A State-Space Approach for Service-oriented SLA Translation

Yong-ming Yan, Bin Zhang, Jun Guo

School of Computer Science and Engineering, Northeastern University, China
yym_sy@163.com, zhangbin@mail.neu.edu.cn, guojun@mail.neu.edu.cn

Abstract

Interaction among components leads to the complication of a service system. In order to satisfy the consults between users and service providers to get the relation between upper targets and lower objectives, Services Level Agreement (SLA) translation is introduced. Previous works often introduce a state-space approach for devices in SLA translation. But little work has been carried out in extending the SLA translation towards the service composition. In this paper, based on the state-space Formulation to model the complex service system, we present a service component framework to optimize resource allocations. In particular, we propose a clustering analysis method, which effectively reduces the analyzing cost, to integrate the variables of service composition. We characterize and analyze the service component to obtain the ranges of their variables meeting SLA, which can be used to conduct resource allocations optimization. A comprehensive set of experiments is finally carried out to demonstrate the effectiveness and efficiency of our proposed approach.

Keywords: SLA, Service composition framework, State-space approach, Cluster analysis

1 Introduction

Service Level Agreements (SLAs) are formalized contracts between service providers and service consumers that are used to define quality of service (QoS) properties [1-2]. And technologies have been created to support autonomous establishment and maintenance of service level agreements in order to guarantee end-to-end quality of service requirements [3-5]. SLAs will guarantee QoS for the customers in SLA commitment contract, and they can simplify the process of negotiation between service consumers and service providers.

The target Service Oriented Architecture (SOA) consists of four main layers [6], namely Operational Resources, Software Components, Business Services and Business Processes. Because there are different criteria for different layers, the translation of standards and parameters through a hierarchy or from one level to another level in a multi-layered SOA environment are called SLA translation [7-8]. It is known as SLA decomposition to convert Service Level Objectives (SLOs) of high-level to policy/system threshold of lower levels, and the results of these translations can be used to design system and monitor system [9]. How to implement the behavior above is one of the major problems of the SLA translation research.

SLA translation methods mainly include three categories: a method based on knowledge and rules, a method based on the queuing analysis model and a method based on statistical learning. The first one is mainly used in network, which is difficult to be built and managed, and the method cannot adapt to changes in the system very well. The second one adopts the queuing network model. Analytical performance model is a powerful mathematical tool, but it is limited in practical ways as a result of its simplified assumptions. The third method is based on statistical learning probability model of reasoning. Cohen et al. [10] have used reasoning probability model to get the relationship between system level merits and SLOs. High level targets include transaction throughput and response time, etc. Statistical methods such as Bayesian network is better than queuing network model. The smaller domain knowledge and black box method they deployed can be widely used for handling conversion problems between the layers.

Multiple devices are needed to accomplish a service, and the service system becomes complicated due to the interaction of equipment. In order to optimize resources and services, we introduce SLA translation based on service compositions to analyze the factors of implement of services and affection of the QoS, as well as quantification of the affects. Kumar et al. [9] proposed a kind of low-level extracted from top target index method based on TAN. They deployed state-space modeling the complicated system. In particular, they divide the state-space variables by equipment, and just analyze each device variables meeting SLA. However, they omit the relationship of variables associated with service compositions, which is significant for the resource and service optimization.

*Corresponding Author: Yong-ming Yan; E-mail: yym_sy@163.com

DOI: 10.3966/160792642018011901019

We integrate different variables, which are completing one service into a service component, with the help of service composition. By achieving the translation of SLA, we can obtain scopes of the service component variables meeting SLA to determine the relationships among variables. Since it will take high energy consumption to discover relationships among the variables by analyzing the whole system without any refinement, we refine the whole system into service components as the sub-spaces. Further, we characterize the sub-spaces and estimate the factors which affect the system performance.

The contributions of this paper can be summarized as deploying the SLA translation method based on a state-space approach in terms of service composition. In this paper, we integrate variables for each service component. And we take a data analysis for each service component and extract the target of the service component level which matches high-level metric.

The rest of the paper is organized as follows. We first introduce something related to solve our problem in Section 2. In Section 3, we will describe the overall framework about our system. The main arithmetic is used to handle the specific questions in Section 4. And then Section 5 is dedicated to get the component SLO. Experimental evaluation is given in Section 6, followed by conclusion in Section 7.

2 Related Work

A customer is more inclined to request a statistical bound on its response time than an average response time. Therefore, Xiong et al. [11] have been concerned with a percentile of the response time that characterizes the statistical response time. And more users tend to ask the response time could meet the SLA. At the same time, the majority of users focus on the situation of every type of service components meeting SLA. During the study, therefore, we need to get decomposition of SLA requirements from the whole system SLA, and the SLA requirements are for all kinds of service components. Service components performances are monitored in the process of running.

Cohen et al. have considered a more sophisticated approach based on statistical modeling and inference to construct these signatures [12]. They have presented a method that successfully clusters system states corresponding to similar problems. And we also considered sophisticated approach to solve our problems. So we deploy Fuzzy Clustering Analysis Method based on Variables to integrate service variables, and analyze service SLA translation in terms of a service component. It is different from previous research on service property analysis. Most of the existing approaches deal with a small subset of system and application level metrics, we focus on service components.

