimate (or underestimate) the coefficients which will greatly influenced the prediction accuracy, the following standard error estimation formula is introduced.

$$S_y = \sqrt{\frac{\sum y^2 - a\sum y - b\sum ty}{n - 2}} \quad (5)$$

Ultimately, the standard error is integrated into the regression equation model $\hat{y}' = \hat{y} \pm S_y$. The probability value of Web service viewed as dependent variable can be quantitatively figured out when the value of invocation duration time is input.

3. The Prototype of Predictive Service Monitoring System

The prototype of our predictive Web service monitoring is illustrated in Fig. 1, which is implemented by JDK, Axis2.0, PRISM and Apache Tomcat on Eclipse plug-in platform. It includes five main components: Service Composition Executor, WS-DTMC Modeler, Service Monitor, Service Predicator and Service Re-Configurator. Each function module is as follow:

1. The Service Composition Executor is responsible for the execution of Web service. It refers to invoke service entry and handles the failure statistics dynamically. After user ends business interactions, the frequency of a probabilistic transition being taken is updated to service log database.
2. As an essential component, we implement the WS-DTMC Modeler by BEPL4WS Translator and Service Logger. It aims to devise suitable models for probabilistic model checking Web service, taking into account both the functional and non-functional behaviors. First, in order to transform the BPEL4WS workflow into a deterministic FSM, we adopt Foster methodology [6] using tool LTSA-WS [7]. Second, by means of investigating the access frequency and failure records in Service Logger, we annotate each state transition with behavior probability in form of WS-DTMC.
3. Service Monitor is the focus of our paper that monitors the available Web service and collects their probabilistic performance data in real-world. We employ model checker PRISM to perform the automated verification of probabilistic behaviors. Fundamental components of PRISM language are modules and variables, supporting commands to be labeled with actions for parallel composition. The syntax is in form of action guard command which forces two or more modules to make transitions simultaneously with different probabilities [15]. $[<action>]<gurad> \rightarrow <prob> : <update> + \cdots + <prob> : <update>$.
4. We implement the Service Predictor based on the unary linear regression analysis method. The related parameters include monitoring interval Γ, invocation duration time $Time$, threshold $\gamma \in [0, 1]$ and comparison parameter μ used to evaluate the coefficient r^2. The historical data used in Service Predictor are recorded in monitoring phase. Using the value calculated by linear regression model to implement Web service reconfigurations in advanced contributes to improvement in the performance of business applications.
5. Service Re-Configurator is to replace one or multiple services to comply with probability-based QoS constrains. Details about these produces are out of the scope of this paper. For completeness, we simply give a short introduce that if the substitution for an individual service cannot be found, we replace multiple services until a satisfactory solution can be found according to the reliability value reported by Service Predictor.

4. Case Study and Experiments

4.1. Case Study

Fig. 2 illustrates a simple example of Online Trip Planning (OTP) with five services. Service state s_0 starts by receiving user login commands. Service state s_1 recommends a travel destination to user. User may preserve a hotel by service state s_3, or book a fight ticket by service state s_2. Followed by accomplishing the fight ticket purchase, user also can order an accommodation conveniently. The end states s_4 is logout service, while end state s_5 is failure information service.

The distributed execution of services emerges different reliability characteristics. To monitor and predict

