Usage Pattern Based Survivability Prediction of Web System

Xun Lee[1,2], Jiaan Zhou[1,2], Hao Fu[1,2]
[1](School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China)
[2](Shanghai Key Laboratory of Computer Software Testing & Evaluating, Shanghai 201112, China)
{xunlee, mujita, fhfuture}@shu.edu.cn

Abstract—With the rapid development of SOA technology, the instability of the distributed Web services has become a hot research topic. The survivability of Web system at a certain period of time is an important and useful indicator for making an appropriate adjustment dynamically to ensure the system runs best while the runtime environment or requirements are changed. However, most researchers only focus on estimating the overall survivability of Web system. The overall survivability cannot reflect the QoS of Web system under different usage patterns actually. In this paper, we propose an approach based on Web log statistic to predict the particular survivability of Web system under a particular usage pattern.

Keywords- distributed Web services; survivability; QoS；usage patterns; Web log statistic;

I. INTRODUCTION

Nowadays, Web applications have been widely used to support various services in our daily life. Although Web applications bring us convenience and improve our lives, distributed Web services is very instable on account of the dynamic and complex network environment. Therefore, techniques to ensure the stability and reliability of distributed Web services have been the research focus in both academia and industry. SOA theory is one of the techniques to solve those problems. The main idea of SOA is that the service is basic units of Web applications in which application components provide service to other components, and we can build the distributed application system by compositing different services.

In the real operation environment, instability of the distributed Web services may result in accidental fault, and lead to the failure of whole system [1]. Therefore, making a rapid response to the changed environment and requirements as well as making an appropriate adjustment dynamically to ensure the system runs at the best performance are very necessary and urgent. To our knowledge, there are two main categories for Web services adaptive adjustment at present, passive adaptive adjustment and active adaptive adjustment. Passive adaptive adjustment refers to adjusting after problems happened while active adaptive adjustment means that the system will be monitored real-time and can forecast the faults through the monitoring data before the service being out of work. The survivability of Web system at a certain period of time is an important and useful indicator for adaptive adjustment of Web service compositions. Survivability means the ability of systems to provide core service under a particular environment, and it also depends on how the system responds [2]. Survivability combines the concepts of reliability, safety, fault tolerance, researchers have not reached an agreement on the accurate definition of survivability, and the measurement and analysis methods of survivability are dependent on the particular condition and environment. Up to now, most existing survivability prediction approaches can only evaluate the overall survivability of Web system. In fact, under specific usage patterns, the particular survivability of a Web system may be different from the overall survivability. Therefore, it is necessary and meaningful to estimate the survivability of different usage patterns. In this paper, we adopt K-Means clustering algorithm [3] to partition the Web users into groups, and construct usage model and corresponding value model for each group, then we describe value model as Discrete Time Markov Chain (DTMC), which can be calculated by the model checking tool. Our approach adopts the idea of value-driven. We judge the survivability of a system by whether the benefits we evaluate can meet the administrator's value expectations.

The remainder of the paper is organized as follows: Section II briefly discusses the related work. Section III and IV present our approach and case study respectively. Finally, we conclude and propose further work in section V.

II. RELATED WORK

Various approaches have been used to predict the quality of Web services, such as collaborative filtering technique, multi agents and model checking. Shao et al. [4] put forward a user-based CF algorithm to predict QoS values and firstly adopted collaborative filtering technique to make QoS prediction, and this method is primarily based on the experience of similar users and is often used to recommend personalized services. Malak et al. [5] applied a forecasting method based on Neural Networks to predict Web services QoS level. Gao et al. [6] proposed a prediction method based on probability model checking to forecast the reliability of the Web services dynamically. Some researchers like Zhou et al. [7] combined the statistical calculation method and model checking technology to predict the survivability of Web system based on log statistics, but they cannot reflect the quality of Web applications under different usage patterns. To overcome this problem, we propose an approach based on Web log statistic to predict the particular survivability of Web applications under a particular usage pattern.

