International Journal of Innovative Computing, Information and Control
Volume **16**, Number **4**, August **2020**

pp. 1409–1423

CONSTRAINTS DECOMPOSITION AND CLUSTERING BASED WEB SERVICES COMPOSITION WITH UNCERTAIN QOS

Hengzhou Ye, Fengyi Huang and Zhidan Hu
Guangxi Key Laboratory of Embedded Technology and Intelligent System
Guilin University of Technology
No. 319, Yanshan Street, Yanshan District, Guilin 541006, P. R. China
2002018@glut.edu.cn

Received March 2020; revised June 2020

Abstract. *Global QoS (Quality of Service) constraint decomposition is an important strategy for service composition, and QoS aggregation calculation is the key to service optimization. A constraint strength aware global QoS constraint decomposition model is proposed by introducing a relaxation factor that can be adjusted adaptively according to fuzzy reasoning rules. The improved K-means algorithm is adopted to avoid the combined explosion problem in the process of QoS aggregation calculation. Simulation experiments show that the constraint strength aware global QoS constraint decomposition model can effectively improve the success rate of service composition when the constraint strength is high, and can significantly improve the effect of reducing solution space when the constraint strength is low. The accuracy and time cost of clustering based QoS aggregation are better than the existing methods.*
Keywords: Services composition, Uncertain QoS, QoS constraint decomposition, QoS aggregation, Clustering

1. **Introduction.** Web services are highly promising for the implementation of the Service-Oriented Computing (SOC) paradigm which has been widely used in many fields such as intelligent train maintenance [1], distributed manufacturing system [2], and fault prediction [3]. Since atomic Web services are often simple and difficult to address the complex needs of users or intricate business processes, creating value-added services by combining many associated Web services to deal with complex needs is an important and effective way to advance the field. With the increase of the number of Web services on the network, the Quality of Service (QoS) of Web services is more and more concerned by users; therefore, one of the key problems in service composition is how to select the services that can satisfy the global constraints and have the optimal global utility efficiently according to QoS. And when considering QoS-aware Web service composition, many studies [4-7] ignore the uncertainty of QoS. However, in an open, heterogeneous, and multi-tenant network environment, the measurement of QoS attributes of services is probabilistic, and is not suitable to be described by a known probability distribution [8].

Global QoS constraint decomposition is an important strategy of Web service composition, which is widely used in solving the problems in Web service composition [9,10]. However, the existing services composition method based on constraint decomposition has poor dynamic scenario adaptability [11]. When the user constraint strength is high [12-15], it is possible to find no feasible solution, and when the user constraint strength is low [16], the elimination effect is very limited and the solution space cannot be significantly reduced. In another aspect, it is not rigorous to describe the uncertainty of service

DOI: 10.24507/ijicic.16.04.1409

QoS with a specific probability distribution [17-19], and the uncertainty of QoS greatly increases the time complexity of QoS aggregation calculation. Therefore, QoS aggregation calculation of composite services is the core problem that QoS-aware service composition faces. Meanwhile, there exist the problems of combination explosion and low accuracy in the process of QoS aggregation calculation [20,21].

In this paper, to deal with the aforementioned problems, we focus on Web services composition with uncertain QoS as the research object. And we proposed a constraint strength-aware QoS constraint decomposition model by introducing a relaxation factor that can be adjusted adaptively according to fuzzy reasoning rules. We also adopt the improved K-means algorithm to avoid the problems of the combination explosion in the QoS aggregation calculation process. The simulation and experimental results show that the model can efficiently improve the success rate of service composition when constraint strength is high; otherwise, it can obviously improve the effect of reducing solution space. Meanwhile, cluster-based QoS aggregation calculation is superior to existing methods in accuracy and time overhead. The contributions of this article can be summed up as follows.

- We proposed a global QoS constraint decomposition model with user constraint strength awareness, and a fuzzy inference rule to adaptively determine the relaxation factor in the model.
- We adopted the clustering strategy to avoid the combination explosion problem in the QoS aggregation calculation process. By improving the K-means algorithm, the accuracy of the QoS aggregation calculation is greatly improved and the time overhead is dramatically reduced.

The rest of the paper is organized as follows. Related work is introduced in Section 2. Section 3 presents the problem statement and preliminaries. Section 4 introduces the details of the proposed algorithm. Section 5 displays experimental analysis. In the end, conclusions are shown in Section 6.

2. **Related Work.** Evolutionary algorithms such as chaos genetic algorithm [5] and particle swarm optimization [22] are adopted to solve QoS-aware service composition problems, but they are difficult to be applied to QoS uncertainty scenarios. The global QoS constraint decomposition strategy, which can transform the service composition problem into a local optimization problem or effectively reduce the solution space, is considered as an effective strategy. The existing global QoS constraint decomposition models can be divided into three categories. The first category is based on experience [9-11]. They are simple and intuitive, but have poor universality and cannot guarantee the global constraints. The second one can guarantee global constraints [12-15], which allows the use of local optimization strategies to ensure global QoS constraints during the service selection stage. However, during constraints decomposition phase, some feasible solutions may be lost and may result in no solution when the user constraint strength is high. The third one can reduce the solution space without losing feasible solutions, but cannot guarantee global constraints [16]. Moreover, when the user constraint strength is low, the reduction effect of solution space is not obvious. Therefore, it is essential to provide a new or revised global QoS constraint decomposition model with good effect under different user constraint strengths.

The calculation of QoS aggregation is one of the keys to solve the service composition problem by using global optimization strategies. Even if a global QoS constraint decomposition strategy is adopted to solve QoS-aware service composition problem, when the model cannot guarantee global QoS constraints, a global optimization strategy still needs to be adopted to satisfy global QoS constraints. The uncertainty of QoS increases the difficulty of constraint decomposition model to guarantee global QoS constraints, and also increases the difficulty and complexity of QoS aggregation. Some literature believes that the QoS of services obeys the normal distribution [17,18] to simplify the calculation of QoS aggregation. However, the response time of services does not obey any well-known probability distribution in [8], and it is not rigorous to describe the uncertainty of service QoS by using specific probability distribution through the investigation and analysis of service QoS in real society [19]. It is not accurate enough to use a few eigenvalues (for example, cloud model [20] is described by expectation, entropy and super entropy) to represent the uncertainty of QoS. In order to describe any distributed QoS, Hwang et al. [21] used Probability Mass Function (PMF) to describe the QoS of atomic or composite services, and then calculated the QoS aggregation of composite services. To solve the problem of combination explosion in the calculation process, they refined an Aggregate Random Variable Discovery (ARVD) problem and used dynamic programming and greedy algorithm to solve it. However, the accuracy of the scheme is low, and time cost is high. In [8], PMF is also used to describe the uncertainty of QoS. By specifying the locations and intervals of the sampling interval, the impact of the combination explosion problem is greatly reduced, and the time overhead and calculation accuracy are improved. However, this method cannot deal with multiplicative QoS aggregation calculations well. On the other hand, due to the strict parameter limitation, the method is difficult to balance the calculation accuracy and time cost when the problem size fluctuates greatly. Therefore, it is necessary to explore a more effective QoS aggregation method with uncertain QoS.

