Research Article

A Dynamical Reliability Prediction Algorithm for Composite Service

Chunli Xie and Jianguo Ren

Department of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, China

Correspondence should be addressed to Chunli Xie; xcl_bhb@163.com

Received 31 March 2014; Accepted 4 June 2014; Published 6 July 2014

Academic Editor: Housheng Su

Copyright © 2014 C. Xie and J. Ren. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Dynamic selection and dynamic binding and rebinding at runtime are new characters of composite services. The traditional static reliability prediction models are unsuitable to dynamic composite services. A new reliability predicting algorithm for composite services is proposed in this paper. Firstly, a composite service is decomposed into composition unites (executing path, composite module and atomic service) according to their constituents. Consequently, a hierarchical graph of all composite units is constructed. Lastly, a new dynamic reliability prediction algorithm is presented. Comparing with the traditional reliability model, the new dynamic reliability approach is more flexible, which does not recompute reliability for all composite units and only computes the reliability of the effected composite units. In addition, an example to show how to measure the reliability based on our algorithm is designed. The experimental results show our proposed methods can give an accurate estimation of reliability. Furthermore, a more flexible sensitivity analysis is performed to determine which service component has the most significant impact on the improvement of composite service reliability.

1. Introduction

With the rapid development and widespread use of web services, more and more businesses are achieved by composite services which can be created by aggregating a number of atomic or composite services (called service components) by following certain composition pattern and rules. Thus, how to ensure the reliability of composite services has become an important problem. In addition, a large amount of function-equivalent services has been in the Internet, which make more services be selected. Users can translate their requirements for services to the constraints of some QoS properties [1]. Then, different users may select different service to realize the same function based on different QoS constraints. There are two methods for web service composition: static composition and dynamic composition. In static web service composition, composition is performed manually; that is, each web service is selected and bound before execution. Static composition is sufficient for constructing applications with well-defined specific requirements that are not likely to change frequently. But, it is not flexible and agile enough

when the requirements are changed frequently. Dynamic web service composition can achieve this goal as, in dynamic composition, services components are selected at runtime from a set of service components and when the selected service components failed, the new selected service components will replace them while keeping the running services constantly available to users [2]. Dynamic service composition provides an ability to rapidly adapt to changes that were not envisioned during design time.

Dynamic binding and rebinding are the basis for the dynamic composition. There are three kinds of binding: early binding, local binding at runtime, and rebinding at runtime [3–6]. In early binding, the service is selected and bound before execution, which is also called static binding. In local binding, when a composite service is running, one of its service components is bound to a selected service according to some QoS properties before it is invoked. Local binding can help service components to avoid binding to failed services and reduce the failure probability of composite services. During the execution of a composite service, when the QoS

of some service components do not satisfy the requirements of the user or the bound services fail, the composite service will pause, and after new services are bound to the service components, the composite service continues to run; the binding in this process is called rebinding at runtime.

To address the above problems of dynamic composition, traditional static reliability model to predict the reliability is not applicable to services with dynamic composition. A new dynamic reliability model is presented for the dynamic service composition in this paper. The new model makes the computation simple when the reliability of some service components is changed frequently.

The remaining sections of the paper are organized as follows. Section 2 describes the related works. Section 3 discusses some concepts about composite units. Section 4 describes the reliability of the composite units. Section 5 presents the dynamic reliability prediction approaches based on the hierarchical graph. Section 6 conducts some experiments to verify our reliability algorithm. Section 7 presents the conclusion and our future works.

2. Related Works

The existing reliability predicting models can be divided into two categories: black-box and white-box [7, 8]. The black-box models, such as the Bayesian models, the SRGM (software reliability growth models), and the statistical models, only consider the functionality of software while ignoring its internal structure. The white-box models estimate the reliability of a software system based on its internal structure. The structures are often described by Markov chain [9–19] or Petri net [20, 21]. These white-box models can be further divided into three categories: state-based models, path-based models, and additive models. As these models are often developed for component-based software systems, they are also called component-based reliability models (CBRM). The state-based models which are the common models usually model the system as a discrete-time Markov chain (DTMC), a continuous-time Markov chain (CTMC), or a semi-Markov process Markov chain (SMP). Wang et al. presented a typical state-based model which described four common architecture styles, including batch-sequential, parallel/pipeline-filter, call-and-return, and fault-tolerance styles, and proposed a transformation from architecture view to state view to perform reliability analysis [10]. S. Gokhale et al. classified the state-based models from three aspects: the architecture of the application, the failure behavior of components, and the analysis methods. The architectures were CTMC or DTMC and the Markov chain was irreducible or absorbing. The failure behavior was represented by reliability, constant failure rate, or time-dependent failure intensity. The analysis method was composite or hierarchical. In sum, they gave seven kinds of DTMC-based models and CTMC-based models.

