International Journal of Services Computing (ISSN 2330-4472) Vol. 4, No. 1, January - March, 2016

WEB SERVICES OPTIMAL COMPOSITION BASED ON IMPROVED ARTIFICIAL BEE COLONY ALGORITHM WITH THE KNOWLEDGE OF SERVICE DOMAIN FEATURES

ZhiZhong LIU, HaiFang WANG, XiaoFei XU, ZhongJie WANG
School of Computer Science and Technology Harbin Institute of Technology, Harbin, China
{ zhizhongliu, wanghaifang, xiaofei,rainy }@hit.edu.cn

Abstract

Web service composition is a key technology for creating value-added services by integrating available services. With the rapid development of Service Computing, Cloud Computing, Big Data and Internet of Things, a fast increasing number of services with similar functionalities but different Quality of Service (QoS) are available on the Internet, and make Web service optimal composition a NP-hard problem. Meanwhile, over the last decade's development and evolution of the service industries, service domain features (such as priori and similarity of services) are gradually formed. These features have valuable domain knowledge for improving Web service optimal composition. However, existing research works on Web service composition don't make full use of the service domain features, and leading to unfavorable results. Therefore, how to improve the efficiency and effectiveness of Web service optimal composition with the knowledge of service domain features becomes a significant challenge. To attack this issue, this paper firstly analyzes the influences of service domain features on Web service optimal composition; then, improves key optimization strategies of ABC with the knowledge of service domain features; finally, proposes the Web service optimal composition method based on improved artificial bee colony algorithm (S-ABC$_{SC}$). The performance of S-ABC$_{SC}$ is verified through simulation experiment; Moreover, the underlying dependencies between service domain features and S-ABC$_{SC}$'s optimality are analyzed and S-ABC$_{SC}$'s parameter settings are determined through several experiments with different service usage data sets.

Keywords: Web Service Composition; Service Domain Features; Artificial Bee Colony Algorithm; Influence Analysis

1. INTRODUCTION

Web service is a type of distributed computing model, with traits of self-contained, modular, loose coupled, standards-based, high capacity, etc. The Service-oriented architecture (SOA) approach allows the integration of service components independently developed into complex business processes and value-added applications to meet the users' requirements and to offer extra business value.

It is expected that with the proliferation of Cloud Computing, Big Data and Internet of Things, more and more Web services become available, thus, a massive number of services with the same functionalities and different Quality of Service (QoS, such as Cost, response time, availability, reliability, reputation, security, throughput, etc.) can be found for each task of the service composition workflow. Massive candidate services make Web services optimal composition a NP-hard problem [1]. How to efficiently build a composite Web service that not only can meet customer requirement but also has the optimal global QoS remains an open challenge [2].

With the development and evolution of service industries, it gradually forms the service domain features, such as priori and similarity. Priori means that, for a typical service request, some services with a higher usage frequency and satisfaction can be found according to historical service usage data. The priori domain knowledge is produced in the long-term service applications. Similarity refers to phenomenon that there are lot of services with the same functionality and similar quality of service (QoS) in the service domain.

Service domain features imply valuable domain knowledge which are useful for the solving the problem of Web service optimal composition problem. For example, for a certain kind of travel service request, service system provides the same one or some composite services to users, and users are satisfied with these composite services. When this kind of travel service request comes again, service system can firstly take the most frequently used composite service as the initial solution. If the initial solution is satisfactory, it needn't to construct new composite service from scratch. In such a situation, based on the priori domain knowledge, service system can quickly find the satisfactory solution. If the solution can not satisfy user request, then, service system can generate new composite services based on this initial solution through service replacement.

When generating new composite services based on the service domain features, it will be more effective if the service system firstly search services in the frequently used service set, because the probability of creating a satisfactory composite service in the frequently used service set is high. If the service system cannot find the suitable services in the frequently used service set, then, it can go to search the similar service set where exist some Web services having the similar QoS with the frequently used services, this mechanism can improve the efficiency of generating better new solutions.

From the above analysis we can find that service domain features imply valuable knowledge for improving the efficiency and effect of Web service optimal composition. However, existing researches on Web service optimal composition (such as service composition based on integral linear programming [3], mixed linear programming [4], heuristic algorithm [5], improved genetic algorithm [6], ant colony algorithm [7], particle swarm optimization [8] and artificial bee colony algorithm [9, 10]) always transform the problem of Web service optimal composition into a pure mathematical problem, and then apply mathematic or swarm intelligent algorithms to solve it. They ignore the influence of service domain features on solving Web service optimal composition. So, the efficiency and effect of Web service composition should be improved.

To tackle this issue, this work firstly analysis the influence of service domain features on solving Web service optimal composition problem; then, improve the key optimization operations of artificial bee colony algorithm [12, 13] with the knowledge of service domain features; finally proposes the algorithm for Web service optimal composition method based on improved artificial bee colony algorithm (named as S-ABC$_{SC}$). The performance of S-ABC$_{SC}$ is verified through simulation experiments; Moreover, the underlying dependencies between service domain features and the optimality of S-ABC$_{SC}$ are analyzed. To measure the richness and confidence level of the domain features, five metrics called domain feature metrics are proposed. According to the experimental results, domain feature metrics are determined and they will help to set the parameters of S-ABC$_{SC}$ so as to achieve better performance.

In summary, this work makes the following contributions:

- It proposes a Web service optimal composition method based on improved artificial bee colony algorithm with the knowledge of service domain features (S-ABC$_{SC}$).
- It verifies the performance of S-ABC$_{SC}$ through simulation experiment.
- It analyzes the underlying dependencies between the service domain features and performance of S-ABC$_{SC}$, and determines the S-ABC$_{SC}$'s parameter settings.

The remainder of this paper is organized as follows. Section 2 presents the related work; Section 3 introduces the problem of Web service optimal composition; Section 4 presents the ABC algorithm; Section 5 presents the service domain features and analyzes their influence on solving Web service optimal composition problem. Section 6 describes Web service optimal composition method based on improved artificial bee colony algorithm with the knowledge of service domain features. Section 7 verifies the performance of S-ABC$_{SC}$; Section 8 analyzes the underlying dependencies between the service domain features and performance of S-ABC$_{SC}$, and determines the algorithm parameter settings. Finally, Section 9 concludes this work and introduces the future work.

2. Related Works

QoS-aware Web service optimal composition is a hot research point in recent years. Many researchers have proposed various methods to solve this problem. Zeng et al proposed a global optimization method based on integer programming [3]. Some researchers choose the optimal subset of each candidate service set from the multidimensional quality space by the skyline method to optimize the service composition problem and reduces the search space effectively [14, 15]. Paper [16] proposes an approximation-based approach for service optimal composition [16]. Canfora et al. uses the Genetic Algorithm to solve the QoS-aware service composition problem [17]. Chen proposes a method based on QoS and PSO [18]. Wang et al. improves the Artificial Bee Colony (ABC) algorithm [19, 20].

