A Parameter Adaptive Genetic Algorithm Based Service Compositions

Huizhou Yang[1], Li Zhang[2*]

School of Computer Science, Xi'an ShiYou University, 710065, Shaanxi, China

Abstract. How to select and combine many services with similar functions reasonably and efficiently to provide users with better service is the main challenge in the service composition problem. This is thorny when the number of the candidate Services is huge. Recently, researches transform the service compositions problem as a multi-objective optimizing task, and then the genetic algorithm is commonly used to tackle this issue. However, the fixed crossover probability and mutation probability settings in genetic algorithm usually result to it falls into a local optimal. To improve the performance of the genetic algorithm in the service composition task, this paper proposes an adaptive parameter adjust strategy, which can adjust the crossover probability and mutation probability automatically. The experiment result shows our method has greatly improved the maximum fitness of the final solutions of traditional genetic algorithm.

1 Introduction

Service composition is to combine a large number of atomic services into a combination services with certain function. As the advance in the Internet, the number of the web service has becoming very large, and due to the complexity of customer needs, the select of a composition service which can meeting the customer's requirement becomes a challenging work. Recent literature tackles this issue by transform it into a multi-objective optimization problem. The main task in this problem is to find out a composition service which has a best QoS performance and meeting the user's functional requirement. However, the QoS constrain based service selection problem is a NP-hard problem, it cannot be solved by the trivial methods. Thus, it is naturally for researches resort to some intelligence optimize algorithm, such as Genetic Algorithm [1, 2], Simulated Annealing Algorithm [3] and Ant Colony Algorithm [4, 5] and so on. Due to the simplicity of genetic algorithm, it gradually becoming the most popular method in tackle the service composition issue. For instance, Vanrompay Y et al. propose use the Genetic Algorithms to search the best service variant in the current context, and they argue that the Genetic Algorithms can meets some main functional demanded by services running on mobile systems [6]. Lin et. at., focus on utilizing the genetic algorithm to find out the optimal cloud web service composition [7]. Gao et al., proposed a Tree-coding genetic algorithm for QoS-aware service composition task, and they confirm that the Tree-coding based genetic algorithms run faster than the one-dimensional coding genetic algorithm [8]. However, the genetic algorithm used in these works also fall into some local optimal when the search space of is very huge [9, 10]. To fill this gap, this paper proposes an improved genetic algorithm, which utilizing the non-linear neuron activation function and the information entropy to realize the adaptive crossover probability and mutation probability settings, the experiment result shows our method has significant performance improvement than the existing used genetic algorithm.

2 Problem description

Web service providers, service requirements, and Service agency are the three major players in the Web services technology architecture. The service agency's responsibility is to select an optimal service composition according the user's needs. Typically, a composition service is consisting of a series of different service class, the service class is a group of services that can satisfied a particular function needs, only multiple service classes work together can meet the user's entire functional needs. And each service class is consisting of a lot of atomic service, these atomic services have the same functionality and different QoS [11]. Usually, different users have different attribute requirements for QoS. The research problem of service composition is to select the optimal composition service which can meets the custom's QoS requirement [12]. Suppose that R = $\{R_1,...R_n\}$ represent a set of customer requirements, each requirement R_i can be accomplished by a service class $C_i = \{A_1, A_2,...A_n\}$,where A_i represent a candidate atomic service, each atomic service have several different QoS attributes, the Qos attribute focus on this paper are price, reliability, response time, reputation and safety. The process of service

* Corresponding author: zhangli0223@163.com

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

composition is to find an atomic service A_i in each service class C_i to formulate a service sequence to complete certain function. Beside this service sequence also need to meet the customer's QoS constrains. The QoS constrain $Cons = \{cons_1, ..., cons_n\}$ reflect the user's specific QoS expectation about the composite service. The QoS attribute calculation method of composition service under different structure as show in table 1 [13, 14], the structure of the composition service in this paper is sequential structure.

Table 1. QoS attribute calculation method of composition service

Combination model	response time(t)	Price(p)	reliability	reputation	safety
Sequential structure	$\sum_{i=1}^{n} q_i^t$	$\sum_{i=1}^{n} q_i^p$	$\prod_{i=1}^{n} q_i^{rel}$	$\sum_{i=1}^{n} q_i^{rep}/n$	$\sum_{i=1}^{n} q_i^{rep}/n$
Concurrent structure	$\max\{q_1^t, \cdots, q_n^t\}$	$\sum_{i=1}^{n} q_i^p$	$\prod_{i=1}^{n} q_i^{rel}$	$\sum_{i=1}^{n} q_i^{rep}/n$	$\sum_{i=1}^{n} q_i^{rep}/n$
Selection structure	$\sum_{i=1}^{n} p_i q_i^t$	$\sum_{i=1}^{n} p_i q_i^p$	$\sum_{i=1}^{n} p_i q_i^{rel}$	$\sum_{i=1}^{n} p_i q_i^{rep}$	$\sum_{i=1}^{n} p_i q_i^{rep}$
Loop structure	$k * q_i^t$	$k * q_i^p$	$(q_i^{rel})^k$	q_i^{rep}	q_i^{rep}