The emergency of web services makes the software system be dynamic discovery and binding, which make the development and the execution be more complex. Then, web services are more prone to failure than traditional distributed software. It is necessary to develop new methods to predict

the reliability of services [17–19]. Tsai et al. extended the components based reliability models to services reliability analysis [20, 21]. The structure of services is also described as Markov chain. Wang et al. proposed a hierarchical reliability model which divided the reliability of services into data reliability, service components reliability, service pool reliability and composite services reliability [22, 23]. Zheng and Lyu proposed a collaborative reliability prediction approach, which employs the past failure data of other similar users to predict the web service reliability for the current user [24, 25]. This method needs large amount of user data; when the number of user is small, it is difficult to classify a user to its similar group. The composition process of web services is actually a workflow process; then a workflow-based model is also a usual method to predict the reliability [26]. Moreover, there are also other approaches to estimate the reliability of composite services from the ports, message delivery, or network transition [27–31]. Web services are different from general software system, as they are dynamic composition. Most of the existing reliability models are based on the service components in the design phase, without considering the dynamic selection and binding. In this paper, we would propose a new dynamic approach to predict the reliability of services.

3. Composite Units

Composite services are divided into *atomic service*, *composite unit*, *execution path*, and *composite service* according to their different constituents. We will discuss the four composite units in detail.

3.1. Atomic Service (AS). An atomic service is the smallest organization unit of web service. In other words, it would be impractical to decompose an atomic service. An atomic service has a fine-grained structure which can be designed, implemented, and tested independently. A composite service is a composite process; in dynamical composition, the service component in composite process is called abstract service. When the service is invoked, the abstract service is bound to the concrete service. There are several concrete services offering the same piece of functionality, which means that several concrete services may match an abstract service in service specification. So, the reliability of atomic service is changed in the execution process.

3.2. Basic Composite Module (BCM). Service composite module is composed of a set of services according to some operations. The constituted services may be atomic services, composite modules, or composite services. The operations are *sequence*, *choice*, *loop*, and *concurrent*. According to the operations, the service composite modules can be divided into four categories, that is, *sequence module (SM)*, *choice module (CM)*, *loop module (LP)*, and *concurrent module (CoM)*. If all services in a composite module are atomic services, the composite module is called a basic composite module (BCM). Let $BCM = \{Ass, OP\}$, Ass is the set of

FIGURE 1: Execution path graph.

atomic services, and OP is the operator of atomic services, $OP \in \{sequence, choice, loop, concurrent\}$.

3.3. Execution Path (Path). All services invoked sequentially are defined as an execution path when completing a task. Let $Path = \{S_1, S_2, \dots, S_n\}$, $S_i \in BCM$, $i = 1, 2, \dots, n$, and the S_i in $Path$ is sequential.

3.4. Composite Service (CS). The execution of a composite service is combined by several paths. Let $CS = \{\langle Path_1, P_1 \rangle, \langle Path_2, P_2 \rangle, \dots, \langle Path_n, P_n \rangle\}$, where $Path_i$ is the i th execution path, $i = 1, 2, 3, \dots, n$, and P_i is the execution probability of $Path_i$. An example of a composite service is shown in Figure 1; the composite service has two execution paths, $Path_1, Path_2$, $CS = \{\langle Path_1, P_1 \rangle, \langle Path_2, P_2 \rangle\}$. Services in an execution path of composite services may be atomic services, service module, or composite services; then they have $AS \in BCM \in Path \in CS$. A composite service includes several paths and the composite modules in a path may be BCM or other composite modules which have some $BCMs$. Any composite service can be finally divided into several execution paths consisting of several $BCMs$. We will prove the theorem.