Recently, some scholars have already begun to apply service domain features to solve service optimization problem. Xu et al. utilizes the Priori and Similarity extracted from historical usage data to solve the problem of concurrent service selection problem [21]. Literature [22] proposes a service selection approach based on composite service execution information. Paper [23] proposes a division based composite service selection approach, in this work, a set of efficient composite service execution instances are extracted from the log repository. Paper [24] obtains user interest on web services and QoS preference by mining history information of Web services usage, then, presents a service recommendation method based on user QoS preferences and user interest points of service. Kang et al. [25] proposes a service recommendation method based on the user service history information, the historical information includes the functional requirement information and the QoS information.

Bravo et al. [26] reduces the problem of designing and implementing Web service compositions to the problem of finding and selecting the composition closest to an initial specification. Similarity is used to measure how close is a given composition with respect to any given specification, in this paper a set of similarity measures are described for Web service composition models. Paper [27] develops the definition of similarity of Web service processes in order to investigate service substitution, it conclude that substituting a service in a composition can be performed independent of the context as long as the new service is similar to the substituted one.

Sharma et al. [28] compute the standard similarity measures based on Cosine Similarity and Euclidean distance, then, retrieve the most relevant services according to the similarity. Paper [29] presents a new context-based solution for Web services discovery. In order to make more efficient discovery and selection, the services context-based selection uses a new quantitative similarity measure to calculate the correspondence degree between the client and the services contexts in order to provide users with appropriate services according to their contexts.

Through analysis it can be found that existing research work only uses single service domain feature to improve the effect of service optimization problem solving, few research has applied these important service domain features comprehensively to guide the creating of Web service optimal composition problem. To fill this gap, this work firstly analysis the influence of service domain features on solving Web service optimal composition problem, then, improve the key optimization operations of artificial bee colony algorithm with the knowledge of service domain features, and propose the Web service optimal composition method based on improved artificial bee colony algorithm, so as to improve the the efficiency and effectiveness of Web service optimal composition.

3. Problem formulation

The process of QoS-ware Web service composition can be described as figure 1, where $T_1, T_2, ..., T_n$ are tasks that comprising the service composition workflow, $S_1, S_2, ..., S_n$ are candidate service sets for tasks, the number of candidate service in each set is $m_i, i \in \{1, 2, ..., n\}$.

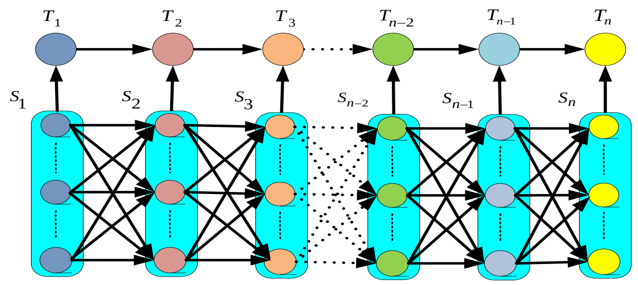


Figure 1. Web service composition process

Usually, for Web service composition, $QoS = \{q_1,...,q_j,...,q_k\}$ indicates the QoS attributes of Web services; $C = \{c_1,...,c_j,...,c_k\}$ indicates global QoS constraints provided by customer, where $c_j (1 \le j \le k)$ is a global QoS constraint for QoS attribute q_j ; $W = \{w_1,...,w_j,...,w_k\}$ represents user's QoS preferences, where $w_j (1 \le j \le k)$ is user's preference for the j-th QoS attribute q_j.

The problem of QoS-aware Web service composition is to select a set of concrete Web services for each task of a service composition so that the composite service constructed by these selected services both satisfy customer's global QoS constraints and also has the optimal global QoS. The mathematical model of the service optimal composition can be described as follows:

$$\text{ObjectiveFunction:} \begin{cases} \text{M}ax\ f(CWS_i) \\ f(CWS_i) = \sum_{j=1}^{k} w_j * q_j' \end{cases} \quad Eq(1)$$

$$\text{Constraints:} \begin{cases} q_1' \le c_1 \\ q_2' \le c_2 \\ \dots\dots\dots\dots \\ q_i' \ge c_i \\ \dots\dots\dots\dots \\ q_k' < c_k \end{cases}$$

Where CWS_i indicates the i-th composite Web service, q_i' is the i-th aggregated QoS value of CWS_i . For the QoS aggregation formula, please refer to literature [30].

4. Artificial Bee Colony Algorithm

Artificial Bee Colony algorithm (ABC) is a new swarm intelligence algorithm inspired by the foraging behaviors of bee colonies [31]. In ABC, the search space is simulated as the foraging environment and each point in the search space corresponds to a food source that the artificial bees could exploit. The nectar amount of a food source represents the fitness of the solution. In ABC, employed bees exploit the specific food sources they have explored before. Onlooker bees receive information about the food sources and choose a food source to exploit depending on the information of nectar quality. The more nectar the food source contains, the larger probability the onlooker bees will choose it. The employed bee whose food source has been abandoned becomes a scout bee. Scout bees search the whole environment randomly. The pseudo code for ABC is listed in Alg.1.

Alg.1: ABC algorithm

```
1: Initialize the food source positions.
2: Evaluate the nectar (fitness) amount of food
   sources
3: repeat
4:     Employed Bees phase
5:     Onlooker Bees phase
6:     Scout Bees phase
7:     Memorize the best solution achieved so far
8: UNTIL(Cycle = Maximum Cycle Number or a
   Maximum CPU time)
```

ABC has better performance [32] and there exist strong mappings between the ABC algorithm and the solving process of Web service optimal composition:

i) The process of searching optimal nectar by bees corresponding to the process of finding the optimal solution of Web service optimal composition problem;

ii) Food sources in ABC corresponding to the feasible solutions of Web service optimal composition problem;

iii) Fitness function in ABC corresponding to the target function of Web service optimal composition problem;

iv) Employed bees phase and onlooker bees phase corresponding to the local search in the solution space of Web service optimal composition problem;

v) Scout bees phase corresponding to the global search in the solution space of Web service optimal composition problem;

vi)Algorithm termination condition in ABC corresponding to the termination condition of solving Web service optimal composition problem.

Due to such strong correlation between the ABC algorithm and the Web service optimal composition, in this paper, we take ABC as the algorithm basis for designing effective algorithm for Web service optimal composition problem.

The food source for service composition problem can be modeled as (2).

$$X_i =< s_{1_t}, ..., s_{j_k}, ..., s_{n_l} > \quad (2)$$

X_i represents a food source constituted by n-dimensional vector, it also indicates a solution for service composition problem; s_{j_k} represents the value of i-th dimensional variable of the food source X_i, $s_{j_k} \in S_j$, s_j is the candidate service set for the j-th dimensional variable of the food source.

