TELKOMNIKA, Vol.12, No.4, December 2014, pp. 1053~1063
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v12i4.786

A Reliable Web Services Selection Method for Concurrent Requests

Guiming Lu*, Yan Hai, Yaoyao Sun
College of Information Engineering, North China University of Water Resources and Electric Power, Zhengzhou, 450000, China
*Corresponding author, e-mail: lgm@ncwu.edu.cn

Abstract

Current methods of service selection based on quality of service (QoS) usually focus on a single service request at a time, or let the users in a waiting queue wait for Web services when the same functional Web service has more than one requests, and then choose the Web service with the best QoS for the current request according to its own needs. However, there are multiple service requests for the same functional web service at a time in practice and we cannot choose the best service for users every time because of the service's load. This paper aims at solving the Web Services selection for concurrent requests and developing a global optimal selection method for multiple similar service requesters to optimize the system resources. It proposes the improved social cognitive (ISCO) algorithm which uses genetic algorithm for observational learning and uses deviating degree to evaluate the solution. Furthermore, to enhance the efficiency of ISCO, the elite strategy is used in ISCO algorithm. We evaluate performance of the ISCO algorithm and the selection method through simulations. The simulation results demonstrate that the ISCO is valid for optimization problems with discrete data and more effective than ACO and GA.

Keywords: *web service selection, ISCO, deviating degree*

1. Introduction

With the development of cloud computing technology and SaaS model, more and more Web services are used under the cloud environment. In such open and dynamic environment, however, there are usually many Web services instances with similar function but great different performance. How to choose the high quality services which are most satisfied for users from the vast amounts of Web services, has become one of the hot issues in the research of service computing and cloud computing field .

Current methods of service selection usually focus on a single service request at a time, or the selection of a service with the best QoS at the user's own discretion, or let the users in a waiting queue wait for Web services when the same functionality of a Web service has more than one service request [1],[2], and then choose the best Web service for the current request. This method does not take the following situation into account where there are multiple user requests for a Web service at the same time. In such situation, conflicts occur when too many requesters select the same best web service because the load capacity of each service is limited. This is likely to lead that the requested service's QoS is reduced, and the entire service system crash and the users' needs cannot be met.

In addition, different Web service users have different QoS attribute preference. So we don't need to choose the best service for each user. The competition and conflict among different users for the same service may lead to the following situations: Some Web services which can satisfy users' Qos requirements are unused and some high QoS Web services serve the users with low QoS requirements, but some users with higher QoS requirements will be in the long time waiting state because it temporarily cannot get the high QoS service, and thus will certainly cause uneven distribution and waste of the service resources, affecting the overall performance of the service system. As you can see, when many users call the same service, the service's QoS and actual load capacity will reduce. Therefore, in order to make the higher QoS service can be used most effectively and the services satisfied users' need not to be idle, effective service selection methods should be adopted. At the same time it can maximize the user's overall satisfaction as far as possible, and guarantee service load balancing, and improve the overall performance of the service system.

Received September 2, 2014; Revised October 15, 2014; Accepted November 4, 2014

From the above research situation, the existing service selection method rarely considers the load balancing problem. This paper mainly studies how to solve the conflict situation of the concurrent service requests, and proposes a reliable Web services selection method for concurrent requests. Because that the optimal solution of the concurrent Web service requests is a set of services sequence number whose value is discrete data, and traditional social cognitive optimization only applies to continuous data. So this paper uses the minimum value of the deviating degree which is the deviation of the candidate Web services' QoS value with concurrent requests' QoS constraints as the objective function, and designs an improved social cognitive algorithm which uses genetic algorithm for observation learning to solve the service optimization problem. And thus balance service resources , improve system performance and better meeting the efficiency of Web service selection.

2. Related Work

With the continuous development of cloud computing technology and the widespread use of a large number of Web services on the network, the increase of the number has brought great convenience for software developers, but meanwhile makes them face the service information overload problem. How to provide the user with the appropriate Web service has become a great challenge for developers.