Proposition 1. Any composite service can be finally divided into some execution paths which consist of several $BCMs$.

Proof. As a composite service constitutes several execution paths by the probability, suppose that a $Path_i$ includes another composite module A which consists of several atomic services and a BCM , $A = \{As_{a1}, As_{a2}, \dots, As_{an}, B, OP_A\}$ and $B = \{As_{b1}, As_{b2}, \dots, As_{bn}, OP_B\}$, where $OP_A \in \{sequence, loop, choice, concurrent\}$ and $OP_B \in \{sequence, loop, choice, concurrent\}$. We use $OP_A = sequence$ and $OP_B = sequence$ as an example to illustrate the problem. If we introduce composite module B into A , we will get $A = \{As_{a1}, As_{a2}, \dots, As_{an}, As_{b1}, As_{b2}, \dots, As_{bn}, sequence\}$. Then, a

composite module A can be deeply divided into $an + bn$ atomic services and the decomposition does not affect its execution path. If one of the operations is *choice*, the execution path $Path_i$ can be replaced by more execution paths. As OP_A, OP_B are one of the four operations, the execution path is not affected except when one of the operations is *choice*. One composite module can be decomposed into one or more $BCMs$, and the corresponding execution path can be extended into more execution paths. So, a composite service can be finally divided into some execution paths which consist of several $BCMs$. $\square$

3.5. Hierarchical Graph of Composite Units. In this section, we will describe a hierarchical graph to represent the composite unit: *composite service*, *path*, *basic composite module*, and *atomic service*, while the following section describes how to calculate the graph attributes.

Definition 2 (hierarchical graph). A hierarchical graph is defined by a 2-tuple $\langle V, E \rangle$, where $\langle V, E \rangle$ is a DAG (directed acyclic graph). V is a set of nodes in the graph, $V = \{v_i\}$, $i = 1 \dots |V|$, and E is a set of directed edges in the graph, $E = \{e_i\}$, $i = 1 \dots |E|$.

Definition 3 (node). A node $v \in V$ models a composite unit i and is defined by a 2-tuple $\langle N_i, R_i \rangle$, where N_i is the name of composite unit i and R_i is the reliability of composite unit i .

Definition 4 (edge). A directed edge $e \in E$ models the constitution of composite units v_i and is defined by the tuple $\langle v_i, v_j, w_{i,j} \rangle$, where v_i, v_j are nodes, the node v_j is one of the components of node v_i , and $w_{i,j}$ is the weight of edge from node v_i to v_j which describes the constitution of v_i and v_j . If $w_{i,j} < 1$, the edge represents the node v_i consisting of v_j according to *choice* structure, if $w_{i,j} = 1$, the edge represents the node v_i consisting of v_j according to *sequence* or *concurrent* structure, and if $w_{i,j} > 1$, the edge represents the node v_i consisting of v_j according to *loop* structure.

As shown in Figure 2, the node $\langle CS, R_1 \rangle$ is the composite service, the nodes $\langle Path_1, R_{path1} \rangle$ and $\langle Path_2, R_{path2} \rangle$ in level 2 are the execution paths of composite service CS , the nodes $\langle BCM_1, R_{BCM1} \rangle$, $\langle BCM_2, R_{BCM2} \rangle$, and $\langle BCM_3, R_{BCM3} \rangle$ in level 3 are the $BCMs$ of CS , the nodes $\langle AS_1, R_{AS1} \rangle$, $\langle AS_2, R_{AS2} \rangle$, $\langle AS_3, R_{AS3} \rangle$, $\langle AS_4, R_{AS4} \rangle$, and $\langle AS_5, R_{AS5} \rangle$ in level 4 are atomic services, and the nodes in the last level are the matched concrete services of atomic service AS_1 .

