Global Optimal Selection of Web Composite Services Based on UMDA

Shuping Cheng, Xiaoming Lu, and Xianzhong Zhou*

School of Management and Engineering, Nanjing University, Nanjing, China
chensp70@163.com, zhouxz@nju.edu.cn

Abstract. QoS model of composite services and Web services selection based on QoS are currently the hot issues in the web service composition area. Services selection based on QoS, which is a global optimal selection issue, has been proved a NP-HARD problem. Takes engine into account, this paper builds the QoS model of service selection in the Web composite services, uses the estimation of distribution algorithm to solve the NP-HARD problem of services selection, and presents a Web services selection method based on the UMDA. Example analysis and experimental analysis based on the UMDA method are performed; it's proved that the method is effective in solving the NP-HARD problem of Web services selection.

Keywords: Qos, Web Composite Services, Service Selection, UMDA.

1 Preface

Researches on Web composite services are attracting more and more attention. The QoS for the composite services and the Web services selection method based on QoS are the hot issues presently.

The present QoS for the composite services only takes the QoS of the member services and the structures of the composite services into account, while the operation of the composite services depends on the engine, which will affect the efficiency of the QoS. The paper, taking the engine into account, rebuilds the QoS for the composite services.

QoS for the Web services depends on QoS of individual service in the combination and the logical structures of the services composited. The current studies show that there are four basic composite structures among the composite services: sequential structure, loop structure, parallel structure and case structure. Based on these four structures, with the known QoS data of the member services, the QoS aggregation method [1] [2] has been put forward. This method supports several common QoS attributes mentioned in the literature [3], such as service price, response time, reliability, and reputation degree. It is common for the academic circle to transform the optimal selection method of the global QoS into a combinatorial optimization problem to find the solutions. There are several optimization methods, such as linear

* Corresponding author.

programming method[4][5], constrained optimization method[6], heuristic method[7][8] and genetic algorithm[9][10][11].

Although, to some extent, the current genetic algorithms have solved the service selection problems, they haven't solved some special problems of the composite services. It is necessary to redesign the genetic algorithms of the Web services from the perspective of QoS. The design of the QoS of the composite services will affect the efficiency of the genetic algorithms deeply [12]. This paper builds the UMDA algorithm which supports the global optimal selection of Web composite services. Compared to the existing achievements, the method, based on the genetic algorithms, gives a more complete and effective QoS scheme of the composite services, which has a good adaptability.

2 QoS for the Composite Services

2.1 Service Price

Due to the introduction of execution engine, the price of the composite services should be defined as the aggregate value of each member service's price adds the price of the engine:

$$Pri(cws) = pri(s) + pri(e)\,. \tag{1}$$

$Pri(cws)$ means the price of the composite services(cws)， $pri(s)$ means the aggregation price of member services， $pri(e)$ means price of the engine.

According to the mainstream toll mode of the Web services, the toll mode of the engine has two different modes: charging by time and charging by month. The charging by time mode happens when users use the engine, and the charging by month mode means that users can use the engine a certain number of times in a certain period after charged.

2.2 Response Time

Besides the aggregate value of the time of the member services, the response time should take the communication time and the treatment time of the engine into account:

$$T(cws) = t(s) + tran(u,e) + pro(e)\,. \tag{2}$$

$T(cws)$ means the response time of the composite services, $t(s)$ means the aggregate value of the time of the member services, $tran(u,e)$ means the communication time between the users and the engine, $pro(e)$ means the treatment time of the engine. The estimation process of response time is as follow:

The first step is estimating the $t(s)$. Engine server suppliers should provide the current network parameters, the users should provide the access amount of every member service, and the member service suppliers provide the environment data when the member services are running.

The second step is estimating the $tran(u,e)$. The $tran(u,e)$ depends on the network environment between the users and the engine, and this data can be calculated by the network performance and the transmission amount caused by the composite services used by the users.

The third step is estimating the $pro(e)$. The $pro(e)$ depends on the complexity of the control logic of the composite services and the performance of the engine.

2.3 Reliability

The reliability of the composite services is the probability of the right response of the services in a longest expected time. The present estimation methods of the Web services just take the aggregation of the reliabilities of the member services into account, while the engine controls the logic and the data flow of the composite services. If the engine fails, the whole operations of the composite services will fail, so the reliability of the engine should be more important than that of the member services. Here is the formula of the reliability of the composite services:

$$Rel(cws) = rel(s) \times rel(e). \tag{3}$$

$Rel(cws)$ means the reliability of the composite services, $rel(s)$ means the aggregate value of the reliabilities of the member services, and $rel(e)$ means the reliability of the engine. The reliability of the engine should be distributed by the engine suppliers after enough tests.

