A Genetic Algorithm with Simplex Optimization Method for QoS-driven Cloud Service Selection

Chengwen Zhang\textsuperscript{1}, Yan Gong\textsuperscript{2}, Jiali Bian\textsuperscript{3}, Bo Cheng\textsuperscript{4}, and Lei Zhang\textsuperscript{5}

\textsuperscript{1}Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts & Telecommunications, Beijing, China
\textsuperscript{2}Chinese Electronic Equipment System Engineering Corporation, Beijing, China
\textsuperscript{3}Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts & Telecommunications, Beijing, China
\textsuperscript{4}State Key Lab of Networking and Switching Technology, Beijing University of Posts & Telecommunications, Beijing, China
\textsuperscript{5}Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education, Beijing, China

Abstract - A special Genetic Algorithm (GA) with simplex optimization method is proposed for the problem of selecting an optimal cloud service composition plan from a lot of composite plans on the basis of some global Quality-of-Service (QoS) constraints. In this GA, some Simplex Method (SM) operations and a fitness function are provided. The design of the SM operations is made in the light of the characteristics of cloud service composition. After analyzing the types of different QoS attributes, objective function and fitness function are built. SM operations enhance local search capability of GA. Because the design of the hybrid algorithm accords with the characteristics of cloud service selection very well, excellent composite cloud service plan can be gotten from a lot of composite plans on the basis of global QoS constraints. Some tests and analyses show that the proposed algorithm can be a good method for QoS-based cloud service selection.

Keywords: Cloud service selection, Genetic algorithm, Simplex method, QoS-aware

1 Introduction

In cloud computing technology [1-3], service is one of the most important concepts. Services in cloud computing environment are cloud services. With the rapid deployment of cloud service, the number of cloud services with same functionalities and different QoS attributes are increasing. These services can combine tens of thousands composite services with same functions and different QoS. Therefore, we need to choose service components from massive cloud services with same functions and different QoS according to user's QoS requirements.

In the field of QoS-based cloud service selection, a lot of international research works were done and some research results were obtained [4-8, 11-21]. There are two kinds of QoS properties calculation methods. One is exhaustive methods and another is approximate algorithms. It is under the scope of combinatorial optimization to find an optimal combination plan that meets some global QoS constraints. QoS-based cloud service selection is in the area of NP-hard. Therefore, approximate algorithm is a good way for cloud service selection. GA is a powerful means for combinatorial optimizing things [9-10]. In GA, a population with a fixed size is used to search the whole solution area. Each individual describes a solution in the population. In GA, the design of its operators and parameters will have significant impact on itself. GA is not advantageous for its local convergence. Thus, its efficiency is not enough and its convergence speed is slow. In order to compensate for GA's local search capability, it is necessary that GA combines a local search algorithm.

To compensate for GA's local search capability, this paper presents a hybrid algorithm of GA and SM. Some SM optimization operations and a fitness function are described.

The remaining sections of this paper are as follows. Section 2 described researches of cloud service selection based on QoS. The proposed hybrid algorithm was discussed in detail in section 3. Section 4 presented some test works and discussed test results. Section 5 came to conclusions and noted that the next step in research content.

2 Cloud service selection based on QoS

It is the area of combinatorial optimization to select the best plan from a large number of cloud service composition plans on the basis of all global QoS constraints. The calculation methods based on QoS attributes are divided into two categories. One category is exhaustive algorithm. In this kind of algorithm, all of candidate plans are calculated according to certain rules in order to choose the best plan. The
methods used in [4-6, 11-12, 19-21] fall into this category. The other is approximate algorithm. In this type of algorithm, an ideal composition plan is infinitely close to the best one. At last, a plan that meets all QoS requirements but is not the best one will be gained. [7, 8, 13-18] used some approximate algorithms.

It is a representative calculation method to calculate QoS properties through the establishment of QoS matrix. [4] adopted a run-time services selection method in dynamic service composition. It could select a single better service, but it could not meet the entire QoS requirements. A local optimization algorithm and a global optimization algorithm were proposed in [5, 6]. The local optimization algorithm could not reach a global optimal solution. When the size of composition services was large, the computation of the global computational algorithm had increased a lot. [12] analysed the conditions of triggering service re-selection. It gave the constraint expression for a stateful cloud service selection and the idea of the service re-selection.

Because a service selection problem according to QoS belongs to NP-hard problem [8], the exhaustive combinatorial optimization method is poor scalability and has large calculation. It is a good way to obtain an approximate solution. Heuristics method can be used. A multidimensional 0-1 knapsack problem model was used for multiple QoS constraints selection in [13]. [7, 8, 14-18] used GA for the optimization of service composition.