4. Reliability of Composite Units

4.1. Reliability of Atomic Services. An abstract service may have several concrete services to be matched; then, different user may select different concrete service to be bound for the same abstract service as they have different QoS requirement [24]. For example, there are an abstract service AS and three matched concrete services AS_1, AS_2 , and AS_3 . The reliability of the three concrete services is $\{0.98, 0.9, 0.75\}$, and one of the QoS $\{high, general, low\}$. The reliability and price of AS_1 are the highest, and AS_3 is the lowest. If a user only

FIGURE 2: Hierarchical graph of composite service.

cares about the reliability and has no claim for price, he may select AS_1 to be invoked. Other users may have different choice considering the price. The binding information can be recorded by monitor mechanism, and the binding probability can be computed from this information [27]. Here, we assume that the binding probability is known. As shown in Figure 3, the reliability of atomic service v_i can be computed by the following:

$$R_{AS} = R_i = \sum_{j=1}^n R_j * w_{i,j}, \quad (1)$$

where R_i is the reliability of an abstract atomic service node V_i , R_j is the reliability of v_j , $j = 1, \dots, n$, which is a concrete atomic service to be bound to v_i , and $w_{i,j}$ is the binding probability of node v_i to node v_j . After the binding changed, the reliability of atomic service AS must be recomputed according to the above formula.

4.2. Reliability of Basic Composite Module. Sequence module (SM), choice module (CoM), loop module (LM), and concurrent module (CoM) are four kinds of composite modules. Next, we will discuss their reliability in a dynamic environment.

4.2.1. Sequence Module/Concurrent Module. As shown in Figure 3(a), node v_i is SM or CM, nodes $v_1, \dots, v_n$ are the atomic services which compose the SM or CoM, and the weight of edge from node v_i to v_j , $j = 1, \dots, n$, is 1. The reliability of SM or CoM is

$$R_{SM} = R_{CoM} = R_i = \prod_{j=1}^n R_j. \quad (2)$$

In the equation, if the binding probability of an atomic service v_k is changed, then the reliability of v_i will be changed, and the reliability of SM will be changed. Let ΔR_k be the changed

FIGURE 3: Hierarchical graph of composite module.

value of v_k ; then the new value of node v_i can be represented by

$$\begin{aligned} R'_i &= \prod_{j=1, j \neq k}^n R_j (R_k + \Delta R_k) \\ &= R_i + \Delta R_k \prod_{j=1, j \neq k}^n R_j \\ &= \left(1 + \frac{\Delta R_k}{R_k}\right) R_i. \end{aligned} \quad (3)$$

From the equation, we can see that when the binding of a service has been changed, it does not need to recalculate the reliability of the entire services. As long as the changed value is calculated, the whole reliability can be known.

4.2.2. Choice Module. As shown in Figure 3(b), the weight of edge from node v_i to node v_j is less than 1; the composite module is CM. The reliability of CM can be computed by the following expression:

$$R_{CM} = R_i = \sum_{j=1}^n R_j w_{i,j}. \quad (4)$$

If a binding probability of an atomic service (e.g., v_k) is changed, the reliability of the atomic service must be recomputed, $R'_k = R_k + \Delta R_k$, where ΔR_k is the changed value of node v_k . Thus, the new reliability of CM can be expressed by the following:

$$\begin{aligned} R'_{CM} &= R'_i = \sum_{j=1, j \neq k}^n (R_j w_{i,j}) + \Delta R_k w_{i,k} \\ &= R_i + \Delta R_k w_{i,k} \end{aligned} \quad (5)$$

and $\Delta R_k = \Delta R_k w_{i,k}$.

4.2.3. Loop Module. As shown in Figure 3(c), the weight of edge from node v_i to node v_j is larger than 1, the composite module is LM, and the weight of edge is the number of times. The reliability of LM can be computed by the following expression:

$$R_{LM} = R_i = R_j^{w_{i,j}}. \quad (6)$$

If the binding probability of v_j is changed, ΔR_j is the changed value of node v_j . The new reliability of LM can be expressed by the following:

$$\begin{aligned} R'_i &= (R_i + \Delta R_j)^{w_{i,j}} \\ &= R_i \left(1 + \frac{\Delta R_j}{R_i} \right)^{w_{i,j}}. \end{aligned} \quad (7)$$