978-1-5090-0806-3/16/$31.00 copyright 2016 IEEE

Although the meanings of survivability vary in accordance with the granularities of the concern abstractions, survivability is believed that could be divided into errors, impact, reliability, cost and benefits [8]. The benefits a Web system can bring is an important factor for measuring the survivability, predicting Web system survivability means to protect the business which it supports and its value [9]. Knight and Sullivan [10] defined a survivability specification as a four-tuple which includes operating environment, essential services of a system and probability distribution of the required services. On the basis of Knight and Sullivan, Zhou et al. [11] applied the system model and the satisfied results of survivability to the definition, and employed model checking technology to analyze system survivability. Literature [7] defined survivability as a triple by reference to Zhou et al. on their approach. This paper is a further study on literature [7], and so we adopt the survivability definition in [7], which is defined as follows.

Definition 1(Survivability definition). Survivability is defined as a tuple K, K = {R, P, M}, where

- R is a requirement about services which must be provided by system;
- P shows that whether the model meets the survivable demand or not. It can be a set of Boolean variables or the probability on the R;
- M is a value model. It is the abstract of the system performance in a specific environment and on the specified period.

We adopt the idea of value-driven, and the expected value, which is in the set of R, is based on the requirement and historical value statistics of the system benefits. We forecast survivability of the Web system by comparing expected value and simulation value. The value model in our approach can be described as a Discrete Time Markov Chain (DTMC), which can be examined by the model checking tool. We use probabilistic model checking technology to calculate simulation value of each value model. Section III will show our approach in detail.

III. USAGE PATTERN BASED SURVIVABILITY PREDICTION

The process of our approach to predict survivability of Web system is different from existing approaches and it is shown in Fig. 1. We need to build more than one statistical model to estimate the survivability of Web system under a particular usage pattern. Each model corresponds to one usage pattern of the Web application. This method has four main steps as follows.

Step 1 is to preprocess the log information and partition the users into groups.

Step 2 is to build a Markov usage model from each group's user sessions.

Step 3 is to add value attribute to corresponding states in usage model to construct the value model.

Step 4 is to use probabilistic model checker PRISM to evaluate the rewards for value model and predict the survivability of the Web system under a particular usage pattern.

This paper adapt Euclidean distance [12] to measure the similarity between two users and use the classic K-Means clustering algorithm [3] to partition the Web users into groups. We build Markov usage model by analyzing the system log data, and adds value attribute to corresponding states to constructs the value model. We adopt quantitative verification technology to predict the expected accumulate value and evaluate the survivability of the system.

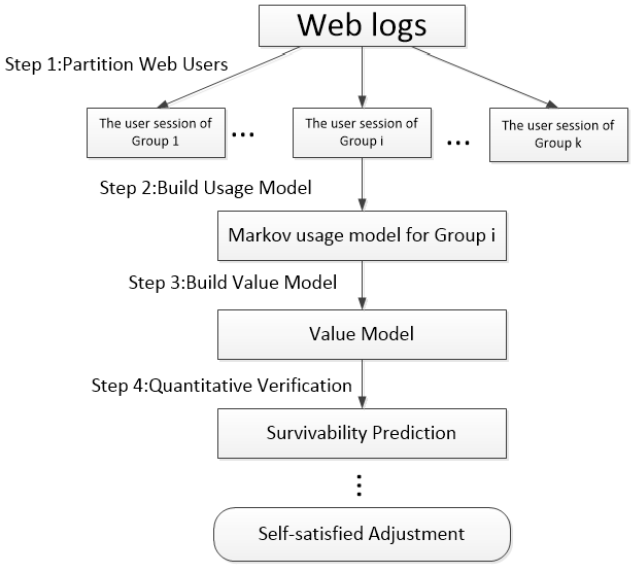

Fig. 1.Process of Our Approach

A. Web Logs Preprocessing and Web User Partition

Web logs refer to some log files which record various information of users, a Web log file contains massive log entries, each log entry includes IP address, Date, States, Resource, Referrer, etc. The IP address and User-Agent are usually used to identify the users of log entries. We define the access log entry as an 8-tuple <remoteHost, agent, reqTime, preSt, reqSt, status, bytes, respTime>. In the tuple, remoteHost denotes user's ip address and agent denotes the browser information of users, reqTime denotes the date and time when user request the Web application, preSt represents the state before Web users request the Web application, reqSt means that what information do the users want to obtain, status denotes the response status of server, bytes denotes the total bytes during one request, respTime denotes the date and time when the Web application responses user.

