Int J Adv Manuf Technol (2011) 56:1167–1179
DOI 10.1007/s00170-011-3230-9

ORIGINAL ARTICLE

An approach for composite web service selection based on DGQoS

Zhi-Jian Wang · Zhi-Zhong Liu · Xiao-Feng Zhou · Yuan-Sheng Lou

Received: 23 September 2010 / Accepted: 9 February 2011 / Published online: 1 March 2011

Abstract Virtual enterprise is a temporary network of independent companies or enterprises that can quickly bring together to fulfill a value-added task. With the development of Web service technologies, the enterprise business systems can be encapsulated as Web services. Establishing a virtual enterprise is actually a process of Web services composition. As more and more Web services are available, many Web services with different quality of service (QoS) can be found for tasks of the composite Web service. Thus, a significant research problem in Web service composition is how to select the correct Web services for tasks such that the composite Web service gives the best overall quality. In this paper, we propose an approach to achieve Web services selection efficiently with excellent comprehensive quality. Firstly, a comprehensive evaluation model based on generic QoS (GQoS) and domain QoS (DQoS) of composite Web service is established. Secondly, a novel optimization algorithm named culture max–min ant system (C-MMAS) is constructed to solve the problem of Web service selection based on DGQoS. Simulations show that when used in Web service selection, C-MMAS is better than ant colony optimization algorithm and MMAS in efficiency.

Keywords Virtual enterprise · Quality of service · Composite web service · Max–min ant system · Culture algorithm

Z.-J. Wang · Z.-Z. Liu (✉) · X.-F. Zhou · Y.-S. Lou
College of Computer and Information Engineering,
Hohai University,
Nanjing 210098, China
e-mail: lzzmff@126.com

1 Introduction

In response to increasing international competition, enterprises have been investigating new ways of cooperating with each other to cope with today's unpredictable market behavior. Advanced developments in information and communication technology enabled reliable and fast cooperation to support real-time alliances. In this context, the virtual enterprise (VE) represents an appropriate cooperation alternative and competitive advantage for the enterprises [1]. VE is a temporary network of independent companies or enterprises that can quickly bring together to fulfill a complex task. In this temporary enterprise organization, the good information interactive ability is the key point in order to enable all participating enterprises to share the corresponding information of designing, producing, and selling and guarantee them to have a better cooperation [2, 3]. Research works on fuzzy control for interconnected systems provide strong technical support for constructing VE [4–6].

Web service is a new type of distributed computing model, with traits of self-contained, modular, loose coupled, standards-based, high capacity, etc. Web service can provide a kind of distributed information sharing computing technology, which can be used on Internet or Intranet by standard XML protocols and message formats to eliminate the discrepancy of different application systems among enterprises [7, 8]. With the development of Web service technologies, the enterprise business systems can be encapsulated as Web services. Establishing a virtual enterprise is actually a process of Web services composition. According to certain logical business, a group of existing Web services can be integrated into a value-added service to complete complex functions.

As more and more Web services are available, many Web services with different quality of service (QoS; such as price, response time, availability, reliability, reputation, security, and so on) can be discovered for each task of the composite Web service; thus, it is necessary to select the correct Web services for tasks to ensure that the overall quality of the composite Web service is better even best. There are two critical issues should be considered in the process of QoS-aware Web service selection.

1.1 Devise efficient web service selection algorithm

QoS-ware Web service selection is a combinatorial optimization problem, and finding the best solution is non-deterministic polynomial time hard [9]. So, the first critical issue for this problem is to devise an efficient Web service selection algorithm. In order to solve this problem, several approaches have been proposed in recent years. In [10], multidimensional multi-choice knapsack problem has been used to model this problem. In works [11–15], linear and nonlinear programming methods are used to find the optimal solution for the problem of Web service selection. Ardagna and Pernici [16] formalize the service selection problem as a mixed integer linear programming problem. Paper [17] adopts integer programming approach to solve the service selection problem. These optimization methods are very effective when the size of the problem is small. However, these methods suffer from poor scalability due to the exponential time complexity.

To achieve higher scalability, genetic algorithm and ant colony optimization algorithm have been used to solve this problem. In [18], authors advocate the use of GA, for its advantage of permitting to deal with QoS attributes having nonlinear aggregation functions and for its good scalability that can solve the problem when the number of concrete services increases. However, in practice, GA emerges some shortcomings, such as premature phenomena and low local optimization ability. So, the traditional GA is not the best solution to a certain question. To conquer those flaws of traditional GA, some improved GAs have been used to solve the problem of QoS-aware Web service selection.

Tang and Ai [19] provide a hybrid genetic algorithm for the Web service selection problem. When selecting Web services, both dependency constraints and conflict constraints between component Web services are considered. Paper [20] uses simulated annealing-based genetic algorithm to solve the problem of quality of experience (QoE)/QoS-aware Web services selection. Simulations show that SGA is better than GA and SA in efficiency.

Fang et al. [21] apply the multi-objective ant colony optimization (MOACO) algorithm to solve the problem of dynamic Web service selection. They propose a model of Web service selection with QoS global optimization and convert this problem into a multi-objective optimization problem with user constraints. They have compared MOACO with multi-objective genetic algorithm [22]; experimental results prove that ACO outperforms GA. However, the ACO algorithm also has some shortcomings such as easily being in a partial optimization and slow convergence speed that ant colony algorithm brings. So, it is necessary to devise an algorithm which is more effective for the problem of QoS-aware Web service selection.

1.2 Design evaluation model for web service

As more and more Web services with the same functionality and different QoS are available, various composite Web service with different QoS can be constructed. How can we get the better even best composite Web service? Many researchers use QoS to resolve this problem. So, the second critical issue of QoS-aware Web service selection is to design a QoS evaluation model that can evaluate composite Web service comprehensively.

QoS attributes presenting non-functional characteristics of Web services include cost, response time, availability, reliability, reputation, security, successful execution rate, and so on. These attributes are public to every Web service, so we call them generic QoS (GQoS). In addition to GQoS, Web services also have domain quality of service (DQoS). DQoS of Web services are attributes of domains that the services belong to [23, 24]. For example, the domain attributes of hotel reservation services are occupancy rate, cleanliness, quiet level, comfort level, security, and transportation convenience. The domain attributes of flight booking services are punctuality, security, comfort level, and so on. The domain attributes of media streaming services may be continuity, integrity, and veracity. DQoS of Web service is also important to users. Take the hotel reservation services for example, users often choose the service with a higher occupancy rate even if the response time of the service is relatively long. So, it is necessary to take the DQoS into consideration when evaluating Web services.

Existing QoS evaluation models [10–18, 21, 22] evaluate services primarily based on the GQoS criteria. Evaluation model provided in paper [14] is the representative of such models. In this model, five GQoS criteria for elementary services are considered: execution price, execution duration, reputation, successful execution rate, and availability, and a five-dimensional GQoS evaluation model based on the five attributes is presented. Wang et al. [25] proposes a QoS-aware service selection model based on fuzzy linear programming technologies, in order to identify their dissimilarity on service alternatives, assist service consumers in selecting most suitable services with consideration of their expectations and preferences.

Gao et al. [20] provides a hybrid evaluation model named QoE/QoS for composite Web service selection. QoE indicates the general acceptability of composite Web service, which represents the customers' subjective perception. Guo et al. [8] have studied the correlations between component Web services and presented a correlation QoS-aware model for Web services. This model has been used to compute the correlation value between cooperative partners of virtual enterprise. Liu et al. [26] propose a domain-oriented service evaluation model; it evaluates the candidate service by considering the evaluation result from domain expert and the constraint information from service consumer. However, this evaluation model needs the participation of experts and only can be used for evaluating elementary Web services.