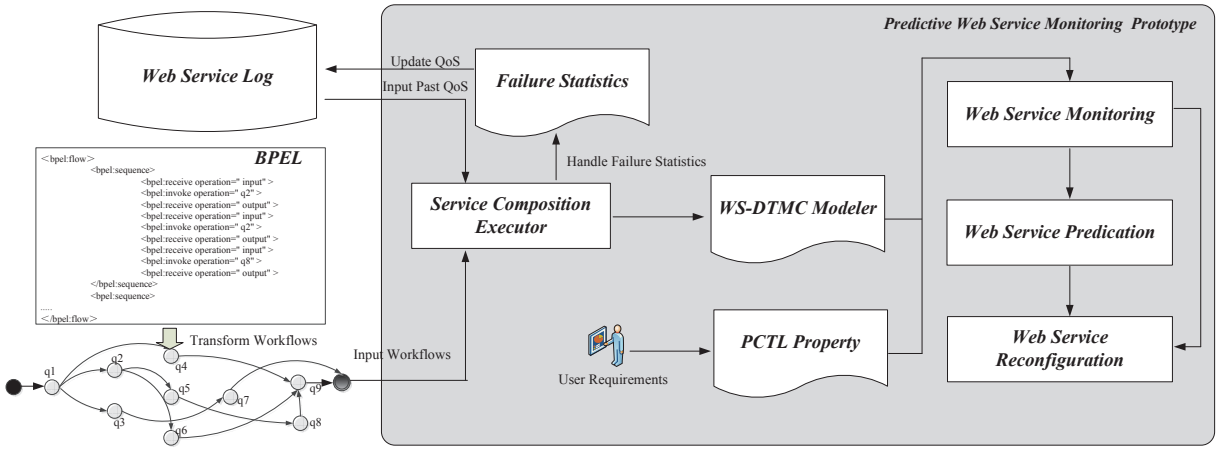

Figure 1 Prototype of Our Approach.

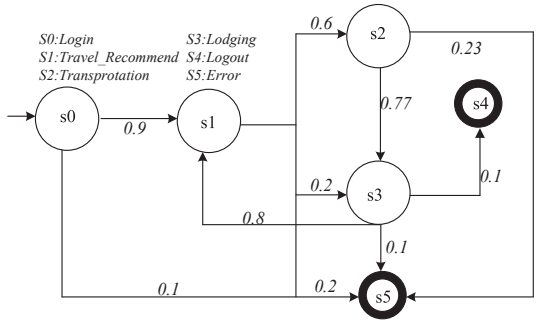

Figure 2 A Motivating Case Study.

Web service reliability, the WS-DTMC model should firstly be translated into language PRISM [10]. The PRISM codes of OTP in Table 1 contain two parts: variables and guarded commands. The variable is defined as enumeration type with initial value, such as "state:[0..5] init 0". And the guarded command specifies the transitions with probability characteristic. For example, in line 8, when service state s_3 is visited, the transition will evolve to service state s_1, service state s_4 and service state s_5 with probability of 80%, 10% and 10%, respectively.

Given a monitoring property $P_{\leq 0.3}(\mathbf{F}\, state = 2)$. It first changes to compute the probability value r of $P_{=?}(\mathbf{F}\, state = 2)$, and then assigns threshold $\gamma = 0.3$ as expected reliability. The comparing result between r and γ reveals the performance of Web service, by which we can confirm service failures.

The generation of quantitative properties for OTP focuses on state coverage using PCTL, which requires that each service state should be visited at least once. By state coverage criteria, properties associated with the reachability operator **F** are produced as shown in Table 2.

Table 1 PRISM Codes

No	Statements
[1]	dtmc
[2]	module otp
[3]	//local variables
[4]	state : [0..5] init 0;
[5]	[]state = 0 → 0.9 : (state' = 1) + 0.1 : (state'=5);
[6]	[]state = 1 → 0.6 : (state'=2) + 0.2 : (state' = 3) + 0.2 : (state' = 5);
[7]	[]state = 2 → 0.77 : (state' = 3) + 0.23 : (state' = 5);
[8]	[]state= 3 → 0.8 : (state' = 1) + 0.1 : (state' = 5) + 0.1 : (state' = 4);
[9]	[]state = 4 → state' = 4;
[10]	[]state = 5 → state' = 5;
[11]	endmodule

Table 2 Formulae using State Coverage

ID	State	Property Formula
1	S_0	$P_{=?}(\mathbf{F}\, state = 0)$;
2	S_1	$P_{=?}(\mathbf{F}\, state = 1)$;
3	S_2	$P_{=?}(\mathbf{F}\, state = 2)$;
4	S_3	$P_{=?}(\mathbf{F}\, state = 3)$;
5	S_4	$P_{=?}(\mathbf{F}\, state = 4)$;
6	S_5	$P_{=?}(\mathbf{F}\, state = 5)$;

Table 3 shows the overall probability values. Each row represents the probability value of reaching the desired state starting from service state s_i under the specified reachability property formula. If the returned value is unsatisfied to user predefined threshold, a dynamic reconfiguration is needed. We get following conclusions:

1.The constant probability 1 or 0 means that it is a universal quantification. The probability reachability