2.4 Reputation Degree

The reputation degree is the satisfaction of the composite services from users. In the problem of the member services selection, the whole reputation degree of the services after composited, as a constraint condition, should be taken into account in estimation of the reputation degree. The paper considers the reputation degree of the engine, besides the reputation degree of the member services. Here is the formula of the reputation degree:

$$Rep(cws) = (1-\alpha) \times rep(s) + \alpha \times rep(e) \tag{4}$$

$Rep(cws)$ means the reputation degree of services, $rep(s)$ means the composite of the reputation degree of member services, and $rep(e)$ means the reputation degree of the engine. α is a weight coefficient and $\alpha \in [0,1]$, and if α is bigger, the reputation degree of the engine is more important. Because the engine is very important and irreplaceable in the solution of the selection of the Web services, the range of α should be $\alpha \in (0.5,1]$.

3 Selection Method of Web Services Based on UMDA

3.1 Selection Method of Web Services Based on UMDA

The users consider the composite services as a whole and focus on satisfaction of the Qos of the whole under the global constraint condition, which is a global optimal selection problem.

Some assumptions are as follow: There are N tasks in the composite package, $task_i (i = 1 \sim N)$ means the NO. i task, Set_i means the candidate service set for $task_i$, M_i means the service amount in Set_i, and $S_{ij} (j = 1 \sim M_i)$ means it is NO. j service in the Set_i. The global optimal selection problem based on the QoS is a multi-objective optimization problem, and a simple solution for this problem is to change the problem to a single-objective optimization problem. The mathematical description for the problem is as follow:

$$MaxF(cws) = \sum_{k=1}^{t} \omega_k \times Qf_k(cws) \tag{5}$$

$$s.t.\ Qf_k(cws) \le B_k \text{ Or } Qf_k(cws) \ge B_k$$
$$cws = (S_{1j}, S_{2j}, \cdots, S_{Nj})$$
$$S_{ij} \in Set_i, i = 1 \sim N, j = 1 \sim M_i$$
$$\sum_{k=1}^{t} \omega_k = 1$$

cws means the composite of the services, $Qf_k(cws)$ means the QoS value of the No.k service in the composite services package, ω_k means weight coefficient, B_k means constraint condition of certain QoS.

In the Web service area, there are lots of researches on the traditional genetic algorithms and their improved versions, but there is little on the estimation of distribution algorithm. The paper discusses the solution of global optimal selection problem based on the QoS with the estimation of distribution algorithm.

3.2 Selection of the Web Services Based on UMDA

(1) Narrowing Solution Space

The users hope the global QoS of the composite services is the best and the QoS of every service corresponding to the task is the best too. If there are too many candidate services, the local optimal selection algorithm can be used to calculate the QoS of every candidate service, and all the services with a QoS value less than a certain threshold will be forsaken, to reduce the amount of the solutions. The threshold is given by the users. It is assumed that after the process above, the amount of the services in the candidate service set is still $\mathrm{M_i}$.

（2） Coding

Binary code system is used in UMDA, and every variable is represented by a binary code. If the binary code system is used in the problem of Web service selection, every candidate service will be represented by a binary code, which will cause a long coding, and this is not suitable to the algorithm processing. In this case, the better solution is using the integer array for coding. Please see the structure in figure 1, and the details are as follow:

The integer array is used for representing the entities of the population, and the length of the array, which is the amount of the gens, is equal to the task amount N in the composite package. $G_i(i=1\sim N)$ means the NO. i gen, corresponding to the related $task_i$ and Set_i . G_i is the number j candidate service in Set_i , $j=\{1,2,\cdots,M_i\}$; $G_i=j$ means the $task_i$ will be implemented by the NO. j service in the candidate service set.