Existing QoS evaluation models are useful to users when they select Web service. Whereas these models do not consider the DQoS of composite Web service, this will cause that DQoS of the selected composite Web service is very poor. Moreover, these models treat all the GQoS constraints as hard constraints. With the growing of categories of GQoS attributes, more and more GQoS constraints will be proposed. If all the GQoS constraints are treated as mandatory constraints, it will be difficult to find a composite Web service that can satisfy all the GQoS constraints. Then, the over-constrained problem will take place. So, how to design a comprehensive evaluation model that can evaluate the DQoS of composite Web services and conquer the over-constrained problem is a burning question.

With the whole consideration of the above two critical issues, this paper devises a comprehensive evaluation model and a novel optimization algorithm for the Web service selection problem. The main contributions of the paper are as follows: Firstly, it designs a comprehensive evaluation model based on evaluation models of DQoS and GQoS. In order to solve the over-constrained problem, a GQoS evaluation model based on constraint hierarchies [27] is constructed. Furthermore, a DQoS evaluation model is designed for the purpose of evaluating the domain quality of composite Web services.

Secondly, it devises a novel optimization algorithm named culture max–min ant system (C-MMAS) for the problem of QoS-aware Web service selection by integrating improved MMAS [28] into the framework of culture algorithm (CA) [29]. Experiment results show that C-MMAS is efficient in solving the Web service selection problem.

The rest of this paper is organized as follows: Section 2 formulates the problem of QoS-aware Web service selection. Section 3 describes the comprehensive evaluation model for composite Web services. Section 4 introduces the C-MMAS algorithm and specifies the process of Web service selection based on DGQoS. Section 5 presents experiment results. Finally, Section 6 concludes the paper.

2 Problem formulation

Web service composition with multiple combination paths can be illustrated as Fig. 1. There are four different

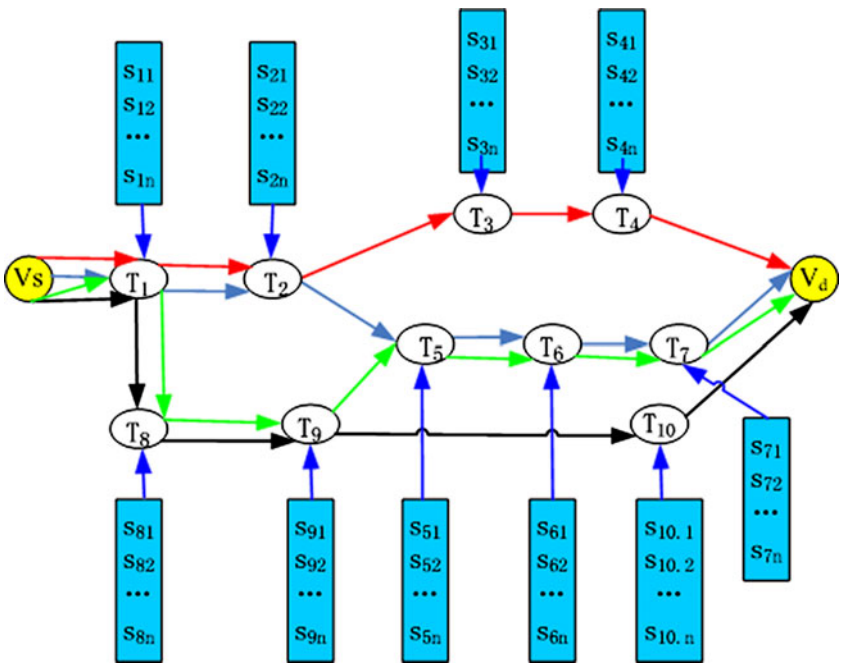


Fig. 1 Web service composition graph

combination paths that can finish the same composite function, where V_s and V_d are virtual starting point and destination point. $t_i(1 \le i \le 10)$ denotes a single task of the composite Web service. Each task t_i has a set of candidate Web services with identical functionalities and different QoS values. The process of constructing a composite Web service can be illustrated as Fig. 2; concrete Web services are selected from the candidate service sets for each task. As Fig. 2 illustrated, service s_{12} is selected for task t_1, service s_{2n} is selected for task t_2, service s_{32} is selected for task t_3, and service s_{4n} is selected for task t_4. Then, $<s_{12}, s_{2n}, s_{32}, s_{4n}>$ is a composite Web service that can complete the composition function.

The number of composite Web services can be constructed from the combination path in Fig. 2 is n^4. If the four combination paths described in Fig. 1 are taken into consideration, even more composite Web services can be constructed. How to find the correct Web services for each task such that the comprehensive quality of the composite Web service is better even best? This is the problem the paper aims to solve.

3 Comprehensive evaluation model

For the purpose of evaluating composite Web services comprehensively, we design a comprehensive evaluation model based on DQoS and GQoS of composite Web services. The DQoS evaluation model is presented in Section 3.1. The GQoS evaluation model is introduced in Section 3.3. Finally, the comprehensive evaluation model is constructed based on the evaluation models of DQoS and GQoS.

3.1 Evaluation model of DQoS

A composite Web service is composed of several component services belonging to different domains, which have different domain attributes because values of different domain attributes cannot be aggregated directly. Thus, we have to compute the evaluation values of DQoS of component services firstly and then compute the evaluation value of DQoS of the composite Web service. The complexity of evaluating DQoS of component services is that DQoS attributes have variety data types, which can be summarized as numeric type, interval, linguistic type, and level type. DQoS attributes also can be divided into the cost type and benefit type. The meaning of the cost type is that the lower the value is, the better the quality will be. The meaning of the benefit type is that the higher the value is, the better the quality will be. In order to unify the dimension and direction of different quality and metric, it is necessary to define a standardized processing to the domain quality before calculating the comprehensive evaluation value of the combination scheme. We defined the standardized processing as follows:

1. Numeric type: Assume that the maximum and minimum values of certain DQoS attribute are $u_{\max}$ and $u_{\min}$, the value of the attribute is x, and then the standardization formula for this kind of data type is defined as Eq. 1.

$$x' = \begin{cases} \dfrac{x - u_{\min}}{u_{\max} - u_{\min}}, \text{Characteristic} = \text{Benefit} \\ \dfrac{u_{\max} - x}{u_{\max} - u_{\min}}, \text{Characteristic} = \text{Cost} \end{cases} \quad (1)$$

2. Interval type: Suppose that $[a, b]$ is the value range of certain DQoS attribute, the value of the attribute is $[x, y]$ and $0 \le a \le x < y \le b$. The standardization formula for this kind of data type is defined as Eq. 2:

$$x_2' = \begin{cases} \frac{x-a}{b-a} + \frac{y-a}{b-a} & \text{Characteristic} = \text{Benefit} \\ \frac{b-x}{b-a} + \frac{b-y}{b-a} & \text{Characteristic} = \text{Cost} \end{cases} \quad (2)$$

3. Linguistic type: Assume that $I=<i_0, \ldots, i_j, \ldots, i_m>$ indicates a set of linguistic values from inferior to supper, such as <poor, fair, fine, excellent>. This kind of data type should be described as triangular fuzzy number firstly [30] and then transforms the triangular fuzzy number to a real number according to Eq. 3,