In order to compensate for the local search capability of GA itself, GA and some kind of local search algorithms need to be combined to enhance its local search capabilities.

SM is a local optimization approach. A combination of GA and SM can form a hybrid algorithm that includes a global optimization algorithm and a local optimization algorithm. The two algorithms complement each other. GA ensures that the hybrid algorithm has a global search capability and can find a global optimal point. SM can add a number of parallel searches in many local areas and it can use local search methods to direct the search. It can not only speed up the process of global optimization, but also solve the “premature” problem of GA to a certain extent. Better convergence speed and search capability can be gotten at the same time.

3 Hybrid algorithm built by GA and SM

In order to solve QoS-aware selection, a special hybrid algorithm is presented in this section. Firstly, some SM operations are proposed on the basis of the decision variable matrix. Lastly, a fitness function is built.

3.1 SM operations

In order to enhance GA’s local search capability, this paper presents a hybrid algorithm that is the combination of GA and SM. The following is the main operation procedure of the hybrid algorithm.

1), building some initial simplexes. Every time a new generation of population is produced in GA, it is necessary to build some local initial simplexes. On the basis of the certain probability, n+1 individuals in a population will be randomly selected to compose a local initial simplex in a n-dimensional space. Each individual's fitness function value shall be the function value of the corresponding vertex in the simplex. NUM is the number of generated initial simplexes. It can be controlled by simplex probability p_sim. It is formula (1).

$$NUM = \text{floor} \left( \frac{\text{SIZE}_{\text{pop}}}{\text{LEN}_{\text{chrom}}} + 1 \right) \times p_{\text{sim}}$$  \hspace{1cm} (1)

In the formula (1), SIZE_{pop} is the number of individuals in each generation of population. The symbol LEN_{chrom} is the length of a chromosome. In the case of a certain population size, the simplex probability p_sim determines the number of simplexes in each generation of population. Its value will have great influence on the local search ability of the hybrid algorithm and the running time of the hybrid algorithm.

2), obtaining the worst individual. Through comparing the function values of n+1 vertices, the vertex with the smallest function value is found and its corresponding individual is denoted by I_{n+1}. I_1, I_2, ..., I_n indicate the individuals corresponding to the remaining n vertices respectively.

3), constructing all decision variable matrixes. The symbol m_{ij} is a decision variable. All of decision variables m_{ij} can constitute a decision variable matrix that is denoted by M. In M, each row represents a decision variable vector of all candidate services of a task. In every decision variable matrix, m_{ij} is 1 only when the jth candidate service of the ith task is selected, otherwise m_{ij} is 0. All decision variable matrixes M_1, M_2, ..., M_s, M_{s+1} are built respectively for I_1, I_2, ..., I_n, I_{n+1}.

4), obtaining a reflection center. n individuals except the worst individual I_{n+1} can decide their reflection center I_c. M_0 is the decision variable matrix of I_0.

5), calculating a reflection point. The reflection point I_0 of the worst individual I_{n+1} can be obtained on the basis of the reflection center I_c. M_0 is the decision variable matrix of I_0.
6), deciding the next operation. The following is to decide whether to terminate simplex operation or not. In \( N_{MU} \) initial simplexes, after every simplex has gained a new individual whose fitness value is better than the worst individual in the simplex and that is able to meet the global user constraints, these new individuals will replace the worst ones and added into the population to participate in the next generation of population genetic manipulations. Otherwise, if the new individual's fitness is less than the worst individual or the new individual does not meet user’s global QoS constraints, the new individual will also replace the worst one in population and form a new simplex to continue with the next iteration of the simplex algorithm. We can end the operation of the simplex until a new individual's fitness is greater than the worst individual and the new individual meets user’s global QoS constraints.

SM can control the evolution direction of GA to make better solutions. The "premature" problem of GA can be solved to a certain extent, because some parallel searches in a number of local solution spaces not only enhance the local search ability but also accelerate the global convergence.

### 3.2 Fitness calculation

The fitness of individual determines the probability of itself being copied to the next generation. Individual's fitness is obtained through a fitness function. The fitness function should take into account an objective function and all global QoS constraints.