3 Parameter adaptive genetic algorithm

Genetic Algorithm (GA) is a most used near-optimal solution find method [15]. Which can find a near-optimal solution in large search spaces by continue iteration. The inspiration of the genetic algorithm comes from the Darwin's theory of evolution, eg. "Survival of the fittest". In Genetic Algorithm, the candidate solutions of the question are called individual, generally, which is represented as a sequence of variables called chromosomes. We evaluated the fitness of each individual in each generation to filtrate individuals thus make sure that the individual which have good fitness value are retained and poor one are eliminated [16]. In genetic algorithm, selection and evolution are two main operation to generate the new individuals. The evolution contains the crossover and mutation operations. The selection process is based on the fitness of each new individual, but don't entirely rely on it, because simply selecting individuals with high fitness will likely result in the algorithm converging to a local optimal solution. Thus the main principle of genetic algorithm is the higher the fitness is, the higher the chance of being chosen is, and the lower the fitness the lower the chance of being chosen is. In the selection step, a predefined number of chromosomes is generated to form the initial generation. After that, the selected individuals will enter the mating process, typically in this step, an invariance crossover probability is set to decide the probability of two selected individuals will mating. Suppose that the mating probability is 0.6 that means 60 percent of the selected "couples" will have offspring. The chromosomes of the mating parent will exchange to generate new chromosomes. A cut-off points it randomly chosen to separate the parent's chromosomes into top and bottom parts. Each individual's chromosome is consisting of the different part of its parents. The mutation operation is another way to generate new individuals, generally, the mutation probability of genetic algorithm is also fixed, based on this probability, a random mutation in the chromosome of a new individual usually changes one byte of the chromosome (0 to 1, or 1 to 0). After the selection, crossover and mutation, the new generation of individuals is generated, those new individuals inherit the information of their parents, and better. These operations repeat circularly until the termination of the conditions are or the maximum number of iterations are reaches [17].

3.1 Service composition by Genetic Algorithm

When deal with the service composition problems, the chromosome in genetic algorithm represent the possible solutions for service composition. We use the one-dimensional integer to encoding the chromosome, so the chromosome can be represented by an integer array, the length of the array represents the number of service classes in a composite service, the value of each element in the array represents the corresponding candidate service, and the order of the element in the array represents the order of the service class. The evolution process in the traditional genetic algorithm as show in table 2.

Table 2. The evolution process in traditional genetic algorithm

Input: Initial population P_I

Output: New generation of individuals

1. Select: $P_s = select(P_I, f(P_I))$
2. Crossover: $P_c = cross(P_s, prob_c)$
3. Mutation: $P_m = cross(P_c, prob_m)$

3.2 Improved crossover and mutation process

The crossover and mutation are two most crucial operation in the genetic algorithm optimize solving process, it is the main method to generate new individuals, the large the probability of crossover $prob_c$, the easier to generat a new individual. The greater the mutation probability, the less likely to fall into the local optimum. However, the increase of these two probabilities will cause the convergence of the algorithm becoming slowly. So far, most of the current genetic algorithms in service composition use fixed crossover and mutation probabilities, this is unreasonable. Intuitively, this One-Fix-All parameter seting strategy certainly not suitable for all data sets, which may have a high performance on some data and poor on another data.

Until now, there have no effectively rules to guide these parameter decision process, the tentative experiments are time consuming. And in practice, we hope these two parameters are adjustable in different conditions. In the initial stage of evolution, a large crossover probability and mutation probability is expect to increase the ability of generating new individual, with the increasing of the iteration, the whole population gradually becomes better, in this case we will expect a small crossover probabilities and mutation probabilities, and when the diversity of the population declines, a large crossover probabilities and mutation probabilities is needed to improving the ability of generating new individuals, correspondingly, when the diversity of the population increases, these two parameters should decrease a bit. Base on above analysis, we propose an adaptive crossover, mutation probabilities adjust strategy as show in Eq. 1 and Eq. 2.

$$P_c = P_{c\max} * (1 - \frac{e^{\alpha} - e^{-\alpha}}{e^{\alpha} + e^{-\alpha}}) \quad (1)$$

$$P_m = P_{m\max} * (1 - \frac{e^{\alpha} - e^{-\alpha}}{e^{\alpha} + e^{-\alpha}}) \quad (2)$$

Where $P_{c\max}$ and $P_{m\max}$ represent the max crossover probabilities and mutation probabilities, separately, α is a variable which can reflect the population iteration number and the population diversity as show in Eq. 3.