From (7), we also can see that when the reliability of an abstract service in LM is changed there is no need to recalculate all reliabilities of services in LM to get the reliability of the entire service. It is easy to get the reliability of the entire service as long as the changed value has been got. In a word, the incremental reliability of BCM is

$$\Delta R_i = \begin{cases} \frac{\Delta R_k}{R_k} R_i & \text{sequence/concurrent} \\ \Delta R_k w_{i,k} & \text{choice} \\ \left[\left(1 + \frac{\Delta R_k}{R_k} \right)^{w_{i,k}} - 1 \right] R_i & \text{loop.} \end{cases} \quad (8)$$

4.3. Reliability of Path. As shown in Figure 2, an execution path is a sequence structure of all its BCMs in the path. The weight of edge from execution paths to its BCMs is 1. So, the reliability of execution path is the same as SM:

$$\begin{aligned} R_{Path} &= R_i = \prod_{j=1}^n R_j, \\ R'_{Path} &= R_i + \Delta R_i, \end{aligned} \quad (9)$$

where $\Delta R_{Path} = \Delta R_i = (\Delta R_k / R_k) R_i$.

4.4. Reliability of Composite Service. The execution of a composite service can be divided into several paths, and

```
(1) {
(2) double R = Ri;
(3) for (vj = vi.firstneighbor();
(4) vj != NULL; vj = vi.nextneighbor())
(5) if (visited[vj] == false);
(6) {
(7) if (wi,j < 1)
(8) R = R + CalRel(G, vj) * wi,j;
(9) else if (wi,j = 1)
(10) R = R * CalRel(G, vj) * wi,j;
(11) else R = R * (CalRel(G, vj))wi,j;
(12) }
(13) return R;
(14) }
```

ALGORITHM 1: Double CalRel (graph G, vertex v_i).

each path is executed according to a probability. Assume that node v_0 is a composite service CoS in Figure 2 that has several execution paths, $Path_1, \dots, Path_n$; their execution probabilities are $P_1, \dots, P_n$ and their nodes are $v_1, v_2, \dots, v_n$. Then, the reliability of composite service CS can be calculated by

$$R_{CS} = R_0 = \sum_{i=1}^n R_i w_{0,i}. \quad (10)$$

If the reliability of $Path_k$ has been changed, and ΔR_k is the changed value of $Path_k$, the new reliability of composite service can be recalculated similar to the CM:

$$R'_{CS} = R'_0 = R_0 + \Delta R_0, \quad (11)$$

where $\Delta R_{CS} = \Delta R_0 = \Delta R_k w_{0,k}$ and ΔR_0 is the changed value of composite service.

5. Reliability Prediction of Composite Service

Composite services can be decomposed into three layers, that is, *path* layer, *BCM* layer, and *atomic service* layer. Thus, the reliability of composite services can be calculated by the reliability of *path*, *BCM*, and *atomic service*. In this section, we will discuss how to compute the reliability of composite services.

5.1. Reliability Algorithm for Composite Service. The nodes in different layer in Figure 2 represent different composite units. We assume that the reliability of atomic services is known, and the reliability of other nodes can be derived from the reliability of their next layer nodes. We can get all the reliabilities of nodes from bottom to up. The algorithm CalRel(G, v) calculates the reliability of all nodes in the hierarchical graph of the composite service CS (Algorithm 1). The parameter G is the hierarchical graph and v_i is a node, $\langle v_i, v_j \rangle \in E$, whose initial value is

$$R_i = \begin{cases} 0 & w_{i,j} < 1 \\ 1 & w_{i,j} \geq 1. \end{cases} \quad (12)$$


```

(1) {
(2) double  $\Delta R = 0$ ;
(3) for ( $v_j = v_i.firstneighbor()$ ;
(4)  $v_j \neq NULL$ ;  $v_j = v_i.nextneighbor()$ )
(5) if (!visited[ $v_j$ ]);
(6) {  $\Delta R_j = CalDeltaRel(G', v_j)$ 
(7) if ( $w_{i,j} < 1$ )
(8)  $\Delta R = \Delta v_j w_{i,j}$ ;
(9) else if ( $w_{i,j} = 1$ )
(10)  $\Delta R = \Delta v_j / R_j R_i$ ;
(11) else
(12)  $\Delta R = [(1 + \Delta v_j / R_j)^{w_{i,j}} - 1] R_i$ ;
(13) }
(14) return  $\Delta R$ ;
(15) }

```

ALGORITHM 2: Double *CalDeltaRel* (graph G' , vertex v_i).

