Optimal Web Service Selection based on Multi-Objective Genetic Algorithm

Junli Wang[1], Yubing Hou[2],
1. The Engineering Research Center of Enterprise Digital Technology, Ministry of Education Tongji University, Shanghai 200092, China
2. Shanghai Branch, China Information Technology Designing and Consulting Institute, Shanghai 200050, China
wangjunli_1029@163.com

Abstract

Considering that there are three aspects of constrains in the service selection process, such as control structure within a composition plan, relationship between concrete services, and tradeoff among multiple QoS indexes, a QoS based optimal Web services selection method by multi-objective genetic algorithm is presented. First we design a chromosome coding method to represent a feasible service selection solution, and then develop genetic operators and strategies for maintaining diversity of population and avoiding getting trapped in local optima. Experimental results show that within a finite number of evolving generations this algorithm can generate a set of nondominated Pareto optimal solutions which satisfy to user's QoS requirements.

1. Introduction

Web service composition problem has received much attention recently, and various solutions have been proposed [1]. Several ongoing research efforts, based on service function property matching, generate some available composition solutions [2]. Further, how to select an optimal solution based on services' non-function property to meet the customers' requirements that is the service selection problem is also an important problem [3].

Generally, three kinds of constrain factors should be considered in the process of service selections: control structure in a composition solution such as sequence, parallel, and choice; relationship between concrete services such as alliance or competition; and tradeoff between several Qos indexes such as cost, respond time, and reliability.

Recently, several service selection optimization methods have been proposed. Ref.[4] formalized the service selection as a integer planning model, then presented an enumerate algorithm to dynamically select optimal composition solutions according to several QoS indexes. This algorithm considered constrain factor of control structure, but did not consider constrain factor of relationship between concrete services. Ref.[5] proposed a service selection optimization method based on genetic algorithm. But in this paper the multiple QoS indexes optimization problem is turned into a single-objective problem using a weighted method, which will inevitably miss some feasible solutions.

To solve these problems, in this paper we design a multi-objective genetic algorithm (MOGA) for the optimal service selection optimization problem. And this paper is organized as follows. Firstly, the Web service selection problem is defined formally in Section 2, the proposed service selection optimization algorithm is given in Section 3, then a set of experiments are conducted, and their results and analysis are shown in Section 4, and finally Section 5 concludes this paper.

2. Description of Optimal Web Service Selection Problem

First, an inductive definition of Web service composition solution (WSCS) is given.

Definition 1: Let $AWS = \{S_1, S_2, \ldots, S_n\}$ be a set of abstract services, where every abstract service S_i belongs to a certain service category. Then,

1) $S_i \in ASW$ is a WSCS, and is called as a basic WCSC;

2) Sequence(S_1, S_2, …), Parallel(S_1, S_2, …)and Choice(S_1, S_2, …)are also WSCSs, and are called as simple WSCSs;

3) if S_1' and S_2' are WSCSs, then Sequence(S_1', S_2', …), Parallel(S_1', S_2', …) are also WSCSs, and are called as complex WSCSs;

978-0-7695-3311-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ISCID.2008.197

4) for each WSCS, it is required that for every abstract service S_i, $|S_i| \leq 1$ holds, where $|S_i|$ is the number of times that S_i occurs in the WSCS.

Because there are three kinds of control structures among abstract services: Sequence, Parallel, and Choice, every WSCS must meet two constrain conditions: in a Choice structure module, one and only one of all abstract services should bind one concrete service; in other structure modules, every abstract service binds one concrete service. It is formally described as follows:

Definition 2: Let $AWS = \{S_1, S_2, \ldots, S_n\}$ be a set of abstract services, *ComPlan* is a WSCS on *AWS*, $CWS = \{s_1, s_2, \ldots, s_m\}$ is a set of concrete services related with *AWS*. Function f associates an abstract service s with a concrete service S, and represents that service s is allocated to service S. Let $Solution = (x_1, x_2, \ldots, x_m)$, where $x_i = 0$ or $x_i = 1$. If $x_i = 0$, then concrete service s_i is not selected, while if $x_i = 1$, then s_i is selected. And m is the number of concrete services.