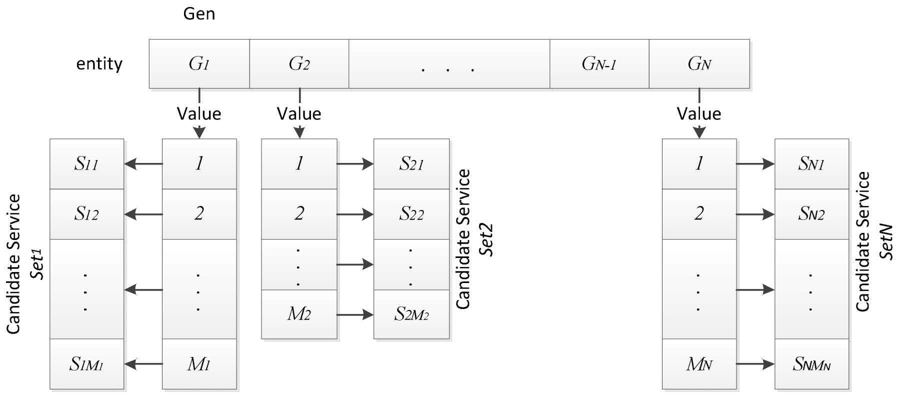


Fig. 1. Entity Structure

（3） Population Initialization

In the first step, the population size K should be determined. According to the research between the learning parameter and problem of the UMDA by Shapiro[14], only when the population size is huge enough, the UMDA algorithm can converge to the global optimal solution. The tests of the practical problems show that the population size should be the square root of the amount of the candidate solution. If there are total $\prod_{i=1}^{N} M_i$ candidate solutions of service selection, in order to make the algorithm converge to the global optimal solution, K should be approximately equal the square root of it.

In the second step, the initial probability of every possible value of every gen G_i should be determined. If users don't have any preference about certain service in the candidate Set_i, the chance of every service is the same. In that case, if the service in Set_i is chosen, the chance of every possible value of G_i is the same, and it will be showed as follow:

$$P(G_i = j) = \frac{1}{M_i}, j = 1 \sim M_i \tag{6}$$

If the users have certain preference, and could give the probability of every candidate, the initial probability of every possible value comes from users. But in the practice, although the users have the preference about some services, the users can't give the exact probability of every candidate service, especially when the amount of the candidate services is huge. The local optimal selection algorithm is used to calculate the QoS of every candidate service and the sum of QoS of every candidate service, and the probability of every candidate service will be calculated by certain formula.

$$P(G_i = j) = \frac{QS_{ij}}{\sum_{j=1}^{M_i} QS_{ij}}, j = 1 \sim M_i \tag{7}$$

In the third step, in every candidate service Set_i, according to the probability of every candidate service, roulette algorithm is used K times for sampling to get the initial population.

（4） Fitness Function

It is applicable that the fitness function is the same with the object function, which means when the constraint conditions are satisfied, and the composite QoS of composite services influenced by the engine is taken into account, the formula is:

$$f(cws) = \sum_{k=1}^{t} \omega_k \times Qf_k(cws) \tag{8}$$

Because the amount of the composite schemes is too huge to calculate all the QoS of the possible composite services, ω_k is given directly by the users, not by the objective data, which is different from the method of the local optimal selection algorithm.

（5） Selection Mechanism.

The selection mechanism is the foundation of the probability updating. Numbers of the distribution relations inherited from the old generation population by the new generation population are called selection pressure, which depends on selection mechanism. In the case that the proper selection mechanism is chosen, the selection pressure will keep population diversity and increase the possibility of the global optimal converging. It can also guarantee the high amount of the optimal entities during the evolution, and the algorithm converges fast. Similar with the genetic algorithm, the selection mechanisms of the EDA include sorting algorithm, roulette algorithm, tournament selection algorithm, and random traversal sampling algorithm.

The paper uses the sorting algorithm. First, the QoS of the composite services of every entity of the population is calculated, and those unsatisfied with the constraint condition will be discarded. Then, after calculating the fitness values of the left entities by the fitness function, the fitness values will be sorted from maximum to minimum, and the excellent ones with high fitness value will be selected according to certain rate. In order to keep the population diversity, the rate should be around 50%.

（6） Probability Updating and Resample

According to values of the gens of the selected optimal entities, the frequencies of the values of every gen are calculated. The probabilities of values of every gen are updated by the value calculated as the frequencies divided by total amount of the optimal entities. According to the new probabilities, every gen will be sampled K times by the roulette algorithm, and there will be K new entities then, which will form a new generation population.

（7） Stopping Rule

There are three common types of terminal conditions: the generation of evolution achieves the set value; the fitness value of the entity achieves the set value; and there is less likely a big change in the next generation. Normally, the generation of evolution of the population is large enough; the evolution should converge to certain optimal entity; or the evolution stops at the set value accepted by the users.