Currently, many approaches are provided to solve the problem of managing server energy and operational cost. Chen et al. [13] have proposed formalism to this problem, and three new online solution strategies based on steady state queuing analysis, feedback control theory, and a hybrid mechanism borrowing ideas from these two. These solutions are adaptive to workload behavior when performing server provisioning and speed control than earlier heuristics towards minimizing operational costs while meeting the SLAs. Meanwhile, one of the most interesting challenges introduced by web services is represented by Quality of Service (QoS)-aware composition and late-binding. They have showed that QoS-aware composition can be modeled as an optimization problem. Gerardo et al. [14] have adopted Genetic Algorithms to this aim. Facing the similar target of resource optimization, a variant of the Bayesian network called Tree Augmented Naive Bayes or TANs is used to probabilistically model the system state space [15]. It completes the SLA translation to get the lower level objectives. We deploy statistics analysis to acquire the high accuracy in data.

Compared with naive Bayes classifier, a classic neural network model prediction effect classifier (based on RBF and Euclidean distance) produces low classification accuracy around 80% [16] about variables belonging to the set meeting SLA or the set not meeting SLA. Based on neural network learning algorithm, how to improve the learning performance of neural networks is currently an important research problem [17]. With the help of Tree Augmented Naive Bayes, we can improve the classification accuracy, but statistical methods have strict requirement for input data.

In SOA environments, service providers need to comply with the service level objectives to satisfy the contracts with customers while minimizing the resource costs. The problem has been formalized by using system identification and control theory. And control-oriented system identification approached is promising to model the system with great incoming workload changes in business. Linear Parameter Varying (LPV) state space system identification algorithms [18] are analyzed for modeling web services system.

State space method assumes that the evolution being studied of the system over time can be determined by a non-observation vector sequence, accompanying with a sequence that can be observed, and the relationship between them can be identified through the state-space model [19]. The target of state-space analysis is to infer about the nature of the unobservable variables from the observation sequence. The state of a system (such as Vi, moment i = 1, 2, 3,…) can be defined as binding appropriate values of variables, which is included in the set V. For a system or device, a measure or influencing metric or factor can be modeled into a state-space variable.

Violations of service level objectives (SLO) in

Internet services are urgent conditions requiring immediate attention. In this system, we implement system SLA conversions to get the scopes of component SLA. When meeting the SLA violation, the sub_model can adjust itself to remove the SLA violation. A similar method was adopted by Steve Zhang et al. They previously explored an approach [20] for identifying which low-level system properties were correlated to high-level SLO violations. For solving the problems, their approach was based on automatically inducing models from data using pattern recognition and probability modeling techniques. And then, they also showed that the ensemble of models captures the performance behavior of the system accurately, which to analyze the possible cause of the performance behavior. Compared with the method of pattern recognition and probability modeling in [20], we analyze the system in the view of state space, it can reveal the relationship between the inner variables and the outer variables better, and may find the unknown crucial character. The system is easier to be observed. It doesn't improve the complexity of system model when the amount of state variables, input variables or output variables are increasing, in other words, it is more controllable.

3 Overall Framework

The whole problem can be divided into four steps to get the service component level targets from the whole system. Firstly we will model the system and define SLA for the system. System space can be denoted by using a set with n variables [21], such as V={v_1, …,v_n}. We can use a sub_set V_Φ to decide whether the system is conforming to SLA or not, and V_Φ is a set whose variables are in running condition.

The system SLA can be defined as a set with m SLOs [15] of service components. Each SLO is denoted by a tuple (o_i, $V_{\omega(i)}$). o_i means the objective criterion of each service component needed, $V_{\omega(i)}$ is a set containing the variables and their states whether they meet the standard or not. ω(i) is mapping from the objective to variables of V_Φ. So the SLA can be expressed by the formulation:

$$L_{V\varphi} = \{(o_1, V_{\omega(1)}), \cdots, (o_m, V_{\omega(m)})\} \quad \textbf{(1)}$$

And Eq. (2) can tell if the whole system conforms to SLA by judging all service components. If f (o_i, $V_{\omega(i)}$) returns true, it means values of $V_{\omega(i)}$ corresponding the objectives, or opposite.

$$F(L_{V\varphi}) = \overset{m}{\underset{i=1}{\wedge}} f(o_i, V_{\omega(i)}) \quad \textbf{(2)}$$

In large and complicated enterprise system, it is a little hard to study the variables of V_Φ, so we simplify the question in sub_models with SLOs. In this case, each SLO can also be divided into independent objectives of independent variables. We can apply the following expression Eq.(3) to represent SLO(o_i, $V_{\omega(i)}$), i=1, ..., m.

$$SLO\,\{(o_1^i, V_{\omega'(1)}), \cdots, (o_k^i, V_{\omega'(k)})\} \subset SLA(V_\gamma) \quad \textbf{(3)}$$

SLA(V_γ) is the targets of set V_γ and V_γ is the set containing the independent variables. As Eq. (4) showing, only when all variables are meeting the objectives we defined, the whole service component is satisfying the component SLO.

$$f(o_i, V_{\omega(i)}) = \overset{k}{\underset{j=1}{\wedge}} f(o_j^i, V_{\omega'(j)}) \quad \textbf{(4)}$$

Component level objectives essentially depend on the value ranges of variables in set V_γ with SLA V_Φ in condition of V running state. This paper can evaluate all variables in set V, set V_Φ with SLA, and sub-space set V_γ having the variables easy to control. After confirming these variables, an enterprise system can use our approach to monitor the system variables, partition the variables by devices and integrate them into the service components by service compositions. Just as the figure 1 showed:

After defining SLA, we partition the state-space by service components, as shown in Figure 1. We integrate the corresponding space variables into a sub-space according to the similarity between variables in service composition. And then a sub_model is established for the sub-space to be analyzed according to Bayesian probability model. Based on the above works, we calculate the sub_SLA.

