An Effective Dynamic Web Service Selection Strategy with Global Optimal QoS Based on Particle Swarm Optimization Algorithm

Guosheng Kang, Jianxun Liu, Mingdong Tang, Yu Xu
Key Laboratory of Knowledge Processing and Networked Manufacture, Hunan University of Science and Technology, 411201, Xiangtan, China
{guoshengkang, ljx529, tangmingdong, xuyu1013}@gmail.com

Abstract—Dynamic Web service selection with global QoS optimization in Web service composition is a critical issue in Web service composition. In order to solve the problem, based on intelligent optimal theory of particle swarm optimization (PSO) algorithm, we propose a strategy PSO-GODSS (global optimization of dynamic Web service selection based on PSO) algorithm to implement Web service selection with QoS global optimization. The basic idea of the algorithm is to transform the original Web service selection problem into a multi-objective services composition optimization problem with global QoS constraints, which is further transformed into a single-objective problem by using the method of ideal point. Then, the theory of intelligent optimization of PSO is applied to produce a set of optimal services composition process with QoS constraints. Theoretical analysis and experimental results indicate the feasibility and efficiency of this algorithm, and the execution efficiency and convergence rate of PSO-GODSS are much better than that of multi-objective genetic algorithm used in prior work.

Keywords—Service Composition; Service Selection; QoS Global Optimal; Multi-objective Optimization; PSO (particle swarm optimization)

I. INTRODUCTION

Web service, as a new distributed computing pattern, has been rapidly developed in recent years and plays an increasingly important role in e-commerce, enterprise application integration, and other applications. As the rapid development of Web service technology, there are more and more stable and easily-used Web services available via the Internet, but the function of a single Web service is limited. To make full use of the shared Web services, it is necessary to integrate them to form a newly value-added and complex Web service to meet requirements of different users. So, how to dynamically integrate the existing Web services to achieve the target, has become a new application requirement and a popular research focus.

Service composition flow model consists of multiple service nodes. Each of them is an abstract concept, which only contains the functional description and interface information instead of a specified Web service. Under a Web environment, Web services, with the same or similar function but different QoS (Quality of Service) attributes (e.g., time, cost, reliability, etc.), are multiple. Each service node in the composite service corresponds to a service group which consists of a set of the same functional Web services offered by different providers. So how to select a specific Web service for each node from each service group to form an executable service process to meet requirements of different users becomes a key problem in service composition, which is called the dynamic Web service selection problem in this paper.

The majority of existing approaches on dynamic Web service selection for service composition are almost local based QoS optimization with single objective function by linear weighted summation based on QoS parameters. They cannot solve Web service selection with QoS global optimization, for example, finding an executable service process which is to be operated with ultra-low cost and ultra-short time under the condition of reliability exceeding 80% [1]. To solve the deficiency of current approaches, S. Liu et al. [1] proposed a strategy GODSS (Global Optimization of Dynamic Web Service Selection) based on multiple-objective Genetic Algorithm. Composite service optimization problem is converted into a multi-objective optimization problem with constraints. However, this paper only focuses on the sequential service composition process model. Then, S. Liu et al. [2] employed the same method to consider other structures in service composition through transforming other process models into sequential models. Based on the multi-objective and multi-constraint optimization problem mentioned in [1,2], this paper proposes a more efficient algorithm PSO-GODSS based on particle swarm optimization. We also suppose that QoS parameters of each specific service are static when PSO-GODSS is running just like in [1,2]. Theoretical analysis and experimental results show that, compared to GODSS, our algorithm is simple, and easy to implement. And the convergence rate of this algorithm is faster. The global optimal service composition process can be searched faster. In general situations, for service requesters, we note that they may not pay much attention to the globally best composite service but the ones which both meet the user's QoS constraints and have better QoS, so evolutionary algorithms are more suitable to resolve the composite optimization problem. In our algorithm, a group of composite services, which meet constraints, can be generated by setting iteration number of the algorithm appropriately.