In recent years, the QoS attribute has been considered in the research of the Web service selection field. Now some researchers make research on a single user requests a functional Web service based on QoS attribute, and proposed some service selection method. Zeng et al [3] present a middleware platform for composition expressed as utility functions over QoS attributes, and using integer programming make users get the best Web services they need, maximizes user satisfaction. Liu, Anne, Ngu and Zeng [4] proposed a QoS computation and policing in dynamic web services selection. In this way, it transforms the problem of service composition into a mixed integer linear problem and users can obtain their best needed services. Yu and Lin [5],[6] proposed a service selection system with end-to-end QoS constraints and put forward a algorithm based on Multiple Choice Knapsack Problem (MCKP) and Multi-dimension Multi-choice 0-1Knapsack Problem (MMKP).Maximilien and Singh raised an independent service selection architecture based on service agent in the literature [7], and considered the credibility of the service in the literature [8]. In the model of Danilo and Barbara proposed [9], the local and global constraints can both be specified and they also proposed optimizing formula to make global optimization, without prior to optimize each possible execution paths. In [10], mixed integer programming was utilized to find the optimal decomposition from QoS global constraints to local constraints, and used a distributed local selection to find the best Web service which can satisfy the local constraints, and then proposed the local optimization and enumeration method. This service composition method designed to filter the candidates of each task through local constraints, to reduce the number of candidate services and improve service quality, and enumerate all compositing solutions to find the optimal one. Approaches proposed in [11] and [12] aim to select Web services for users , but these existing services selection studies only focus on a single service request a functional type of Web service, or is intended to find an effective method to solve the composite service selection model they proposed .

The above methods do not consider the case where many users concurrently request the same function Web service at the same time. Although [13] and [14] consider the concurrent service requesters request the same service at the same time, the solution is only to let the multiple requesters to wait in the line of queue in the event of conflict. In fact, because the different requests have different QoS constraints, multiple concurrent requests can be assigned to multiple services simultaneously. In order to avoid the Web service overloading, and improve the performance of the overall system. This paper proposes a reliable service selection method for the problem of concurrent users request for the Web services.

3. The description of Web services selection for concurrent requests

Web service QoS guidelines mainly include cost, response time, service availability, service reliability, and Successful Rate, etc. General Web service selection methods are given below. Assuming that the set of candidate Web services which can satisfy user functional

requirements already exists, and there are n Web services meeting user demand through function matches. The QoS attribute sets of the candidate Web services is: $< q_1,...,q_i,...q_k >$. General Web services selection process can be divided into the following main steps:

(1) QoS Standardization of Candidate Web services and concurrent requests

Web service QoS attributes can be divided into two types [15]: One is the positive criteria, i.e., the greater the value the better its quality, such as reliability, availability, credibility, etc.; another is negative criteria, i.e., the greater the value the poorer the quality, for example, reaction time and cost and so on. To make the Web service QoS values comparable, it is necessary to standardize all the QoS attribute values that all the QoS values are converted into a unified value interval. Commonly used QoS standardization formula has the following two kinds:

the positive ones as defined in (1):

$$V_{ij} = \begin{cases} \dfrac{Q_{ij} - Q_j^{\min}}{Q_j^{\max} - Q_j^{\min}} & if \;\; Q_j^{\max} - Q_j^{\min} \neq 0 \\ 1 & if \;\; Q_j^{\max} - Q_j^{\min} = 1 \end{cases} \tag{1}$$

the negative ones as defined in (2):

$$V_{ij} = \begin{cases} \dfrac{Q_j^{\max} - Q_{ij}}{Q_j^{\max} - Q_j^{\min}} & if \;\; Q_j^{\max} - Q_j^{\min} \neq 0 \\ 1 & if \;\; Q_j^{\max} - Q_j^{\min} = 0 \end{cases} \tag{2}$$

In the formula (1) and (2), $Q_j^{\max}$ is the maximum of the j-th attribute in the mass matrix, $Q_j^{\max} = Max(Q_{ij}), 1 \le i \le n$; $Q_j^{\min}$ is the minimum of the j-th attribute in the mass matrix, $Q_j^{\min} = Min(Q_{ij}), 1 \le i \le n$. By formula (1) and (2) can put all QoS values to the uniform value interval, and normalized QoS values indicate that the greater the value the better the quality. By the methods of attribute value standardization, unified the direction of the QoS values, makes for a Web service QoS evaluation has the scientific nature and rationality. Similarly, for QoS constraints of user requests to do the same standardization.