5. Influences of Service Domain Features on Web Service Optimal Composition

Service domain features are important objective law in service domain and they have strong influences on solving Web service optimal composition problem, making use of this domain knowledge is helpful for improving creating optimal composite Web services. In the following sections, influences of service domain features on solving Web service optimal composition are analyzed in details.

(1) Affecting service space division

With the knowledge of priori and similarity, candidate service space can be divided into some small service space. Such as priori service composition schemes set (PriSS), priori service set (PriS), similar service set (SimS) and generic service set (GenS); Service set PriSS, PriS, SimS and GenS are produced as follows. Based on the priori knowledge, some frequently used and satisfactory service composition schemes can be collected to form the PriSS. For an initial service set S for a task note in the service composition process, firstly determine the priori service set PriS for the task note according historical service usage data; then, find out the similar services with services in PriS and construct the similar service set SimS; finally, reduce PriS and SimS from S and take the rest service to form the general service set GenS. Once these service sets are determined, the service system can maintain and update these service sets regularly, so as to support the solving of service composition.

(2) Affecting the service space search strategy

According to the priori feature, the satisfactory solution is most likely to appear in the PriSs, and then in SimSs, the last is in the GenSs. Therefore, when solving service composition problem, instead of searching in the whole candidate service space wholly, it is better to first search in the PriSs, then in SimSs and GenSs sequentially.

(3) Affecting initial food sources generation strategy

With the domain knowledge of priori and similarity, adaptive initial solution generation method can be proposed according to the confidence of priori. That is, when the confidence of priori is strong, most initial solutions canbe generated based on PriSS and PriSs; otherwise, intial solutions can be generated proportionately based on the four service spaces.

(4) Affecting the neighborhood search strategy

With the domain knowledge of priori and similarity, hybrid neighborhood search method can be designed in PriSs and SimSs. Neighborhood search with knowledge guidance is useful for avoiding blind and random neighborhood search, so as to improve algorithm's search capability.

6. Web Service Optimal Composition Based on Improved Artificial Bee Colony (S-ABC$_{SC}$)

This section firstly presents the improvements of ABC based on the influences of service domain features on solving Web service optimal composition problem, and then, describes Web service optimal composition method based on S-ABC$_{SC}$.

(1) Improved search strategies

In the service domain, when the confidence of the priori is abundant, the promising area where the most satisfactory or optimal solution can be found is the PriSs, then is SimSs, and the last is the GenSs. In this case, it is better to search in PriSs, SimSs and GenSs sequentially, instead of searching the whole candidate service space randomly. Here let us take this space search strategy as service space priority search strategy, which is denoted as P . On the other hand, when the confidence of the priori is low, the probability of finding the most satisfactory or optimal solution in the PriSs is relatively small, but the probability of finding the satisfactory solution within the three service spaces (PriSs, SimSs, GenSs) is relatively larger. In this case, it is better to carry out search in the three service spaces simultaneously. Here we call this space search strategy as the service space equilibrium search strategy, which is denoted as E .

(2) Improved initial food source generation methods

For the problem of Web service optimal composition, some high quality initial food sources are generated with the prior knowledge. For service space search strategies P and E , two initial food sources generation strategies are proposed respectively.

(a) Initial food source generation strategy based on priori

This strategy refers to generating initial food sources based on PriSS and PriSs, i.e., selecting the better prioir service schemes as the initial food sources and randomly generating food sources based on PriSs. The proportion of food sources are generated based on PriSS is α, the proportion of food sources are generated based on the PriSs is β, and $\alpha + \beta = 1$.

(b) Initial food source equilibrium generation strategy

This strategy refers to generating a certain amount of initial food sources based on the PriSs, SimSs and GenSs respectively, Here the proportions of the three parts for the food sources are set as α, β and χ, where $0 \le \alpha, \beta, \chi \le 1$ and $\alpha + \beta + \chi = 1$.

(3) Improved employed bees phase

With the partial order among services in PriSs and SimSs, a heuristic neighborhood search method with search direction and search step is proposed.

The heuristic neighborhood search method in PriSs (denoted as $NS(PriS)$) is described as follows: firstly, we determine the optimal search direction of the i-th dimensional variable; then, along the determined direction and search step size η, we select a new service and replace with the old service. The search direction can be determined by the testing method. The neighborhood search strategy in SimSs ($NS(SimS)$) is identical with $NS(PriS)$. Since there is no useful domain knowledge in GenSs that can be used to guide the neighborhood search, the random neighborhood search method is adopted.

(4) Improved onlooker bees phase

For algorithm S-ABC$_{SC}$, the roulette wheel selection method is adopted to select food source for onlooker bees. After getting a food source, onlooker bees become the employed bees and begin the searching according to the neighborhood search strategies.

(5) Improved scout bees phase

For algorithm S-ABC$_{SC}$, two kinds of new food sources generating methods for scout bees are proposed. New food source generation method for P (the service space priority search strategy):

(a) When the searching space is PriSs, then, generate new food sources in the PriSs;

(b) When the searching space is SimSs, then, generate new food sources in the SimSs;

(c) When the searching space is GenSs, then, generate new food sources in the GenSs.

New food source generation method for E (service space equilibrium search strategy): generate new food sources randomly based on PriS, SimS and GenS. Web service optimal composition method based on improved artificial bee colony algorithm is described as Alg. 2.

Alg.2: Web service optimal composition based on improved artificial bee colony algorithm (S-ABC$_{SC}$)

Input: PriSS, PriSs, SimSs, GenSs, QoS Data, QoS constraints, User QoS preferences; values of parameters α, β, χ, η, Limit and SN;