When the algorithm is invoked, parameter v_i is the composite service node v_0 . The calculation process is the traverse process and all reliabilities of nodes are calculated in traverse of graph. Then the time complexity is the same as the traverse process, which is $O(n + e)$, n is the number of nodes, and e is the number of edges.

5.2. Dynamic Reliability Algorithm for Composite Services. If the reliability of an atomic service is changed, the composite service will be affected. In this section, we will discuss how to perform our dynamic reliability model to evaluate the reliability of composite services. Not all composite units are affected by the atomic service; we only recalculate the reliability of the influenced nodes. For example, in Figure 4, the nodes with bold line are the affected nodes. The first step is to find all affected nodes, for example, CS, *Path*₁, *Path*₂, *Path*₃, *Path*₄, *BCM*₁, and *AS*₁ in Figure 4. Then, all the affected nodes are extracted to a subgraph G' in Figure 5. We present an algorithm *CalDeltaRel* to calculate the reliability of the affected nodes (Algorithm 2). The parameter G' is the subgraph and v_i is a node in subgraph G' . Variable R_i is the reliability of node v_i . The returned result (ΔR_{CS}) is the incremental reliability of a composite service and $R'_{CS} = \Delta R_{CS} + R_{CS}$.

6. Case Study

6.1. Experimental Setup. We demonstrate the reliability prediction model with a typical example of a service for travel. In this example, the composite consists of 11 atomic services, and we assume that their reliability is known. As shown in Figure 6 the composite service can be divided into 4 execution paths and 7 basic composite modules. Its hierarchical graph is Figure 7.

6.2. Comparison of Prediction Accuracy. The prediction reliability of the composite units in travel service according to formula (2) to (10) is shown in Table 1.

FIGURE 4: Hierarchical graph of composite service.

FIGURE 5: Subgraph of composite service.

To study our method performance, we compare our prediction approach with the user-object reliability model (UORM). As shown in Figure 8, success and failure states are added to the state diagram of UORM, the transition matrix is P , and the reliability is

$$R_{UORM} = R_{END}(I - Q)_{1,n}^{-1} = 0.422. \quad (13)$$

From Table 1, we can see that the result of our approach is the same as that of UORM.

6.3. Dynamic Reliability Prediction. If the reliability of atomic service AS_3 is changed, the affected composite units are

FIGURE 6: Travel service.

TABLE 1: Reliability of composite units.

Composite unit	R
AS ₁	0.9
AS ₂	0.9
AS ₃	0.9
AS ₄	0.85
AS ₅	0.85
AS ₆	0.9
AS ₇	0.9
AS ₈	0.85
AS ₉	0.9
AS ₁₀	0.85
AS ₁₁	0.95
BCM ₁	0.5267
BCM ₂	0.9
BCM ₃	0.9
BCM ₄	0.85
BCM ₅	0.9
BCM ₆	0.85
BCM ₇	0.95
path ₁	0.4029
path ₂	0.4266
path ₃	0.4029
path ₄	0.4503
CS	0.422

BCM₁, Path₁, Path₂, Path₃, and Path₄. According to (11), we have

$$\Delta R_{CS} = \sum_{i=1}^4 (\Delta R_{Path_i} w_{0,i}),$$

$$\begin{aligned} \Delta R_{Path_i} &= \frac{\Delta R_{BCM1}}{R_{BCM1}} R_{Path_i}, \\ \Delta R_{BCM1} &= \frac{\Delta R_{AS3}}{R_{AS3}} R_{BCM1}. \end{aligned} \quad (14)$$

If $\Delta R_{AS3} = 0.02$, the new reliability of composite service v_0 is $R'_{CS} = R_{CS} + \Delta R_{CS} = 0.41298$. The computing process is very simple, but, according to UORM, the transition matrix P' must be rebuilt. The new reliability $R'_{UORM} = R_{END}(I - Q)^{-1}_{1,n} = 0.41298$. From the results of two methods, we can see that our dynamic approach is simpler than UORM, especially when the atomic services are not fixed; then our approach is more applicable to dynamic web services.