The rest of this paper is organized as follows: Section II introduces related work. Section III presents the problem of service composition optimization and introduces the basic models of composite service process and QoS computation methods for these flow models. Section IV outlines particle swarm optimization algorithm and designs PSO-GODSS algorithm to resolve MCOOP problem. Section V describes experiment results to verify the feasibility and effectiveness of the algorithm and Section VI concludes the paper.
II. RELATED WORK

Since Ran [3] proposed a new Web service discovery model in which the functional and non-functional requirements are taken into account for service discovery, the study of Web service selection based on QoS has become one of the research focuses in the field of Service Computing, and many achievements have been made. However, the majority of current approaches on dynamic Web service selection for service composition are usually QoS locally optimal based [3-6]. These approaches only select the service with best QoS parameters summation for each single service node in composite service. Among these approaches, services in each service node are mutually independent, which means that they cannot resolve the problem of QoS global optimization for composite service. Although there are still some work globally based [7-11], they all are the problem of single objective consisting of classical weighted summation of QoS parameters, not considering the problem of multi-objective optimization. And many of them proposed IP approach to solve the problem of global optimization. Wang et al. [12] proposed an efficient and effective QoS-aware service selection approach. It employs cloud model to compute the QoS uncertainty for pruning redundant services while extracting reliable services. Then, mixed integer programming is used to select optimal services. Lin [13] aims at enhancing the credibility of service composition plan, taking advantage of a Web service’s QoS history records, rather than using the tentative QoS values advertised by the service provider, but at last the composition optimization problem is also instantiated into an Integer Programming problem. Therefore, there are still some drawbacks among these works. For one thing, the solution of weighted summation is highly sensitive to the weight vector and also the optimization result is overly simplistic, so they cannot solve the multi-objective optimization problem in a proper way. At the mean time, they can only generate one optimal solution, so users can not choose other substitutions when the solution becomes inaccessible for some reason. For another, linear programming is an NP-complete problem. As the increase of the number of optional services, performance of service composition may be affected significantly. And these approaches require that objective functions and constraints must be linear, which is restricted in some practical situations, even though some non-linear equations can be transformed into linear ones. To avoid solving the NP-complete problem, Alrifai and Risse [9] use mixed integer programming to find the optimal decomposition of global QoS constraints into local constraints. Then the distributed local selection is employed to find the best web services that satisfy the local constraints. Further, Alrifai et al. [14] proposed an approach based on the notion of skyline to effectively and efficiently select services for composition, reducing the number of candidate services to be considered. And Tang et al. [15] proposed a heuristic service composition method, named local optimization and enumeration method. It aims at filtering the candidates of each task to a small number of promising ones by local selection, and then enumerates all the composite solutions to pursue a near-to-optimal one. However, most importantly, all these approaches cannot solve Web service selection with QoS global optimization with multi-objective and multi-constraint, for example, finding an executable service process which is to be operated with ultra-low cost and ultra-short time with constraint conditions on reliability and reputation respectively. To solve the deficiency of the existing approaches, papers [1,2] designed a GODSS algorithm to resolve the dynamic Web service selection with QoS global optimal in Web service composition. The essence of the algorithm is that the problem of dynamic Web service selection with global optimal QoS is transformed into a multi-objective service composition with QoS constraints. The theory of intelligent optimization of multi-objective genetic algorithm is employed to produce a set of optimal Pareto service composition process with constraint principle by means of optimizing various objective functions simultaneously [2], which solved the problem that could not be solved in traditional approaches. However, the execution efficiency of the algorithm still needs to be improved further, which is our main motivation in this paper.

III. PROBLEM DESCRIPTION OF SERVICE COMPOSITION

Def (Multi-Constrain and Multi-Objective Optimal Path, MCOOP) [1]: Let G(N,E,W) to be a Directed Graph, N is the set of vertexes, E is the set of edges, W is the weight of the edge. In G(N,E,W), there are \( k(k \geq 2) \) constraints \( C_i(i = 1,\ldots,k) \), each path \( P \) from the start vertex \( S \) to the end vertex \( T \) (two virtual services \( S \) and \( T \) are added to the graph) has \( m(m \geq 2) \) nonnegative measure rules \( f_1,\ldots,f_m \). \( P \in \Omega \), where \( \Omega \) is the set of paths. Path \( P^* \) is a multi-objective optimal path with multi-constraints iff there is no another path \( P \). When \( P \) and \( P^* \) are content with \( C_i \) for each measure rule, \( f_i(P^*) \geq f_i(P)(i = 1,\ldots,m) \), there is at least one rule \( i \) which meet \( f_i(P^*) > f_i(P)(i = 1,\ldots,m) \), then \( P^* \) is called the Pareto optimal solution under the multi-constraint and multi-objective problem. Here, \( \geq \) and \( > \) show non-inferior and priority relations between measure rules, respectively.