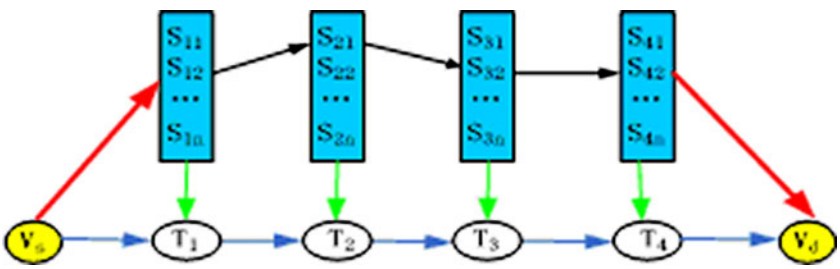


Fig. 2 Instantiation of one combination path

where $\mu(x)$ is the membership function of R and S is the integral interval.

$$R = \frac{\int_S x\mu(x)\mathrm{d}x}{\int_S \mu(x)\mathrm{d}x} = \frac{\int_{R^L}^{R^M} x\mu(x)\mathrm{d}x + \int_{R^M}^{R^U} x\mu(x)\mathrm{d}x}{\int_{R^L}^{R^M} \mu(x)\mathrm{d}x + \int_{R^M}^{R^U} \mu(x)\mathrm{d}x} \tag{3}$$

Assume that i_j is the value of the DQoS attribute, R_0 and R_m are the real numbers corresponding to i_0 and i_m, and R_j is the real number corresponding to i_j. Standardization formula for this kind of data type is defined as Eq. 4.

$$R_j^{'} = \frac{R_j - R_0}{R_m - R_0} \tag{4}$$

4. Level type: Suppose that $G=<g_0, \ldots, g_i, \ldots, g_m>$ is a set of evaluation values ordered from inferior to supper and g_i is the metric value of a DQoS attribute; the standardization formula for this kind of metric type is defined as Eq. 5, where i is the ordinal number of g_i and m is the total number of the elements of set G.

$$g_i^{'} = \frac{i}{m} \tag{5}$$

When evaluating the DQoS of a component Web service, we should firstly standardize all the values of DQoS attributes and then calculate the evaluation value of DQoS according to formula 6.

$$\mathrm{DQoS} = \sum_{i=1}^{k} w_i \times \mathrm{d}q_i \tag{6}$$

where $\mathrm{d}q_i$ is the standardized value of the ith DQoS attribute and w_i represents the importance of the ith DQoS attribute. The value of w_i can be provided by user or extracted automatically by preference learning method [23].

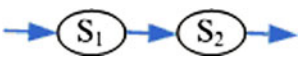
3.2 Aggregation formulas for DQoS evaluation values

Evaluation value of DQoS of a composite Web service can be obtained according to evaluation values of DQoS of component Web services and the combination patterns. Presently, there are four main combination patterns, namely sequential, parallel, switch and loop. DQoS aggregation formulas under these combination patterns are defined as follows:

1. The sequential pattern is illustrated as Fig. 3. Aggregation formula for this combination pattern is defined as Eq. 7.

$$\mathrm{DQoS}_{\mathrm{Sequ}} = \mathrm{DQoS}_1 + \mathrm{DQoS}_2 \tag{7}$$

Fig. 3 Sequential



2. The parallel pattern is described as Fig. 4. Aggregation formula for this combination pattern is defined as Eq. 8.

$$\mathrm{DQoS}_{\mathrm{Para}} = \min\{\mathrm{DQoS}_1, \cdots, \mathrm{DQoS}_n\} \tag{8}$$

3. The switch pattern is similar to the parallel pattern and illustrated as Fig. 5. Aggregation formula for this combination pattern is defined as Eq. 9, where p_i is the probability that the service s_i will be selected and $\sum_{i=1}^{n} p_i = 1$.

$$\mathrm{DQoS}_{\mathrm{Swit}} = \sum_{i=1}^{n} p_i \times \mathrm{DQoS}_i \tag{9}$$

4. The loop pattern is described as Fig. 6. Aggregation formula for this combination pattern is defined as Eq. 10, where k is the times of the loop.

$$\mathrm{DQoS}_{\mathrm{Loop}} = K(\mathrm{DQoS}_i) \tag{10}$$

3.3 Evaluation model of GQoS

In order to conquer the over constrained problem, a GQoS evaluation model based on constraint hierarchies is designed. In this model, GQoS constraints are described as hard GQoS constraint hierarchy and soft GQoS constraint hierarchies according to user preferences. Constraints in the hard constraint hierarchy are constraints that user concerns mostly and must be satisfied. Those in soft constraint hierarchies are user preferred requirements; these constraints are not required to be satisfied necessarily, but system must try to satisfy them if possible.

Suppose that there are n global GQoS constraints proposed by user, these GQoS constraints are described as constraint hierarchies $H = \{H_0, H_1^{w_1}, \ldots, H_k^{w_k}\}$ according to user GQoS preferences. $H_0 = \{c_{01}, c_{02}, \ldots, c_{0n_0}\}$, $H_1^{w_1} = \{c_{11}, c_{12}, \ldots, c_{1n_1} | w_1\}$, $\ldots$, $H_k^{w_k} = \{c_{k1}, c_{k2}, \ldots, c_{kn_k} | w_k\}$, and $n_0 + n_1 + \ldots + n_k = n$. H_0 is the hard constraint hierarchy; $H_1^{w_1}, H_2^{w_2}, \ldots, H_k^{w_k}$ are soft constraint hierarchies. $w_i (1 \le i \le k)$ denotes user GQoS preference for the ith soft constraint hierarchy. Assume that CWS is a composite Web service, the function for calculating the deviation value

Fig. 4 Parallel

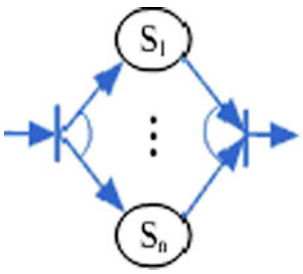
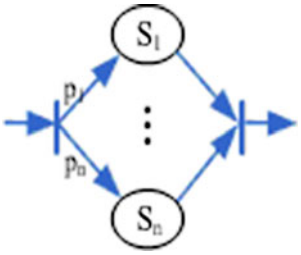


Fig. 5 Switch



between GQoS values of cws, and soft GQoS constraints is defined as Eq. 11.

$$\mathrm{Dev}(\mathrm{CWS}) = \sum_{i=1}^{k} w_i \times d_i \times \frac{1}{N_i + 1} \tag{11}$$

$$d_i = \sum_{j=1}^{n_j} e_{ij}(c_{ij}, \mathrm{cws}.q_j), e_{ij}(c_{ij}, \mathrm{cws}.q_j) = \frac{|c_{ij} - \mathrm{cws}.q_j|}{c_{ij}} \times \theta,$$

$$\theta = \begin{cases} 0 & \text{if cws}.q_j \text{ satisfy } c_{ij} \\ 1 & \text{if cws}.q_j \text{ cannot satisfy } c_{ij} \end{cases}$$

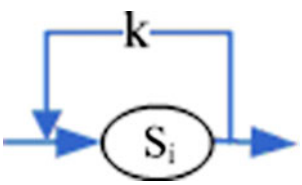
where c_{ij} is the jth GQoS constraint in the soft constraint hierarchy H_i and cws.q_j is the value of the GQoS attribute involved in constraint c_{ij}. e_{ij} indicates the difference between constraint c_{ij} and GQoS value cws.q_j. If the GQoS value cws.q_j can satisfy the constraint c_{ij}, then θ=0. Otherwise, θ=1. d_i is the sum of all the differences between GQoS values of CWS and GQoS constraints in H_i. N_i is the number of constraints in H_i satisfied by GQoS values of CWS.