Table 3 Probability Values of Execution Result

Property Formula	Start State	Probability
$P_{=?}(\mathbf{F}\, state = 0)$	state=0	1
	state=1	0
	state=2	0
	state=3	0
	state=4	0
	state=5	0
$P_{=?}(\mathbf{F}\, state = 1)$	state=0	0.899
	state=1	1
	state=2	0.629
	state=3	0.79
	state=4	0
	state=5	0
$P_{=?}(\mathbf{F}\, state = 2)$	state=0	0.622
	state=1	0.688
	state=2	1
	state=3	0.581
	state=4	0
	state=5	1
$P_{=?}(\mathbf{F}\, state = 3)$	state=0	0.596
	state=1	0.666
	state=2	0.769
	state=3	1
	state=4	0
	state=5	0
$P_{=?}(\mathbf{F}\, state = 4)$	state=0	0.115
	state=1	0.13
	state=2	0.172
	state=3	0.198
	state=4	1
	state=5	0
$P_{=?}(\mathbf{F}\, state = 5)$	state=0	0.873
	state=1	0.849
	state=2	0.841
	state=3	0.788
	state=4	0
	state=5	1

will not be changed during the iterative calculation process.

2. If the probability reachability drops down to the predefined threshold value, we need to execute reconfigurations for Web service optimization.
3. The $P_{=?}(\mathbf{F}\, state = 4)$ and $P_{=?}(\mathbf{F}\, state = 5)$ are used to compute the probability reachability of terminal states. In the former, it represents the probability of reaching failure state. Contrarily, the latter represents the probability of reaching a terminal state. Studying the probability value of reaching terminal states can quantitatively verify the deadlock or liveness property.
4. The remaining properties are used to compute the probability reachability of inner states. It helps to estimate the probability relationship between two service states for improving the performance of business logic. Thus, forecasting these probabilities contributes to maximize the benefit of e-commerce.

To introduce the procedure of generating the linear regression equation for predicating the reliability of Web service, we first discuss the property $P_{=?}(\mathbf{F}\, state = 3)$ with starting service $state = 1$ as an example.

Table 4 Data For Building Linear Equation

ID	t	y	ty	t^2	y^2
1	1	0.666	0.666	1	0.443556
2	2	0.626	1.252	4	0.391876
3	3	0.591	1.773	9	0.349281
4	4	0.549	0.2.196	16	0.301401
5	5	0.51	2.55	25	0.2601
Total	15	2.942	8.437	55	1.746214

According to Section 2.2, the coefficient a and b can be computed from Table 4 where each data is recorded at each invocation duration time $t \in \{1, 2, 3, 4, 5\}$ during monitoring phase.

$$b = \frac{n\sum ty - (\sum t \sum y)}{n\sum t^2 - (\sum t)^2} = \frac{5*8.437 - (15*2.942)}{5*55 - (15)^2} = -0.0389$$

$$a = \frac{\sum y}{n} - b\frac{\sum t}{n} = \frac{2.942}{5} - b\frac{15}{5} = 0.7051$$

Thus, the linear regression equation model is $\hat{y} = 0.7051 - 0.0389t$. Initially, service state s_1 has probability of $a = 0.7051$ to reach state s_3. With the invocation duration time increasing, the probability is decreasing with 0.0389.

Table 5 Data For Building Linear Equation

ID	t	y	$\hat{y}$	$y - \hat{y}$	$y - \bar{y}$
1	1	0.666	0.6662	-0.0002	0.0776
2	2	0.626	0.6273	-0.0013	0.0376
3	3	0.591	0.5884	-0.0026	0.0026
4	4	0.549	0.5495	-0.0005	-0.0394
5	5	0.51	0.5106	-0.0006	-0.0784

Furthermore, to validate the effectiveness of sample, the correlation coefficient r needs to be estimated. It is clear that the goodness of fit $r = 0.9$ is appropriate, which is figured out by using decision coefficient r^2 formula and data from Table 5. In addition, the standard error estimation is 0.0017. Thus, the final equation