In dynamic service composition, an ordinary service composition flow model consists of multiple service nodes. Each service node (SN) corresponds to a service group (SG). The services in same service group have same or similar function but different QoS. The problem of dynamic Web service selection with global optimal QoS in Web service composition is to select a specific service from SG for each service node to construct an executable service process on the precondition that the service process meets the specific QoS constraints, and multi-objective functions are maximally optimized. According to the definition, the problem of dynamic service selection with QoS global optimization can be converted into finding a multi-objective optimal path satisfied QoS constraints from \( S \) to \( T \). In other words, it is a problem to find a solution of MCOOP in service composition graph (SCG). We propose PSO-GODSS to solve the problem in this paper.
A service composition flow model is a workflow based on Web service composition to the four basic models defined by Workflow Management Coalition. Most service composition flows can consist of the four models. There are some research achievements [27,9] on reduction rules of service composition models and QoS computation methods for service composition based on the four basic models. Generally, an aggregation functions is proposed to aggregate QoS of each service node. This paper describes composite service flows and computes the QoS of composite service according to the following equations. We suppose that each service contains four QoS parameters in this paper which are \( T(\text{time}) \), \( C(\text{cost}) \), \( \text{Rep}(\text{reputation}) \), \( \text{R(reliability)} \).

(1) Sequential:
\[
T_n = \sum_{i=1}^{n-1} T_i \quad C_n = \sum_{i=1}^{n} C_i \quad \text{Rep}_n = \sum_{i=1}^{n} \text{Rep}_i / n \quad R_n = \prod_{i=1}^{n} R_i
\]

(2) Parallel:
\[
T_n = \max\{T_i\} \quad C_n = \sum_{i=1}^{n} C_i \quad \text{Rep}_n = \sum_{i=1}^{n} \text{Rep}_i / n \quad R_n = \min\{R_i\}
\]

(3) Conditional: where \( a_i \) is the probability of branch \( i \) being chosen, \( \sum_{i=1}^{n} a_i = 1 \).
\[
T_n = \sum_{i=1}^{n} T_i \quad C_n = \sum_{i=1}^{n} C_i \quad \text{Rep}_n = \sum_{i=1}^{n} \text{Rep}_i / n \quad R_n = \prod_{i=1}^{n} R_i
\]

(4) Loop: where \( k \) is the number of loop.
\[
T_n = k \sum_{i=1}^{n} T_i \quad C_n = k \sum_{i=1}^{n} C_i \quad \text{Rep}_n = \sum_{i=1}^{n} \text{Rep}_i / n \quad R_n = \prod_{i=1}^{n} R_i
\]

IV. ALGORITHM DESCRIPTION OF PSO-GODSS

A. Formalized Description of MCOOP

This paper takes \( T \) and \( C \) as two objective functions for the sake of simplicity. The process model can be carried out by our algorithm effectively with ultra-short \( T \) and ultra-low \( C \). \( \text{Rep} \) and \( R \) can be regarded as two constraints. A model of multi-objective service composition with multi-constraint can be formalized as follows:

\[
\begin{align*}
\text{Min } F(P) &= [(T(P) - T^*)^2 + (C(P) - C^*)^2]^{1/2} \\
\text{s.t.} \{ &\text{Rep}(P) \geq \text{Rep}_0 \\
& \text{R}(P) \geq R_0
\end{align*}
\]

(3)

Where \( T(P), C(P), \text{Rep}(P) \) and \( \text{R}(P) \) correspond to the computations of QoS parameters in composite service mentioned in Section III. Formula (1) is to minimize the vector to make the objective functions to be minimized simultaneously. Actually, there may be many QoS parameters for a Web service composition, but the main thought is to take them as an objective function or a constraint condition in the optimization model, so this model can be extended to a service composition with any number of objective functions and constraint conditions.