We need to measure the similarity between users before partitioning the Web users, and we identify the characteristics of one user by the transitions and their occurrence probabilities. Table 1 shows a sample of a User-Transfer matrice, and Table 2 shows the corresponding probability matrix. We adopt Euclidean distance to measure the similarity between two users, if the distance is smaller, then the two users is more similar, when the distance is zero, we can regard the two users have no differences. We believe the similar users have approximately equal probability to access the same service. The Euclidean distance can be calculated as follows:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(ptr_i - qtr_i)^2} \quad (1)$$

In the equation 1, ***p***, ***q*** are two users, ptr_i and qtr_i denote the occurrence probability of the ***i*** th transition for user ***p*** and ***q*** respectively. Based on the definition of similarities of the Web users, we adapt the K-Means clustering algorithm to partition the Web users into groups, and we can regard the users of each group have the similar usage pattern.

TABLE 1. A Sample of a User-Transfer Matrices

	S1	S2	S3	SUM
S1	-	4	6	10
S2	3	-	5	8
S3	-	5	-	5

TABLE 2. A Sample of a User-Transfer probability matrix

	S1	S2	S3
S1	0	0.4	0.6
S2	0.375	0	0.625
S3	0	1	0

B. Building Usage Model

Different from most of the researchers, our Usage Model in this part refers to the behavior of a particular kind of users in the system. As is known to all, Markov usage model [13] is very suitable to describe the navigation model of a Web System. Our approach adopts Markov usage model to describe the usage model of users. If we regard one Web service as the state of a Markov usage model, and regard the occurrence probability as the state transitions, then the usage pattern of a Web system can be described easily by a Markov usage model.

There are some methods to define the Markov usage model, for example, [13] defines a Markov usage model as a quintuple. Our approach is based on probability model checking technology, so we adopt probabilistic labeled transition system (PLTS) to describe the Markov usage model, PLTS can be defined as follows.

Definition 2(Probabilistic Labelled Transition System) $M = (S, Init, \Sigma, T, AP, L)$, where

- S is a finite nonempty set of states;
- $Init \in S$ is the initial state;
- Σ is a finite set of actions;
- $T \subseteq S \times \Sigma \times Distr(S)$ is the translation relation, Distr (S) is the probability distribution function set of the state sample space S, the state sample space $\in$ state space;
- AP is a finite set of atoms;
- $L: S \to 2^{AP}$ is a proposition assignment function.

A Markov usage model, which includes some nodes and arcs, can be described by a directed graph. The nodes represent one service of a Web application in our approach, and the arcs represent the transition probability. We can easily build the Markov usage model with our algorithm 1 designed below.

Algorithm 1 BuildMarkovModel

Input: the log entries of one user group //log files;
Output: a directed graph G //the Markov usage model;

Step 1. Get the set of states.
List<node> **StateList** = new ArrayList()
for each log entry **Data**$_i$ do
if the previous / next State of **Data**$_i$ is not in **StateList** **then**
StateList.add(State)
end if
end for
Step 2. Count the transition labels
int N= StateList.lenght
G=new [N][N]
for each log entry **Data**$_i$ do
statePre=Data.pre
stateNext=Data.next
int i= **StateList**.getIndex(statePre)
int j= **StateList**.getIndex(stateNext)
G[i][j]++;
end for
Step 3. Calculate the probability of the transition labels
Total=new [N]
for(**G**[][])
for(**G**[])
Total[i]=count(**G**[])
end for
end for
for (**G**[][])
for(G[])
G[j][i]=**G**[j][i]/Total[i]
end for
end for

return G;

C. Building Value Model

The value of a system is an important criterion for survivability, predicting Web system survivability means in some degree to predict the services which it supports and its value. The value of system business can be affected by many factors, in order to simplify the problem and illustrate our methods more straightforward, we add corresponding value attributes to each state in the usage model. We suppose the service can produce a certain amount of value for the system when one user accesses the service. That is to say, if the value is 1, the system could gain one dollar when one user accesses the service.