Output: Optimal Web service composition scheme

```
Step1: Determining service space search strategies
1.    If (the confidence of the priori $\ge \mathrm{P}$)
2.        Select the service space search strategy $\mathrm{P}$;
3.    Else
4.        Select the service space search strategy $\mathrm{E}$;
5.    EndIf
Step2: Initial food sources generation
6     If(the service space search strategy== $\mathrm{P}$)
7         Generate the initial food sources with the initial food
          sources generation strategy based on priori;
8     EndIf
9     If(the service space search strategy== $\mathrm{E}$)
10        Generate the initial food sources with the equilibrium
          initial food sources generation strategy;
11    EndIf
Repeat
Step3: Improved employed bees phase
12    If(the service space search strategy== $\mathrm{P}$)
13      For(i=1 to $SN$)
14        If(the searching service space==PriSs)
15          Generate new food sources with the neighborhood
            search method $NS(PriS)$;
16        EndIf
17        If(the searching service space==SimSs)
18          Generate new food sources with the neighborhood
            search method $NS(SimS)$;
19        EndIf
20        If(the searching service space==GenSs)
21          Generate new food sources with the neighborhood
            search method $NS(GenS)$;
22        EndIf
23        Evaluate the new food source with formula (1);
24        Keep the better food sources;
25        If(the fitness of food source is not improved)
26          Counteri=Counteri+1;
27        EndIf
28      EndFor
29    EndIf
30    If(the service space search strategy== $\mathrm{E}$)
31      For (i=1 to $SN$)
32        Generate new food sources with the hybrid neighborhood
          search method in the service candidate set;
33        Evaluate the new food source with formula (1);
34        Keep the better food sources;
35        If(the fitness of the food source is not improved)
36          Counteri=Counteri+1;
37        EndIf
38      EndFor
39    EndIf
Step4: Improved onlooker bees phase
40    For(i=1 to $SN$)
41      Select a food source for the i-th onlooker bee;
```

```
        Carry out the employed bee phase;
    **EndFor**
**Step5: Improved scout bees phase**
    **For**(i=1 to $SN$ )
        **If**(Counteri>Limit&&seraching in PriSs)
            Generate a new food source based on PriSs;
        **EndIf**
        **If**(Counteri>Limit&& seraching in SimSs)
            Generate a new food source based on SimSs;
        **EndIf**
        **If**(Counteri>Limit&& seraching in GenSs)
            Generate a new food source based on GenSs;
        **EndIf**
    **EndFor**
**Step5: Memorize the best solution achieved so far**
    **Until**
    **If** (the given arbitration criteria is satisfied)
        Output the memorized solution;
    **EndIf**
```

7. Performance Verification of S-ABC$_{SC}$

Experiments are conducted to verify the performance of S-ABC$_{SC}$. In these experiments, a Web service composition process that includes six tasks is taken as an example, and each task in the composition process has 200 candidate services. The QWS dataset [33] was used as the QoS dataset of the candidate services, QoS attributes including ResponseTime, Reliability, Throughput and Success Rate. In our experiments, we set the user preferences of QoS attributes are 0.5, 0.3, 0.1 and 0.1 respectively.

The PriSS and PriSs for this service composition problem are generated by iterating the Genetic Algorithm (GA) 100 times during solving the problem. 200 feasible service composition schemes were collected, and 20 service composition schemes were randomly selected and taken as the PriSS. Then, the PriSs, SimSs and GenSs for tasks were constructed. The scale of the three service spaces were set as 20, 40 and 140 respectively. In this experiment, the scale of the initial food sources SN is set as 100, we also set $\alpha = 0.2$, $\beta = 0.8$, and $limit = 3$. The value of F was set as 0.7. S-ABC$_{SC}$ algorithm was realized using C++. The experiments were conducted on a PC with the following configuration, OS: Microsoft Windows XP 2002; CPU: Intel(R), G550 @ 2.60GHZ; Memory: 3 GB.

In our experiments, we compared S-ABC$_{SC}$, Discrete ABC (D-ABC) [34] and GA for solving the service composition problem. The experiment platform and programming language of the three algorithms are identical. For the D-ABC, we set $SN = 100$ and $limit = 3$. For GA, the scale of the initial population was set as 100, the probability of the crossover operation and the mutation operation were 0.8 and 0.5 respectively. Then, the performances of the three algorithms are compared under the three different algorithm arbitration criteria.

(1) Optimal solution under the time constraint

This arbitration criteria means that when the running time of the algorithm reaches the maximum time proposed by user, then, output the current optimal solution that the algorithm has found. In order to overcome the time error due to the uncertainty of the experimental platform and increase the accuracy of the experiment results, under each time constraint, algorithms are executed for 20 times independently, and take the average of the results as the final results. The experimental results are shown as figure 2, where the vertical axis represents the fitness of solutions found by the three algorithms, the horizontal axis denotes time constraints; the time unit is millisecond (ms). The fitness is the evaluation values of Web service composition scheme; it can be calculated according to formula (1).

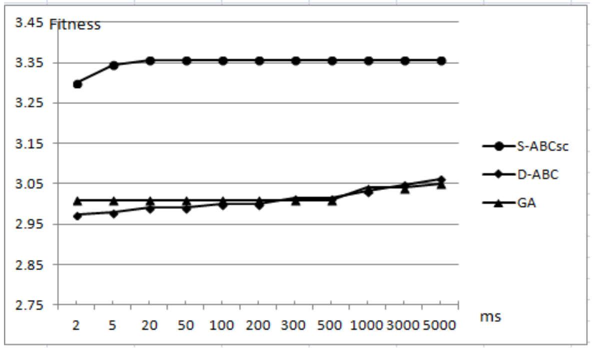


Figure2. Optimal solution under different time constraints

From figure 2 it can be seen that, under the same time constraint, solutions found by S-ABC$_{SC}$ are better than solutions found by D-ABC and GA. This means that S-ABCSC can find better solution within a shorter period of time under the assumed situation.

(2) Optimal solution under maximum iterations

This arbitration criteria means that when the algorithm has been iterated for the maximum iteration times, then, output the optimal solution the algorithm has found. In this experiment, the three algorithms are executed independently, the experiment results are shown as figure 3, and where the vertical axis represents the fitness of solutions found by the three algorithms, the horizontal axis denotes different maximum iteration times.

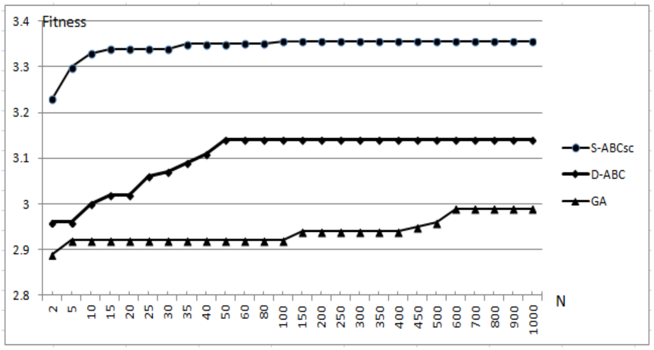


Figure 3. Optimal solutions under maximum iteration times

8. Analyze the Underlying Dependencies between Service Domain Features and S-ABC$_{SC}$'s Optimality

8.1 Experiment Preparation

The preceding chapters give a brief introduction to two domain features introduced into S-ABC$_{SC}$ algorithm, *Priori* and *Similarity*. The two domain features will be used in the form of priori knowledge and similarity knowledge extracted from the historical usage data, so the priori knowledge and the similarity knowledge can represent the corresponding historical usage data. In order to describe the properties of various historical usage data, we propose five domain feature metrics including Size of Priori Service Set (*SP*), Confidence of Priori Service Set(*CP*), Frequency of Priori Service Set(*FP*), Size of Similarity Service Set(*SS*) and Similarity Degree of Similarity Service Set(*SDS*).These metrics can be measured by the following expressions.