$$\alpha = \frac{gen}{\max_{gen}} (\frac{entro_t}{entro_{\max}}) \quad (3)$$

In the left part of α , we use gen to denote the population iteration number, $\max_{gen}$ is a regularization constant, and the right part of α indicates the population diversity of each iteration. The $entro_t$ and $entro_{\max}$ represent the entropy of the t-th population and the max entropy of the population respectively, which can be calculated according Eq. 4, and Eq. 5.

$$entro_t = -\sum_{i=1}^{m} \frac{k_i}{n} \log \frac{k_i}{n} \quad (4)$$

$$entro_{\max} = -\sum_{i=1}^{m} N \quad (5)$$

Where, N represent the population scale, m represent the kinds of different chromosome, k_i denote the number individual in k-th class. Different with the traditional one-fix-all parameter setting method, the improved method can adjust the crossover probabilities and mutation probabilities according the condition of the number of population iteration and the diversity of the population automatically.

4 Experiment evaluation

In this section, a series of experiments were designed to demonstrate the effectiveness of the proposed method. All the experiments are performed in Matlab (2015a) on computer with Inter Core (TM) i7-6700 CPU (3.40GHz and 8G RAM). In our experiments, the QoS attributes we focused are price, safety, reliability, response time and reputation, the service class in the composition service is set to 45, and each class has 30 candidate atomic services, the corresponding QoS value for each atomic service is generated randomly based on the parameter ranges in table 2. The parameter settings of the genetic algorithm in our experiment is followed [18] eg. crossover probability 0.7, mutation probability 0.1.

First, we investigate the changes of maximum fitness along with the iterations number of the population, the experiment result is show in figure 1, from which we can find that the improved genetic algorithm can significantly increase the maximum fitness of the population, and as the number of population iterations increases, the advantages become more obvious.

Table 3. The generation interval of different QoS values

price	reliability	response time	reputation	safety
[100,500]	[0.3 0.7]	[0.2, 4]	[0.2,0.8]	[0.1, 0.9]

In the second part of the experiment, we investigate the change of maximum fitness along with the number of candidate services, the experiment result is show in figure2.

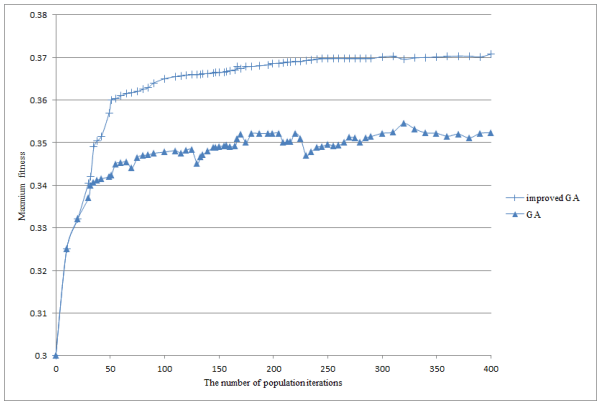

Figure 1. The change of maximum fitness along with the population iterations number

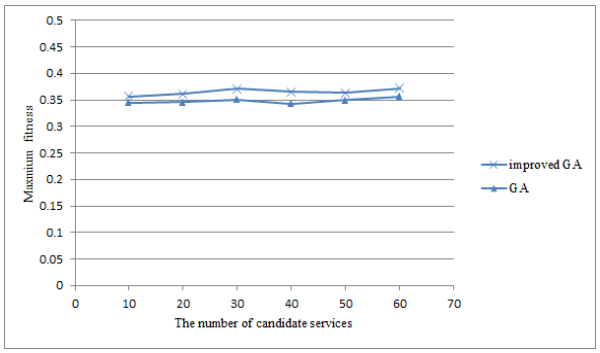

Figure 2. The change of maximum fitness along with the number of candidate service

From figure 2, we can find that although with the increase of the number of atomic services in the service class the maximum fitness of the improved genetic algorithm does not increase significantly, its

performance is also better than traditional genetic algorithm. After that we changes the number of the services class in the composition service, to investigate the changes of maximum fitness, the experiment result as show in figure 3

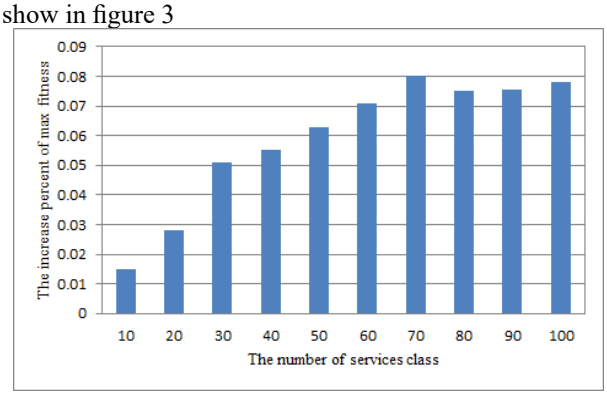

Figure 3. The change of maximum fitness along with the number of service classes

It is easy to conclude from figure 3 that with the increase of number of the service class in the services combination process, the maximum fitness of improved genetic algorithms has been continuously improved, which means that even the number of service classes in the composition service is increased, it superiority of improved genetic algorithms will not decreased.