The comprehensive evaluation model of composite Web services based on evaluation models of DQoS and GQoS is defined as Eq. 12.

$$\mathrm{DG}_{\mathrm{QoS}}(\mathrm{CWS}) = w \times \mathrm{Dev}(\mathrm{CWS}) + (1 - w) \times (1 - \mathrm{DQoS}_{\mathrm{CWS}}) \tag{12}$$

where w indicates the importance of the deviation value between GQoS values and GQoS constraints and $(1-w)$ represents the importance of domain quality of the composite Web service. $\mathrm{DQoS}_{\mathrm{CWS}}$ is the evaluation value of DQoS of the composite Web service. $\mathrm{DG}_{\mathrm{QoS}}$(CWS) is the comprehensive evaluation value of the composite Web service; the smaller the value of $\mathrm{DG}_{\mathrm{QoS}}$(CWS) is, the better the composite Web service will be.

Fig. 6 Loop



4 C-MMAS for web service selection based on DGQoS

In order to solve the problem of Web service selection more efficiently, a novel optimization algorithm named C-MMAS is constructed by integrating improved MMAS into the framework of CA and is applied to solve the problem of Web service selection based on DGQoS. In this section, we firstly present the improvements we have made on the MMAS algorithm. Then, introduce the computing process of C-MMAS. Finally, C-MMAS for Web service selection based on DGQoS is briefly described.

4.1 Improved max–min ant system

Max–min ant system comes from ACS [31] and makes three improvements on it: (1) The pheromone intensity is limited between $\tau_{\max}$ and $\tau_{\min}$; (2) in MMAS, only the best ant in iteration or the best ant so far is permitted to deposit the pheromone; and (3) before MMAS starts searching, let the pheromone intensity of all the ways be the maximum value and combined with a pheromone evaporation rate. It was reported that MMAS was the best performing ant system approach. However, the transition and pheromone updating rules of MMAS still have some flaws.

In the MMAS algorithm, ants transfer to the next node with the largest transition probability. If always adopting this transition rule, majority of the ants will flock to the same path and will weaken the searching capability of MMAS. In order to overcome this flaw, we take the roulette wheel model as the transfer rule.

$$P_{ij} = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\gamma}}{\sum_{s \in \text{ allowed}} [\tau_{is}(t)]^{\alpha}[\eta_{is}(t)]^{\gamma}} & j \in \text{allowed} \\ 0 & j \notin \text{allowed} \end{cases} \tag{13}$$

For the problem studied in this paper, Eq. 13 is the formula for calculating the selection probabilities of candidate services, where $\tau_{ij}(t)$ and $\eta_{ij}(t)$ are pheromone intensity and heuristic information between two candidate Web services with combination relationship. Here, define $\eta_{ij}(t) = \frac{1}{\mathrm{DG}_{\mathrm{QoS}}(\mathrm{CWS}_{\mathrm{sub}})+1}$, where $\mathrm{CWS}_{\mathrm{sub}}$ is a sub-composite Web service. $\mathrm{DG}_{\mathrm{QoS}}(\mathrm{CWS}_{\mathrm{sub}})$ is the comprehensive evaluation value of $\mathrm{CWS}_{\mathrm{sub}}$. α and γ denote the importance of pheromone intensity and heuristic information, respectively.

The MMAS algorithm only allows the best-so-far or iteration-best ant to deposit pheromone. This will cause that the pheromone of a solution which is not the global optimal one constantly increasing and will bring down the convergence speed. In order to overcome this flaw, a new global pheromone updating rule is designed: After the ant colony completing a full search, increase the pheromone intensity of the top λ ants according to Eq. 14 and attenuate other solutions' pheromone intensity according to Eq. 16. The

implication of this updating rule is that the better the solution is, the greater increment of pheromone will be added to the path of the solution.

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{k}(t) \tag{14}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{1}{\mathrm{DG_{QoS}}(\mathrm{CWS_{sub}})+1}, & \text{if } e_j \in L_k \\ 0, \quad \text{otherwise} \end{cases} \tag{15}$$

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) \tag{16}$$

where $\rho(0<\rho<1)$ is the retention factor of global pheromone. $\Delta\tau_{ij}^{k}(t)$ denotes the increment of pheromone of the kth solution. $\mathrm{DG_{QoS}}(\mathrm{CWS}_k)$ is the comprehensive evaluation value of the kth solution. L_k is the path of the kth ant. e_j is the edge of L_k. Maximum and minimum pheromone intensity formula are defined as follows [29]:

$$\tau_{\max} = \begin{cases} Q, & \text{at beginning} \\ \dfrac{1}{\rho}\dfrac{1}{\mathrm{DG_{QoS}}(\mathrm{CWS_{best-so-far}})}, & \text{otherwise} \end{cases} \tag{6}$$

$$\tau_{\min} = \frac{\tau_{\max}}{20} \tag{7}$$

where Q is a constant, $\mathrm{CWS_{best\text{-}so\text{-}far}}$ is the best solution found so far.

4.2 Culture max–min ant system

Cultural algorithm is a class of computational models derived from observing the cultural evolution process in nature. It is a dual inheritance system that characterizes evolution in human culture at both the macro-evolutionary level, which takes place within the belief space, and at the micro-evolutionary level, which occurs in the population space. Knowledge produced in the population space at the micro-evolutionary level is selectively accepted or passed to the belief space and used to adjust the knowledge structures there. This knowledge can then be used to influence the changes made by the population in the next generation. Peng [32] has proved that evolution under the influence of culture is far better than the evolution solely relying on biological genetic evolution. In this paper, C-MMAS optimization algorithm is constructed by integrating improved MMAS into the framework of CA. The calculation framework of C-MMAS is described as Fig. 7.

The computing process of C-MMAS can be described as follows: Firstly, solutions are generated by the MMAS algorithm in population space, and then all the solutions are evaluated by the Evaluate() function. After that, several excellent solutions are absorbed as knowledge to the belief space by Accept() function, and the knowledge in the belief space is updated with the Update() function. Next, mutation (Evolve()) is carried out on all the solutions in the population space. Then, evaluate the new solutions produced by mutation, absorb excellent solutions to the belief space, and update the knowledge again. After the knowledge has been updated for every η times, use the knowledge to guide the search of ants through Influence() function. All the functions mentioned above are defined as follows:

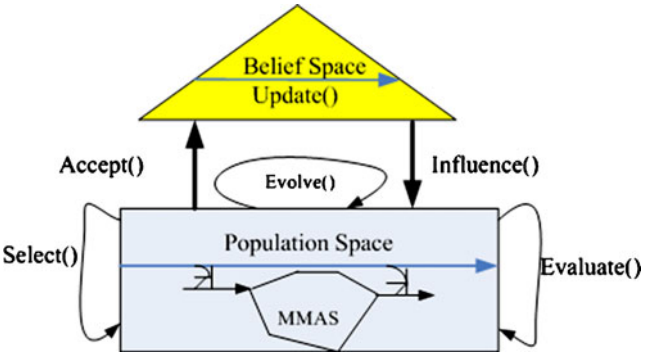


Fig. 7 Computing framework of C-MMAS algorithm

1. Evaluate(): Compute the comprehensive evaluation value of each solution, and the function is defined as Evaluate()=$\mathrm{DG_{QoS}}$(CWS).
2. Accept(): Absorb k excellent solutions to the belief space. Here defines Accept()=ceil($m\times10\%$), where m is the number of solutions and ceil(x) is an integral function.
3. Update(): Replace better solutions absorbed from the population space with the poor solutions in the belief space.
4. Evolve(): Carry out mutation operation on all the solutions in the population space. That is to randomly select a component service s_{ik} from solution $<s_{1i}, s_{2j},\ldots, s_{ik},\ldots,s_{mt}>$, then randomly select a service s_{iq} from the candidate service set which the service s_{ik} belongs to and replace s_{iq} with s_{ik}. Two-point mutation operation is adopted here.
5. Influence(): Replace solutions in belief space with the poor solutions in population space.