(2) Mathematical model for global optimal

Web services selection need to make choices for each concurrent user a candidate Web services. In order to maximize the satisfaction of users and optimize the deployment of Web services resources, the problems need to be solved is how to assign requests Web service, making the deviating degree minimal between each candidate services and their corresponding requests. Selection model shown in Figure 1:

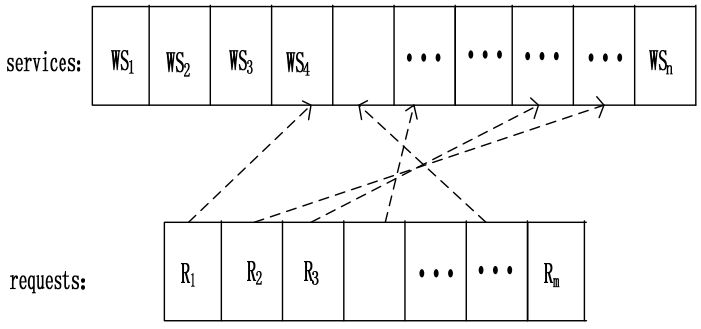

Figure 1. Selection model of candidate services and requests

Suppose n represents the number of Web services, m represents the number of requests, and $n \geq m$. The user preference for QoS attributes is $W = (W_1, W_2, \ldots, W_k)$, and $\sum_{i=1}^{k} W_i = 1$. So the global optimal mathematical model for concurrent user requests with QoS constraints can be defined as:

The objective function:

$$Min \sum_{i=1}^{m} Dev(R_i WS_j)$$

$$Dev(R_i WS_j) = (r_{i1} - q_{j1})w_1 + (r_{i2} - q_{j2})w_2 + \ldots + (r_{ik} - q_{jk})w_k \quad (3)$$

$Dev(R_i WS_j)$ shows the QoS deviating degree between the i-th requests and the j-th candidate Web service. Where r_{ik} is the k-th QoS attribute constraint value for the i-th user requests, q_{jk} is the k-th QoS attribute constraint value for the j-th candidate service, $WS_{k1}, WS_{k2}, \ldots, WS_{kn}$ is a solution,

i.e., $WS_{k1}, WS_{k2}, \ldots, WS_{kn}$ is offered respectively to $R_1, R_2, \ldots, R_m$. Need to be aware of is the attribute values in formula are standardized, values between 0-1.

As the solution of the Web service selection for the concurrent requests is a set of sequence number, and the sequence number belongs to the discrete data issues. This paper for this problem using genetic algorithms to observational learning, at the same time, using elite reserved strategy, proposed an improved social cognition algorithm (ISCO) to solve the above optimization problem.

4. Service selection method based on ISCO

Given the evolutionary algorithm with advantages of parallel computing, group optimization, does not need to provide information associated with application background, etc., so a variety of evolutionary algorithms can be used to solve the problem of Web service selection based on QoS. This paper proposes an improved social cognition algorithm (ISCO) to solve the above optimization problem.

4.1. Improvement of the social cognitive algorithm

1)The use of genetic algorithm:

Genetic Algorithm (GA) simulates the process of natural selection and biological evolution by computer, and searches the relatively optimal solution through survival of the fittest mechanism. It adopts global parallel optimization search method, and does not depend on gradient information, has the global search ability. Compared with traditional optimization methods, it is simple to use, with good global search ability and other characteristics, and is one of the most effective ways to solve the optimization problem today.

Since SCO algorithm is based on the field search, and the optimization problem with continuous solution space within the scope of its search rules, it does not apply to solve the problems of discrete solution space. ISCO algorithm respectively uses crossover and mutation operator of genetic algorithm for learning agents and the new solution obtained through imitation learning. After each crossover and mutation operations is performed, select better solution both before and after the operation can make search space expansion, the rapid increase solving range diversity, at the same time avoiding the local optimum.

The service options of this algorithm need to determine the chromosome representation, the calculation of fitness function, as well as the crossover and mutation operations, etc. Each chromosome is represented as a Web service solution, namely a Web

service sequence identity, composed by a number of genetic loci. Each gene position corresponds to service identification; the value of the i-th gene bit is the number of the service selected by the i-th request in the candidate set of services. The fitness function is the deviating degree in 5.5.Crossover operation is to take a crossover method to two chromosomes (two different kinds of service composition scheme) while mutation operation is to change the value of certain genes bits in these two chromosome, resulting in two new chromosomes.

2)The use of elite strategy:

GA genes do not necessarily reflect the true nature of the problem to be solved, because the crossover and mutation operators could lead to the best individual of the current population lost in the next generation, and this phenomenon will cyclically appear in the evolutionary process. So it cannot reach the purpose of cumulating better genetic, but may undermine a good combination, and cannot converge to the global optimum. For elite strategy, the elite individual (the best individual appeared so far in the evolutionary process) directly copied to the next generation instead of matching and crossover, and can avoid the best individual being destroyed for hybrid operation.