B. Transforming MCOOP into Single Objective Problem

One characteristic of multi-objective optimization is that the objective functions always collide with one another, i.e., improvement of some functions will decline the capability of other functions. Generally, there is no mutual minimal point among all the objective functions. The basic method to solve multi-objective programming is the method of evaluation function. Its essence is to construct an evaluation function by using the visualized background in geometry or application. In that way, the multi-objective problem can be transformed into a single-objective optimization problem. Then using sophisticated methods to solve the problem of single-objective optimization, the solution can be taken as the solution of multi-objective optimization, which is especially effective for solving the problem of service composition optimization based on the fact that service requesters may not pay much attention to the globally best composite service but the ones which both meet the user’s QoS constraints and have better QoS. So we employ the method of ideal point to construct the evaluation function of MCOOP problem, which is listed as follows.

\[
\begin{align*}
\text{Min } F(P) &= [(T(P) - T^*)^2 + (C(P) - C^*)^2]^{1/2} \\
\text{s.t.} \{ &\text{Rep}(P) \geq \text{Rep}_0 \\
& \text{R}(P) \geq R_0
\end{align*}
\]

Where \( (T^*,C^*) \) is the ideal point. \( T^* \) and \( C^* \) can be obtained respectively by objective function \( T(P) \) and \( C(P) \) under the constraint conditions.

\[
T^* = \min\{T(P) | \text{Rep}(P) \geq \text{Rep}_0, \text{R}(P) \geq R_0\}
\]

\[
C^* = \max\{C(P) | \text{Rep}(P) \geq \text{Rep}_0, \text{R}(P) \geq R_0\}
\]

As for the problem of multi-objective service composition optimization in this paper, according to the QoS aggregation method in composite service, \( T^* \) can be obtained by selecting the service for each service node which owns the shortest execution time in the service group. Similarly, \( C^* \) can be obtained by selecting the service for each service node which owns the minimum cost in the service group.

C. Principle of Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO), as an optimization technique, was first proposed by Kennedy and Eberhart in 1995 [16,17]. This algorithm was first inspired by the regularity of cluster activities of birds and fish. It uses organized social behavior to replace the mechanism of natural selection in evolutionary algorithms. Through individual collaboration between populations, the optimal solution of the problem can be searched. PSO algorithm generates an initial population by random and initializes each particle a random velocity. During the process of flying, velocities of particles are dynamically adjusted according to the flying experience of itself and companions. The population of particles always enable to fly to the better search area. Specific evolution equation, parameter meaning and parameter setting can refer to the references [16,17]. M.R. Carey et al. [18] have proved that MCOOP is an NP-Complete problem. Particle swarm optimization, as an intelligence optimization algorithm, has features of parallel computing and population optimization. Hence, particle swarm optimization is widely applied in solution of various NP-Complete problems.

D. Algorithm Design of PSO-GODSS

As for the optimization model in part B, based on particle swarm optimization, we set the corresponding
parameters. We suppose that there are m service groups and there are \( n_i \) Web services in \( S_{Gi} \), i.e., \( S_{Gi} = (s_{i1}, \ldots, s_{in_i}) \) \((i = 1, \ldots, m)\). \( S_{ij} \) is the identification number of the service in \( S_{Gi} \). We select a service for each service node, i.e., \( x_k = s_{ijk} \). A service composition candidate is constructed as an m-dimension vector \( x = (x_1, \ldots, x_i, \ldots, x_m) \), \( x_i \in [1, n_i] \).

So the service composition candidate can be coded as an m-dimension positive integer vector which is the position vector in our PSO optimization algorithm. Decoding is just the reverse process.