3. **Problem Statement and Preliminaries.**

3.1. **Statement of uncertain QoS service composition framework.**

The overall framework of uncertain QoS Web services composition based on global QoS constraint decomposition and clustering strategy is shown in Figure 1. Service composition problems involve many issues, such as user demand analysis, workflow modeling, service discovery

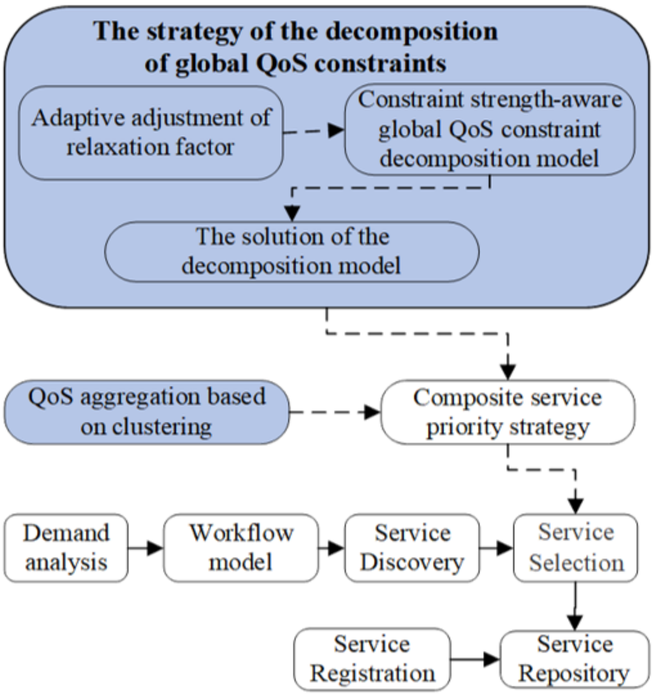

FIGURE 1. Uncertain QoS service composition framework

and service selection, which can be studied separately. For example, [23] focuses on workflow modeling, [24] focuses on service discovery, and [9] only focuses on service selection. This paper only focuses on the service selection process, that is, the workflow model, service repository, etc., are known. The service selection process is divided into two phases: the global QoS constraint decomposition phase and the composite service optimization phase.

Different from the existing constraint decomposition strategy, the global QoS constraint decomposition should be able to adapt to various constraint strengths. When the constraint strength is high, the main purpose of constraint decomposition is to reduce the loss of feasible solutions and improve the success rate of service composition. When the constraint strength is low, the main purpose of constraint decomposition is to reduce the solution space and reduce the time overhead of service composition. Therefore, on the basis of the global QoS constraint decomposition model proposed in [12], a global QoS constraint decomposition model with constraint strength awareness is proposed, and an adaptive adjustment algorithm for relaxation factors based on fuzzy inference rules is designed by adding support for QoS uncertainty and introducing relaxation factors to adapt to different constraint strengths.

When the user constraint strength is high, the global QoS constraint cannot be guaranteed due to the introduction of relaxation factors. It is still necessary to adopt the global optimization strategy in service composition, and uncertain QoS aggregation calculation is the key to achieve the global optimization strategy. When the user constraint strength is low, although global QoS constraints can be guaranteed, if global utility optimization is to be considered, it also depends on uncertain QoS aggregation calculation. By improving the K-means algorithm, a QoS aggregation calculation method based on the clustering strategy is designed to improve the calculation accuracy and reduce the time complexity.

For specific combined service optimization strategies, there are some existing local optimization methods [25] or global optimization methods [5,22], which will not be discussed in this article.

3.2. **Statement of uncertain QoS service composition problem.** Let the workflow include task sequence $T = \{t_1, t_2, \ldots, t_n\}$, and task t_i $(i = 1, 2, \ldots, n)$ has a candidate service set $S(t_i) = \{s_{i1}, s_{i2}, \ldots, s_{im}\}$, $|S(t_i)|$ represents the number of candidate services owned by t_i. Each service involves multiple QoS, $Q(s, q)$ is used to describe the value of the attribute q of the service s. The QoS of a service can be gainful or deductive. Considering that the gain attribute can be converted into the depreciation attribute by multiplying by -1, only the depreciation QoS is considered.

Considering the QoS uncertainty, the QoS of the service can be regarded as a random variable and described by its newer monitoring samples. Let x be a random variable and $est(x)$ be an operator to evaluate x. The specific form can be determined by the user and the service provider through consultation. If the user proposes an upper bound constraint qc to the attribute q of the service s, then $est(Q(s, q)) \leq qc$.

We assume that there is a composite service, $cs^{\#} = \left\{s_1^{\#}, s_2^{\#}, \ldots, s_n^{\#}\right\}$, $Q\left(s_i^{\#}, q_r\right) = \frac{1}{|s(t_i)|} \sum_{s \in S(t_i)} E(Q(s, q_r))$, $E(X)$ represents the expectation of the random variable X, and then the user's constraint strength ω_r $(r = 1, 2, \ldots, u)$ for q_r is determined by Formula (1).

$$\omega_r = 1 - \frac{qc_r}{Q\left(cs^{\#}, q_r\right)} \tag{1}$$

where $Q\left(cs^{\#}, q_r\right)$ represents the aggregated value of $cs^{\#}$'s r-th QoS.

3.3. **Constraint strength-aware QoS constraint decomposition model.** Suppose that the user proposes an upper constraint $QC = \{qc_1, qc_2, \ldots, qc_u\}$ for u QoS attributes such as $q_1, q_2, \ldots, q_u$, and the upper bound constraint on the r-th QoS of task t_i is x_{ir}. The number of services $N(t_i)$ that satisfy the x_{ir} constraint among the candidate services of t_i can be calculated by Formula (2).

$$N(t_i) = \#\{s|est(Q(s, q_r)) \leq x_{ir} \wedge s \in S(t_i), r \in [1, u]\} \tag{2}$$

In Formula (2), $\#\{A\}$ represents the number of elements in set A.