1), objective function

The following is how to build an objective function. There are two types of QoS attributes. One is decrease-type (the smaller the value of QoS attribute is, the better the impact of the QoS attribute is, such as Price), and another is increase-type (the larger the value of QoS attribute is, the better the impact of the QoS attribute is, such as Reputation). Different QoS attributes tend to have different dimensions or units. When choosing services, it is difficult to directly compare the relative importance of various QoS attributes. Therefore, it needs for QoS attributes to be normalized. The specific practice is to map the various range of different QoS attributes to [0, 1] interval. The formula (2) is for the decrease-type QoS. The formula (3) is for the increase-type QoS.

\[
Q_{q}^{' \downarrow} = 1 - \frac{Q_{q}}{Q_{qMax}} \tag{2}
\]

\[
Q_{q}^{' \uparrow} = \frac{Q_{q}}{Q_{qMax}} \tag{3}
\]

\( Q_{qMax} \) is maximum of the \( q \)th QoS attribute among all of instances of composite cloud services.

In this paper, \( W_{q} \) is used to indicate the weight value of the \( q \)th QoS attribute. The value of \( W_{q} \) signifies different degree of importance of QoS property. After QoS properties values are normalized, the weight value calculation will be done. Then, the objective function of integrated QoS for the entire composition cloud services can be established. It is formula (4).

\[
Q = \sum_{y=1}^{n} W_{q} Q_{q}^{' \downarrow} \tag{4}
\]

In formula (4), \( n \) is the number of QoS properties.

Through the above steps, an objective function can be made. On the basis of the definition of the equation of the objective function, the bigger the objective function value is, the better the selection result is.

2), fitness function

Some global QoS constraints will be established on the basis of the global QoS requirements that users may bring forward. Penalty function method is a common way of dealing with constrained optimization problems. This way can ensure the population diversity to avoid the algorithm into local convergence. This paper adopts penalty function method to integrate all global constraints and the objective function together. It is formula (5).

\[
\textit{Fit} = Q - \lambda \sum_{j=1}^{n} (\Delta Q_{j})^{2} \tag{5}
\]

In the formula (5), \( Q \) is the objective function, \( n \) is the number of constraint conditions, \( \lambda \) is a coefficient and an experience value. \( Q_{j} \) is the global value of the \( j \)th QoS attribute. The definition of \( \Delta Q_{j} \) is shown as follows.

If \( Q_{j} \) is less than 1, the formula of \( \Delta Q_{j} \) is (6).

\[
\Delta Q_{j} = \begin{cases} 
Q_{j} & \text{if } Q_{j} \geq R_{jMax} \\
0 & \text{if } R_{jMin} \leq Q_{j} \leq R_{jMax} \\
1 - Q_{j} & \text{if } Q_{j} < R_{jMin}
\end{cases} \tag{6}
\]

If \( Q_{j} \) is equal to 1 or larger than 1, the formula of \( \Delta Q_{j} \) is (7).
In the formula (6) and (7), $\Delta Q_j$ is the difference between the maximum and minimum in the $j$th restrictive condition respectively.

### 4 Tests and analyses

In this paper, a more powerful and efficient hybrid search algorithm is composed by GA and SM. The hybrid algorithm will have better search ability and faster convergence speed. Through some tests and test analyses, the capacity and efficiency of presented hybrid algorithm will be validated. In order to verify the effect of services choice done by the hybrid algorithm, some comparison tests between simple GA and the hybrid algorithm were made.

#### 4.1 Test data preparation

Aiming at a fair comparison between two algorithms, they would run in the same hardware and software operating environment, including CPU, memory, OS, development language and IDE, etc. In addition, the simple GA (SGA) and the hybrid algorithm used initialization parameters as following. The population size is 500. The crossover probability is 0.7 and the mutation probability is 0.1.

The values of specific QoS attributes were randomly generated within a certain range. Some global limits for a part of QoS properties were randomly generated. The overall limits were applied to all specific cloud service compositions through the above penalty function method. In the comparison tests, simple GA and the hybrid algorithm were run for 50 times respectively. They would all be used to solve different scale of problems (that is, the number of different tasks and different number of candidate services). In every comparison test, the two algorithms solved the cloud service selection problems with the same size of cloud services combination. The number of tasks in cloud service composition was same. Their tests and analyses will be made in three parts. One is search ability. Another is convergence ability. The last one is running time. At last, different values of simplex probability are analyzed.

#### 4.2 Tests and analyses of search ability

Search capability is that the algorithm can find the optimal solution in a solution space. It can be measured by the quality of the solution that the algorithm searches. In GA, the algorithm search capability can be measured through the fitness value of the final selected individual. In the hybrid algorithm, the bigger the fitness value is, the better the selection result is. The average values of the final fitness values at all running time were taken. A few of test data are listed in Tab.1. In Tab.1, the value of the simplex probability $p_s$ is 0.7.