The basic model of PSO algorithm aims at the continuous numerical solution. From the encoding method of the candidate, it can be known that service composition optimization problem in this paper is a discrete problem. The velocity and shift models require to be improved to ensure the particles to fly inside the integer space. Therefore, to ensure that the evolutionary search is done inside the integer space, the velocity and shift models are improved as follows:

\[
\begin{align*}
  v_{id}(t + 1) &= \text{int}(\omega v_{id}(t)) + \varphi_1 + \varphi_2 \\
  x_{id}(t + 1) &= x_{id}(t) + v_{id}(t + 1)
\end{align*}
\]

Where \( \varphi_1 \in [a_1, b_1] \), \( p(\varphi_1) = 1/m_1 \), \( m_1 = [b_1 - a_1 + 1] \); \( \varphi_2 \in [a_2, b_2] \), \( p(\varphi_2) = 1/m_2 \), \( m_2 = [b_2 - a_2 + 1] \). \( 0 \leq \varphi_1 \leq 1; 0 \leq \varphi_2 \leq 1 \).

\[
\begin{align*}
  a_1 &= \begin{cases} c_1 (p_{id} - x_{id}(t)) & \text{if } p_{id} > x_{id}(t) \\
                     0 & \text{otherwise} \end{cases} \\
  b_1 &= \begin{cases} c_1 (p_{id} - x_{id}(t)) & \text{if } p_{id} > x_{id}(t) \\
                     0 & \text{otherwise} \end{cases} \\
  a_2 &= \begin{cases} c_2 (p_{gd} - x_{gd}(t)) & \text{if } p_{gd} > x_{gd}(t) \\
                     0 & \text{otherwise} \end{cases} \\
  b_2 &= \begin{cases} c_2 (p_{gd} - x_{gd}(t)) & \text{if } p_{gd} > x_{gd}(t) \\
                     0 & \text{otherwise} \end{cases}
\end{align*}
\]

In formula (5), both \( \varphi_1 \) and \( \varphi_2 \) are integers generated by uniform distribution. Equation (5) and (6) together represent the essence of basic PSO algorithm and control evolutionary search inside the integer space effectively at the same time. According to the QoS aggregation of composite service models, encoding and decoding methods, and the improved velocity and displacement modes, combined with the PSO algorithm, we design the PSO-GODOSS algorithm to solve the MCOOP problem.

The corresponding algorithm description is as follows:

**Step1:** Transform MCOOP into a single objective problem based on the method mentioned in part B;

**Step2:** Initialize \( n \) particles and set the corresponding parameters;

**Step3:** Decode the position of the particle into a candidate of composite service. Check whether the candidate satisfies the constraint conditions. If it is true, compute the fitness value of the particle with equation (3); otherwise, assign the worst fitness value for the position of the particle;

**Step4:** For the current position \( X_i \) of the particle, compare its fitness value with that of best position \( P_i \) that it experienced before. If the current fitness value is better, then let \( X_i \) be the best position \( P_i \) that it experienced;

**Step5:** For the best position \( P_i \) of the particle, compare its fitness value with that of global best position \( P_g \) that all particles experienced before. If the current fitness value is better, then let \( P_i \) be the global best position \( P_g \) that all particles experienced;

**Step6:** Update the velocity and position of the particle with equation (5) and (6). Then check whether \( X_i \) and \( V_i \) are out of boundary, which is to confine particle’s search scope and maximum velocity;

**Step7:** Repeat Step3-Step6 until all the particles are executed;

**Step8:** \( t = t + 1 \);

**Step7:** Check whether the algorithm meets the condition of termination (iteration number). If it is not the case, then return to Step3; otherwise, stop the algorithm.

The time complexity of PSO-GODOSS mainly consists of the time of its iteration. Let the iteration number be \( T \), number of particles be \( n \), number of service groups be \( m \), then the time complexity for computing fitness values is \( O(nm) \); the time complexity for updating velocity and shift is \( O(nm) \); the time complexity for comparison of finding the local and global best position is \( O(n) \). It can be seen, the main time complexity of the algorithm focuses on the computation for fitness value, because it must decode the position into a composition service candidate at first, then it computes the composite service’s QoS, and computes the fitness value at last. Therefore, the total time complexity of PSO-GODOSS algorithm is \( O(Tn^2m) \). From this theoretical analysis, time complexity of PSO-GODOSS algorithm is lower than GODSS whose time complexity is \( O(T(n^2m + n^*n)) \) where \( n^* \) is the number of auxiliary population. In GODSS algorithm, when computing the fitness value of each chromosome, they need to be sorted first according to priority relations, which makes the algorithm complexity increase. So PSO-GODOSS algorithm is simpler and easier to implement. Its convergence rate is faster, which will be verified by simulation experiments in section V.