As the other evolutionary algorithms, the solution of the EDA depends on the related parameters and methods, and the solution may be not the optimal one. For the problem of the selection of the Web services, if there are too many composite schemes, the suboptimal solution calculated by the EDA is acceptable.

4 Simulation and Comparison

Some tests with the simulation data will be used in this part to verify that EDA is efficiency to solve the selection of the Web services based on QoS under the big solution space. Both the efficiency of the EDA and the efficiency of the genetic algorithm will be analyzed in this part.

There are 4 types of the QoS attributes, the values of the attributes are randomly generated, and the weight is $\omega = (0.4, 0.2, 0.2, 0.1, 0.1)$. The related data of the engine here is the same as that in the real example. There are N tasks in the composite services, the amount M of the candidate services of every task is the same, the generation size is

K, and the generation of the evolution is P. Different Ns and Ms are used in different experiments, and the algorithm will run 50 times in every experiment. If the convergence fails even the operation of the algorithm is over, the maximal fitness value of the entity in the last generation is used as the statistical data.

4.1 Simulation Result

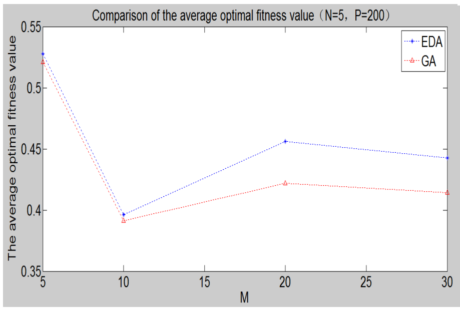
The related data is as follow:

(1) N is 5, K is 100, the value of M is changing, and the stopping rule is that P is 200 or convergence finishes in advance.

（2）N is 5, K is 100, the value of M is changing, and the stopping rule is that the maximal fitness value achieve the set value or convergence finishes in advance.

（3）M is 10, K is 100, the value of N is changing, and the stopping rule is that P is 200 or convergence finishes in advance.

（4）M is 10, K is 100, the value of N is changing, and the stopping rule is that the maximal fitness value achieve the set value or convergence finishes in advance.



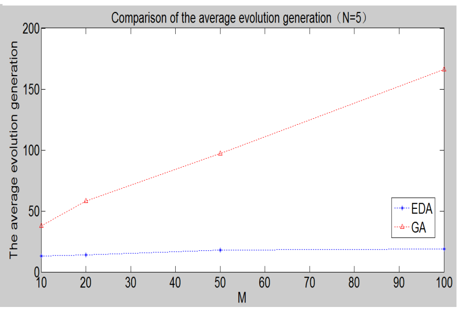
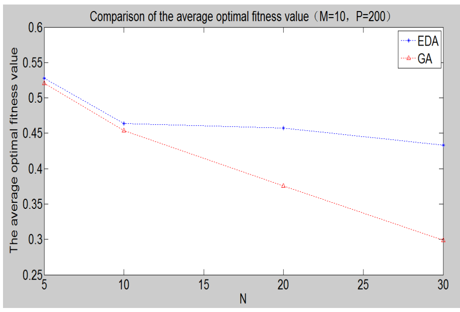


Fig. 2. Comparison of the average optimal fitness value

Fig. 3. Comparison of the average evolution generation



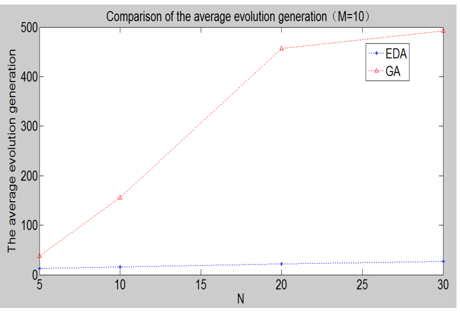


Fig. 4. Comparison of the average optimal fitness value

Fig. 5. Comparison of the average evolution generation

4.2 The Efficiency of EDA

From the comparison:

（1）When the stopping rule is the evolution generation, if the N is fixed, the fluctuation of the convergence evolution generation is small when M increases; if the M is fixed, the convergence evolution generation increases when N increases; and the EDA converges very fast to a good fitness value.

（2）If the stopping rule is a maximal fitness value, N is fixed, and the fluctuation of the evolution generation of EDA when the maximal fitness value achieves is small when M increases; M is fixed, and the evolution generation of EDA when the maximal fitness value achieves increases when N increases; and the maximal fitness value of EDA achieves fast under a small evolution generation, and it sometimes happens that the algorithm converges fast to the value less than the maximal fitness.