If the following two conditions hold, then *Solution* is called as a Web service selection solution (WSSS) corresponding to *ComPlan*.

1) for any Choice structure in *ComPlan*, Choice(S_{i1}, S_{i2}, …, S_{ik}), and any concrete service s_u satisfying $f(s_u) = S_{il}$, where $1 \leq l \leq k$, it is required that $\sum_u x_u = 1$ holds;

2) for any other abstract service S_i, and any concrete service s_v satisfying $f(s_v) = S_i$, it is required that $\sum_v x_v = 1$ holds.

Definition 3: Qos index of a concrete service is defined as a triple $QoS(s) = (C(s), T(s), R(s))$, where $C(s)$ denotes cost of service s, $T(s)$ denotes respond time of service s, and $R(s)$ denotes reliability of service s.

Similar to definition 3, Qos index of a WSSS *Solution* can be defined as $QoS(S) = (C(S), T(S), R(S))$. Since every concrete service s_i has a QoS index, QoS(S) will be computed according to the equations in Table 1[4], where s_i is a concrete service in a certain control structure module, w is the number of services in the module, and p_i represents whether s_i is selected whose value is 0 or 1.

Moreover, to represent constrain relationships between concrete services, the following relational matrix is defined.

Definition 4: Let $CWS = \{s_1, s_2, \ldots, s_m\}$ be a set of concrete services, $R = \{r_{ij}\}$ is defined as a relational matrix on *CWS*, where r_{ij} denotes a constrain relationship between s_i and s_j, and its value is assigned as follows.

$$r_{ij} = \begin{cases} 1 & \text{exist alliance relationship between } s_i \text{ and } s_j \\ -1 & \text{exist competitive relationship between } s_i \text{ and } s_j \\ 0 & \text{other} \end{cases}$$

Definition 5: Let $Solution = (x_1, x_2, \ldots, x_m)$ be a WSSS on *ComPlan*, R be a relational matrix on *CWS*, $QoS(S) = (C(S), T(S), R(S))$ be QoS index of *Solution*, the optimal Web service selection problem is formalized as:

$$\begin{cases} \text{Minimize } C(S) \\ \text{Minimize } T(S) \\ \text{Maximize } R(S) \end{cases} \quad (1)$$

$$\text{s t. } \begin{array}{ll} x_i = x_j & \text{if } r_{ij} = 1; \\ x_i \neq x_j \text{ or } x_i = x_j = 0 & \text{if } r_{ij} = -1 \end{array}$$

3. Optimal Web Service Selection based on Multi-Object Genetic Algorithm

As a powerful search strategy based on natural selection and population genetics, genetic algorithm[6] outperforms conventional optimization methods such as the gradient ascent and simulated annealing.

The implementation of MOGA in application of the Web service selection problem incorporates three basic steps so that the algorithm is formulated for the specific application: the presentation of individual, the formulation of the fitness function, and genetic operators.

Table 1 Qos index computing method of WSSS *Solution*

QoS	Sequence structure	Parallel structure	Choice structure
Cost	$C(S) = \sum_{i=1}^{w} C(s_i)$	$C(S) = \sum_{i=1}^{w} C(s_i)$	$C(S) = \sum_{i=1}^{w} p_i C(s_i)$
Respond time	$T(S) = \sum_{i=1}^{w} T(s_i)$	$T(S) = \max_{i=1,\cdots w} (T(s_i))$	$T(S) = \sum_{i=1}^{w} p_i T(s_i)$
Reliability	$R(S) = \prod_{i=1}^{w} R(s_i)$	$R(S) = \prod_{i=1}^{w} R(s_i)$	$R(S) = \prod_{i=1}^{w} p_i R(s_i)$

3.1 Representation of Solution Strings

According to the Definition 2, a WSSS is defined as $Solution=(x_1, x_2, \ldots, x_m)$, so a corresponding string (called chromosome in GA), $C=c_1c_2\ldots c_m$, is used to represent every possible solution that meets user's request. And for any i, $1\le i\le m$, $c_i= x_i$. In this solution string, every bit corresponds to a concrete service whose value is 0 or 1. If the concrete service is selected, then the value is 1, otherwise, the value is 0. So a solution string is represented as a binary string of length m.