6.4. Sensitivity Analysis. Sensitivity analysis helps to understand the impact of the reliability of service components with respect to the system or composite service reliability [28]. The sensitivity of composite service reliability based on our approach with respect to a given atomic service AS_i can be defined as

$$C_i = \frac{\Delta R_{CS}}{\Delta R_{ASi}},$$

P

$$= \begin{bmatrix} Q & D \\ 0 & I \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0.158 & 0.369 & 0 & 0 & 0 & 0 & 0 & 0 & 0.474 \\ 0 & 0 & 0 & 0.54 & 0.36 & 0 & 0 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0.45 & 0.45 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0.15 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0.15 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.95 & 0 & 0.05 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

P'

$$= \begin{bmatrix} Q' & D' \\ 0 & I \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0.154 & 0.360 & 0 & 0 & 0 & 0 & 0 & 0 & 0.474 \\ 0 & 0 & 0 & 0.54 & 0.36 & 0 & 0 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0.45 & 0.45 & 0 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0.15 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9 & 0 & 0.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.85 & 0 & 0.15 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.95 & 0 & 0.05 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

(15)

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is supported partially by the National Natural Science Foundation of China under Grant nos. 61304174 and 61304117.

References

- [1] V. Agarwal and P. Jalote, "From specification to adaptation: an integrated QoS-driven approach for dynamic adaptation of web service compositions," in *Proceedings of the IEEE 8th International Conference on Web Services (ICWS '10)*, pp. 275–282, Miami, Fla, USA, July 2010.
- [2] I. Satoh, "Spatial connector: loosely binding contextual changes and non-context-aware services," in *Proceedings of the 8th International Joint Conference on Software Technologies*, pp. 50–57, Reykjavik, Iceland, 2013.
- [3] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, "A framework for QoS-aware binding and re-binding of composite web services," *Journal of Systems and Software*, vol. 81, no. 10, pp. 1754–1769, 2008.
- [4] G. Canfora, M. D. Penta, R. Esposito et al., "Service composition (re)binding driven by application-specific QoS. Service-oriented computing," in *Proceedings of the 4th International Conference Service-Oriented Computing*, pp. 141–152, Chicago, Ill, USA, December 2006.
- [5] G. Canfora, M. di Penta, R. Esposito, and M. L. Villani, "QoS-aware replanning of composite web services," in *Proceedings of the IEEE International Conference on Web Services (ICWS '05)*, pp. 121–129, Orlando, Fla, USA, July 2005.
- [6] M. D. Penta, R. Esposito, M. L. Villani, R. Codato, M. Colombo, and E. Di Nitto, "WS-Binder: a framework to enable dynamic binding of composite web services," in *Proceedings of the International Workshop on Service-Oriented Software Engineering*, pp. 74–80, Jhongli, Taiwan, December 2008.
- [7] K. Goševa-Popstojanova and K. S. Trivedi, "Architecture-based approach to reliability assessment of software systems," *Performance Evaluation*, vol. 45, no. 2-3, pp. 179–204, 2001.
- [8] S. S. Gokhale, "Architecture-based software reliability analysis: overview and limitations," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 1, pp. 32–40, 2007.
- [9] R. C. Cheung, "A user-oriented software reliability model," *IEEE Transaction on Software Engineering*, vol. 6, no. 2, pp. 118–125, 1980.
- [10] W.-L. Wang, D. Pan, and M.-H. Chen, "Architecture-based software reliability modeling," *Journal of Systems and Software*, vol. 79, no. 1, pp. 132–146, 2006.
- [11] S. S. Gokhale and K. S. Trivedi, "Analytical models for architecture-based software reliability prediction: a unification framework," *IEEE Transactions on Reliability*, vol. 55, no. 4, pp. 578–590, 2006.
- [12] Q. Yu, X. Liu, A. Bouguettaya, and B. Medjahed, "Deploying and managing Web services: issues, solutions, and directions," *Very Large Data Bases Journal*, vol. 17, no. 3, pp. 537–572, 2008.