4.3 Process of web service selection based on DGQoS

For the convenience of further discussions, some hypotheses are declared first: The directed graph of Web service composition has been given. Candidate Web services for each task have been discovered. Values of GQoS attributes of candidate services are provided by service providers. Values of DQoS attributes of candidate services have been

C-MMAS for Web Service Selection Based on DGQoS

Step1 Initialization

1: Input: $A, B_{cs}, C_{GQoS}, D_{DQoS}, H$

2: Initialize: $P_{pl}, S_{ij}, m, \alpha, \gamma, \rho, Q, \lambda, \eta, N_{\max}$

Step2 One Complete Search of All the Ants

3: **For** k=1 to m **Do** /*m is the number of ants*/
4: {Set TN[k]= starting point} /* Firstly, put all the ants at the starting vertex of the graph. TN[k] is used to record the task that the k^{th} ant gets to.*/
5: **End For**
6: **For** t=1 to p **Do** /*p is the total number of tasks of the composition Web service*/
7: **For** i=1 to m **Do** /*m is the number of ants*/
8: **While** (The task recorded in TN[i] is not the destination of the combination graph) **Do**
9: {Find tasks that have combination relationship with the task recorded in TN[i] according to the adjacency matrix A} /*determine the tasks that the i^{th} ant will visit*/
10: **For** (j=1 to q) **Do** /* for all the candidate Web services of the tasks which have combination relationship with the task recorded in TN[i]*/
11: {Calculate GQoS values of the sub-CWS constructed by the j^{th} candidate service and services recorded in the path table of the i^{th} ant }
12: **If** (GQoS values of the sub-CWS can satisfy GQoS hard constraints) **Then**
13: {Calculate the evaluation value of DQoS of the j^{th} candidate service and computer the comprehensive evaluation value of the sub-CWS }
14: **End If**
15: **Else**
16: {Make a mark on the j^{th} service to indicate that it can't be selected by the i^{th} ant}
17: **End If**
18: **End For**
19: **If** (Exist candidate Web services that are not marked) **Then**
20: {Calculate the selection probability of these services according to Eq(13) and use roulette wheel model to select a service and put it into the path table of the i^{th} ant; Put the task corresponding to the selected service into TN[i]}
21: **End If**
22: **If** (All the candidate Web services are marked) **Then**
23: {The i^{th} ant stops searching}
24: **End If**
25: **End While**
26: **While** (The task recorded in TN[i] is the destination of the combination graph) **Do**
27: {The i^{th} ant stops searching; create a solution by services stored in the path table of the i^{th} ant.}
28: **End While**
29: **End For**
30: **End For**

Step3 Evolution under the Guidance of Knowledge

31: Evaluate all the solutions obtained form step 2 with the *Evaluate()* function;
32: **Then**
33: {Accept *k* excellent solutions from the population space to the belief space and update the knowledge in the belief space}
34: **For** t=1 to h **Do** /* h is the number of solutions obtained form step 2*/
36: {Carry out mutation on the t^{th} solution and compute the GQoS values of the new solution produced by mutation}
37: **End For**
38: **For** t=1 to h **Do**
{Check the GQoS values of the t^{th} solution produced by mutation whether can satisfy GQoS hard constraints; If yes, calculate the comprehensive evaluation value of the t^{th} solution; If no, let the comprehensive evaluation value of the t^{th} solution be a larger value}
39: **End For**
40: **Then**
41: {Carry out the Accept() and Update() operations}
42: **If** (The knowledge in the belief space has been updated for η times) **Then**
43: {Execute the *Influence()* operation and update the global Pheromone intensity according to Eq(14) and Eq(16)}
44: **End If**
45: **Else**
46: {Update the global pheromone intensity according to Eq(14) and Eq(16)}
47: **End If**

Step4 Determine Whether to Stop

48: **If** (Iteration Num = N_{max}) **Then**
49: {Output the optimal solution in the belief space and stop computing}
50: **End If**
51: **Else**
52: {Iteration Num =Iteration Num+1 and Go to Step2}
53: **End If**

standardized according to Eqs. 1–5. GQoS constraint hierarchies have been constructed. Here, the adjacency matrix is used to describe the Web service composition graph. In the process of Web service selection, the adjacency matrix is used to determine whether two tasks have combination relationship. In the matrix, when a_{ij}=1, it denotes that task t_i and task t_j have sequential combination relationship. When a_{ij}=0, it denotes that task t_i and task t_j do not have combination relationship.

Firstly, put all the ants on the starting point of the combination graph, and then all the ants begin to search solutions along the directed path of the combination graph. Every ant has a path table used to record the services selected for tasks that it has visited. After getting to the destination of the combination graph, services recorded in the path table of each ant can construct a feasible solution for the problem. After the stop criterion is verified, take the best solution in the belief space as the optimal solution. C-MMAS for Web service selection based on DGQoS is described in detail below.

Where A is the adjacency matrix of Web service composition graph, B_{cs}, C_{GQoS}, D_{DQoS}, P_{PI}, S_{ij} are binary arrays. B_{cs} is used to store candidate services, C_{GQoS} and D_{DQoS} store GQoS values and DQoS values of candidate Web services, P_{PI} records the pheromone intensity between candidate Web services which has combination relationship, and S_{ij} are path tables of ants. H is the GQoS constraint hierarchies. m is the number of ants. N_{max} is the maximum iteration number of the C-MMAS algorithm.

5 Experiments

5.1 Experiment design

In this experiment, C-MMAS algorithm is applied to solve the problem described in Section 2. There are ten sets of candidate services for the ten tasks. Each set has 50 services. Nine kinds of GQoS attributes including Cost, Response Time, Availability, Reliability, Reputation, Security, Successful Execution Rate, Load and Throughput are considered. GQoS values of candidate services are randomly generated within certain ranges. Value ranges of those QoS attributes are 0≤Cost≤$100, 0<RT≤10Sec, 0.75<Rel≤1, Rep ∈ {1, 2, 3, 4, 5}, 0.75<SER≤1, Rep ∈ {1, 2, 3, 4, 5}, Sec ∈ {1, 2, 3, 4, 5}, 0≤Load≤500, 0≤ ThrP≤500 kbyte/s. Aggregation formulas under the sequential pattern are described in Table 1. Suppose that GQoS constraint hierarchies is $H = \{H_0, H_1^{0.85}, H_2^{0.5}\}$. $H_1^{0.85} = \{\text{Ava} \geq 0.80, \text{SER} \geq 0.80, \text{Rep} \geq 4 | 0.85\}$, $H_2^{0.5} = \{\text{Sec} \geq 4, \text{Load} \leq 250, \text{Thrp} \geq 350 | 0.5\}$. H_0 is the hard constraint hierarchy. $H_1^{0.85}$ and $H_2^{0.5}$are soft constraint hierarchies with weight 0.85 and 0.5. Assume that each class of Web service have five kind of DQoS attributes, to simplify the experiment, values of DQoS attributes are randomly generate in interval [0, 1], and the weight of each DQoS attribute is 0.2. The C-MMAS algorithm is realized with C++ programming language, and the experimental environment is personal computer with following configures: CPU Pentium(R) 4, 2.66 GHz; Memory 512M; and OS Windows XP, 2002.