As for the partitioning by devices, we just simplify the question and record it in a set D that will be explained in Section 4. Main works focus on integrating variables of service composition and probabilistic system modeling analysis in the following section. Based on the upper level SLA metrics, we get allowed ranges of variables to show the relationship between one variable and another. The above formulations are used to restrain the designed system meeting SLA. Meanwhile, we will adjust sub_models under the condition of making the whole system meeting SLA constantly when SLA violations turn up or the sub_models need changes.

4 Main Arithmetic

All involved devices are recorded in a set D (d_1,…, d_p), p is the number of devices, d_p is one device we need to monitor. We put corresponding variables into sub_spaces by each device, and apply a tuple (v_i, d_j) to record every state variable. The tuple is representing that the variable v_i belongs to device d_j. This step can be one aspect to guide resource provisioning and deploying to gain the best exploitation, but this works would not be done in my paper. Then we use corresponding algorithm to integrate related variables

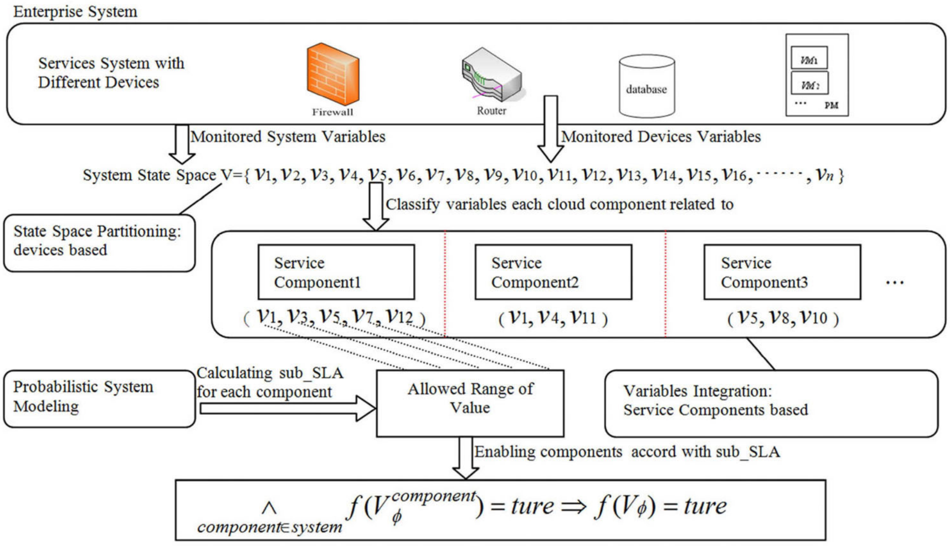
Enterprise System
Services System with Different Devices
Firewall
Router
database
VM1
VM2
...
PM
Monitored System Variables
Monitored Devices Variables
System State Space V={ $v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}, v_{16}, \cdots\cdots, v_n$ }
Classify variables each cloud component related to
State Space Partitioning: devices based
Service Component1
($v_1, v_3, v_5, v_7, v_{12}$)
Service Component2
(v_1, v_4, v_{11})
Service Component3
(v_5, v_8, v_{10})
...
Probabilistic System Modeling
Calculating sub_SLA for each component
Allowed Range of Value
Variables Integration: Service Components based
Enabling components accord with sub_SLA
$\underset{component \in system}{\wedge} f(V_\phi^{component}) = ture \Rightarrow f(V_\phi) = ture$

Figure 1. Overall structure

of one service component and build the sub-space model.

4.1 Service Component Variables Integration

Spatial data mining seeks to discover meaningful patterns from data where a prime dimension of interest is geographical location. A novel variable resolution approach [22] has been presented to cluster discovery which acts in the first instance to define spatial concentrations within the data thus allowing the nature of clustering to be defined. In PLS approach [23], it is frequently assumed that the blocks of variables satisfy the assumption of unidimensionality. In order to fulfill at best this hypothesis, they use clustering methods of variables. Fuzzy Clustering Analysis Method based on Variables (FCAMV) [24], the method integrates every variable one service composition related according to the similarity between the state variable and another. Now we define one state variable as v_j є V_Φ, as the following:

$$v_j = (x_{1j}, x_{2j}, \cdots, x_{mj}), \; j = 1, 2, \cdots, n \tag{5}$$

Here, x_{ij} represents the value that the variable gets in i second. The cosine of vi and vj can be used as similarity coefficient, as shown in Eq. (6). But before using the similarity coefficient to calculate the similarity between one variable and another, the values of variables are needed to be done with normalizing process, just as Eq. (7) and Eq. (8) showing. Eq. (7) is for a nominal variable or an ordinal variable, and m is the number of monitored values of one variable. Eq. (8) is for interval-scaled variables. They are normalized by the rate between the different distances to the minimum.

$$r_{ij} = \sum_{k=1}^{n} x_{ki} \cdot x_{kj} \Big/ \sqrt{\sum_{k=1}^{n} x_{ki}^2 \sum_{k=1}^{n} x_{kj}^2} \qquad i = 1, \cdots, n, \; j = 1, \cdots, n \tag{6}$$

$$z_{ij} = (x_{ij} - 1)/(m-1) \qquad i = 1, \cdots, m; \; j = 1, \cdots, n \tag{7}$$

$$z_{ij} = \left| x_{ij} - \min x_{ij} \right| \Big/ (\max x_j - \min x_j) \qquad i = 1, \cdots, m; \; j = 1, \cdots, n \tag{8}$$

As for our question, some variables we monitored are continuous variables as time is changing, such as the search depth. The ranges of their values are continuous and there is a maximum value and a minimum value, so they should be calculated by Eq. (8). But for other variables, they can be calculated by Eq. (7), because their values are discrete.