The number of composite services that satisfy the entire workflow (WF) can be calculated by Formula (3).

$$N(WF) = \Pi_{i-1}^{n} N(t_i) \tag{3}$$

Suppose that there are composite services $cs^* = \{s_1^*, s_2^*, \ldots, s_n^*\}$, s_i^* can complete the task t_i, and the value of the r-th QoS: $Q(s_i^*, q_r) = x_{ir}$. The r-th QoS aggregation value cs^* is recorded as $Q(cs^*, q_r)$, then the constraint decomposition model can be described as the following formula.

$$\text{MAX } N(WF) \tag{4}$$

$$\text{s.t.} \quad Q(cs^*, q_r) \leq (1 + \gamma_r) * qc_r, \quad r \in [1, u] \tag{5}$$

$$x_{ir} \in \left[\min_{s \in S(t_i)} est(Q(s, q_r)), \max_{s \in S(t_i)} est(Q(s, q_r))\right] \tag{6}$$

Formula (4) describes the optimization goal, which is to maximize the number of remaining candidate schemes after constraint decomposition. Formula (5) shows the global QoS constraint condition, and γ_r is the introduced relaxation factor. When the constraint strength ω_r is high, γ_r should take a value greater than 0 to reserve more candidate services so as to improve the success rate of finding a combination solution. When ω_r is low, γ_r can be a value of 0 or less to ensure global constraint or eliminate some candidate services and reduce the solution space. Formula (6) specifies the fluctuation range of x_{ir}.

3.4. **Adaptive adjustment of relaxation factors based on fuzzy reasoning.** The selection of relaxation factor γ_r is related to many factors, such as the number of tasks n in the workflow, the number of constraints u, constraint strength ω_r, the expected value $N(WF)$, and the number of candidate services owned by each task $|S(t_i)|$. Experiments show that γ_r is mainly affected by n, u, and ω_r. Because u is usually a small natural number, we can use a fuzzy reasoning method to design the value of γ_r for each value n and ω_r. Since u usually has only a few possible values, the value of γ_r can be adjusted according to n and ω_r for a particular u. Fuzzy reasoning is an imprecise reasoning using fuzzy knowledge, which is relatively simple and effective to solve the above problem [26,27].

Firstly, the input variables n and ω_r and output variables γ_r should be fuzzy. According to the experimental results, set the fuzzy subset of n as {N1, N2, N3}, and the fuzzy subset of ω_r as $\{\text{W1}, \text{W2}, \ldots, \text{W13}\}$, the fuzzy subset of γ_r is $\{\text{S1}, \text{S2}, \ldots, \text{S15}\}$. By adopting the triangle and trapezoid membership functions, the above three variables can be fuzzy and their membership functions are shown in Figures 2-4 respectively.

Then, the rules of fuzzy reasoning should be determined. Adopt the form of if-then to set the following 15 rules.

1) if (n is N1 or N2) and (ω_r is W1), then γ_r is S1;
2) if (n is N3 and ω_r is W1) or (ω_r is W2), then γ_r is S2;
3) if (n is N1 or N2) and (ω_r is W3), then γ_r is S3;
4) if (n is N3) and (ω_r is W3), then γ_r is S4;
5) if (ω_r is W4), then γ_r is S5;

6) if (ω_r is W5), then γ_r is S6;
7) if (ω_r is W6), then γ_r is S7;
8) if (n is N1) and (ω_r is W7), then γ_r is S8;
9) if (n is N2 or N3) and (ω_r is W7), then γ_r is S9;
10) if (ω_r is W8), then γ_r is S10;
11) if (ω_r is W9), then γ_r is S11;
12) if (ω_r is W10), then γ_r is S12;
13) if (ω_r is W11), then γ_r is S13;
14) if (ω_r is W12), then γ_r is S14;
15) if (ω_r is W13), then γ_r is S15.

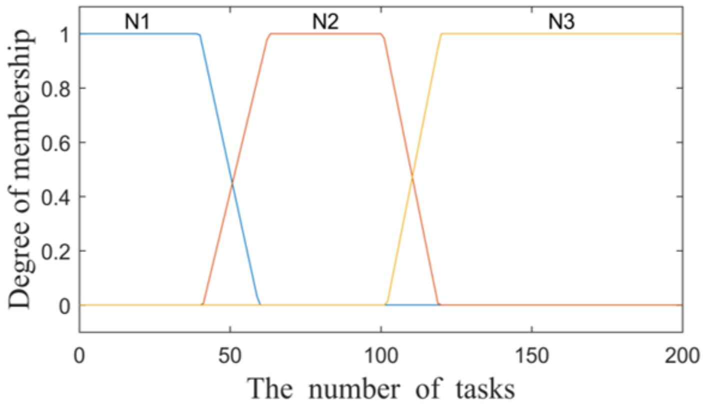

FIGURE 2. Membership function of the number of tasks

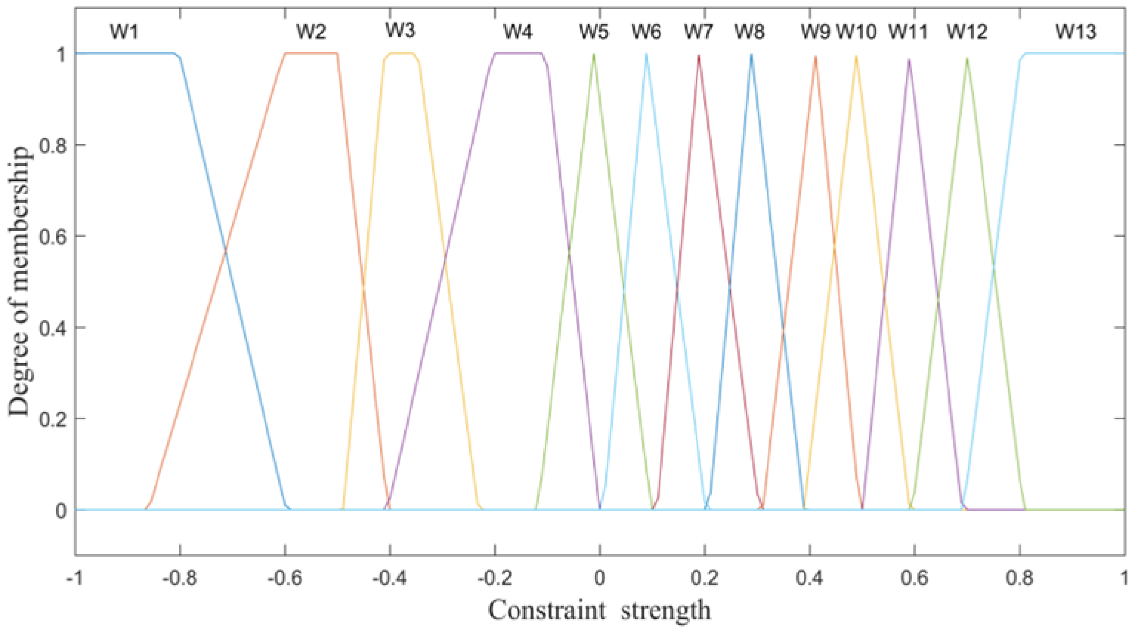

FIGURE 3. Constraint strength membership function

Table 1 describes the above rules in another form.