5.2 Values determination for key parameters

As the performance of C-MMAS depends on the values of key parameters, it is necessary to determine suitable values for these parameters firstly. The key parameters of C-MMAS are α, γ, ρ, λ, and η. α indicates the importance of pheromone intensity. γ denotes the importance of heuristic information. ρ is the retention factor of global pheromone, and λ is the number of solutions involved in the pheromone updating rule. η is a constant used to decide whether to apply the knowledge to guide the search of the ants. In this experiment, testing method is adopted to determine most suitable values for these parameters. Through experimenting, it is found that setting different values for these parameters only affect the searching capability of C-MMAS but do not affect its running speed. So, in the following experiments, we only record comprehensive evaluation values of solutions searched by the C-MMAS algorithm and do not record the running times.

Initially, set m=50, ρ=0.5, Q=1, and N_{max}=100. Define α=1.0, $\gamma=\mu\alpha$, and test to determine the appropriate ratio between α and γ, that is, to find a suitable value for μ. Set μ=1, 2, 3, 4, 5, 6, 7, 8, 9, 10, respectively. Implement the algorithm and record the solutions searched by the algorithm when setting different values for μ. Experimental results are described in Fig. 8, where the vertical axis represents the comprehensive evaluation values of solutions and the horizontal axis denotes values set for μ. From

Table 1 Aggregation formulas under the sequential pattern

(1) Cost	(2) ResponseTime	(3) Ava	(4) Rel	(5) SER
$C = \sum_{i=1}^{k} C_i$	$T = \sum_{i=1}^{k} T_i$	$\text{Ava} = \prod_{i=1}^{k} \text{Ava}_i$	$\text{Rel} = \prod_{i=1}^{k} \text{Rel}_i$	$\text{SER} = \sum_{i=1}^{k} \text{SER}_i$
(6) Rep		(7) Sec	(8) Load	(9) Thrp
$\text{Rep} = \min\{\text{Rep}_1, \ldots, \text{Rep}_k$		$\text{Sec} = \min\{\text{Sec}_1, \ldots, \text{Sec}_k$	$\text{Load} = \max\{\text{Load}_1, \ldots, \text{Load}_k$	$\text{Thrp} = \min\{\text{Thrp}_1, \ldots, \text{Thrp}_k$

Fig. 8, we can find that when set μ=4, the algorithm has the best searching capability. So it can be determined that 4 is the most suitable value for μ.

After determining μ=4, we continue to find the most suitable value for parameter ρ. Set ρ=0.2, 0.4, 0.6, 0.8, 1.0, respectively. Execute the algorithm and record the solutions searched by C-MMAS when setting different values for ρ. Experimental results are depicted in Fig. 9, where the vertical axis represents the comprehensive evaluation values of solutions and the horizontal axis denotes values set for ρ. As Fig. 9 indicates, parameter ρ has great impact on C-MMAS. If the value of ρ is too small, it is difficult for ants to determine which path is better. Otherwise, it is likely to fall into local optimum. According to the experimental results, it can be determined that 0.6 is the most suitable value for ρ.

Then, set μ=4, ρ=0.6, and proceed to determine the most suitable value for λ. Set λ=1, 2, 3, 4, 5, 6, 7, respectively. Run the algorithm and record the solutions searched by C-MMAS when setting different values for λ. Experimental results are described in Fig. 10, where the vertical axis represents the comprehensive evaluation values of solutions and the horizontal axis denotes values set for λ. From Fig. 10, it can be found that 5 is the most suitable value for λ.

Finally, set μ=4, ρ=0.6, and λ=5, and we use the same method to observe the influence of η to the algorithm and determine how often to apply the knowledge to guide the search of ants would make the algorithm perform well. Set η=1, 5, 10, 15, 20, respectively. Implement the algorithm and record the solutions searched by C-MMAS when setting different values for η. Experimental results are shown in Fig. 11, where the vertical axis represents the comprehensive evaluation values of solutions and the horizontal axis denotes values set for η. From Fig. 11, it can be found that when set η=5, the searching capability of the algorithm is strongest. When $\eta \geq 10$, the value of η does not influence the algorithm's performance anymore. So, it can be determined that the most suitable value for η is 5.

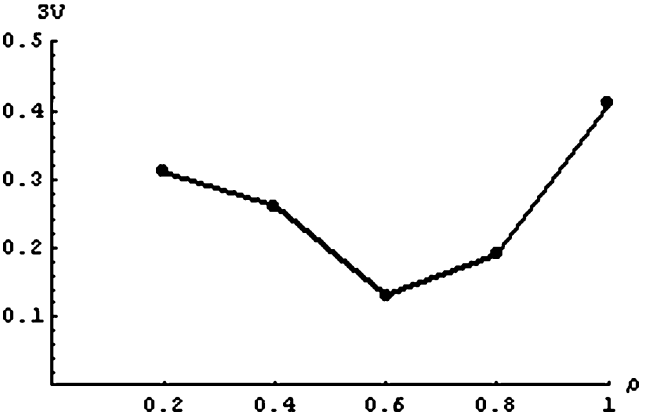


Fig. 9 Influence of ϕ to C-MMAS

5.3 Verify the validity of evaluation models

Experiments have been done to verify the effects of evaluation models of GQoS and DQoS. Firstly, the GQoS model is used to evaluate Web services, the C-MMAS algorithm in iterated for 20, 40, 60, 80, 100, and 120 times. Evaluation values of solutions searched by C-MMAS are recorded. Experiment results are described in Fig. 12, where the vertical axis represents the evaluation values of solutions searched by C-MMAS and the horizontal axis denotes the iteration times. As Fig. 12 illustrates, the deviation values of solutions are more than zero. That is to say, if all the nine GQoS constraints are treated as hard constraints, no composite Web service could be found. And yet, by using the GQoS evaluation model, we can find a composite Web service that not only can satisfy user's most concerned GQoS constraints(hard QoS constraints) but also can maximize the satisfaction of user's preferred GQoS constraints(soft GQoS constraints).