If having calculated every statistic r_{ij} (i=1,2,…,n, j=1,2, …,n) for the variables, the similarity relation R of state-space V is ascertained, as denoted by Eq.(9). n is the number of the variables in V.

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{bmatrix} \tag{9}$$

And then we need deploy transitive closure to construct Matrix equivalence:

$$R \to R^2 \to R^4 \to \cdots \to R^{2^k} = t(R), \text{ k=1, 2, …} \tag{10}$$

The Eq. (10) shows that a value of k will make the set R are an equivalent matrice, which is used to classify the variables set V_Φ. The matrix of fuzzy similar relation R must be a fuzzy equivalence relation. The variables are classified in accordance with R_λ, and λ gets value from [0, 1]. The clusters will differ with different λ. The work makes our question have different particle-sizes for space partition. Consequently, we set λ a appropriate value to integrate the variables into several clusters, and we adopt one as the subspace V_σ. In other words, more related variables will be more possible to be in the cluster.

For example, the method FCAMV is used to evaluate a small-sized service system. Through an analysis about service performance variables, services attributes field consists of Accuracy, Cache refresh time and so on. We use these attributes to form an attribute set AU, AU= {*Accuracy*, *Cache-refresh-time*, *Searching-depth*, *Distribution-server*, *Response-time*, *Throughput*, *Concurrent-users*}. In other way, AU={x_1, x_2, x_3, x_4, x_5, x_6, x_7}. After determining the theory domain and putting the records of all variables into a data matrix, we count on Eq. (7) and Eq. (8) to normalize the data, so that all kinds of variables can be calculated together.

The similarity relation R can be got by Eq. (6) and Eq. (9):

$$R=\begin{bmatrix} 1.00 & 0.63 & 0.71 & 0.65 & 0.73 & 0.68 & 0.78 \\ 0.63 & 1.00 & 0.79 & 0.88 & 0.68 & 0.78 & 0.66 \\ 0.71 & 0.79 & 1.00 & 0.79 & 0.79 & 0.76 & 0.70 \\ 0.65 & 0.88 & 0.79 & 1.00 & 0.80 & 0.87 & 0.75 \\ 0.73 & 0.68 & 0.79 & 0.80 & 1.00 & 0.84 & 0.89 \\ 0.68 & 0.78 & 0.76 & 0.87 & 0.84 & 1.00 & 0.82 \\ 0.78 & 0.66 & 0.70 & 0.75 & 0.89 & 0.82 & 1.00 \end{bmatrix}$$

We need to apply transitive closure to construct equivalence matrix t(R):

$$t(R)=R^2=R^4$$

Here it is a transitive closure approach to solve the cluster question, and we can also adopt maximal tree [25]. But we just use one of them to handle our problems.

$$t(R)=\begin{bmatrix} 1.00 & 0.78 & 0.78 & 0.78 & 0.78 & 0.78 & 0.78 \\ 0.78 & 1.00 & 0.79 & 0.88 & 0.84 & 0.87 & 0.84 \\ 0.78 & 0.79 & 1.00 & 0.79 & 0.79 & 0.79 & 0.79 \\ 0.78 & 0.88 & 0.79 & 1.00 & 0.84 & 0.87 & 0.84 \\ 0.78 & 0.84 & 0.79 & 0.84 & 1.00 & 0.84 & 0.89 \\ 0.78 & 0.87 & 0.79 & 0.87 & 0.84 & 1.00 & 0.84 \\ 0.78 & 0.84 & 0.79 & 0.84 & 0.89 & 0.84 & 1.00 \end{bmatrix}$$

After these works, we set up different values of threshold λ to get different clusters, as the Table 1 shows.

Table 1. Clustering result

λ value [0,1]	Variables of the cluster
0.78	{x_1, x_2, x_3, x_4, x_5, x_6, x_7}
0.87	{x_2, x_4, x_5, x_6, x_7}
0.88	{x_5, x_2, x_4, x_7}
0.89	{x_5, x_7}

We know that the cluster includes all variables when the value of λ is a value from 0 to 0.78. And when the value of λ is 0.89, the cluster just contains x_5, x_7, and it means that

Response-time has a deep relationship with *Concurrent-users* in attribute set AU. Meanwhile, *Response-time* and *Concurrent-users* are related to our question about service performance. As a result, we put the variables into the following set that we will study.

4.2 Subspace Model Construction

We construct the subsystem model for sub-space V_σ, called sub_model, with the help of Tree Augmented Naive Bayesian Classifier (TANC). This method is establishing in U= {A_1, …, A_n, C}. A_1, …, A_n are discrete attribute variables, and C is a class variable. The following Figure 2 shows the relation between the variables. It is one example of TAN, and the probability of A_2 appearing is affected by A_1 in contrast to Naive Bayesian [7].