Finally, according to the fuzzy rules determined in Table 1, the projection surface of γ_r can be obtained and be shown in Figure 5.

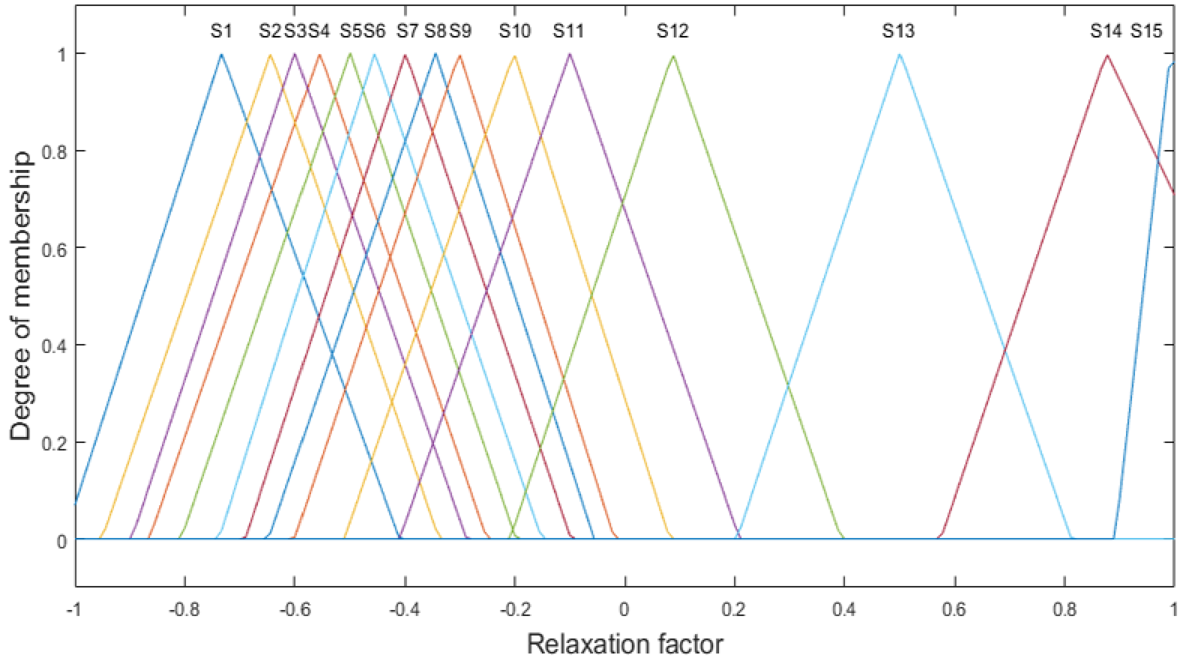

FIGURE 4. Membership function of relaxation factor

TABLE 1. Fuzzy reasoning rules

Constraint strength / Number of tasks	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13
N1	**S1**	**S2**	**S3**	**S5**	**S6**	**S7**	**S8**	**S10**	**S11**	**S12**	**S13**	**S14**	**S15**
N2	**S1**	**S2**	**S3**	**S5**	**S6**	**S7**	**S9**	**S10**	**S11**	**S12**	**S13**	**S14**	**S15**
N3	**S2**	**S2**	**S4**	**S5**	**S6**	**S7**	**S9**	**S10**	**S11**	**S12**	**S13**	**S14**	**S15**

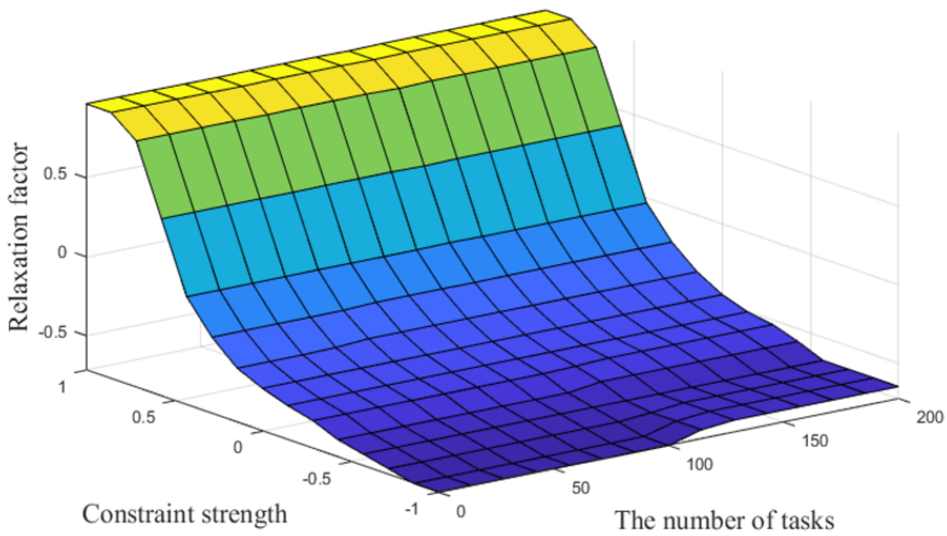

FIGURE 5. (color online) Surface projection of relaxation factor

4. Algorithm Design.

4.1. Constrained model solution. Referring to the greedy algorithm used in [12], we design Algorithm 1 to solve the constraint decomposition model. In Algorithm 1, if $isFit(x)$ is true, that indicates x satisfies the condition (5). If $isBoundFit(x)$ is false, that means x satisfies condition (6). The action $adjust(x)$ indicates that all tasks and all QoS of each task are adjusted once in steps Δx, $adjust(x, `+\text{'})$ is to increase $x$, and $adjust(x, `-\text{'})$ is to decrease x. The action $adjust(x, i, `+\text{'})$ means to increase all QoS of task $t_i$ by $\Delta x_i$, and $adjust(x, i, r, `+\text{'})$ means to increase the r-th QoS of task t_i by Δx_{ir}. The function $bestTask()$ returns the number of the tasks that leads to the maximum increment. The function $bestQoS(i)$ returns the QoS attribute that brings the maximum increment to t_i.