Our value model can be also described by probabilistic labeled transition system (PLTS), and it can be defined as follows.

Definition 3(Value model) $M =< Q, Init, \sum, T, AP, L >$, where

- Q=(S,V),S is a finite nonempty set of states, V is a finite set of values;
- $Init \in S$ is the initial state;
- Σ is a finite set of actions;
- $T \subseteq S \times \Sigma \times Distr(Q)$ is the translation relation, $Distr(Q)$ is the probability distribution function set

of the state sample space S, the state sample space $\in$ state space;

- AP is a finite set of atoms;
- $L: S \rightarrow 2^{AP}$ is a proposition assignment function.

D. Quantitative analysis

There are many factors such as errors, reliability, cost and benefits can affect the survivability of system, our approach is based on the idea of value-driven. We use quantitative model checking technique to simulate the value of system under a particular usage pattern. We judge the survivability of a system by whether the benefits we evaluate can meet the administrator's value expectations. The system would be not survivable if the simulative value below administrator's expected value, and it is necessary to adjust the system structure and services to ensure the Web system can gain most benefits.

Quantitative model checking technique is an automated analysis technique which combines model checking with quantitative data analysis. The usual strategy of quantitative model checking technique has three main steps [6]:

-Step 1. Construct service behavior model by analyzing the usage information of a Web application.

-Step 2. Generate probabilistic timing properties from the usage model.

-Step 3. Implement quantitative model checking according to corresponding properties.

As we mentioned previously, we construct the value model by probability behavior transfer information and add price-return constraint to some states. In the step 3, we use the probabilistic model checker PRISM [15] to calculate the cumulative value of Web system. PRISM is a quantitative probabilistic model checking tool which supports quantitative analysis and property checking. We adopt PRISM to evaluate the rewards for our value models in this paper.

IV. CASE STUDY

In this section, we present the results of a bookstore Web application, and show how our approach works. The bookstore Web application is an open source e-commerce application which can be downloaded from gotocode.com, and many researchers [16][17] adopt the bookstore Web application as their Web experiments.

The Web application is deployed in a LAN in our experiment, and we use Tomcat 7.0 as the Web server. We invited our classmates as volunteers to access the bookstore Web application. We implement our approach according to the four steps mentioned in Section III, to present and discuss survivability of the bookstore Web system under different usage patterns.

A. Preprocess Web Logs and Partition Web Users

As is known to all, IP address and User-Agent are usually used to identify the Web user, in our experiment, 2563 log entries and 43 Web users are found. We group the log entries of one user within 15 minutes into a user session, and we get 64 user sessions. We adopt Euclidean distance to calculate the similarity between users. Users in a group have approximate probability to the same service, and so if the distance is small, the two users are more similar.

We adapt the K-Means clustering algorithm to partition 64 user sessions into 5 groups according to the similarity between users. Table 3 lists some frequent transitions which can reflect the specific characteristics of each group.

In Table 3, the transition "BookDetail → BookDetail_Form_Order_Insert", "Orders" is the name of a form, "Insert" is the name of a form action. Usage pattern of each group has its own characteristic, for example, the members of Group 1 are the new users of this bookstore, and they only register a new account and then browse few other pages. The users of Group 2 are administrators whose responsibility is to manager books. The users of Group 3 are the customers who always place orders and also likely to cancel orders. The users of Group 4 are customers who always purchase the books and pay in a short time. The users of Group 5 are just likely to search some information about books.