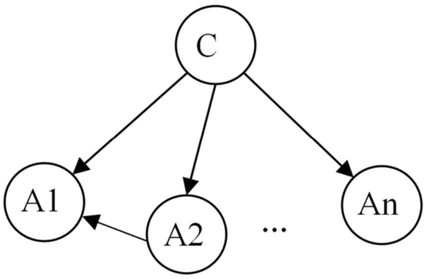

Figure 2. Tree augmented naive Bayesian

As for sub_model, this paper can achieve SLA translation with sub_model. That is to say, we let service component variables satisfy lower level target and it also means getting the value ranges of variables meeting SLA. The TAN method returns LV_γ of sub_SLA and a probability p. LV_γ represents the state vector of a system state satisfying sub_SLA, and p is credibility of getting SLA LV_γ. The state can get the acceptance of LV_γ by comparing p with threshold [15]. The specific steps are as follows:

(1) The values of each monitored variable are put into training data set V_θ. The return of the system-state function F (LV_Φ) serves as class variable c (including SLA-satisfying and SLA-dissatisfying). We monitor the whole system variables to get the values with a time Sequence, and the values of all variables in V_σ at one point form a state variable V_i={a_1, a_2, ..., a_d}, i=1, ..., d. Properties a_1, a_2, ..., a_d are from training data set V_θ. These variables are the input of the following Eq. (11). According to the formula, we can judge whether the system state V_i conforms to SLA or not, and get the probability [26].

$$\begin{aligned} p &= P(c \mid a_1, a_2, \cdots, a_d) \\ &= \frac{P(c) \cdot P(a_1, a_2, \cdots, a_d \mid c)}{P(a_1, a_2, \cdots, a_d)} \\ &= P(c) \prod_i P(a_i \mid c) \end{aligned} \quad \textbf{(11)}$$

(2) For ensuring the SLA, it needs to determine the value ranges of variables in V_σ. We search the set V_σ for sub_model, and come out all possible value ranges of controllable variables, denoted by the set V^σ:

$$V^\sigma = \{V_1^\sigma, \cdots, V_d^\sigma\} \quad \textbf{(12)}$$

Then the above set is placed with the pregenerated ones into the current system-state set Vnow, and we create a set of possible system state $\{V_1, \ldots, V_d\}$. In this paper, class variable c needs to be adjusted to meet SLA, and we estimate probability p for each possible system-state. When p is bigger than the threshold κ, the state reaches the credibility of corresponding to SLA through judging by TANC, and the state will be recorded in Vκ. The state can be also to decide sub-SLA of the service component. If a SLA violation is detected or Vnow needs to change sub_model, the sub-SLA has to be recalculated.

5 Component SLO

When we have acquired the value set V_κ that all component variables values are meeting SLA, what we should do next is to catch the sub-SLA (SLO) for each component. The controllable variables are isolated to search the appropriate ranges for service components. We should make each controllable variable be independent with each other according to the service component and the scopes of the variables values. Every variable and its range of values constitute the sub-SLA (SLO) of each component. The emphasis is searching the independent value ranges in V_κ.

When there is only one controllable variable or one state variable for the sub-space, we need no considering the question. In V_κ, every value of the controllable variable can be expressed as a point [15], and then we connect all the points together, so that we can get a clique about each variable. At the same time, we can also obtain some cliques between the value sets of variables. Afterwards we choose the clique with the biggest value of expression χ, as shown in Eq. (13):

$$\chi = \prod_{j=1}^{|V_\sigma|} N_j \quad \textbf{(13)}$$

N_j is the number of values of variable $v_{j\sigma}$ ($v_{j\sigma} \in V_\sigma$) appearing in the clique. When χ gets the biggest values, the set of values appearing can be as the acceptance values range of component variable if corresponding to selected clique.

In Figure 3, there are three related variables, which are part of our whole variables set. The variables have overlapping value sets, which reflect the relation between controllable variables. Only if the value can make the variable meet the SLA, we put the value into its value set to get the biggest value of χ. As Figure 3 shown, variable A has 9 values and C has 12, and they have a common value, which means they have association with each other. There are three values of other variables except for the three variables in the picture. As for our system, it has another variables these possess their values. We regard the controllable variables and the ranges of their own values as the component SLO.

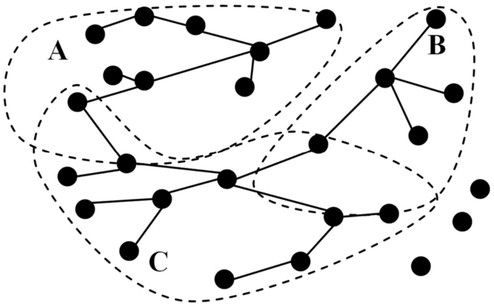

Figure 3. Values of Controllable Variables

Note that, we can restrain the uncontrollable variables by binding the easy controllable variables in order to ensure the service component meeting sub-SLA. If all components are sacrifying their own SLOs, the whole system will be meeting SLA. In other words, if we want to make the service system meet the SLA we defined, all service components have to meet their own SLOs. We have the following SLA conversion algorithm sub_SLA ().

Algorithm 1 returns V_T (v_i, S_i), i=1, …, n, which is the scopes of variables to make Eq. (13) get the big value. Si is the values range of variable vi under condition of meeting SLA. At line 1, we monitor service composition attributes, which contain response time, equipment throughput and so on. Each attribute is defined as a variable, which to form a state vector V= (v_1, v_2, …, v_n). And then we monitor the system to get a time series of state variables V_i, i=1, …, m. Every element is the value of j variable in the moment i. At line 2, we define the class variable C (c_1, c_2). c1 is conforming to the SLA, while c2 does not accord with the SLA. For example, we can define the SLA as {response time v1 ⩽ t μm}. At line 7, we put the parameters into the Bayesian formulation Eq.(10) and get the probability p of each state Vi. At line 8, When p ⩾ κ, the confidence of Vi belonging to c1 is big enough that Vi is meeting SLA, and then Vi will be put into set Vα; When p < κ, Vi is not meeting SLA. At line 13, the scopes of variables are isolated respectively according to that χ gets the biggest value.