The elite strategy [16] (Maintain the best solution found over time before selection) is put forward by De Jong for the genetic algorithm. For Genetic algorithms, the ability to converge to the global optimal solution is the primary problem. De Jong makes the definition for elite selection methods as follows [16]:

When set to the first t-generation, a (t) is the best individual in the group. And set A (t + 1) as the new generation of groups, if A (t +1) in the absence of a (t), then add a (t) to A (t + 1) as the n + 1 individual in the A (t + 1), where n is the size of the groups.

Elite strategy generated a significant role to improve the global convergence ability of standard genetic algorithm, and Rudolph has theoretically proved the standard genetic algorithm with elite strategy is global convergence. Here put forward an improved social cognitive algorithm (ISCO algorithm) ,which takes each agent with the knowledge point to do crossover and mutation, produce the better solution into the next generation of knowledge base, regard the optimal solution of each iteration as elite individuals, and avoid the best individual being destroyed because of hybrid operation.

3)The use of dynamic learning agent:

According to SCO algorithm, the selection of the agent's own knowledge point is static each time library updates itself. Because that the knowledge base is constantly updated, to enhance the learning ability of the algorithm, this paper adopts dynamic learning agent, that is, randomly select knowledge learning as the new learning agents after each update of the base so that the agent own knowledge points evolve with the evolution of the knowledge base and the algorithm can improve the overall learning ability.

4.2. The algorithm description of Web services selection for concurrency requests

In Web service selection optimization problems for concurrent users' demands, the knowledge points corresponds to the sequence number of Web service selection, and position level corresponds to the evaluating Value of Web service selection which is the deviating degree. This paper uses a random method to generate initial solutions. Web service selection process based on ISCO algorithm can be described as:

Algorithm. Web service selection algorithm for concurrent requests
input: QoS values of candidate Web services and
user requests，the Maximum number of iterations
$N_{\max}$， The number of initial solution N_{POP}
output：the optimal solution of Web service selection

Step1: the initialization process:

(1) N_{POP} solutions are randomly generated where each solution is a set of sequence number for Web service, and the deviating degrees of requests with Web services are calculated as evaluation value of the solution.

(2) N_c is randomly selected from N_{POP} solutions as the learning agents, but the same solution is not allowed to be assigned to multiple agents.

Step 2: Vicarious learning process

for i=1 to N_c // For N_c learning agents

{

Imitation learning // A number of solutions are randomly generated, and find N_c better solutions from them.

Observational learning // Let X_1 be the N_c learning agency's own solutions, X_2 be the N_c better solutions, then divided them into several sections. Select intersection randomly and do crossover operation for X_1, X_2 to get two new solutions X_1', X_2' using genetic algorithm, and then choose the better one as X_3, and continue to mutate X_3 to get new solution X_3' which is stored in the database.

(Where the observational learning processes diagram as shown in Figure 2)

}

Step 3: database update process

Remove the same amount of worst solution with learning agent; meanwhile, choose the optimal solution of this with the previous generation.

Step4: Dynamic Proxy

A plurality of mutually different learning agents is randomly generated.

Step5: End of algorithm judgment

IF (to achieve a pre-determined number of iterations)

Output optimal solution

ELSE

Increase the number of iterations 1, return to the step 2.