- *SP*: $SP = \frac{1}{D} \times \sum_{i=1}^{D} N_{Pi}$, N_{Pi} represents the amount of priori services of the *i*-th activity node.
- *CP*: $CP = \frac{1}{D} \times \sum_{i=1}^{D} CP_i$, CP_i stands for the function calculating the confidence of the priori service set of the *i*-th activity node, $CP_i = \frac{1}{N_{Pi}} \times \sum_{j=1}^{N_{Pi}} \frac{US_j}{PU_j}$, where US_j represents the amount of the records that utilize the *j*-th priori service and satisfy user requirements, PU_j stands for the amount of the records that contain the *j*-th priori service.
- *FP*: $FP = \frac{1}{D} \times \sum_{i=1}^{D} FP_i$, $FP_i = (\sum_{j=1}^{N_{Pi}} PU_j)/hdSize$, where *hdSize* represents the record amount of the historical usage data.
- *SS*: $SS = \frac{1}{D} \times \sum_{l=1}^{D} N_{Si}$, N_{Si} stands for the amount of similarity services of the *i*-th activity node, $N_{Si} = \left| U(\sum_{j=1}^{N_{Pi}} Sim(s_{Pj})) \right|$, where s_{Pj} represents the *j*-th priori service, *Sim* is the function calculating all the similarity services that are similar to s_{Pj} ,*U* is the function that combines all the similarity services of N_{Pi} priori services and eliminate the repeated similarity service.
- *SDS*: $SDS = Avg(\sum_{j=1}^{N_{Pi}} \sum_{l=1}^{N} CED(s_{Pj}, s_l))$,*CED* is the function calculating the *Euclidean Distance* between two services, *Avg* is the function calculating the average value of the data set which only contains the data satisfying the following condition: $CED(s_{Pj}, s_l) \geq t_s$, t_s is the similarity threshold.

For example, given a set of historical usage data concerning the flight information that user *U* went to Shanghai by air from Beijing last year. It has 350 records in all. Statistics show that *U* most often ordered 3 flights. There are 15 flights (including the 3 ones presented above) totally. In addition, there are 6 flights similar to the 3 ones introduced above among the remaining 12 flights. Then *SP*, *CP*, *FP*, *SS*, *SDS* are defined as follows.

- *SP* represents the size of priori service sets. Here, it represents the amount of the flights *U* most often ordered, *SP*=3.
- *CP* is on behalf of the confidence of priori service sets, indicating the proportion that the records satisfying user requirements make up in the records containing the priori services. Here, it reflects the probability of the 3 flights *U* usually ordered satisfy the user requirements.
- *FP* is the frequency of priori service sets. The larger *FP* indicates the bigger amount of records containing the priori services in the historical data. Here, it reports the times of ordering the 3 flights presented above by *U* last year.
- *SS* indicates the size of similarity service sets. Here, it indicates the amount of the flights which are similar (include similar prices, similar flying time) to the 3 ones recommended above among the remaining 12 ones, so here *SS*=6.
- *SDS* stands for the similarity degree of similarity service sets. In this paper, the similarity degree between two services is measured by their QoS *Euclidean Distance*. The smaller *SDS* between two services indicates the services are more similar. Here, it reflects the similarity degree between the 3 flights *U* most often ordered and the 6 ones similar to them.

All the experiments in this section are concerning the service composition problem illustrated as figure 4. The service process includes 11 task nodes, and each node represents a kind of activity. The directed flows among the nodes include sequential structure, selective structure and parallel structure.

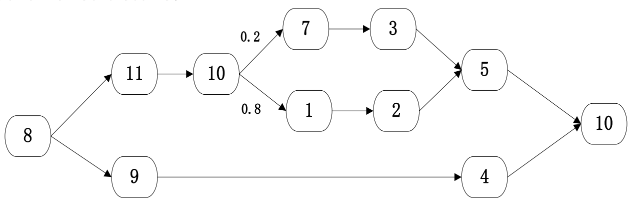
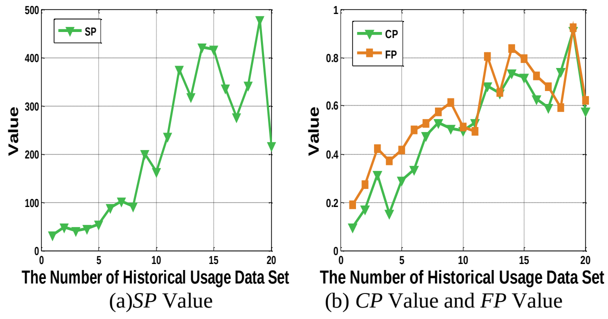


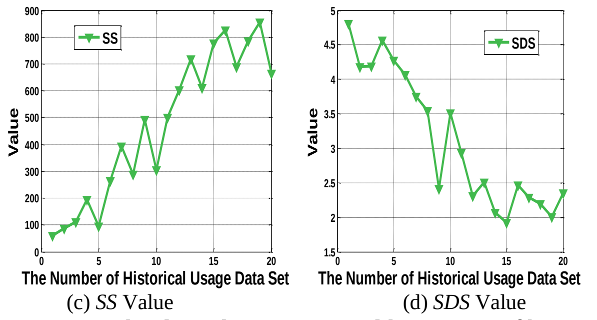
Figure4. Web service composition workflow

For each task node included in the service process, there are 5000 candidate services. In addition, each candidate service has five QoS indicators, namely response time, price, availability, reputation and throughput, and the value ranges of the five indicators are [1,10], [20,100], [0.6,0.999], [0.8,0.999], [1,20] respectively. The requirements raised by users are QoS constraints. It is a five-dimension vector which can be expressed as *UR*= {50,400,0.2,0.85,3}. The five dimensions are corresponding to the five QoS indicators.

We utilize the ABC algorithm to solve the same composition problem with various *Maximum Cycle Number* (*MCN*) values randomly, and record the optimal solution obtained, so as to constitute various sets of historical usage data. A set of historical usage data can be represented as a five tuple <*SP*, *CP*, *FP*, *SS*, *SDS*>. Because it is not easy to depict the dependencies on the basis of huge scales of historical usage data, so we select out 20 sets of historical data which are most representative from 200 sets. Although these data fails to fully reflect the actual data, but these 20 sets of data help characterize the dependencies to a great extent, and it basically reflects the features of all the historical data. Each set has 50000 records among the 20 sets of historical data. In addition, we number the 20 sets as 1,2,...,20, and call them HD1, HD2,...,HD20, respectively. The disciplines among them are shown in Fig.5. We can see the value ranges of *SP*, *CP*, *FP*, *SS*, *SDS* are [0,500], [0,1], [0,1], [0,900], [0,5] respectively. To note that, in this paper, all the data in figure 5 is the average results after executing the algorithms for 50 times.



(a)*SP* Value (b) *CP* Value and *FP* Value



(c) *SS* Value (d) *SDS* Value

Figure 5. The disciplines presented by 20 sets of historical usage data.