However, because there are several constrain conditions in the process of service selection, not every chromosome is a feasible selection solution. So the feasibility of chromosome should be judged by the following method: if a chromosome $C=c_1c_2\ldots c_m$ satisfies the constrain conditions both in Definition 2 and in Definition 5, then it is called a feasible chromosome, that is an individual.

3.2 Fitness function

In order to determine the quality or performance of each encoded solution string in the population, the GA associates a fitness measure with each solution string. For the problem of Web service selection, performance of every service selection solution is measured by its Qos index. Here we adopt the three Qos optimization functions defined in Definition 5 as three fitness functions in GA.

3.3 Genetic operators

Generally, three kinds of genetic operators are involved in GA, that is selection operator, crossover operator, and mutation operator.

In this paper, ranking selection method is adopted as the selection operator[7]. First, it explicitly uses Pareto dominance[8] in order to determine the reproduction probability of each individual. Basically, it consists of assigning rank 1 to the nondominated individuals and removing them from contention, then finding a new set of nondominated individuals, ranked 2, and so forth. Then, fitness of the individual is assigned by its rank. Hence, in the ranking based selection process, the more forward an individual ranks, the more opportunity there is for the individual entering into the next generation.

For the crossover operator, traditional one is implemented by choosing a point at random, and exchanging the segments to the right of this point. But, because there is a constrain that every abstract service should bind one and only one concrete service, here a set of concrete services corresponding to an abstract service should be taken as a unseparated substring, so crossover operator should occur only between substrings, as shown in Fig.1. Similarly, mutation operator should also be adjusted, and it is implemented by choosing a substring randomly, then operating it with the traditional mutation operator, shown in Fig.2.

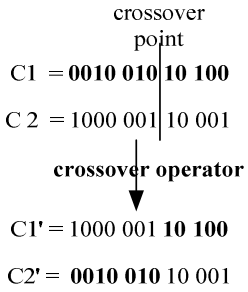

Fig. 1 crossover operator

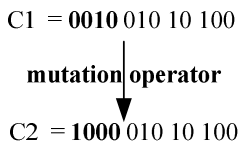

Fig. 2 mutation operator

3.4 MOGA for optimal service selection problem

When specific representation scheme and fitness functions to evaluate the solutions have been decided, the final MOGA for the optimal service selection problem will be developed. And the proposed genetic algorithm is given in Table 1, where elitist strategy[7] is employed to ensure that the most highly fit individuals in the population are passed on to the next generation without being altered by genetic operators.

Table 1. MOGA for optimal service selection problem

Begin
Initialize genetic parameters (the maximal number of iterations *maxgen*, probability of crossover p_c, probability of mutation p_m, probability of selection p_s)
Generate randomly an initial population of *pop_size* individuals /*As described in Sect.3.1, every individual is a feasible chromosome. */
while (certain termination criterion is not met) do /* such as the number of generations is less than a given value of *maxGen**/
{
Evaluate every individual and compute its Qos function values according to Equation (1);
Rank individuals and assign their fitness values by the ranking selection method in Sec.3.3;
Random select *pop_size/2* individuals according to selection probability p_s to form a temporary population;
For the individuals in the temporary population, adopt crossover and mutation operators to form new *pop_size/2* individuals;
Generate the next new population by collecting the new *pop_size/2* individuals and the *pop_size/2* individuals in the temporary population;

```
        The best individual in current population will be
          reserved into the next population;/*Elitist strategy is
          employed*/
}
End
```