Algorithm 1 Solve the constraints decomposition model

Input: $QC = \{qc_1, qc_2, \ldots, qc_u\}$, $levelNum$
Output: x_{ir} $(i = 1, 2, \ldots, n,\ r = 1, 2, \ldots, u)$
S1: Init ω_r, γ_r, x_{ir}, and Δx_{ir};
S2: **If** $(isFit(x))$ **then**
  Do
  $adjust(x, `+\text{'})$;
  While $(isFit(x)$ && $!isBoundFit(x))$;
  $adjust(x, `-\text{'})$;
  Else if $(!isFit(x))$ **then**
  Do
  $adjust(x, `-\text{'})$;
  While $(!isFit(x)$ && $!isBoundFit(x))$;
  If $(isBoundFit(x))$
  output ("No solution!") and return;
  End if
S3: $i = bestTask()$;
  While $(i! = -1)$ **do**
  $adjust(x, i, `+\text{'})$;
  $i = bestTask()$;
  End while
S4: **For** $(i = 1$ to $n)$ **do**
  $r = bestQoS(i)$;
  While $(r! = -1)$ **do**
  $adjust(x, i, r, `+\text{'})$;
  $r = bestQoS(i)$;
  End while
  End for

The initial value of x_{ir} $(i = 1, 2, \ldots, n,\ r = 1, 2, \ldots, u)$ is the expected average of the r-th attribute values of all candidate services of the i-th task. Δx_{ir} can be determined by Formula (7), where $levelNum$ is a constant and can be set according to experience values.

$$\Delta x_{ir} = \frac{\max_{s \in S(t_i)} E(Q(s, q_r)) - \min_{s \in S(t_i)} E(Q(s, q_r))}{levelNum} \tag{7}$$

The algorithm first sets an initial value for x (S1) and then optimizes x from three levels in turn. Firstly, adjust x to the critical value for all tasks and all QoS attributes (S2). Then, repeatedly look for the most valuable task and adjust x for all its QoS attributes

(S3). Finally, repeatedly look for the most valuable QoS attribute and adjust x (S4). Consider that x is a matrix with n rows and u columns. S2 means adjusting all the elements in the matrix, S3 means adjusting all the elements of a row, while S4 means adjusting an element of a row and a column. S2 emphasizes integrity, that is, the number of remaining services for all tasks is expected to be close, while S4 highlights individual differences.

The main time overhead of Algorithm 1 is the adjustment of x in S2, S3, and S4. S2, S3, and S4 need to be adjusted x times no more than $levelNum$, $n * levelNum$, $n * u * levelNum$ respectively, where n is the number of tasks and u is the number of QoS constraints. The worst time complexity of Algorithm 1 is O $(n * u * levelNum)$.

4.2. **Cluster-based QoS aggregation operation.** Similar to [8], considering the four common modes of sequence, concurrency, selection, and cycle in workflow, the common QoS is divided into five categories: summation, weighted average, product, maximum, and minimum. When a random variable is used to represent the QoS of a service, the QoS aggregation operation involves basic operations such as the sum, weighted average, product, maximum, and minimum of multiple random variables. Because these operations have commutative and associative laws, only two random variables need to be considered.

Let the distribution of the random variable X be $P\{X = x_i\}$, $i = 1, 2, \ldots, a$, the distribution of Y is $P\{Y = y_j\}$, $j = 1, 2, \ldots, b$, and the distribution of Z is $P\{Z = z_l\}$, $l = 1, 2, \ldots, a * b$. When $Z = X + Y$, z_l can be expressed by Equation (8).

$$z_l = x_i + y_j, \quad i \in [1, a],\ j \in [1, b] \tag{8}$$

When $Z = w_1X + w_2Y$ (w_1, w_2 are weights, $w_1 + w_2 = 1$, $0 \le w_1, w_2 \le 1$), z_l can be represented by Formula (9).

$$z_l = w_1 \cdot x_i + w_2 \cdot y_j, \quad i \in [1, a],\ j \in [1, b] \tag{9}$$

When $Z = XY$, z_l can be expressed by Formula (10).

$$z_l = x_i \cdot y_j, \quad i \in [1, a],\ j \in [1, b] \tag{10}$$

When $Z = \max(X + Y)$, z_l can be represented by Formula (11).

$$z_l = \max(x_i + y_j), \quad i \in [1, a],\ j \in [1, b] \tag{11}$$

When $Z = \min(X + Y)$, z_l can be expressed by Formula (12).

$$z_l = \min(x_i + y_j), \quad i \in [1, a],\ j \in [1, b] \tag{12}$$

It can be seen that no matter which aggregation operation is adopted by X and Y, there are ab values (repetition is allowed) for Z. Obviously, there is a combination explosion problem, and the number of values after aggregation needs to be controlled within the specified range.

4.3. **Improved K-means clustering algorithm.** After each QoS aggregation calculation, if the number of aggregated values exceeds the expected value, a clustering algorithm similar to K-means [28,29] can be used to control the number of aggregated values. Considering that the data set to be clustered is one-dimensional, the classic K-means algorithm is adjusted as follows.

1) The initial clustering center ics is determined by Formula (13), where k is the number of clustering centers, and max and min are the maximum and minimum values of the data set respectively. ε is randomly selected on $[-\varepsilon_r, \varepsilon_r]$, which is used to introduce a certain randomness for the selection of ics. ε_r can be based on experience, such as 0.2.

$$ics[i] = (i + 0.5 + \varepsilon) * \frac{\max - \min}{k}, \quad i \in [0, k) \tag{13}$$

2) After ics is determined, the clustering center $p[j]$ to which the j-th number $d[j]$ belongs is determined by the following rules: if $d[j] \le ics[0]$, then $p[j] = 0$, and if $d[j] \ge ics[k-1]$, then $p[j] = k-1$; if $\exists i \in [1, k-2]$ satisfies $ics[i] \le d[j] \le ics[i+1]$ and $d[j] - ics[i] \le ics[i+1] - d[j]$, then $p[j] = i$; if $\exists i \in [1, k-2]$ satisfies $ics[i] \le d[j] \le ics[i+1]$ and $d[j] - ics[i] \ge ics[i+1] - d[j]$, then $p[j] = i+1$.