8.2 Analyzing the Influence of Service Domain Features on S-ABC$_{SC}$'s Optimality

Based on the 20 sets of historical usage data, quantities of attempts are carried on to verify the historical usage data and find the critical ranges which help algorithm designers judge whether the given historical usage data has positive impact on S-ABC$_{SC}$.

(1) The verification of the historical usage data

Firstly, we verify that if all the historical usage data would help the S-ABC$_{SC}$ algorithm to obtain a better solution, namely comparing S-ABC$_{SC}$ with ABC. The values of *SN*, *Limit*, *MCN* of S-ABC$_{SC}$ are the same as those of ABC, and the values of *PPSS* and *SL* are 0.8, 3 respectively. After lots of attempts, we can find out that among the 20 sets of data, the optimality of the final solution obtained by S-ABC$_{SC}$ on the basis of HD1 or HD2 or HD3 or HD4 or HD5 is no better than that of ABC, just as shown in figure 6. That is to say, the historical usage data called HD1, HD2, HD3, HD4, HD5 have negative effects on the ABC-based service composition algorithm.

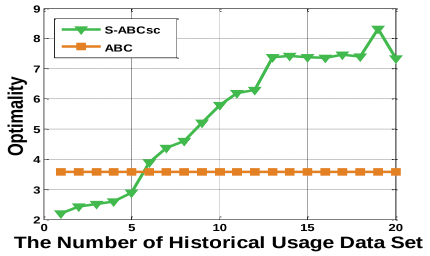


Figure 6. The contrast between the results of S-ABC$_{SC}$ and ABC.

Based on the 20 sets of data above, we summarize the effects that domain features exert on the optimality of S-ABC$_{SC}$ respectively from different metrics *SP*, *CP*, *FP*, *SS* and *SDS*, just as shown in figure 6. It is not difficult to find the approximately proportional relationships between the five domain features metrics and the optimality of S-ABC$_{SC}$. Along with *SP*, *CP*, *FP* or *SS* increasing, the optimality of S-ABC$_{SC}$ decreases firstly and then increases. After a certain value range, S-ABC$_{SC}$ performs much better than ABC. However, with the continuous increase of the *SDS* value, the optimality of S-ABC$_{SC}$ increases firstly and then decreases. After a certain value range, ABC performs much better than S-ABC$_{SC}$. There are critical value ranges of *SP*, *CP*, *FP*, *SS* and *SDS*, which help decide whether domain features have a positive impact on the optimality of S-ABC$_{SC}$ or not.

(2) The critical ranges

On the basis of the experimental results above, we summarize the critical ranges which are shown in figure 7.

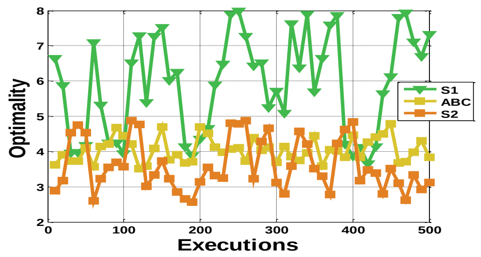
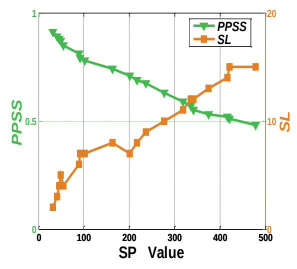


Figure. 7 The verification of the critical ranges.

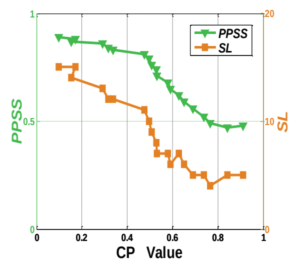
According to the figure 7, we find that with the increase of *SP*, the optimality of S-ABC$_{SC}$ gradually increases. When *SP* is less than the values belonging to the interval [54, 67], due to the small size of the priori service set, the algorithm isn't able to find a solution better than that of ABC until arriving to the stop condition. However, along with the size of the priori service set increasing, the search space in the priori service set becomes larger, making it easier to find the global optimal solution. With the increase of *CP*, the optimality of S-ABC$_{SC}$ increases constantly. It is not difficult to understand that with the confidence of the priori knowledge increasing, the percentages that priori service sets satisfying the user requirement are larger, so it is easier to find out optimal solutions. When *CP* is less than the values belonging to the interval [0.18, 0.24], it is very difficult to find better solutions than those of ABC for S-ABC$_{SC}$ because the services which do not meet the user requirement take up a larger percentage of the priori service set.

The optimality of S-ABC$_{SC}$ becomes larger as *FP* increases continuously. This means that the amount of priori services existing in the corresponding historical usage data increases, which indicates that there are many users with similar requirements utilizing these priori services, so these services are more likely to meet the user requirement. While *FP* is less than the values coming from the interval [0.37, 0.43], the amount of the priori services used in the historical usage data is too small, and these services are more likely to fail to satisfy the requirement and the optimality of the final solution obtained by S-ABC$_{SC}$ is smaller than that of ABC. With the increase of *SS*, the optimality of S-ABC$_{SC}$ increases gradually. Similar to the situation of *SP*, when the search space of S-ABC$_{SC}$ in the similarity service set becomes larger, S-ABC$_{SC}$ is more likely to find out the global optimal solution. When *SS* is less than the values extracted from the interval [33, 49], it indicates that the number of similarity services is limited, so the optimality of S-ABC$_{SC}$ is no better than that of ABC. It is noted that the optimality of S-ABC$_{SC}$ is becoming smaller as *SDS* increases constantly. The increase of *SDS* indicates that the number of services which are most similar to priori services becomes smaller. As a result, it becomes extremely difficult for S-ABC$_{SC}$ to find optimal solutions in the similarity service set.

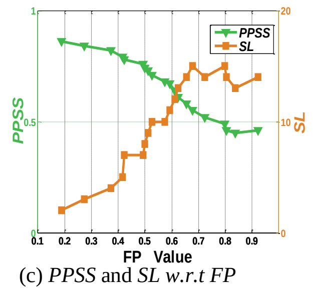
To verify these rules, we construct different sets of historical usage data. We utilize S-ABC$_{SC}$ to solve the service composition problem introduced in this paper. The experiment results are shown in figure 8, where S1 represents the results of S-ABC$_{SC}$ based on the historical data <156, 0.25, 0.672, 307, 1.934>, and S2 stands for the results of S-ABC$_{SC}$ based on the historical data <14, 0.19, 0.275, 103, 3.17>. From the experiments, we summarize the accuracy of the critical ranges presented above is about 70%. In addition, we find that *SS* and *SDS* exert smaller effects on the optimality of S-ABC$_{SC}$.