TABLE 3. Frequent Transitions of 5 Groups

Group No.	Frequent Transitions
Group 1	Login → Registration
	Registration → Registration_Form_Reg_Insert
	Registration → Login
Group 2	AdminBooks → BooksMaint
	BooksMaint → AdminBooks
	BooksMaint → BookMaint_Form_Books_Update
Group 3	Main → Books
	BookDetail → BookDetail_Form_Order_Insert
	ShoppingCart → ShoppingCartRecord
	ShoppingCartRecord → ShoppingCartRecord_Form_Delete
Group 4	Main → Books
	BookDetail → BookDetail_Form_Order_Insert
	ShoppingCart → ShoppingCartRecord_Form_Pay
	ShoppingCart → Books
Group 5	Main → Books
	Books → BookDetail
	BookDetail → Books

B. Usage Model Construction

In Table 3, each group has its own particular usage pattern, and we build a Markov usage model for each group from the user sessions. Algorithm 1 shows the three steps to construct the usage models. Fig.2 is an example Markov usage model for Group 5, which contains 10 states and 31 transitions.

C. Building Value Model

In our experiment, we believe every business affair can produce a certain amount of profit. To simplify the business value model of the bookstore Web system, we add value attributes to some states in the usage model to build the value model. Fig. 3 shows the value model of group 5, state s0 is the login page of the Web system. State s1 is the

register user page. As we can see from Fig. 3, the transition

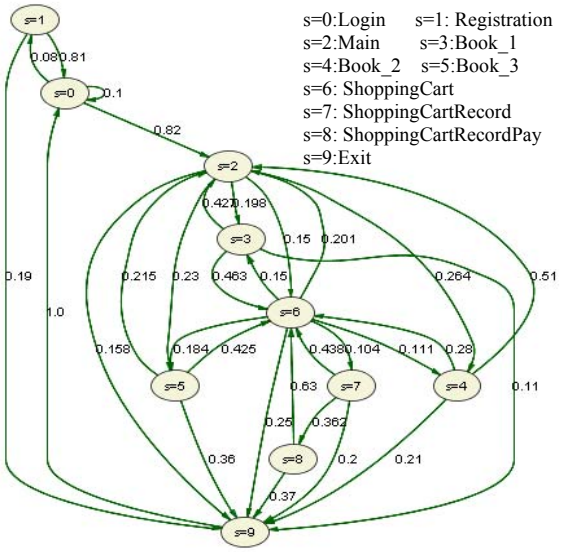

Fig. 2.Part of the Markov usage model for Group 5

probability from s0 to s1 is 0.08, which may reflect that there are 8 percent of new users in Group 5, we suppose the system will gain 0.8 dollar when a new user registers the Web system. State s2 is the main page which contains many book links, and state s3, s4 and s5 are the corresponding book details, the value-reward v=0.1 means the system will gain 0.1 dollar when a user browse one book. In addition, this paper adds value-reward v=0.2 to states s6, v=0.3 to state s7 and v=1.0 to state s8. We regard the value as the reward constraints of states. The state s9 is the exit which means a user exit from the Web system, the users must return to login page once exit, and so the transition probability from s9 to s0 is 1.0.