3) When the cluster center ics is updated to ncs, if $p[j] = i$, $d[j]$ can only be adjusted to $ncs[i-1]$, $ncs[i]$, or $ncs[i+1]$, that is, the value of $p[j]$ can only be $i-1$, i, or $i+1$. The specific adjustment rules are: if $d[j] < ncs[i]$ and $d[j] - ncs[i-1] < ncs[i] - s[j]$, then $p[j] = i-1$; if $d[j] > ncs[i]$ and $d[j] - ncs[i] > ncs[i+1] - d[j]$, then $p[j] = i+1$, otherwise, $p[j] = i$.

4) The condition for stopping the update of the cluster center is that the maximum distance that the cluster center moves is less than $(\max - \min)/(10 * k)$.

5. Experimental Analysis.

The service composition problem with response time as a constraint is considered. The workflow is randomly generated according to the number of services involved, involving three modes of order, concurrency and selection, and their ratio is about 2 : 1 : 1. The ws-dream dataset2 data set [19] was used as the data source. It involves 4,500 Web services, each of which involves two QoS, response time and throughput. Each QoS of each service involves 64 real measurements from 142 users, that is, each service contains approximately 9,000 records. From this data set, randomly take 100 records for the response time of each Web service as a QoS sample describing the response time of the service. In the experiment, the QoS samples of the candidate services are selected from the 4,500 services in a cyclic order, and $levelNum = 30$.

The fuzzy reasoning rules are implemented by Matlab, other programs are coded by Java, run on 64-bit Windows7 operating system, which has Intel (R) Core (TM) i5-2450M CPU with 4 GB of memory. The average of 20 tests was adopted.

5.1. Analysis of constraint decomposition model effect.

Two scenarios are considered: one is a scenario without considering the relaxation factor, that is, the relaxation factor is always taken as 0 (denoted as M1), and the other is the scenario where the relaxation factor is adaptively adjusted according to the method of Section 3.3 (denoted as M2). It is agreed that $est(x)$ is the expectation of x plus 3 times the mean square deviation of x.

The goal of service composition is to find a combination scheme that meets the constraints. Whether a feasible combination scheme can be found is an important measure. The basis for judging whether a feasible solution can be found is: first, filter the candidate services of each task according to the local constraints obtained by the constraint decomposition process, and then, if there are no remaining candidate services of a task, it is considered that no feasible solution can be found, otherwise, select s_{ij} from its remaining candidate services for task t_i, minimize $est(Q(s_{ij}, t))$, and synthesize services from these selected candidate services. If $est(Q(s, t))$ is not greater than the response time constraint specified by the user, it is considered that a feasible solution can be found, otherwise, it is considered that a feasible solution cannot be found. Here $Q(s, t)$ represents the QoS of the response time of services.

In the experiments, we have verified a variety of scenarios when the number of tasks $n \in [10, 100]$ and the number of services $m \in [20, 200]$, and the results were similar. Due to the limitation of space, only the case when $n = 100$ and $m = 200$ is taken into considered. As for the constraint strength ω, we set $\omega \in [0.4, 0.85]$, since, both methods are 100% efficient at finding a feasible solution when $\omega < 0.4$ and cannot find a feasible solution when $\omega > 0.85$. Figure 6 compares the probability that the two methods can find

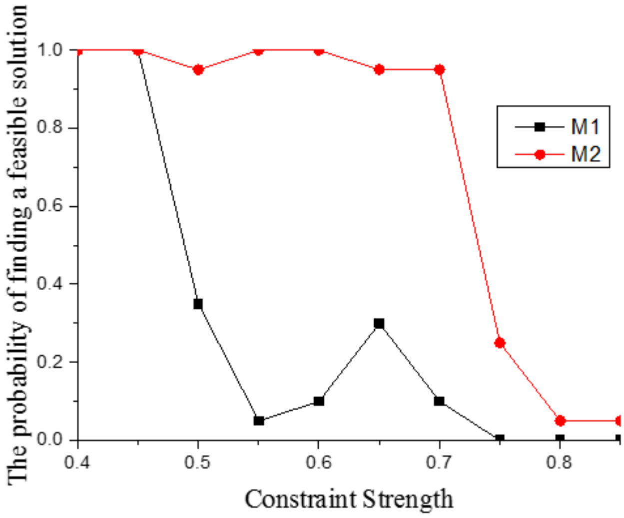

FIGURE 6. Find the probability of feasible solution changes with constraint strength

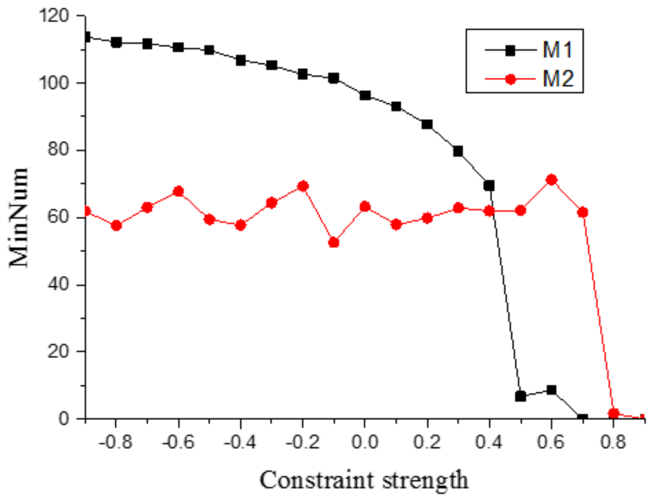

FIGURE 7. MinNum changes with constraint strength

a feasible solution. For M1, when $\omega \geq 0.5$, the probability of finding a feasible solution is less than 35%, and when $\omega \geq 0.75$, basically no feasible solution can be found. For M2, when $\omega \leq 0.7$, the feasible solution can be always found, and when $0.7 \leq \omega \leq 0.85$, it is still possible to find a feasible solution. Figure 6 clearly shows that when the constraint strength is greater than 0.5, the constraint decomposition model with relaxation factors can significantly improve the probability of finding a feasible solution.

When the user constraint strength is low, one of the goals of constraint decomposition is to eliminate some candidate services to reduce the solution space. Taking the minimum value of the number of local constraints (denoted as MinNum) obtained by each task satisfying the decomposition as the measurement index, Figure 7 compares the changes of MinNum obtained by two methods when the number of tasks is 100, the number of candidate services is 200, and the constraint strength ω increases from -0.9 in steps of 0.1 to 0.9. For M1, MinNum decreases as ω increases, for M2, when $\omega \leq 0.7$, MinNum is roughly stable at about 60. When ω is not greater than 0.4, compared with M1, the MinNum obtained by M2 is smaller, that is, the solution space can be significantly reduced on a larger scale.