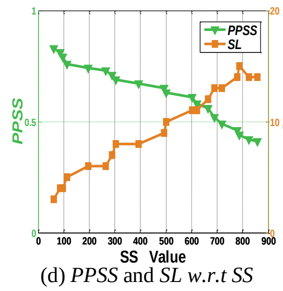
(a) *PPSS* and *SL w.r.t SP*



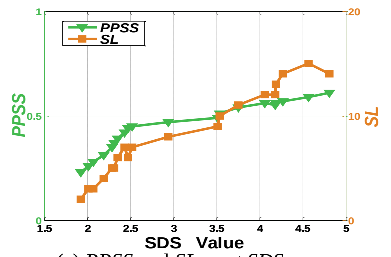
(b) *PPSS* and *SL w.r.t CP*



(c) *PPSS* and *SL w.r.t FP*



(d) *PPSS* and *SL w.r.t SS*



(e) *PPSS* and *SL w.r.t SDS*

Figure 8. The Dependencies between Five Domain Features and S-ABC$_{SC}$ Optimal Parameter Settings

(3) Dependencies between Domain Features and S-ABC$_{SC}$ Parameter Settings

To identify the underlying dependency between domain features and S-ABC$_{SC}$ parameter settings, we need to collect different experiment results of S-ABC$_{SC}$. According to the 20 sets of data, we assign various parameter settings of *PPSS* and *SL*, and execute S-ABC$_{SC}$. When the algorithm finds out the optimal solution, we need to record the optimality and execution time. Totally we collect 12000 records with all the values of *PPSS* and *SL* belonging to [0,1], [1,20] respectively.

To separate the sub-space of parameter settings that will achieve acceptable performances from the other sub-spaces leading to tolerance or unacceptable results, we utilize the C4.5 algorithm [17] to make the classification on the results calculated for each set of data, and then we can get the curtailed ranges of *PPSS* and *SL*. Here we take HD3 as an example, and the classification result of optimality is a decision tree. According to the decision tree, the overall conclusion drawn from the classification results is:

- $PPSS \in [0.437,1]$
- $SL \in [3,11]$

Similar to this, we classify the other data sets and obtain the curtailed ranges of *PPSS* and *SL* finally.

Obviously, the above classifications only identify the approximate ranges of *PPSS* and *SL*, and the ranges are different with various domain feature metrics. So we need to concrete the dependencies to facilitate algorithm designers to assign the values of *PPSS* and *SL* easily, so as to achieve optimal results. Figure 9 shows the variations of optimal S-ABC$_{SC}$ parameter settings with respect to each of five domain features metrics.

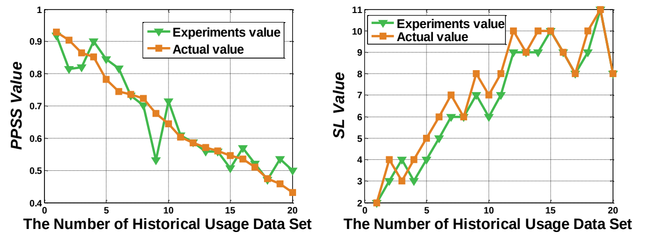


Figure 9. The Fitting Curve of Two S-ABC$_{SC}$ Parameters between Predicated and Actual Optimal Parameter Setting

According to figure 9, it is obvious that there exist approximate linear relations. The difference is that *PPSS* has negative correlations with *SP*, *CP*, *FP* and *SS*, but has positive correlation with *SDS*. Meanwhile, *SL* has positive correlations with *SP*, *FP*, *SS* and *SDS* and has negative correlation with *CP*. Therefore we use the multiple linear regression method to find the optimal fitting formulas, and we obtain formulas (3) and formula (4). Then we utilize the formulas to calculate the values of *PPSS* and *SL* respectively, and compare the calculations with the actual data, just as shown in Figure 8. Through the experimental analysis, we can find that accurate rates of the formula (3), (4) are about 75% and 85% respectively. Although the calculation is not fully corresponding to the actual data, but the fit results are acceptable.

$$PPSS = 0.0004\times SP\text{-}0.1837\times CP\text{+}0.2466\times FP \\ \text{-}0.0001\times SS\text{+}0.1836\times SDS \qquad (3)$$

$$SL = 0.0038\times SP + 5.7618\times CP + 2.0423\times FP \\ + 0.0027\times SS + 0.3425\times SDS \qquad (4)$$

Based on the experiments results above, in terms of the service composition problem, given a set of historical usage data, firstly algorithm designers can calculate the values of five domain features metrics and express the data set as <*SP*, *CP*, *FP*, *SS*, *SDS*>, and then, according to the method introduced in this paper determine whether the set of data is effective. If it is effective, algorithm designers will solve the service composition problem by the S-ABC$_{SC}$ algorithm, otherwise, they will take use of the ABC algorithm. While using the S-ABC$_{SC}$ algorithm, algorithm designers can assign values of *PPSS* and *SL* expediently according to the formula (1), (2) without a lot of tedious attempts.

9. Conclusions

With the analyzing the influence of service domain features on solving the problem of Web service optimal composition, in this paper, we firstly improved the key optimization operations of the artificial bee colony algorithm, and propose a new Web service optimal composition method based on the improved artificial bee colony (S-ABC$_{SC}$), then, the performance of S-ABC$_{SC}$ is verified through stimulation experiments, and the underlying dependencies between service domain features and the optimality of S-ABC$_{SC}$ are analyzed. S-ABC$_{SC}$'s parameter settings are determined through several experiments with different historical service usage data sets. Experimental results show that S-ABC$_{SC}$ is feasible and effective. In our future work, we will study the service domain features deeply, dig the valuable knowledge of service domain features and propose optimization algorithm paradigm for solving service optimization problems (such as service selection, service composition and resource scheduling) with the knowledge of service domain features.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable feedback on this work. This work is supported by: The National Natural Science Foundation of China (No.61472106); The National Natural Science Foundation Youth Fund China (No.61300124, 61403128); Major projects of science and technology in Shandong Province (2015ZDXX0201B02);

References

[1] Liao, J., Liu, Y., Zhu, X., & Wang, J. (2014). Accurate sub-swarms particle swarm optimization algorithm for service composition. *Journal of Systems and Software*, 90, 191-203.

[2] Laili, Y., Tao, F., Zhang, L., Cheng, Y., Luo, Y., & Sarker, B. R. (2013). A Ranking Chaos Algorithm for dual scheduling of cloud service and computing resource in private cloud. *Computers in Industry*, 64(4), 448-463.

[3] Zeng, L., Benatallah, B., Ngu, A. H., Dumas, M., Kalagnanam, J., & Chang, H. (2004). Qos-aware middleware for web services composition. *IEEE Transactions on Software Engineering*, 30(5), 311-327.

[4] Ngoko, Y., Goldman, A., & Milojicic, D. (2013). Service selection in web service compositions optimizing energy consumption and service response time.Journal of Internet Services and Applications, 4(1), 1-12.

[5] Yu, T., Zhang, Y., & Lin, K. J. (2007). Efficient algorithms for Web services selection with end-to-end QoS constraints. *ACM Transactions on the Web (TWEB)*, 1(1), 6.