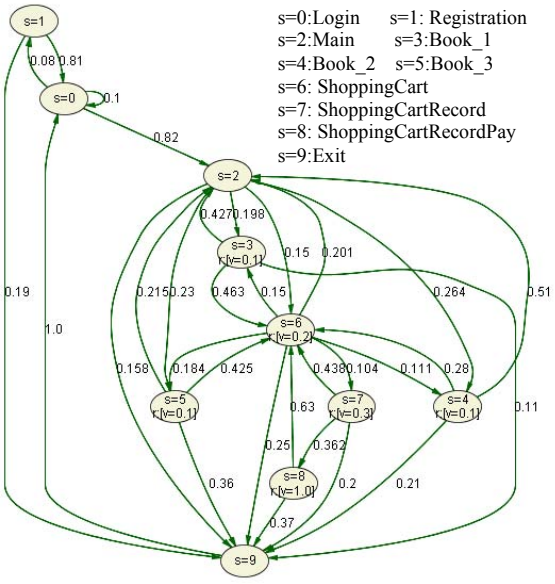

Fig. 3.Part of the value model in for Group 5

In order to calculate the cumulative value of the system by tool PRISM, we take the value of services as the reward constraints of states in Discrete Time Markov Chain (DTMC). The value model of Fig. 3 can be described with DTMC language as follows.

```
dtmc
module m

s : [0..9] init 0;

[] s=0 -> 0.1:(s'=0) + 0.08:(s'=1) + 0.82:(s'=2) ;
[] s=1 -> 0.81:(s'=0) + 0.19:(s'=9) ;
[] s=2 -> 0.198:(s'=3) + 0.264:(s'=4) +0.23:(s'=4) + 0.15:(s'=6)+ 0.158:(s'=9) ;
[] s=3 -> 0.427:(s'=2) + 0.463:(s'=6)+ 0.11:(s'=9);
[] s=4 -> 0.51:(s'=2) + 0.28:(s'=6)+ 0.21:(s'=9) ;
[] s=5 -> 0.215:(s'=2) + 0.425:(s'=6)+ 0.36:(s'=9);
[] s=6 -> 0.201:(s'=2) + 0.15:(s'=3)+ 0.111:(s'=4)+ 0.184:(s'=5)+ 0.104:(s'=7) + 0.25:(s'=9) ;
[] s=7 -> 0.438:(s'=6) + 0.362:(s'=8)+ 0.2:(s'=9) ;
[] s=8-> 0.63:(s'=6) + 0.37:(s'=9) ;
[] s=9-> 1.0:(s'=0);
endmodule
rewards "v"

s=1 : 0.8;
s=3 : 0.1;
s=4 : 0.1;
s=5 : 0.1;
s=6 : 0.2;
s=7 : 0.3;
s=8 : 1.0;

endrewards

label "exit"=s=9;
```

D. Quantitative Verification

In this part, we forecast survivability of the Web system by comparing expected value and simulation value. The expected value is very corresponding to the historical value statistics and requirements. In this paper, we count total benefits which is produced by a group in a certain period of time and regard the average benefits of every user as the expected value, and the expected value of Group 5 is 0.57 from the historical data statistics.

We use probabilistic model checker PRISM to calculate the simulation value of models, the corresponding logic property can be described as “filter (sum, R{ “v” } =? [F(“exit”)], ”init”)”, which shows the expected cumulative value from state “login” to “exit”. Fig. 4 shows the calculation result of Group 5 which means the system would gain 0.54 dollar in expectation when one user of Group 5 visits the system. However, the expected value of users in Group 5 is 0.57, so we conclude that the system under the usage pattern of Group 5 is not survivable at that certain period of time, which also means the survivability of uses in Group 5 in some degree.

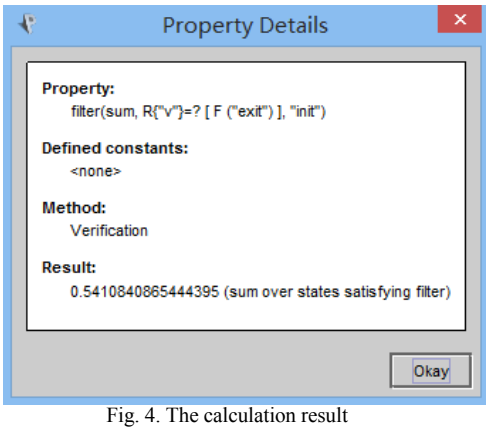

Fig. 4. The calculation result

Table 4 shows the corresponding results of each group and the overall Web system. Each group is a particular usage pattern of the whole system and can represent some corresponding services, so we believe our real-time results are very useful for reliability analysis and services dynamic adjustment.