Algorithm 1 sub_SLA ()

INPUT: State variable V, Credibility threshold κ, response time t, the number of variables n, the number of time sequence m;

OUTPUT: thresholds values set V_T(vi, Si) containing ranges of all controllable variables;

```
01 Initialization: $V_i = (v_1^i, \cdots, v_n^i)$ ; i = 1;
02                 C= {$c_1$, $c_2$}; $c_1$=1; $c_2$=0;
03 Monitor the values of the state variables;
04 Put them into data set D;
05 For i=1 to m
06     For j=1 to n
07         p = P($c_1$| $v_1^i, \cdots, v_n^i$)= P($c_1$) Π P( $v_j^i$ |$c_1$);
08     if p >= κ then
09         Put $V_i$ into data set V;
10 For k=1, i=1 to M, n
11     if $\alpha_k$ belongs to $v_i$ then
12         $N_i$++;
13 For L=1 to |$V_\sigma$|
14      $\chi = \chi * N_L$;
15 If $\chi$ is the biggest value then
16      return $V_T$;
17 Else Goto(10);
```

6 Experimental Design and Analysis of Results

6.1 Experimental Setup

For testing our designed system, we setted up the hardware and software environment. As for hardware system, we setted up 5 Emulab(a network testbed, which is opening for universities and research institution) stations with Intel Pentium Dual CPU E2200@2.20GHz; Memory 2GB; Disc 120GB. Emulab is designed to provide unified access to a variety of experimental environments. It provides a web-based front end through which users create and manage experiments, a core which manages the physical resources within a testbed, and numerous back ends which interface to various hardware resources [27]. So the system architecture consists of hardware assembly, software system and state machine. There are test nodes, servers, switch, and router and so on, and the applications are running within the Xen [28]. The software use opensource codes, and we setted up kernel-XEN-2.6.18-53.e15.i686.rpm in our devices. We used a application system RUBiS with 4 VMs, one supporting Apache server, two supporting Tomcat server and the left one supporting MySQL server instance. The 4 VMs are placed in 4 different sites. RUBiS client and monitoring program were placed in the fifth site [15]. Due to its low performance loss, XEN gradually becomes one of the most popular virtual management tools. But its SEDF scheduling algorithm under the SMP does not solve global load balancing problem. Data connection used Gigabit Ethernet links.

Workload means users' requests. EPA-HTTP web traffic [29] is used to trace data we needed to generate workload for the RUBiS applications, and the data is from the LBL Repository. We traced users' requests in an hour. Custom monitoring program were being used to monitor Apache server, Tomcat server and MySQL server. It can return their state values (Apache_status, Tomcat_status and Mysql_status). About one hundred Monitored state variables of all equipment included response time, user concurrency, throughput, memory, CPU share, request rate and so on. And we put all the monitored variables values into training data set.

We setted up the parameters λ and κ. Data were constantly taken from the training data set. We analyzed the substitution algorithm, and obtained the experimental results. And then we designed a program of integrating service component variables and used a program of Bayesian classifier to classify variable scopes. With these procedures we dealt with the data to verify the usability and effectiveness of the system analysis model.

6.2 Experimental Results

The following experiments illustrate the stability and the validity of the proposed approaches. We first investigate the effect of variation in workload. We monitored the variables number of users' requests per minute in an hour and reflect the result in the Figure 4. As the Figure 4 shows, the number of users' requests is 1,487 in the 5th minute. And it becomes 9,011 in the 31th minute. So we can draw a conclusion that the number is more than 90 around the 32th minute, which means the system pressure was heavier than other time. But in the ten minutes, the number of user requests is floating in 1,500 up and down, which reflects the light load for the designed system. This allows us to see the operation condition of different load pressure system. On the basis we carried out the other experiment to study our problem.

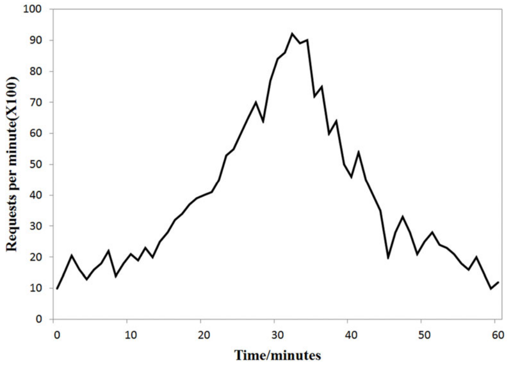

Figure 4. Workload: requests per minute

Secondly, we study the changes in number of cluster variables to search the correlative variables of a service composition, which affects the cost of analysis. In the beginning, we setted up the value of λ. After cluster analysis with a special value of λ, we then got the result and reflect it in Figure 5. As shown in Figure 5, when λ is 0.2, the number of variables is 79. And when λ is 0.9, it decreases to 6. We can see that the variables number of the cluster decreases as λ is close to 1. The cluster we needed will keep too many variables if λ is too small, and some variables may affect service quality lightly. But it will leave too fewer research variables if λ is too big. So we choose one service variables set that containing 5 or 6 key variables (such as response time, Concurrent users, CPU share, memory share, throughput rate) to study. In conclusion, when integrating service component variables, if λ has different values, we can get the different numbers of clustering. In order to achieve high classification accuracy, we should select a proper λ value.