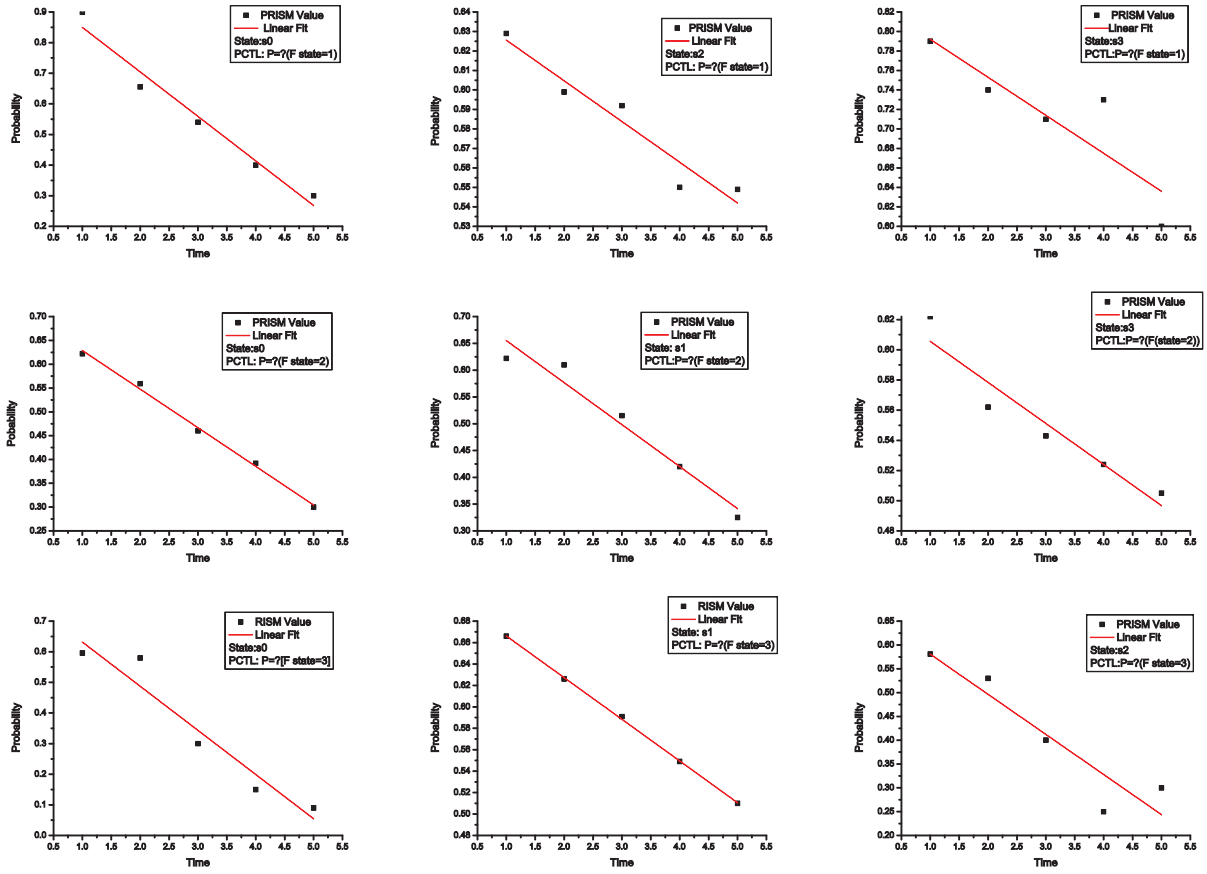

Figure 3 Linear Fit Equation Simulation for Web Service.

$$\hat{y} = 0.7051 - 0.0389t \pm 0.0017.$$

$$r = \sqrt{1 - \frac{9.1 \times 10^{-6}}{0.0151412}} \approx 0.9 \tag{6}$$

$$S_y = \sqrt{\frac{\sum y^2 - a\sum y - b\sum ty}{n-2}} \approx 0.0017 \tag{7}$$

Using this equation can check whether the observed reliability deviates from the performance promised in user's predefined requirements. For example, given threshold $\gamma = 0.2$, the invocation duration time t belonged to region [12.94, 13.03] will make the threshold unsatisfied. In this case, we should take out a reconfiguration until time $t = 12.94$ is reached.

4.2. Experiments

In order to show the efficacy of our proposed approach, we then conduct experiments at a simulation environment that 2.0GHz CPU and 2GB RAM with Windows 7 Operating System. In experiment, we vary the Internet stability from 90% to 40% with a step value of 10%. The primary idea of these experiments is to record the reliability value of each service state for future prediction. Table 6 shows experiment results. The column denoted as Time is the invocation duration time. Each cell is probability value calculated by PRISM. Using linear regression analysis method, we get following results of Fig.3 from Table 6's data.