It shows that the satisfied suboptimal solution can be steadily achieved in a small evolution generation, and the better composite scheme can be found for the selection of the Web services with EDA.

4.3 Comparison

It shows that EDA is more efficiency than the traditional genetic algorithm in literature[11] in the selection of the Web services:

The convergence speed of EDA is fast than that of GA. If the generation size and the stopping rules (the evolution generation) are the same, the optimal fitness value of EDA is better than that of GA. If the generation size and the stopping rules (the optimal fitness value) are the same, the evolution generation of EDA is much less than that of GA.

Additionally, this experiment just proves that the EDA in this paper is better than the traditional genetic algorithm in literature[11] in the selection of the Web services, which doesn't mean the EDA is better than the traditional genetic algorithm. The operations of the two algorithms depend on the coding, evolution and the parameters.

5 Conclusion

The paper discusses the selection of the web services based on QoS, introduces the EDA which is a global optimal selection scheme to solve the NP-HARD problem. The paper introduces the solution of the selection of the Web services based on the UMDA, designs the algorithm, performs example analysis and experimental analysis, and verifies that the UMDA is efficient to solve the problem.

Acknowledgement. This research was supported by: (i) the National Planning Office of Philosophy and Social Science under Grant 11&ZD169; (ii) the NSFC under Grant 70971061 and 71171107.

References

1. Aalst, W.M.P.V.D.: Don't go with the flow: Web services composition standards exposed. IEEE Intelligent Systems 18(1), 72–76 (2003)
2. Yin, K.: Research on QoS-aware Services Composition in Internet Environment. Zhejiang University, Hangzhou (2010)
3. Michael, C.J., GeroMuhl, G.G.: QoS Aggregation for Web Service Composition using Workflow Patterns. In: EDOC, pp. 149–159 (2004)
4. Zeng, L.Z., Benatallah, B., et al.: QoS-Aware Middleware for Web Services Composition. Transactions on Software Engineering 30(5), 311–327 (2004)
5. Ardagna, D., Pernici, B.: Adaptive Service Composition in Flexible processes. IEEE Transactions on Software Engineering 33(6), 369–384 (2007)
6. Rosenberg, F., Celikovie, P., Michlmayr, A., Leitner, P., Dustdar, S.: An End-to-End Approach for QoS-Aware Service Composition. In: 2009 IEEE International Enterprise Distributed Object Computing Conference, pp. 151–160 (2009)
7. Tao, Y., Lin, K.: Service selection algorithms for Web services with end to end QoS constraints. Information Systems and e-Business Management 3(2), 103–126 (2005)
8. Berbner, R., Spahn, M., Repp, N., Heckmann, O.: Ralf Steinmetz. Heuristies for QoS-aware Web Service Composition. In: ICWS, pp. 72–82 (2006)
9. Liu, K., Wang, H., Xu, Z.: A Web Service Selection Mechanism Based on QoS Prediction. Computer Technology and Development 17(8), 103–109 (2007)
10. Xia, Y.: Research on Some Key Issues of Dynamic Service Composition. Beijing University of Posts & Telecommunications, Beijing (2009)
11. Wu, C.: Research on Dynamic Web Service Composition and Performance Analysis with QoS Assurences. Wuhan University, Wuhan (2007)
12. Ignacia, R., Jesu, S.G., Hector, P., et al.: Statistical analysis of the mainparameters involved in the design of a genetic algorithm. IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews 32(1), 31–37 (2002)
13. Shapiro, J.L.: Drift and scaling in estimation of distribution algorithms. Evolutionary Computation 13(1), 99–123 (2005)
14. Al-Masri, E., Mahmoud, Q.H.: Discovering the best web service. In: Proc. of Int'l Conf. on World Wide Web (WWW), pp. 1257–1258 (2007)
15. Al-Masri, E., Mahmoud, Q.H.: QoS-based Discovery and Ranking of Web Services. In: Proc. of Int'l Conf. on Computer Communications and Networks (ICCCN), pp. 529–534 (2007)
16. Al-Masri, E., Mahmoud, Q.H.: Investigating Web Services on the World Wide Web. In: Proc. of Int'l Conf. on World Wide Web (WWW), Beijing, pp. 795–804 (2008)
17. Michael, C., Jaeger, G., Rojec-Goldmann, G.: QoS Aggregation in Web Service Composition using Workflow Patterns. EEE, 181–185 (2005)