- [13] C. Li, X. Fu, Q. Tian, W. Wang, and Y. Xia, "Optimal reliability allocation with minimum cost for Web service composition," in *Proceedings of the 25th Chinese Control and Decision Conference (CCDC '13)*, pp. 58–63, Guiyang, China, May 2013.
- [14] R. Albu and F. Popentiu-Vladicescu, "A prototype for web services reliability prediction," *International Journal of Information and Communication Technology*, vol. 5, no. 1, pp. 64–74, 2013.
- [15] X. Liu and J. Cao, "Robust state estimation for neural networks with discontinuous activations," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 40, no. 6, pp. 1425–1437, 2010.
- [16] X. Liu and J. Cao, "On periodic solutions of neural networks via differential inclusions," *Neural Networks*, vol. 22, no. 4, pp. 329–334, 2009.
- [17] R. Yingxin, Q. Gu, Q. Jingxian, and D. Chen, "Reliability prediction of web service composition based on DTMC," in *Proceedings of the 3rd IEEE International Conference on Secure Software Integration Reliability Improvement (SSIRI '09)*, pp. 369–375, Shanghai, China, July 2009.
- [18] G. Fan, H. Yu, L. Chen, and D. Liu, "Petri net based techniques for constructing reliable service composition," *Journal of Systems and Software*, vol. 86, no. 4, pp. 1089–1106, 2013.
- [19] Y. N. Xia, X. Luo, J. Li, and Q. S. Zhu, "A petri-net-based approach to reliability determination of ontology-based service compositions," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 43, no. 5, pp. 1240–1247, 2013.
- [20] W. T. Tsai, D. Zhang, Y. Chen, H. Huang, R. Paul, and N. Liao, "A software reliability model for web services," in *Proceedings of the 8th IASTED International Conference on Software Engineering and Applications*, pp. 144–149, MIT Cambridge, November 2004.
- [21] V. Grassi and S. Patella, "Reliability prediction for service-oriented computing environments," *IEEE Internet Computing*, vol. 10, no. 3, pp. 43–49, 2006.
- [22] L. Wang, X. Bai, L. Zhou, and Y. Chen, "A hierarchical reliability model of service-based software system," in *Proceedings of the 33rd Annual IEEE International Computer Software and Applications Conference (COMPSAC '09)*, pp. 199–208, Seattle, Wash, USA, July 2009.
- [23] Y.-S. Dai, Y. Pan, and X. Zou, "A hierarchical modeling and analysis for grid service reliability," *IEEE Transactions on Computers*, vol. 56, no. 5, pp. 681–691, 2007.
- [24] Z. B. Zheng and M. R. Lyu, "Collaborative reliability prediction for service-oriented systems," in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*, pp. 35–44, Cape Town, South Africa, May 2010.
- [25] Z. B. Zheng and M. R. Lyu, "Personalized reliability prediction of web services," *ACM Transactions on Software Engineering and Methodology*, vol. 22, no. 2, article 12, 2013.
- [26] L. Coppolino, L. Romano, N. Mazzocca, and S. Salvi, "Web services workflow reliability estimation through reliability patterns," in *Proceedings of the 3rd International Conference on Security and Privacy in Communication Networks (SecureComm '07)*, pp. 107–115, Nice, France, September 2007.
- [27] H. Zo, D. L. Nazareth, and H. K. Jain, "End-to-end reliability of service oriented applications," *Information Systems Frontiers*, vol. 14, no. 5, pp. 971–986, 2012.
- [28] Z. H. Ding, M. Y. Jiang, and A. Kandel, "Port-based reliability computing for service composition," *IEEE Transactions on Services Computing*, vol. 5, no. 3, pp. 422–436, 2012.
- [29] C. J. Prakash, P. M. Rohini, R. B. Ganesh, and V. Maheswari, "Hybrid reliability model to enhance the efficiency of composite web services," in *Proceedings of the IEEE International*

Conference on Emerging Trends in Computing, Communication and Nanotechnology (ICE-CCN '13), pp. 79–83, Tuticorin, India, March 2013.

- [30] S. Hwang and C. Lee, “Reliable Web service selection in choreographed environments,” *Decision Support Systems*, vol. 54, no. 3, pp. 1463–1476, 2013.
- [31] M. Silic, G. Delac, and S. Srbljic, “Prediction of atomic web services reliability based on K-means clustering,” in *Proceedings of the 9th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pp. 70–80, Saint Petersburg, Russia, August 2013.