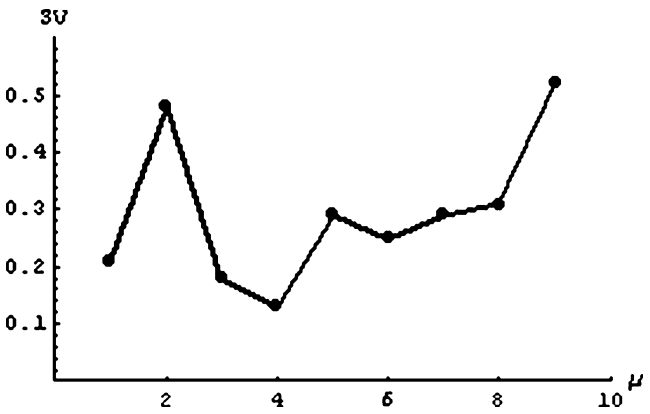


Fig. 8 Influence of μ to C-MMAS

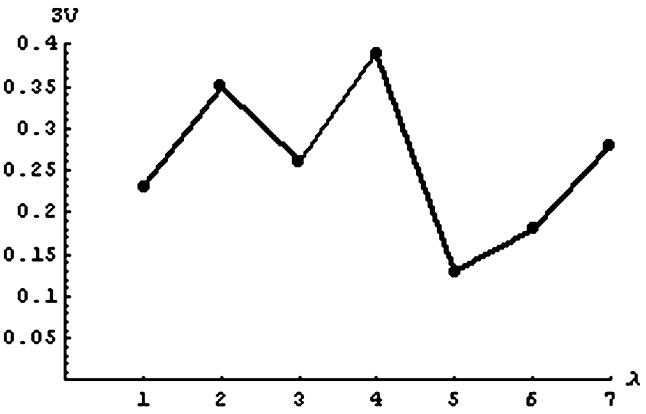


Fig. 10 Influence of λ to C-MMAS

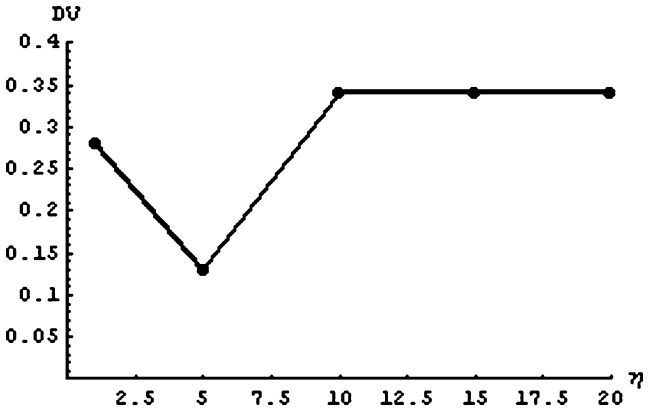


Fig. 11 Influence of η to C-MMAS

The GQoS evaluation model not only can describe user required GQoS requirements but also can describe their preferred ones. It can reduce the number of hard GQoS constraints and allows the soft GQoS constraints can be violated. So, it is helpful to conquer the over-constrained problem in the process of Web service selection.

In order to verify the effect of DGQoS evaluation model, experiments have been done under two scenarios. In the first scenario, only take the GQoS of Web services as the evaluation factors. Use the evaluation value of GQoS of composite Web services as selection criteria. C-MMAS is iterated for 20, 40, 60, 80, 100, and 120 times. Compute the evaluation values of DQoS of the solutions searched by the algorithm when it is iterated for different times. In the second scenario, take the GQoS and DQoS of Web services as evaluation factors and use the comprehensive evaluation values of composite Web services as selection criteria. Iterate C-MMAS for 20, 40, 60, 80, 100, and 120 times. Compute the evaluation values of DQoS of the solutions searched by the algorithm when it is iterated for different times.

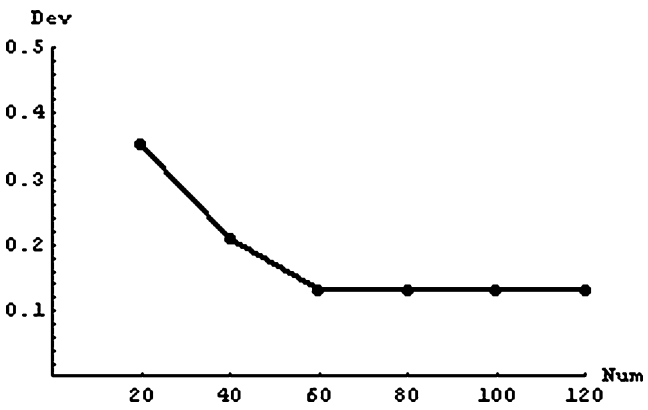


Fig. 12 Deviation values of solutions

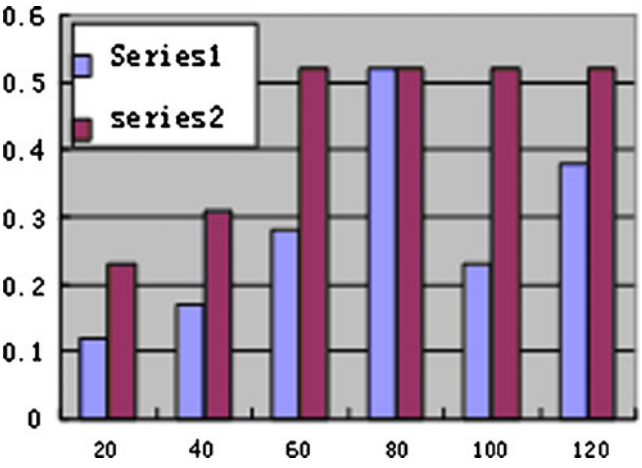


Fig. 13 Evaluation values of DQoS of solutions

Experiment results are described in Fig. 13, where the vertical axis represents the evaluation values of DQoS of solutions searched by C-MMAS. The horizontal axis denotes the iteration times. Series 1 represents the evaluation values of DQoS of solutions searched in first scenario; series 2 represents the evaluation values of DQoS of solutions searched in the second scenario. As Fig. 13 illustrates, the evaluation values of DQoS of solution searched in the second scenario are better than that of solutions searched in the first scenario. So, it can be concluded that taking the DGQoS of composite Web service as evaluation factor is helpful to find the composite Web service with better domain quality, and this can greatly improve users' satisfaction.

5.4 Comparison of algorithms

In order to verify the efficiency of C-MMAS algorithm, the ACS, MMAS, and C-MMAS are applied to solve the same problem described in Section 5.1. ACS algorithm and

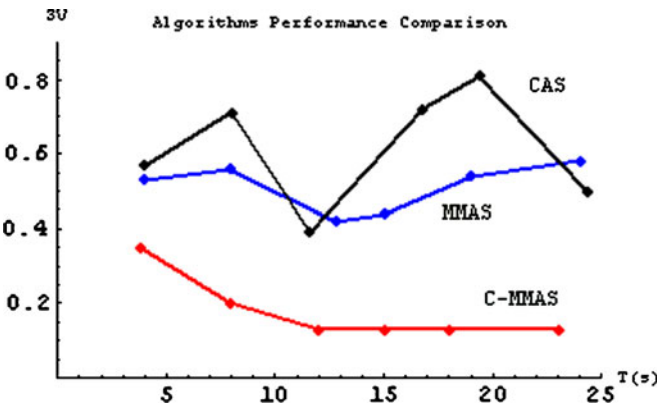


Fig. 14 Performance comparison of different algorithms

MMAS algorithm are also realized with C++ programming. Experimental environments of the three algorithms are identical. Each algorithm is iterated for 20, 40, 60, 80, 100, and 120 times, respectively. Solutions searched by each algorithm and corresponding running times are recorded. Experimental results are described in Fig. 14, where the vertical axis represents the comprehensive evaluation values of solutions and horizontal axis denotes the running times of the three algorithms when they iterated for 20, 40, 60, 80, 100, and 120 times.

As Fig. 14 illustrates, the differences between the running times of the three algorithms are small when they are iterated for the same times, but solutions searched by them are very different. As Fig. 14 illustrates, solutions searched by the C-MMAS algorithm are much better than solutions searched by the other two algorithms. Moreover, C-MMAS tends to convergence after it has been iterated for 60 times. And yet, the ACS and MMAS algorithm do not show convergence tendency all the time. Experimental results show that the C-MMAS algorithm demonstrates better performances than that of the ACS and MMAS algorithm, and it is efficient in solving problem of Web service selection based on DGQoS.