TABLE 4.Survivability Prediction Results of the Bookstore

	Gr.1	Gr.2	Gr.3	Gr.4	Gr.5	Overall
Simulation Value(M)	1.79	0.61	1.48	2.13	0.54	1.43
Expected Value(R)	1.74	0.60	1.64	2.57	0.57	1.76
Suv.(P)	Y	Y	N	N	N	N

V. CONCLUSION AND FUTURE WORK

Web services QoS prediction is a significant concern for many researchers. The main contribution of this paper is proposing an approach to estimating the particular survivability for each user group. We use Euclidean distance to measure the similarity between users and adapt K-Means algorithm to partition Web log entries into session groups, and construct usage model and value model for each group. We use PRISM to calculate the simulation value of value models, and compare expected value with the simulation value to predict the survivability of the Web system under a particular usage pattern. The system will be not survivable if the simulation value below the expected value. So we believe it is time to make an appropriate adjustment to ensure the Web system can gain most benefits and make sure the core services are running well.

For the future works, we plan to optimize our value assessment approach and explore other efficient clustering algorithm. What is more, we believe the survivability for each user group has important reference values for making dynamic adjustment for Web services, but we do not provide a strategy for adjustment in this paper, and this is also one part of our future work.

REFERENCES

[1] Lei Z P, Qiu H P, Yang Z C. Research on Web services Adjustment Strategy[J]. Computer Technology & Development, 2014.

[2] Moitra S D, Konda S L. A Simulation Model for Managing Survivability of Networked Information Systems[R]. Pittsburgh,USA: [s. n.], Tech. Rep.: CMU/SEI-2000-TR-020, 2000.

[3] Han J, Kamber M, Pei J. Data mining: concepts and techniques[M]. Elsevier, 2011

[4] Shao L S, Li Z, Zhao J F, et al. Web services QoS Prediction Approach[J]. Journal of Software, 2009, 20(8):2062-2073.

[5] Malak J S, Mohsenzadeh M, Seyyedi M A. Web services QoS Prediction Based on Multi Agents[C]// Proceedings of the 2009 International Conference on Computer Technology and Development - Volume 01. IEEE Computer Society, 2009:265-269.

[6] Honghao Gao. Web services dynamic adaptive configuration based on probability model checking [D]. Shanghai: Shanghai University. 2012.

[7] Zhou J, Miao H, Kai J, et al. Survivability prediction of Web system based on log statistics[C]//Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015 16th IEEE/ACIS International Conference on. IEEE, 2015: 1-6.

[8] Jha S, Wing J, Linger R C, et al. Survivability Analysis of Network Specifications[C]// Dependable Systems and Networks, 2000. DSN 2000. Proceedings International Conference on. IEEE, 2000:613-622.

[9] Bihuan Chen. Software system adaptive method based on demand and system structure [D]. Shanghai: Fudan University. 2014.

[10] J. C. Knight and K. J. Sullivan. On the definition of survivability. Technical Report CS-TR-33-00, University of Virginia, Department of Computer Science, 2000.

[11] Zhou Q L, Zhang B, Lin X I. System Survivability Analysis Based on Model Checking[J]. Computer Engineering, 2012, 38(17):38-41.

[12] Deza M M, Deza E. Encyclopedia of distances[M]. Springer Berlin Heidelberg, 2009.

[13] Kallepalli C, Tian J. Measuring and modeling usage and reliability for statistical Web testing[J]. Software Engineering IEEE Transactions on, 2001, 27(11):1023-1036.

[14] Yan J, Wang J, Chen H W. Deriving Software Markov Chain Usage Model from UML Models[J]. Journal of Software, 2005, 16(8).

[15] Marta Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0: Verification of Probabilistic Real-time Systems. In Proceedings of the 23rd International Conference on Computer Aided Verification (CAV'11).Cliff Lodge, Snowbird, Utah, LNCS, Vol. 6806, 2011, 585-591.

[16] Sant J, Souter A, Greenwald L. An exploration of statistical models for automated test case generation[C]. Proceedings of the Third International Workshop on Dynamic Analysis, New York, USA: ACM Press, 2005: 1–7.

[17] Sprenkle S, Pollock L, Simko L. A Study of usage-based navigation models and generated abstract test cases for Web applications[C]. Proceedings of the Fourth IEEE International Conference on Software Testing, 2011: 230–239.