4. Experiments

In this section, the performance and effectiveness of the proposed algorithm is tested by some experiments, and in these experiments, all related parameters in GA are assigned as follows: size of population is 100; the maximal number of iterations is 500; crossover probability, mutation and selection probability are respectively 0.85, 0.01 and 0.65. Due to no guidelines or standard services testing datasets for web service problem, we create three testing datasets, and the number of services is respective 35, 70 and 105. Seven abstract services are involved. The range of three Qos indexes are given as service cost (0-100), respond time (0-30) and reliability (0-100). The service relational matrix records relationship between any two concrete services and its elements are randomly assigned the value of 0, 1, or -1.

When facing the multi-objective optimization problem, enumeration method will enumerate every possible solution, then for each objective compute its value, and find out the best objective value. But in fact, the best solution for one single objective is possibly not the best one for other objectives. In this experiment, the best objective value for each objective is computed and shown in Table 2.

The experimental results we got are a set of nondominated solutions, which are considered to be equally good. Also the performance of these solutions is evaluated, and for every Qos index Table 2 gives percentages of the number of these solutions whose value equals to the best objective value to the number of nondominated solutions. Here with 50 tests, the average percentages are recorded.

It is shown in Table 2 that the proposed algorithm will find out the optimal value for every index with more than 91 percent within 500 times iterations, so it is feasible for solving the multiple Qos indexes optimization problem in Web service selection process. And also because genetic algorithm is a widely concerned global search algorithm, which has good parallel search ability, this algorithm has strongly ability in global stochastically searching and can be extended to cope with the problem involving a great number of concrete services.

Table 2. Percentage of optimal solutions

The number of concrete services	Relevance %	Percentage of optimal solutions% (the optimal value)		
		Cost	Respond time	Reliability
35	30	100 (426.5)	96 (136.42)	100 (84.26)
70	50	97 (353.0)	94 (97.68)	98 (89.51)
105	80	93 (326.0)	91 (85.93)	92 (93.38)

5. Conclusion

In this paper, we have presented a multi-objective genetic algorithm (MOGA) for solving optimal service selection optimization problem to generate a set of Pareto optimal solutions that meet users' requirements. A set of experiments are conducted to analyze and evaluate the performance of the proposed method, and the experimental results shown that our proposed algorithm can approximately converge to the optimal of each index, and generate a set of Pareto optimal solutions within a certain generation.

6. References

[1]Brahim Medjahed, Athman Bouguettaya, Ahmed K.Elmagarmid. Composing Web Services on Semantic Web. The VLDB Journal, Vol.12, No.4, 2003, pp. 333-351

[2]Brahim Medjahed and Athman Bouguettaya. A Multilevel Composability Model for Semantic Web Services. IEEE Transactions on Knowledge and Data Engineering, Vol.17, No.7, 2005, pp. 954-968

[3]E.Michael Maximilien and Munindar P.Singh. A Framework and Ontology for Dynamic Web Services Selection. IEEE Internet Computing, Vol.8, No.5, 2004, pp.84-93

[4]Liangzhao Zeng, Boualem Benatallah, and Anne H.H. QoS-Aware Middleware for Web Services Composition. IEEE Transactions on Software Engineering, Vol. 30, No. 5, 2004, pp.311-327

[5]Liangjie Zhang and Bing Li. Requirements Driven Dynamic Services Composition for Web Services and Grid Solutions. Journal of Grid Computing, Vol.2, No.2, 2004, pp.121-140

[6] DeJong, K. A., "An Analysis of the Behavior of a Class of Genetic Adaptive Systems", Ph.D. thesis. University of Michigan Press, Ann Arbor, 1975.

[7]Goldberg D E. Genetic Algorithms in Search, Optimization and Machine Learning. Massachusetts: Addison -Wesley, 1989

[8]Vanveldhuizen D A, Lamont, G B. Multiobjective evolutionary algorithms: analyzing the state of the art. IEEE Transaction on evolutionary computation, Vol.8, No.2, 2000, pp.125-147