6 Conclusions

As the market competition is fierce day by day, enterprises must respond to the demands of the market and be adapted to the change of it. In this case, virtual enterprise has appeared and will become the future trends. With the development of Web service technologies, the enterprise business systems can be encapsulated as Web services. Establishing a virtual enterprise is actually a process of Web services composition. As more and more Web services with same functionality and different QoS are available, QoS-aware Web service selection becomes an active research problem.

This paper firstly designs a comprehensive evaluation model based on the evaluation model of DQoS and GQoS. The DQoS model is used to evaluate the DQoS of composite Web services, and the GQoS evaluation model based on constraint hierarchies is used to conquer the over-constrained problem of Web service composition. Experiment results have shown the validity of the comprehensive evaluation model.

For the purpose of solving the problem of Web service selection more efficiently, this paper devises a novel optimization algorithm named C-MMAS by integrating improved MMAS into the framework of culture algorithm. Experimental results prove that C-MMAS has better performances than that of the ACS and MMAS algorithms and is effective in solving Web service selection problem.

The problem of QoS-aware Web service selection is a typical combinatorial optimization problem. Experiment results have shown that the C-MMAS algorithm demonstrates great performance in solving this problem. So, C-MMAS algorithm proposed can be used to solve other important manufacturing or industrial problems, such as traveling salesman problem, quadratic assignment problem, job scheduling problem, and so on.

Acknowledgments This work is supported by the National Natural Science Foundation of China under Grant No. 60805022 and the National High-Tech Research and Development Plan of China under Grant No. 2007AA01Z178.

References

1. Lee JY, Kim K (2007) A distributed product development architecture for engineering collaborations across ubiquitous virtual enterprises. Int J Adv Manuf Technol 33:59–70
2. Deng H, Chen L, Wang CT, Deng QN (2006) A grid-based scheduling system of manufacturing resources for a virtual enterprise. Int J Adv Manuf Technol 28:137–141
3. Sari B, Amaitik S, Kilic SE (2007) A neural network model for the assessment of partners' performance in virtual enterprises. Int J Adv Manuf Technol 34:816–825
4. Chen CY, Lin JW, Lee WL, Chen CW (2010) Fuzzy control for an oceanic structure: a case study in time-delay TLP system. J Vib Control 16(1):147–160
5. Chen CW (2009) Modeling and control for nonlinear structural systems via a NN-based approach. Expert Syst Appl 36:4765–4772
6. Hsiao FH, Chen CW, Liang YW, Xu SD, Chiang WL (2005) T-S fuzzy controllers for Nonlinear interconnected systems with multiple time delays. IEEE Trans Circuits Syst 52(9):1883–1893
7. Fung RYK, Chen TH, Sun X, Tu PYL (2008) An agent-based infrastructure for virtual enterprises using Web-services standards. Int J Adv Manuf Technol 39:612–622
8. Guo H, Tao F, Zhang L, Su SY, Si N (2010) Correlation-aware web service composition and QoS computation model in virtual enterprise. Int J Adv Manuf Technol 51:817–827
9. Canfora G, Penta MD, Esposito R, Villani ML (2004) A lightweight approach for QoS-aware service composition. In Proc. 2nd International Conference on Service Oriented Computing (ICSOC'04), New York, USA, pp 36–47
10. Yu T, Zhang Y, Lin KJ (2007) Efficient algorithms for Web services selection with end-to-end QoS constraints. ACM Trans Web 1(1):1–26
11. Ardagna D, Pernici B (2005) Global and local QoS guarantee in web service selection. In: Proc. of Business Process Management Workshops, 2005, pp 32–46
12. Cardellin V, Casalicchio E, Grassi V, Francesco LP (2007) Flow-based service selection for web service composition supporting multiple QoS classes. In: ICWS 2007. IEEE Intl. Conf. Web Services, pp 743–750
13. Yu T, Lin KJ (2005) Service selection algorithms for composing complex services with multiple QoS constraints. In: Proc. of 3rd Int'l Conf. on Service Oriented Computing, pp 130–143
14. Zeng L, Benatallah B, Ngu AHH, Dumas M, Kalagnana J, Chang H (2004) QoS-aware middleware for web services composition. IEEE Trans Softw Eng 30(5):311–327

15. Menascé DA, Casalicchio E, Dubey V (2010) On optimal service selection in service oriented architectures. Perform Eval 67:659–675
16. Ardagna D, Pernici B (2007) Adaptive service composition in flexible processes. IEEE Trans Software Eng 33:369–384
17. Hhuang AFM, Lan CW, Yang SJH (2009) An optimal QoS-based Web service selection scheme. Inf Sci 179:3309–3322
18. Gerardo C, Penta MD, Esposito R, Villani ML (2005) An approach for QoS-aware service composition based on genetic algorithms. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, pp 1069–1075
19. Tang ML, Ai LF (2010) A hybrid genetic algorithm for the optimal constrained web service selection problem in web service composition. In: Proceeding of the 2010 World Congress on Computational Intelligence, pp 1–8
20. Gao ZP, Chen J, Qiu XS, Meng LM (2009) QoE/QoS driven simulated annealing-based genetic algorithm for Web services selection. J China Univ Posts Telecommun 16:102–107
21. Fang QQ, Peng XM, Liu QH, Hu YH (2009) A global QoS optimizing web service selection algorithm based on MOACO for dynamic web service composition. 2009 International Forum on Information Technology and Application, pp 37–42
22. Liu SL, Liu YX, Jing N, Tang GF, Tang Y (2005) A dynamic Web services selection strategy with QoS global optimization based on multi-objective genetic algorithm. Proc. Grid and Cooperative Computing (GCC 2005), Springer, Berlin, pp 84–89
23. Yang WJ, Li JZ, Wang JZ (2005) Domain-adaptive service evaluation model. Chniese J Comput 28(4):514–523
24. Buccafurri F, Meo PD, Fugini MG, Furnari R, Goy A, Lax G, Lops P, Modafferi S, Pernici B, Redavid D, Semeraro G, Ursino D (2008) Analysis of QoS in cooperative services for real time applications. Data Knowl Eng 67(3):463–484
25. Wang P, Chao KM, Lo CC (2010) On optimal decision for QoS-aware composite service selection. Expert Syst Appl 37:440–448
26. Liu GQ, Zhu ZL, Wang Q, Li YQ (2009) A domain-oriented evaluation model for QoS in Web service. Ninth International Conference on Hybrid Intelligent Systems, pp 319–321
27. Borning A, Freeman-Benson B, Wilson M (1992) Constraint hierarchies. Lisp Sympolic Comput 5(3):223–270
28. Stutzle T, Hoos HH (2000) MAX–MIN ant system. Future Gener Comput Syst 16(8):889–914
29. Reynolds RG (1994) An introduction to cultural algorithms. Proceedings of the Third Annual Conference on Evolutionary Programming. World Scientific, River Edge, pp 131–139
30. Deng HP, Yeh CH (2006) Simulation-based evaluation of defuzzification-based approaches to fuzzy multiattribute decision making. IEEE Trans Syst Man Cybern 36(5):968–977
31. Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66
32. Peng B (2005) Knowledge and population swarms in cultural algorithms for dynamic environments [D]. Wayne State University, USA