[6] Sellami, K., Ahmed-Nacer, M., Tiako, P. F., & Chelouah, R. (2013). Immune genetic algorithm for scheduling service workflows with QoS constraints in cloud computing. *South African Journal of Industrial Engineering*, 24(3), 68-82.

[7] Zhang, C., Mu, G., Chen, H., Sun, T., & Pang, L. (2014). Distributed Service Discovery Algorithm Based on Ant Colony Algorithm. *Journal of Software*, 9(1), 70-75.

[8] Yin, H., Zhang, C., Zhang, B., Guo, Y., & Liu, T. (2014). A hybrid multiobjective discrete particle swarm optimization algorithm for a sla-aware service composition problem. *Mathematical Problems in Engineering*, 2014.

[9] Wang, X., Wang, Z., & Xu, X. (2013). An Improved Artificial Bee Colony Approach to QoS-Aware Service Selection. *2013 IEEE 20th*

International Conference on Web Services (ICWS), Santa, Clara, Marriott, June, 2013, pp. 395-402.

[10] Huang, K., Fan, Y., & Tan, W. (2012, June). An empirical study of programmable web: a network analysis on a service-mashup system. *2012 IEEE 19th International Conference on Web Services (ICWS)*, pp. 552-559.

[11] Karaboga D. An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, ErciyesUniversity, (2005).

[12] Karaboga, D., Gorkemli, B., Ozturk, C., & Karaboga, N. (2014). A comprehensive survey: artificial bee colony (ABC) algorithm and applications. *Artificial Intelligence Review*, 42(1), pp. 21-57.

[13] He, J., Chen, L., Wang, X., & Li, Y. (2013). Web Service Composition Optimization Based on Improved Artificial Bee Colony Algorithm. *Journal of Networks*, 8(9), pp. 2143-2149.

[14] Alrifai, M., Skoutas, D., & Risse, T. (2010, April). Selecting skyline services for QoS-based web service composition. *In Proceedings of the 19th international conference on World wide web*, pp.11-20, ACM.

[15] Chan, C. Y., Jagadish, H. V., Tan, K. L., Tung, A. K., & Zhang, Z. (2006, June). Finding k-dominant skylines in high dimensional space. *In Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 503-514. ACM.

[16] Boissel-Dallier, N., Benaben, F., Lorré, J. P., & Pingaud, H. (2015). Mediation information system engineering based on hybrid service composition mechanism. *Journal of Systems and Software*, 108, 39-59.

[17] Yilmaz, A. E., & Karagoz, P. (2014). Improved Genetic Algorithm Based Approach for QoS Aware Web Service Composition. *2014 IEEE International Conference on Web Services (ICWS)*, pp. 463-470.

[18] Liu, L., Li, Y. F., Zhang, M., & Liu, S. T. (2014). QoS Oriented PSO Algorithm for Cloud Service Composition Modeling. *In Applied Mechanics and Materials*,Vol. 577, pp. 931-934.

[19] Wang, X., Wang, Z., & Xu, X. (2013). An Improved Artificial Bee Colony Approach to QoS-Aware Service Selection. *2013 IEEE 20th International Conference on Web Services (ICWS)*, pp.395-402.

[20] Min, X., Xu, X., & Wang, Z. (2014). Combining Von Neumann Neighborhood Topology with Approximate-Mapping Local Search for ABC-Based Service Composition. *2014 IEEE International Conference on Services Computing (SCC)*, pp. 187-194.

[21] Xu, X., & Liu, Z. (2014). S-ABC-A Service-Oriented Artificial Bee Colony Algorithm for Global Optimal Services Selection in Concurrent Requests Environment. 2014 IEEE International Conference on Web Services (ICWS), pp. 503-509.

[22] ZHANG M.W., WEI W.J., ZHANG B., ZHANG X.Z., ZHU Z.L. (2008). Research on Service Selection Approach Based on Composite Service Execution Information. *CHINESE JOURNAL OF COMPUTERS*. 31(8), pp.1398-1411. (in chinese)

[23] Zhang M.W., Zhang B., Zhang X.Z., Zhu Z.L. (2012). A Division Based Composite Service Selection Approach. *Journal of Computer Research and Development*, 49(5), pp.1005-1017. (in chinese)

[24] Kang G., Tang M., Liu J., et al. Diversifying Web Service Recommendation Results via Exploring Service Usage History. IEEE TRANSACTIONS ON SERVICES COMPUTING. DOI 10.1109/TSC.2015.2415807.

[25] Kang, G., Liu, J., Tang, M., Liu, X., Cao, B., & Xu, Y. (2012, June). AWSR: Active web service recommendation based on usage history. 2012 IEEE 19th International Conference on Web Services (ICWS), Honolulu, Hawaii, USA, pp. 186-193.

[26] Bravo, M. (2014). Smiilarity Measures for Web Service Composition Models. *International Journal on Web Service Computing,* 5(1), 1.

[27] LI X.T., FAN Y.S. (2009). Analyzing Compatibility and Similarity of Web Service Processes. *CHINESE JOURNAL OF COMPUTERS*, 12: 2429-2437, (in chinese)

[28] Sharma, V., & Kumar, M. (2012). Comparative Analysis of IR Based Web Service Similarity Measures Using Vector Space Model. *In Global Trends in Information Systems and Software Applications*, pp. 752-760. Springer Berlin Heidelberg.

[29] M'hamed, B. F., & Abdelkader, B. (2013). Context-Aware Web Service Discovery Based on A Quantitative Similarity Measure. *International Journal of Modern Education and Computer Science* , 5(8), pp. 27.

[30] He, J., Chen, L., Wang, X., & Li, Y. (2013). Web Service Composition Optimization Based on Improved Artificial Bee Colony Algorithm. *Journal of Networks*, 8(9), pp. 2143-2149.

[31] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

[32] Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation*, 214(1), pp.108-132.

[33] Al-Masri, E., & Mahmoud, Q. H. (2008, April). Investigating web services on the world wide web. In Proceedings of the 17th international conference on World Wide Web , pp. 795-804. ACM.

[34] Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., & Chua, T. J. (2011). A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information sciences*, 181(12), pp. 2455-2468.

Authors

XiaoFei XU, Born in 1962, Ph. D., professor, Ph. D. Supervisor. His research interests include Service Computing and Software Service Engineering, Enterprise Computing, Data Mining and Business Intelligence, and so on.

ZhiZhong LIU, Born in 1981, Postdoctoral. His research interests include service computing and swarm intelligence algorithm.

ZhongJie WANG, Born in 1978, Ph. D., professor, Ph. D. Supervisor. His research interests include Service computing, service engineering, software architecture and its evolution.

HaiFang WANG, Born in 1988, she is doctoral candidate; Her research interests include Service computing, service value, and so on.