V. SIMULATION AND ANALYSIS

This paper presents simulation experiments aiming at the example mentioned in reference [2]. The composite service flow graph of the experiment is as follows:

![Simulation example of service composition flow](image)
Each service node in the service composition flow above corresponds to a service group. QoS of Web services in each group are generated at random within a certain range. Noting that the evaluation grades are fuzzy, the set of fuzzy evaluations may as well be considered as \{very good, good, general, and bad\}. Their corresponding numerical values are \{3, 4, 3, and 2\}. It is easily known that users are more sensitive to dissatisfaction than satisfaction, i.e., users may complain much for a decreased reputation grade but are not satisfied for a same margin increased reputation grade [19]. Hence, based on this fact, the partial large Cauchy distribution membership function to quantize the grades is used by following equation.

\[
f(x) = \begin{cases} 
(1 + \alpha(x - \beta)^2)^{-1}, & 1 \leq x \leq 3 \\
\alpha \ln x + b, & 3 \leq x \leq 5 
\end{cases}
\] (7)

Where \(\alpha, \beta, a,\) and \(b\) are undetermined constants. Let \(f(5)=1, f(3)=0.8,\) and \(f(1)=0.01;\) then the quantization value of fuzzy evaluation set is \{1, 0.9126, 0.8, 0.5245\}. According to the basic service composition models and QoS aggregation method mentioned in section III, we can obtain the objective function and constraint conditions easily. According to the QoS aggregation methods, it is known that the reliability constraint should not be set too high. Supposing \(R_{0} = 0.8, R_{0} = 0.5\) in this experiment, we verify the feasibility and effectiveness of PSO-GODSS algorithm through both execution time and convergence rate of the algorithm. The configuration of our microcomputer is as follows: Core i3 550 3.2GHz Dual Core processor, 2GB memory, Windows 7 32-bit operating system. The algorithm is implemented by Matlab 7.8. We set that the number of service groups is 10, number of iterations are 100, 200, 300, and 400 respectively in our experiment. In this experiment, we compare the execution time of PSO-GODSS algorithm and GODSS algorithm mentioned in reference [2], which is showed in Figure 2.

![Figure 2. Computation time of PSO-GODSS VS GODSS](image2)

It can be seen that the execution time of our method is lower than GODSS, which proves the feasibility and effectiveness of PSO-GODSS algorithm from the aspect of execution efficiency. Furthermore, we compare the convergence rate by showing the change of average fitness value with iteration by both algorithms. The convergence comparison is illustrated in Figure 3 which shows that the convergence rate of our approach is faster than that of GODSS. To be more specific, our approach is convergent at 51st generation but GODSS at 83rd generation which further proves the feasibility and effectiveness of PSO-GODSS algorithm from the aspect of convergence rate.

![Figure 3. Convergence rate of PSO-GODSS VS GODSS](image3)

VI. CONCLUSION

Web service selection is a critical issue in dynamic Web service composition. To enhance the execution time of existing algorithm on dynamic Web service selection with global optimal QoS to satisfy user’s requirement better, based on particle swarm optimization, we proposed the PSO-GODSS algorithm to achieve the goal. The problem of dynamic Web Service selection with global optimal QoS is transformed into a multi-objective service composition optimization problem with QoS constraints, which is further transformed into a single-objective problem by using the method of ideal point, so that the time execution of our algorithm can be improved considerably. A set of optimal service composition processes satisfying the QoS constraints can be generated by our approach through setting the number of iteration appropriately. Theoretical analysis and experimental results show the feasibility and effectiveness of our algorithm, and the execution efficiency and convergence rate of PSO-GODSS is much better than the existing multi-objective genetic algorithm.

ACKNOWLEDGMENTS

The work was supported by the National Natural Science Foundation of China under grant No.90818004 and No.61100054, Program for New Century Excellent Talents in University under grant No.NCET-10-0140, Excellent Youth Foundation of Hunan Scientific Committee No.11JJ1011, and Scientific Research Fund of Hunan Educational Committee under grant No.09K085 and No.11B048.
REFERENCES