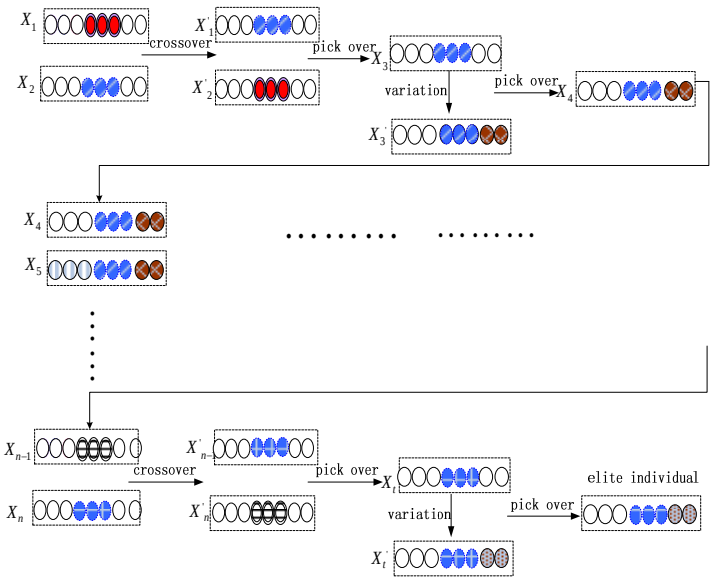

Figure 2. Schematic diagram of observational learning

5. Simulations

5.1 The experiment design

Suppose there are 300 candidate Web services. Consider the five common QoS attributes: Cost, Response Time, Availability, Reliability, Load, and QoS values of candidates Web services are generated randomly within a certain range. The range of them as follows: $0 \le C \le 300\ \$$, $0 < T \le 30\ sec$, $0.7 < Ava \le 1$, $0.75 < Rel \le 1$, $0 \le Loa \le 300$. Meanwhile set 6 concurrent users, and its QoS values are fixed as shown in Table 5.1, the attribute weights were set to: $W = \{0.2, 0.1, 0.15, 0.25, 0.3\}$; The experiment achieves the ISCO algorithm through C + + programming language , and the configuration of the PC which is run by the algorithm is: CPU：Pentium(R)4 2.66GHZ, Memory：4G；OS：Microsoft Windows 7。

Table 1. The QoS constraints of concurrent users

UserID	C	T	Ava	Rel	Loa
1	50	10	0.75	0.80	200
2	150	20	0.85	0.90	100
3	80	15	0.80	0.85	150
4	200	10	0.95	0.80	250
5	75	25	0.90	0.95	50
6	150	10	0.85	0.75	100

5.2 The determination of ISCO algorithm parameters

The values of key parameters in the ISCO algorithm have a great influence on the performance of the algorithm, and so the appropriate values should be set in the algorithm parameters before applying ISCO to solve Web service selection problem in order to give full play to the algorithm's ability. The key parameters of ISCO algorithm include: population size, namely the number of initial solutions N_{POP}, the number of learning agents N_c, maximum number of iterations N_{MAX}, the number of optimal solutions what are draw in the imitation learning competition λ.

1) Determine the parameters: Population size N_{POP}

Population size N_{POP} is the number of initial solutions. In the experiment, set the number of excellent solutions presented as λ = 3; Because $N_{POP} = 3 * N_c$ in ISCO algorithm, so N_{POP}, N_c are considered setting different values of the following:

(1) $N_{POP} = 100, N_c = 30;$
(2) $N_{POP} = 150, N_c = 50;$
(3) $N_{POP} = 200, N_c = 70;$
(4) $N_{POP} = 250, N_c = 80;$
(5) $N_{POP} = 300, N_c = 100;$

The maximum numbers of iterations N_{MAX} are taken respectively 10,100,1000,10000, then execute the algorithm and record the searched solution in algorithm. The results are shown in Figure 5.4, abscissa axis represents different values of N_{POP}, vertical axis represents the evaluation values of solutions that are searched by algorithm when N_{POP} have different values.

It is found in experiments that regardless what is the value of maximum number of iterations of the algorithm N_{MAX}, when $N_{POP} = 200, N_c = 70;$ the algorithm search ability is strongest, that is to say N_{MAX} has no effect on the selection of N_{POP}.

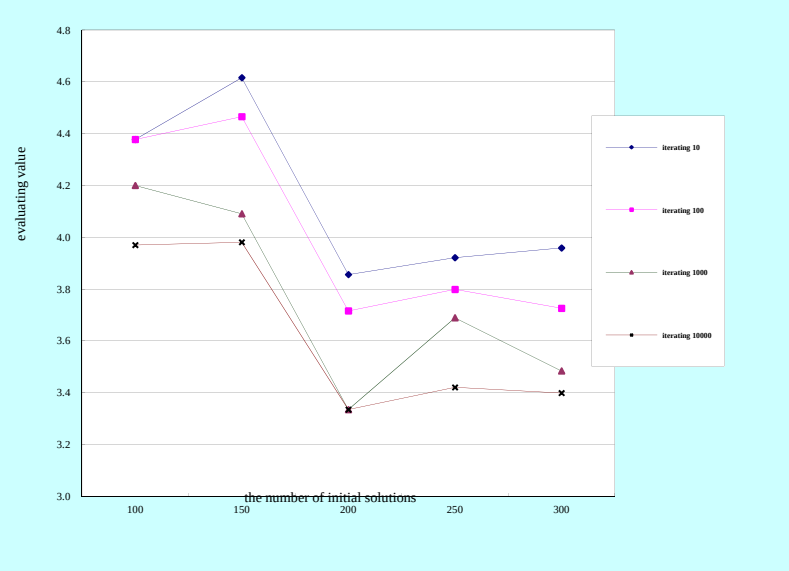

Figure 3. The influence of the number of initial solutions on the performance of the algorithm