Table 6 The Historical Probability Log

Time	State								
	$P_{=?}(\mathbf{F}\ state=1)$			$P_{=?}(\mathbf{F}\ state=2)$			$P_{=?}(\mathbf{F}\ state=3)$		
	0	2	3	0	1	3	0	1	2
1	0.899	0.629	0.79	0.622	0.688	0.581	0.596	0.666	0.769
2	0.656	0.599	0.74	0.559	0.61	0.562	0.58	0.626	0.53
3	0.54	0.592	0.71	0.46	0.515	0.543	0.3	0.591	0.4
4	0.4	0.55	0.73	0.392	0.42	0.524	0.15	0.549	0.25
5	0.3	0.549	0.6	0.3	0.325	0.505	0.09	0.51	0.3

Fig.3 shows the linear regression equation for inner states, from which the probability value of Web service

can be calculated in some time. We find that at the top of Fig.3 the service state s_0 performs worse than service state s_2 and service state s_3. In fact, when Time is larger than 2, the trends of probability decrease slowly because they have a loop relationship back to service state s_1. In the middle of Fig.3, the success rate of reaching service state s_2 from service state s_3 is higher than those of the others. At the bottom of Fig.3, when the time clock increases, the service state s_0 will be unavailable due to its probability trend goes down quickly.

4.3. Evaluation

The real service failure may occur before the predicted time point. Thus, we need to consider the availability of service-oriented software when the prediction method is employed. In the second set of experiments, it evaluates the impact of parameters γ and Γ on availability. We vary the value of threshold γ given in the PCTL formulae from 0.6 to 0.95 with a 0.05 interval. For each experiment, we select a prediction period Γ from 5, 10, 15, 25, 30. This involves two steps: (1) The monitoring is running to collect probabilities during Γ. (2) The prediction is carried out at the boundary of Γ. According to the prediction value, we can devise a reconfiguration in some time for avoiding service failure when our predictive monitoring is applied.

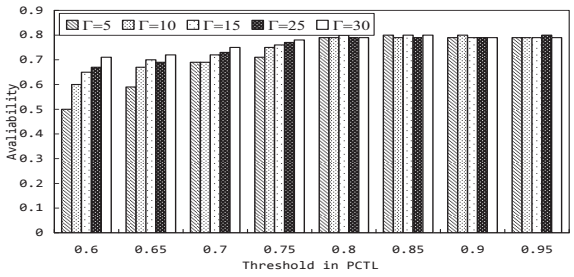

Figure 4 Impact of Parameters γ and Γ on Availability.

Fig.4 shows that the availability rises more distinctly with parameter γ increasing, but, it is smooth and steady at the value of 0.8. On the other hand, Fig.4 shows that the availability strengthens when parameter Γ increases because the time consumption on the reliability computation is related to Γ. The bad scenario is that the failure occurs when the prediction computation is unfinished, which may cause service failure. These experiments also demonstrate that frequently performing probabilistic model checking is not an effective solution to monitor Web service. It needs reliability prediction supporting. In this experiment, the parameters with $\Gamma = 15$ and $\gamma = 0.8$ are optimal options.

5. Related Works

Web service is considered as the basic infrastructure of choice for managing business process and workflow activities over Internet [1]. For presenting the nonfunctional characteristics of Web service, many QoS researches have been discussed at various perspectives, spanning the use of QoS ontologies, the definition of ad-hoc methods in QoS-aware frameworks, and the application of optimization algorithms [9].

Typical QoS metrics at the application level include throughput, response time, cost, reliability, fidelity, etc. Shao [19] proposed a collaborative filtering based approach to make similarity mining and prediction from consumers' experiences. Considering that service-aware computing is an important part of pervasive computing for Web-based mobile application with uncertainty, Zhang [25] gave a new service-aware computing approach for mobile application with uncertainty. Malak [17] gave a service QoS prediction architecture to predict the quality level during service selection and monitoring phase. Addouche [2] attempted to translate UML specifications into models to be solved by probabilistic model checkers. Cardoso [4] proposed two different metrics to evaluate the control-flow complexity of BPEL processes before their actual implementation. Zhao [26] modeled WS-CDL with QoS information and presented several QoS estimation methods. Compared to our work, their work focuses on at design phase.

Yu [24] designed a broker-based architecture to facilitate the selection of QoS-based services. The objective of service selection is to maximize an application-specific utility function under the end-to-end QoS constraints. Xiong [21] build a Web service Configuration Net based on Petri-nets under single and multiple QoS attributes to exhibit Web service configuration. However, they have only focused on the static scenario by assuming that each Web service has a deterministic QoS.