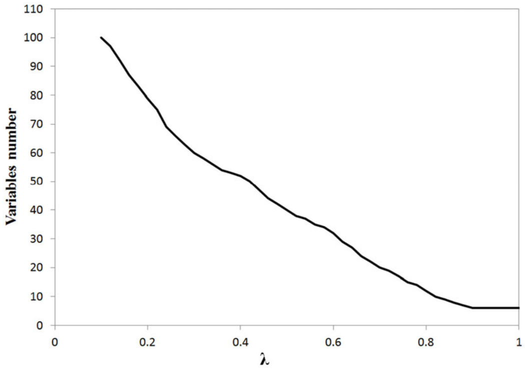

Figure 5. Variables cluster

Thirdly, the changes in classification accuracy were calculated in our experiments. The test illustrated that our approach was more effective and high accuracy than previous works for analyzing service composition. When doing the SLA classification experiments in sub_model, the threshold κ was setted up different values. The result is shown in Figure 6. From the figure, we observe that when the value of threshold κ is 0.75, classification accuracy is 82.4%. When the κ value is 0.9, classification accuracy is 93.3%. Through the experiments we concluded the different influence that different thresholds κ have on probabilistic model. As shown in Figure 6, classification accuracy of the probability model is greater as the value of threshold κ is increasing.

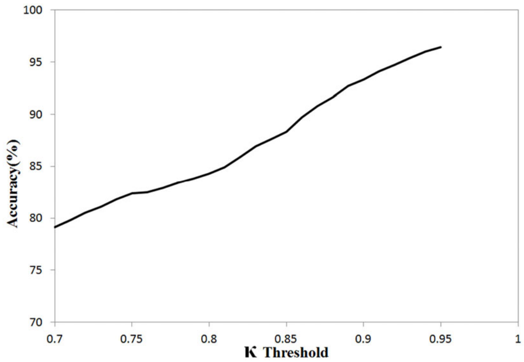

Figure 6. Classification accuracy

Finally, we monitored the changes in SLA violations rate. In 1 minute we load a pulse, and then it will turn up more SLA violations. In a experiment that contained our design system, when a SLA violation turned out, we need to adjust the sub_model in accordance with SLA to eliminate the violation. In Table 2, we recorded the result of our experiments. We drew from experiments that SLA violation had accounted for 15.5% in all request-responses system without our designed system, while the non-violations are just 74.5%. But with our designed system, the rate of SLA violation is 0.1%, which is nearly 0. From the experiments, we can find the proportion of SLA violations with or without the designed system. Because we design the system that can automatically adjust it when meeting SLA violation, so service SLA violation ratio reduces.

Table 2. Service SLA violation ratio changes with or without the design system

Service	With the system	Without the system
SLA violation	0.1%	15.5%
SLA non-violation	99.9%	74.5%

7 Conclusion

The service-oriented SLA translation based on state-space approach models and analyzes the complex service system. And the system level SLA conversion is accomplished for the component level. In other words, it analyzes the relations between the controllable variables and the uncontrollable to determine the scopes of the respective thresholds. The naive Bayesian classification algorithm is used to handle the problem about classifying state variables according to whether they are meeting SLA or not. Compared with neural network method, naive Bayesian classification has strict requirements on the input data, but the classification accuracy is higher. The goal of integrating state variables by equipment or service composition is analyzing specific problems of a component affecting the quality of service. Through the thresholds of the variables of service composition, we can find the state variables with big influence. The experiments prove the method is effective

Acknowledgments

This work was supported by the National Natural Science Foundation Program of China (No.61572117, 61370155, 61300019), the Special Fund for Fundamental Research of Central Universities of Northeastern University (No.N140406002), the National key Techonlogy R&D Program of the Ministry of Science and Technology (2015BAH09F02) and the Provincial Scientific and Technological Project (2015302002).

References

[1] A. M. Abbas, O. Kure, Quality of Service in Mobile Ad Hoc Networks: A Survey, *International Journal of Ad Hoc and Ubiquitous Computing*, Vol. 6, No. 2, pp. 75-98, July, 2010.

[2] M. Varela, P. Zwickl, P. Reichl, M. Xie, H. Schulzrinne, From Service Level Agreements (SLA) to Experience Level Agreements (ELA): The Challenges of selling QoE to the user, *2015 IEEE International Conference on Communication Workshop*, London, England, 2015, pp. 1741-1746.

[3] A. Rufini, M. Mellia, E. Tego, F. Matera, Multilevel Bandwidth Measurements and Capacity Exploitation in Gigabit Passive Optical Networks, *IET Communications*, Vol. 8, No. 18, pp. 3357-3365, December, 2014.

[4] Y. Gao, J.-J. Duan, W.-N. Shu, A Novel Ant Optimization Algorithm for Task Scheduling and Resource Allocation in Cloud Computing Environment, *Journal of Internet Technology,* Vol. 16, No. 7, pp. 1329-1338, December, 2015.

[5] V. Saritha, V. M. Viswanatham, Approach for Channel Reservation and Allocation to Improve Quality of Service in Vehicular Communications, *IET Networks*, Vol. 3, No. 2, pp. 150-159, June, 2014.

[6] U. Zdun, C. Hentrich, W. M. P. Van Der Aalst, A Survey of Patterns for Service-Oriented Architectures, *International Journal of Internet Protocol Technology*, Vol. 1, No. 3, pp. 132-143, May, 2006.

[7] J.-H. He, X.-Y. Bai, R.-L. Li, Z.-S. Cui, Service Oriented Infrastructures Based on SLA, *Telecommunication Engineering*, Vol. 51, No. 9, pp. 100-105, September, 2011.

[8] H. Li, *Challenges in SLA Translation*, SLA@SOI European Commission Seventh Framework Programme (2007–2013) SAP Research, December, 2009.