<table>
<thead>
<tr>
<th>Tasks Number</th>
<th>Average Maximum Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.141:0.148</td>
</tr>
<tr>
<td>25</td>
<td>0.053:0.165</td>
</tr>
<tr>
<td>30</td>
<td>0.036:0.112</td>
</tr>
</tbody>
</table>

As described in Tab.1, comparison of data can fully verify that the hybrid algorithm can get better results of cloud service selection than the simple GA. The efficiency of simple GA is still unsatisfactory, although to a certain extent it solved the service selection problem. When the selection problem is the same size of combination cloud services, the hybrid algorithm can get higher average final fitness value than the simple GA. When the scale of the composition problem is small, the advantage of the hybrid algorithm is not clear. But, when there are a larger number of tasks in a combined service flow, the hybrid algorithm can get much better solutions than the simple GA. In the test conditions of this article, when the number of tasks is more than 25, the hybrid algorithm clearly has stronger search capabilities. This shows that the SM operations have better search capabilities in the larger scale of cloud service selection. Because, the more the number of tasks is, the more the number of individuals in each local initial simplex. The local search ability is stronger. Thus, the search capabilities of the hybrid algorithm are more prominent. This means that the presented hybrid algorithm has greatly enhanced the local search capability on the basis of the combination of SM and GA.

#### 4.3 Tests and analyses of convergence ability

The test results were analyzed on their convergence speeds. Algorithm convergence rate refers to the generation where the final fitness value is reached. In order to validate whether the proposed hybrid algorithm increased the convergence speed, simple GA and the hybrid algorithm were run for many times respectively. The average running generations were taken. A few of result data are listed in Tab.2. The value of the simplex probability $p_s$ is 0.7.

<table>
<thead>
<tr>
<th>Tasks Number</th>
<th>Average Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>356:287</td>
</tr>
<tr>
<td>25</td>
<td>372:268</td>
</tr>
<tr>
<td>30</td>
<td>383:312</td>
</tr>
</tbody>
</table>

As described in Tab.2, comparison of data can fully verify that the hybrid algorithm has faster convergence than the simple GA. This shows that the presented hybrid algorithm has greatly quickened the local convergence rate on the basis of the combination of SM and GA.
4.4 Tests and analyses of simplex probability

The simplex probability $p_s$ determines the number of the simplexes in each generation of population. Tab.3 shows different fitness values that the hybrid algorithm obtains when the value of $p_s$ is different.

Tab. 3. Fitness Comparison (different Simplex Probabilities)

<table>
<thead>
<tr>
<th>Tasks Num</th>
<th>$p_s = 0.1$</th>
<th>$p_s = 0.5$</th>
<th>$p_s = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.132</td>
<td>0.139</td>
<td>0.165</td>
</tr>
<tr>
<td>25</td>
<td>0.089</td>
<td>0.157</td>
<td>0.196</td>
</tr>
<tr>
<td>30</td>
<td>0.056</td>
<td>0.082</td>
<td>0.157</td>
</tr>
</tbody>
</table>

In Tab.3, the bigger the value of $p_s$ is, the greater the value of fitness is. Because, with an increasing value of $p_s$, the number of the simplex is added in each generation of population. As a result, the local search ability is stronger. But, the running time of the hybrid algorithm will be raised too. So, a median value may be accepted.

5 Conclusions

Now, more and more easily used cloud services with the stability characteristics are shared on network. A single atomic cloud services can provide limited functionalities. In order to more fully utilize the shared cloud services, it is necessary to combine shared cloud services to form a new combination of cloud services. It is inevitable for a function to appear a large number of candidate services with the same function property and different non-functional attributes (mainly referring to QoS attributes). It has become an urgent problem that how to fast and flexibly select the best services to meet user’s needs from massive candidate services.

Based on the analyses of composite cloud service selection problem, a simple GA combines a local optimization algorithm – SM. In the result, the search ability and convergence speed can be improved at the same time. The proposed algorithm also includes a fitness function. Through the realization of the above-mentioned algorithm, some strong validations of the proposed algorithm in capacity and efficiency effects were done. The hybrid algorithm can be a good solution for QoS-driven cloud services selection.

The number of individuals in populations is same when the combination sizes in the above experiments are different. If the populations with different sizes can be adopted for different composition scales, the efficiency of algorithm will be greatly improved. Therefore, the next study will examine the dynamic adaptive mechanism of population size.

Acknowledgments: This work was supported by NSFC under Grant No. 60872042, the Fundamental Research Funds for the Central Universities (2011RC0203) and the Co-construction Special Funds of Beijing.

6 References