Zheng [27] recognized this issue, and proposed an approach for predicting QoS values of Web services by systematically combining the user-based PCC approach and the item-based PCC approach. Gallotti [9] presented a supporting tool ATOP, through which QoS properties can be specified and formally analyzed for service composition. Aiello [3] proposed an approach for choosing services from a group of candidates to implement an abstract composition while ensuring some important QoS criteria. But they don't consider the uncertainty which will lead to reliability dynamic change in Internet. This problem attributes to data mining research of service prediction. Xiao [20] presented a novel prediction recovery method through the integration of regression model and grey theory, which can guarantee real-time data services available by means of providing predictive values of damaged data to application activities that have to access these data immediately. Kim [13] suggested a practical approach to learning user's route

patterns from their histories and using that information to predict specific routes.

Different from above existing works, our paper employs probabilistic model checking to monitor Web services both at all of the functional and nonfunctional verification level, and uses unary linear regression analysis method to predict the reliability of Web service. The advantage of our method enables the adaptability and flexibility of Web service-based applications in the changeful Internet environment.

6. Conclusions

In this paper, we concentrated on studying the predicative Web service monitoring approach for service-oriented software. Our approach first transformed the high-level composition description BEPL4WS workflow into WS-DTMC model for performing probabilistic model checking. Then, the PCTL formulae extracted by state converge were used as the quantitative property of Web service. After executing formal verification in model checker PRISM, we introduced a statistics method to show the linear equation for revealing the relationship between the reliability and the invocation duration time. It contributed to predict the reliability of Web service in advance. Finally, a case study was discussed, and experiment results shown that our approach had a good scalability and efficacy for monitoring Web service.

Composing Web services for service dynamic reconfiguration handling the failure service may give rise to state-space explosion. Thus, in the future, we will investigate the probabilistic abstraction of Web service at semantic compatibility aspect.

Acknowledgement

This work is supported by National Natural Science Foundation of China (NSFC) under grant No. 61170044 and No. 60970007, Project of Science and Technology Commission of Shanghai Municipality (No. 10510704900) and Shanghai Leading Academic Discipline Project (Project Number: J50103).

References

[1] W.V.D. Aalst and K.V. Hee. Workflow Management: Models, Methods, and Systems. MIT press (2004).

[2] N. Addouche, C. Antoine, and J. Montmain. Combining Extended UML Models and Formal Methods to Analyze Real-Time Systems. In Proc. SAFECOMP 2005. Lecture Notes in Computer Science, vol.**3688**, pp.24-36. Springer-Verlag, Berlin, Heidelberg (2005).

[3] M. Aiello, F. Rosenberg, C. Platzer, A. Ciabattoni,and S. Dustdar. Service QoS composition at the level of part names. In Proc. the 3rd international conference on Web Services and Formal Methods. Lecture Notes in Computer Science, vol. **4184**, pp. 24-37. Springer-Verlag, Berlin, Heidelberg (2006).

[4] J. Cardoso. Complexity analysis of bpel web processes. Software Process: Improvement and Practice. **12(1)**, 35-49 (2007)

[5] G. Diaz, M.E. Cambronero, J.J. Pardo, V. Valero, and F. Cuartero. Model checking techniques applied to the design of web services. CLEI Electron. J. **10(2)**, 12-25 (2007)

[6] H. Foster, S. Uchitel, J. Magee, and J. Kramer. Model-based verification of web service compositions. In Proc. the 18th IEEE Int. Conf. on Automated Software Engineering Conference. pp. 152-163. IEEE Computer Society, Washington, DC, USA(2003)

[7] H. Foster, S. Uchitel, J. Magee, and J. Kramer. LTSA-WS: a tool for model-based verification of web service compositions and choreography. In Proc. the 28th international conference on Software engineering. pp. 771-774. ACM, New York, NY, USA (2006)

[8] X. Fu, T. Bultan, and J. Su. Analysis of interacting bpel web services. In Proc. the 13th international conference on World Wide Web. pp. 621-630. ACM, New York, NY, USA (2004)

[9] S. Gallotti, C. Ghezzi, R. Mirandola, and G. Tamburrelli. Quality prediction of service compositions through probabilistic model checking. In Proc. the 4th International Conference on Quality of Software-Architectures,Models and Architectures. Lecture Notes in Computer Science, vol. **5281**, pp. 119-134. Springer-Verlag, Berlin, Heidelberg (2008)