5.2. **Analysis of QoS aggregation effect.** Comparing the sampling method (labeled as Sample), the method in [8] (labeled as PDF), and the method in this article (labeled as Cluster), we take the aggregated QoS expectations and mean square deviation as the measurement indicators. The number of samples is 2000 times of the number of services. In the PDF method, the sampling starting point is 0, and the sampling interval is 200. In the Cluster method, the number of cluster centers is 50.

The sampling method has a high number of sampling times, and its results should be relatively reliable. Taking the expectation and mean square deviation of aggregated QoS obtained by the sampling method as a reference, the relative difference between the expectation and mean squared deviation of aggregated QoS obtained by PDF and Cluster methods can be calculated.

Figure 8 compares the relative differences between the expectation of aggregated QoS of the two methods and the Sample method. It can be seen from the figure that the relative difference obtained by Cluster is not more than 1%, while that obtained by PDF is more than 2%. The expectation value obtained by Cluster is slightly high, while PDF is slightly low. The absolute value of the relative difference of expectations of two methods is slightly increased with the number of tasks. In general, the expectations of two methods are relatively accurate, and the accuracy of Cluster is more than double that of PDF.

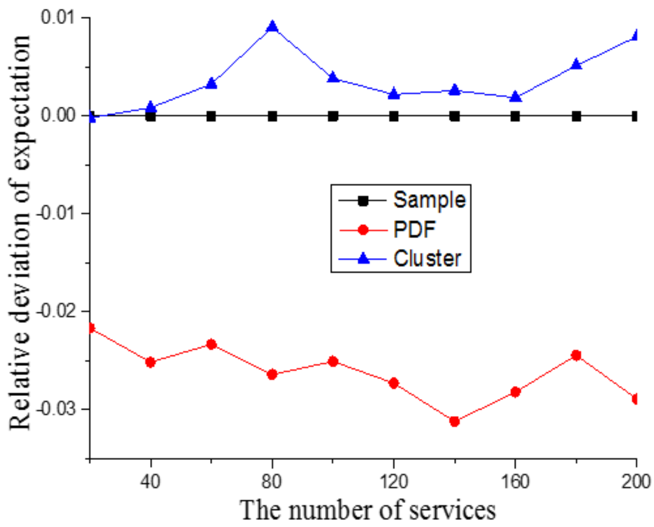

Figure 8. Relative difference of expectation varied with the number of services

Figure 9 compares the relative difference between the aggregated QoS mean squared deviation of the two methods and the Sample method. It can be seen from the figure that the maximum amplitude of the relative difference of the mean square deviation between PDF and Cluster is not much different, about 0.3%. The mean square deviation obtained by PDF is slightly low, and the mean square deviation obtained by Cluster is also slightly low, but occasionally it is high. In terms of the average fluctuation situation, the stability of the mean square deviation obtained by Cluster is better than PDF, and the accuracy is also high.

The time overhead of the three methods shown in Figure 10 varies with the number of services. It can be seen that the time overhead of the Cluster is much lower than the other two methods, and its time overhead roughly increases linearly with the number of services.

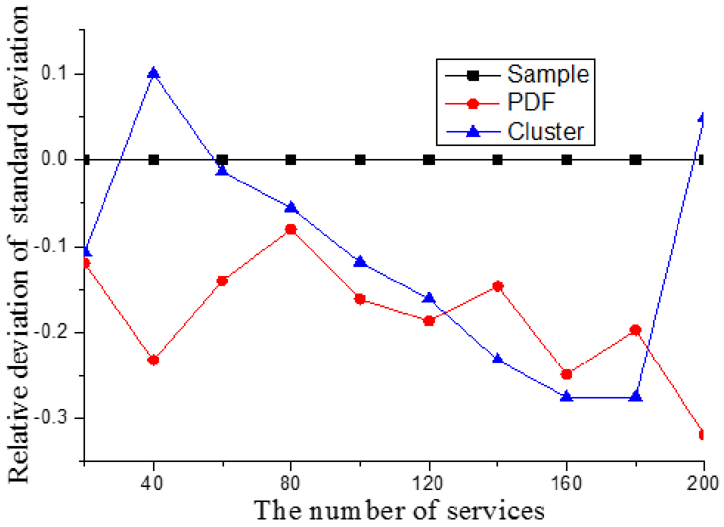

FIGURE 9. Relative difference of mean variance varied with the number of services

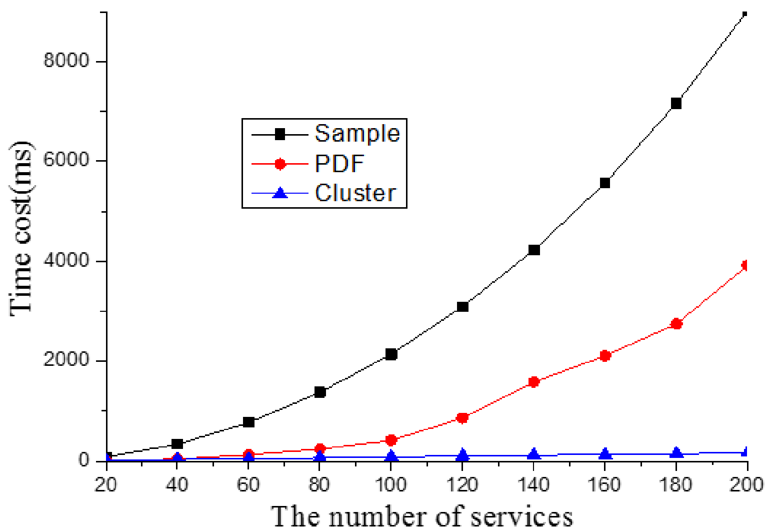

FIGURE 10. Time overhead varied with the number of services

Experimental analysis shows that, compared with PDF method, Cluster method has obvious advantages in both calculation accuracy and time overhead.

6. **Conclusions.** Based on the constraint decomposition strategy, the uncertain QoS service combination is divided into two stages: constraint decomposition and service optimization. In the constraint decomposition stage, by constructing a global QoS constraint decomposition model with constraint strength awareness, the success rate of service composition is improved when the constraint strength is high. When the constraint strength is low, the solution space can be reduced more effectively, thereby, reducing the time overhead of service optimization. In the service optimization stage, we focus on the calculation of uncertain QoS aggregation, and use experimental records to represent the QoS distribution. Improved K-means clustering algorithm is used to avoid the combination explosion problem in the aggregation calculation process. Compared with the existing methods, the calculation accuracy and time overhead are significantly improved.

The experimental results show that the relaxation factor is also influenced by the number of services and QoS distribution of services to some extent. Since it is difficult to apply the fuzzy reasoning method to more than 2 input variables, the next step is to explore a more effective method for adaptive adjustment of relaxation factor.