[9] Y. Chen, S. Iyer, X. Liu, D. Milojicic, A. Sahai, Translating Service Level Objectives to Lower Level Policies for Multi-tier Services, *Cluster Computing*, Vol. 11, pp. 299-311, July, 2008.

[10] I. Cohen, M. Goldszmidt, T. Kelly, J. Symons, J. S. Chase, Correlating Instrumentation Data to System States: A Building Block for Automated Diagnosis and Control, *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*, Berkeley, CA, 2004, pp. 1-16.

[11] K.-Q. Xiong, H. Perros, Service Performance and Analysis in Cloud Computing, *2009 World Conference on Services-I*, Los Angeles, CA, 2009, pp. 693-700.

[12] I. Cohen, S. Zhang, M. Goldszmidt, J. Symons, T. Kelly, A. Fox, Capturing, Indexing, Clustering, and Retrieving System History, *Proceedings of the twentieth ACM symposium on Operating systems principles*, New York, NY, 2005, pp. 105-118.

[13] Y.-Y. Chen, A. Das, W.-B. Qin, A. Sivasubramaniam, Q. Wang, N. Gautam, Managing Server Energy and Operational Costs in Hosting Centers, *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, New York, NY, 2005, pp. 303-314.

[14] G. Canfora, M. D. Penta, R. Esposito, M. L. Villani, An approach for QoS-aware Service Composition Based on Genetic Algorithms, *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, New York, NY, 2005, pp. 1069-1075.

[15] V. Kumar, K. Schwan, S. Iyer, Y. Chen, A. Sahai, A State-space Approach to Sla Based Management, *IEEE Network Operations and Management Symposium*, Salvador, Brazil, 2008, pp. 192-199.

[16] S. Ghosh-Dastidar, H. Adeli, N. Dadmehr, Principal Component Analysis-enhanced Cosine Radial Basis Function Neural Network for Robust Epilepsy and Seizure Detection, *IEEE Transactions on Biomedical Engineering*, Vol. 55, No. 2, pp. 512-518, February, 2008.

[17] S.-L. Hung, H. Adeli, A Parallel Genetic/neural Network Learning Algorithm for MIMD Shared Memory Machines, *IEEE Transactions on Neural Networks*, Vol. 5, No. 6, pp. 900-909, November, 1994.

[18] M. Tanelli, D. Ardagna, M. Lovera, L. Zhang, Model Identification for Energy-aware Management of Web Service Systems, *Proceedings of the 6th International Conference on Service-Oriented Computing*, Berlin, Germany, 2008, pp. 599-606.

[19] J. Durbin, S. J. Koopman, *Time Series Analysis by State Space Methods*, Oxford University Press, 2012.

[20] S. Zhang, I. Cohen, J. Symons, A. Fox, Ensembles of Models for Automated Diagnosis of System Performance Problems, *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, Washington, DC, 2005, pp. 644-653.

[21] S.-W. Liang, H.-P. Lu, T.-K. Kuo, A Study on Using the Kano Two-Dimensional Quality Model to Evaluate the Service Quality of Government Websites, *Journal of Internet Technology*, Vol. 15, No. 2, pp. 149-162, March, 2014.

[22] A. J. Brimicombe, A Variable Resolution Approach to Cluster Discovery in Spatial Data Mining, *Proceedings of the 2003 International Conference on Computational Science and Its Applications*, Heidelberg, Germany, 2003, pp. 1-11.

[23] V. Stan, G. Saporta, Conjoint Use of Variables Clustering and PLS Structural Equations Modeling, *Handbook of Partial Least Squares*, pp. 235-246, November, 2009.

[24] M. Verma, M. Srivastava, N. Chack, A. K. Diswar, N. Gupta, A Comparative Study of Various Clustering Algorithms in Data Mining, *International Journal of Engineering Research and Applications*, Vol. 2, No. 3, pp. 1379-1384, June, 2012.

[25] L. A. Zadeh, Fuzzy Sets, *Information and control*, Vol. 8, No. 3, pp. 338-353, November, 1964.

[26] P. A. Flach, N. Lachiche, N. Bayesian Classification of Structured Data, *Machine Learning,* Vol. 57, No. 3, pp. 233-269, December, 2004.

[27] D. Johnson, T. Stack, R. Fish, D. M. Flickinger, L. Stoller, R. Ricci, Mobile Emulab: A Robotic Wireless and Sensor Network Testbed, *25th IEEE International Conference on Computer Communications*, Barcelona, Spain, 2006, pp. 1-12.

[28] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, A. Warfield, Xen and the Art of

Virtualization, *ACM SIGOPS Operating Systems Review*, Vol. 37, No. 5, pp. 164-177, December, 2003.

[29] S. Kumar, V. Talwar, V. Kumar, P. Ranganathan, K. Schwan, vManage: Loosely Coupled Platform and Virtualization Management in Data Centers, *Proceedings of the 6th international conference on Autonomic computing*, New York, NY, 2009, pp. 127-136.

Biographies

Yong-ming Yan received his MS degree in software engineering from the Northeast University in 2007. Now he is a PhD candidate at the School of Computer Science and Engineering, Northeastern University. His current research interests include cloud computing and service optimization.

Bin Zhang received his PhD degree in computer science from the Northeastern University. Now he is full professor of Northeastern University. His current research interests include service computing, cloud computing and web engineering.

Jun Guo received his PhD degree in computer science from the Northeastern University. Now, he is an associate professor of Northeastern University. His main research interests include software testing and optimizing performance on the cloud.