[10] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In Proc. 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Lecture Notes in Computer Science, vol.**3920**, pp.441-444. Springer-Verlag, Berlin, Heidelberg (2006)

[11] R.L. Huang and Y.H Guan. Data Statistics Analysis: SPSS Theory and Applications. The Higher Education Press, Beijing (2010), (in Chinese)

[12] S.Y. Hwang, H. Wang, J. Srivastava, and R.A. Paul. A probabilistic QoS model and computation framework for web services-based workflows. In Proc. the 2004 Int. Conf. Conceptual Modeling-ER 2004. Lecture Notes in Computer Science, vol. **3288**, pp. 596-609. Springer-Verlag, Berlin,Heidelberg (2004)

[13] J.M. Kim, H. Baek, and Y.T. Park. Probabilistic graphical model based personal route prediction in mobile environment.Appl. Math. Inf. Sci. **6(2S)**, 651-659 (2012).

[14] M. Kova, J. Bentahar, Z. Maamar, H. Yahyaoui. A formal verification approach of conversations in composite web services using nusmv. In Proc. the 2009 conference on New Trends in Software Methodologies, Tools and Techniques. pp. 245-261. IOS Press, Amsterdam, The Netherlands (2009)

[15] M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. Performance analysis of probabilistic timed automata using digital clocks. Form. Methods Syst. Des. **29**, 33-78 (2006)

[16] S. Makridakis, S. Wheelwright, and R. J. Hyndman. Forecasting: methods and applications (3rd edition). New York: John Wiley and Sons (1998)

[17] J.S. Malak, M. Mohsenzadeh, and M.A Seyyedi. Web service QoS prediction based on multi agents. In Proc. the

2009 International Conference on Computer Technology and Development. vol. **1**, pp. 265-269. IEEE Computer Society (2009)
[18] S. NAKAJIMA. Model-checking verification for reliable web service. In Proc.OOPSLA'02 Workshop on Web Services. pp. 1-5. IEEE Computer Society, Washington, DC, USA (2002)
[19] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei. Personalized QoS prediction forweb services via collaborative filtering. In Proc.the 2007 IEEE International Conference on Web Services. pp. 439-446 (2007)
[20] Y.Y. Xiao, H. Zhang, G.Q. Xu, and J.S. Wang. A prediction recovery method for supporting real-time data services. Appl. Math. Inf. Sci. **6(2S)**, 363-369 (2012)
[21] P. C. Xiong, Y. S. Fan, and M. C Zhou. QoS-aware web service configuration. IEEE Transactions on Systems, Man, and Cybernetics, Part A **38**, 888-895 (2008)
[22] H. L. Yang, L Zhou, K He, C Deng, X. P. Zhao, and Z. Y. Qiu. A probabilistic QoS model-checking for dynamic routing protocol. In Proc.the 10th International Conference on Quality Software. pp. 441-448. IEEE Computer Society, Washington, DC, USA (2010)
[23] Y.W. Yang, W.H. Chen, and H.P. Lu. A fuzzy-grey model for non-stationary time series prediction. Appl. Math. Inf. Sci. **6(2S)**, 445-451 (2012)
[24] T. Yu, Y. Zhang, and K.J. Lin. Efficient algorithms for web services selection with end-to-end QoS constraints. ACM Transactions on the Web. **1(1)**, 1-26 (2007)
[25] D. G. Zhang and X. D Zhang. A new service-aware computing approach for mobile application with uncertainty. Appl. Math. Inf. Sci. **6(1)**,9-21 (2012)
[26] X. P Zhao, S. C. Qin, H.L. Yang, and Z.Y. Qiu. A QoS view of web service choreography. In Proc.the 2007 IEEE International Conference on e-Business Engineering. pp. 607-611.IEEE Computer Society, Washington, DC, USA (2007)
[27] Z.B Zheng, H Ma, M. R. Lyu, and I. King. QoS-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. **4(2)**, 140-152 (2011)

Honghao Gao received the Ph.D degree in the School of Computer Engineering and Science of Shanghai University, Shanghai, China, in 2012. His research interests include Web service and model checking.

Huaikou Miao. He is currently a professor in Computer Engineering and Science at Shanghai University, China. His research interests include formal methods and software engineering.

Hongwei Zeng ,Ph.D. He is currently a senior research fellow in Computer Engineering and Science at Shanghai University, China. His research interests include formal methods and software testing.