Acknowledgment. This research work was supported by the National Natural Science Foundation of China (Grant No. 61762031), Guangxi Universities Key Laboratory Director Fund of Embedded Technology and Intelligent Information Processing (Grant No. 2018A-05) and Foundation of Guilin University of Technology (Grant No. GUTQD-JJ2002018).

REFERENCES

[1] B. B. Traore, B. K. Foguem, F. Tangara et al., Service-oriented computing for intelligent train maintenance, *Enterprise Information Systems*, vol.13, no.1, pp.63-86, 2019.

[2] U. D. Atmojo, Z. Salcic, K. I. Wang et al., A service-oriented programming approach for dynamic distributed manufacturing systems, *IEEE Trans. Industrial Informatics*, vol.16, no.1, pp.151-160, 2020.

[3] G. P. Bhandari, R. Gupta and S. K. Upadhyay, An approach for fault prediction in SOA-based systems using machine learning techniques, *Data Technologies and Applications*, vol.53, no.4, pp.397-421, 2019.

[4] Y. W. Zhang, G. M. Cui, Y. T. Yan et al., Quality constraints-aware service composition based on task granulating, *Journal of Computer Research and Development*, vol.55, no.6, pp.1822-1829, 2018.

[5] W. A. Tan and Y. Zhao, Web service composition based on chaos genetic algorithm, *Computer Integrated Manufacturing Systems*, pp.1822-1829, 2018.

[6] M. M. Fazeli, Y. Farjami and M. Nickray, An ensemble optimisation approach to service composition in cloud manufacturing, *International Journal of Computer Integrated*, vol.32, no.1, pp.83-91, 2019.

[7] H. Mezni and M. Kbekbi, Reusing process fragments for fast service composition: A clustering-based approach, *Enterprise Information Systems*, vol.13, no.1, pp.34-62, 2019.

[8] H. Zheng, J. Yang and W. Zhao, Probabilistic QoS aggregations for service composition, *ACM Trans. the Web*, vol.10, no.2, pp.1-36, 2016.

[9] S. G. Wang, Q. B. Sun, F. C. Yang et al., Web service dynamic selection by the decomposition of global QoS constraints, *Journal of Software*, vol.22, no.7, pp.1426-1439, 2011.

[10] Z. Z. Liu, H. Peng and L. Cao, Research on QoS discomposing and service status monitor for composite service with global QoS constraint, *Acta Electronica Sinica*, vol.44, no.4, pp.886-892, 2016.

[11] S. Y. Hwang, C. C. Hsu and C. H. Lee, Service selection for web services with probabilistic QoS, *IEEE Trans. Services Computing*, vol.8, no.3, pp.467-480, 2015.

[12] H. Ye, T. Li and C. Jing, Decomposition of global constraints for QoS-aware web service composition, *International Journal of Innovative Computing, Information and Control*, vol.12, no.6, pp.2053-2066, 2016.

[13] S. X. Sun, A decomposition-based approach for service composition with global QoS guarantees, *Information Sciences*, vol.199, no.15, pp.138-153, 2012.

[14] C. Surianarayanan and G. Ganapathy, An approach for selecting best available services through a new method of decomposing QoS constraints, *Service Oriented Computing & Applications*, vol.9, no.2, pp.107-138, 2015.

[15] Z. Z. Liu, X. Xue, J. Q. Shen et al., Web service dynamic composition based on decomposition of global QoS constraints, *International Journal of Advanced Manufacturing Technology*, vol.69, nos.9-12, pp.2247-2260, 2013.

[16] I. Guidara, I. A. Jaouhari and N. Guermouche, Dynamic selection for service composition based on temporal and QoS constraints, *IEEE International Conference on Services Computing*, pp.267-274, 2016.

[17] L. F. Ren, W. J. Wang and X. Xu, Uncertainty-aware adaptive service composition in cloud computing, *Journal of Computer Research and Development*, vol.53, no.12, pp.2867-2881, 2016.

[18] H. Z. Ye and X. P. Lu, Robust web services composition based on discrete particle swarm optimization, *Journal of University of Electronic Science and Technology of China*, vol.47, no.3, pp.443-448, 2018.

[19] Z. Zheng, Y. Zhang and M. R. Lyu, Investigating QoS of real-world web services, *IEEE Trans. Services Computing*, vol.7, no.1, pp.32-39, 2014.
[20] S. G. Wang, Q. B. Sun, G. W. Zhang et al., Uncertain QoS-aware skyline service selection based on cloud model, *Journal of Software*, vol.23, no.6, pp.1397-1412, 2012.
[21] S. Y. Hwang, H. J. Wang, J. Tang et al., A probabilistic approach to modeling and estimating the QoS of web-services-based workflows, *Information Sciences*, vol.177, no.1, pp.5484-5503, 2007.
[22] H. Fekih, S. Mtibba and S. Bouamama, An efficient user-centric web service composition based on harmony particle swarm optimization, *International Journal of Web Services Research*, vol.16, no.1, pp.1-21, 2019.
[23] Klimek and Radoslaw, Pattern-based and composition-driven automatic generation of logical specifications for workflow-oriented software models, *Journal of Logical and Algebraic Methods in Programming*, vol.104, pp.201-226, 2019.
[24] A. Ahmad, A. Abdulaziz, A. M. Mohammed et al., Location assisted delay-less service discovery method for IoT environments, *Computer Communications*, vol.150, pp.405-412, 2020.
[25] H. Z. Ye, T. S. Li and Y. H. Guan, A self-healing QoS-aware web service composition method, *Journal of University of Electronic Science and Technology of China*, vol.46, no.3, pp.618-624, 2017.
[26] J. N. Zhang, S. G. Wang, Q. B. Sun and F. C. Yang, Fast and reliable fault-tolerance approach for service composition in integration networks, *Journal of Software*, vol.28, no.4, pp.940-958, 2017.
[27] Sasmoko, W. Budiharto and H. Prabowo, Self-diagnostic using fuzzy logic for teaching learning quality improvement in universities, *ICIC Express Letters*, vol.13, no.3, pp.247-253, 2019.
[28] X. X. Chen, Y. Q. Wei, M. Ren et al., Weighted K-means clustering algorithm based on firefly algorithm, *Application Research of Computers*, vol.35, no.2, pp.466-470, 2018.
[29] J. Yang, T. Liu, R. Chen, G. Zhang, H. Peng and J. Wang, A kernel-based clustering algorithm under the framework of membrane computing for image segmentation, *ICIC Express Letters, Part B: Applications*, vol.10, no.4, pp.311-317, 2019.