5 Conclusion

Genetic Algorithm is the most popular used method to solve the multi-objection optimize problem, especially in the field of service composition optimization. However, the traditional one-fix-all crossover probability and mutation probability setting strategy result it often fall into a local optimal, which has significant reduces the quality of combination service. To address this issue, this paper proposes an adaptive crossover probability and mutation probability adjust method, which can adjust these two perimeters automatically to prevent the algorithm fall into a local optimal, the experiment result shows our method have an obvious performance improvement than previous genetic algorithm. It should be noticed that the method proposed in this paper is a general one, which can be generalized to other practical applications, as long as it can be formulated as a multi-objective optimize problem.

References

1. Liu, H., Zhong, F., Ouyang, B., & Wu, J. (2011). An Approach for QoS-Aware Web Service Composition Based on Improved Genetic Algorithm[C] International Conference on Web Information Systems and Mining. IEEE, 2011:123-128.
2. Maulik U, Bandyopadhyay S. Genetic algorithm-based clustering technique[J]. Pattern Recognition, 2004, 33(9):1455-1465.
3. Zeb A, Khan M, Khan N, et al. Hybridization of simulated annealing with genetic algorithm for cell formation problem[J]. International Journal of Advanced Manufacturing Technology, 2016, 86(5-8):1-12.
4. Qian X, Huang M, Gao T, et al. An improved ant colony algorithm for winner determination in multi-attribute combinatorial reverse auction[C] Evolutionary Computation. IEEE, 2014:1917-1921.
5. Duan H B, Wang D B, Zhu J Q, et al. Development on ant colony algorithm theory and its application[J]. Control & Decision, 2004, 19(12):1321-1320.
6. Dong Y Y, Hong N I, Deng H J, et al. Service Selection Strategy Offering Global Optimal Qulity of Service[J]. Journal of Chinese Computer Systems, 2011, 32(3):455-459.
7. Alrifai M, Skoutas D, Risse T. Selecting skyline services for QoS-based web service composition[C] International Conference on World Wide Web. ACM, 2010:11-20.
8. Mishra B S P, Dehuri S, Mall R, et al. Parallel Single and Multiple Objectives Genetic Algorithms: A Survey[J]. International Journal of Applied Evolutionary Computation, 2011, 2(2):21-57.
9. Yun Y, Nakayama H. Utilizing expected improvement and generalized data envelopment analysis in multi-objective genetic algorithms[J]. Journal of Global Optimization, 2013, 57(2):367-384.
10. Xue C, Dong L, Li G. An Improved Immune Genetic Algorithm for the Optimization of Enterprise Information System based on Time Property[J]. Journal of Software, 2011, 6(3):436-443.
11. Canfora G, Penta M D, Esposito R, et al. An approach for QoS-aware service composition based on genetic algorithms[C] Conference on Genetic and Evolutionary Computation. 2005:1069-1075.
12. Strunk A. QoS-Aware Service Composition: A Survey[C] Eighth IEEE European Conference on Web Services. IEEE Computer Society, 2010:67-74.
13. Lei L, Dong Y. Multi-objective genetic optimization algorithm for SLA-aware service composition problem[J]. Journal of Jilin University, 2015, 45(1):267-273.
14. Lin C H. Study of Optimizing Web Service Composition - Using Genetic Algorithm and Case-based Reasoning[J]. 2010.
15. Todd S J. Genetic optimization method and system: US 9047569 B2[P]. 2015.
16. Schuller D, Polyvyanyy A, García-Bañuelos L, et al. Optimization of Complex QoS-Aware Service Compositions. [C] Service-Oriented Computing -, International Conference, ICSOC 2011, Paphos, Cyprus, December 5-8, 2011 Proceedings. DBLP, 2011:452-466.
17. Ye Z, Zhou X, Bouguettaya A. Genetic algorithm based QoS-aware service compositions in cloud computing[C] International Conference on Database Systems for Advanced Applications. Springer-Verlag, 2011:321-334.

18. Mueller-Bady R, Kappes M, Palomo-Lozano F, et al. Maintaining Genetic Diversity in Multimodal Evolutionary Algorithms using Population Injection[C] on Genetic and Evolutionary Computation Conference Companion. ACM, 2016:95-96.