2) Determine the parameters: The number of excellent solutions proposed λ

At the initial moment, set the amount of the population, namely the number of initial solutions $N_{POP} = 200$, the number of learning agent $N_c = 70$; the maximum number of iterations of the algorithm $N_{\max} = 5000$ .In observational learning, each time do crossover on 2 nodes and then do variation on that 2 nodes where crossover and variation points are randomly generated, in order to improve the algorithm's search ability and convergence speed, it need to determine the value of λ(the number of excellent solutions extracted).λ is set from 1 to 10, execute the algorithm and record the solutions that are searched by the algorithm. The experimental results are shown in Figure 4,, and the abscissa axis represents different values of λ, the ordinate axis represents the evaluation values of the solutions that are searched by the algorithm. As can be seen from Figure 4, when λ = 7 the algorithm has the strongest searching capabilities, and therefore determine λ= 7.

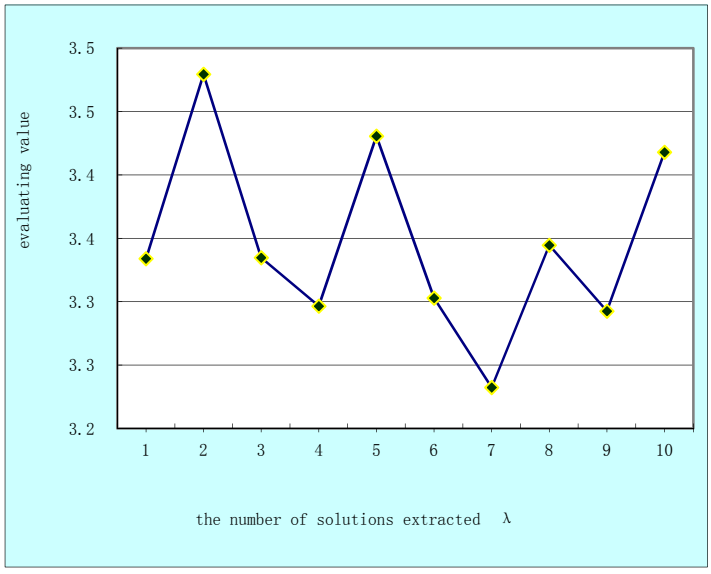

Figure 4. The influence ofλ(the number of solutions extracted) on the performance of the algorithm

5.3 Comparison with other algorithms

Basic genetic algorithm (GA algorithm) has a strong parallelism and global search capability in solving Web service selection problem. Ant colony optimization (ACO algorithm) using meta-knowledge and heuristics to guide the search, is an evolutionary algorithm based on swarm intelligence having the advantages of robustness, parallelism and easy to implement, etc. In view that these two evolutionary algorithms for solving optimization problems perform relatively more excellently, the experiments compare the accuracy and the running time adopting three algorithms to verify the effect of ISCO algorithms. After determining the values of the parameters, this experiment at the initial time set $N_{POP} = 200$, $N_c = 70$,λ=7, while using the improved social cognitive algorithms (ISCO algorithm), genetic algorithm (GA algorithm), ant colony algorithm (ACO algorithm) to solve the above service selection problem. Three algorithms had the same experimental environment, and all of them use C + + programming language. We randomly recorded three algorithms' running time, the corresponding number of iterations, the evaluation values of solutions searched by three algorithms, where the units of evaluation value and running time are seconds. Experimental results are shown in Table 2:

During the experiment, each algorithm was iterated 10,100,500,1000,2000,4000,5000 times, and the searched solutions and the running time of the algorithms were recorded. Comparison on searching performance of three algorithms as shown in Figure 5:

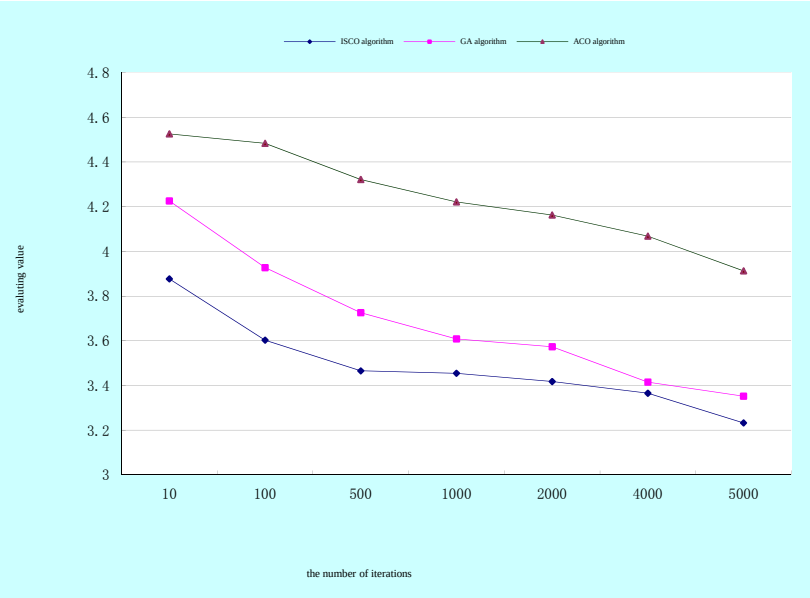

Figure 5. Comparison on searching performance of three algorithms

Comparison on running time of the three algorithms in the case of the same number of iterations as shown in Figure 6:

	ISCO algorithm		**GA algorithm**		**ACO algorithm**	
iterations	Evaluation value	Running time	Evaluation value	Running time	Evaluation value	Running time
10	3.87672	0.016	4.2256	0.009	4.52553	0.064
100	3.60277	0.156	3.92734	0.101	4.48333	0.756
500	3.46542	0.797	3.72554	0.617	4.32179	3.781
1000	3.4545	1.563	3.60842	1.264	4.22138	7.428
2000	3.41747	3.125	3.57311	2.581	4.16251	13.951
4000	3.36525	6.265	3.41491	5.129	4.06817	28.001
5000	3.23241	7.796	3.35182	6.451	3.91255	35.279
	convergence 3.23241		convergence 3.35182		convergence 3.91255	

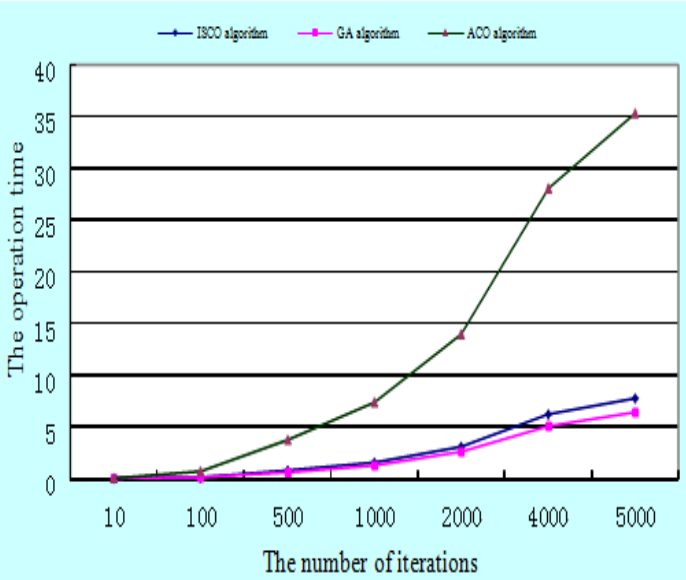

Figure 6. Comparison on running time of the three algorithms with the same number of iterations

As can be seen from Table 1 and Fig. 6, ISCO algorithm outperforms GA algorithm and GA algorithm outperforms ACO algorithm through the analysis from the searching performance; GA algorithm outperforms ISCO and ISCO algorithm outperforms the ACO algorithm through the analysis from the convergence rate; It can be seen from the calculation process of ISCO that ISCO have advantages of low complexity, less parameters, and easy to implement.

As can be seen from Fig. 6, in the whole experiment, when three algorithms are in the same situation iterations, the time used by ISCO algorithm to search the optimal solution is 7.796 s, the time used by GA algorithm is 6.451s, and the time of ACO algorithm is 35.279s, the difference between the running time of ISCO and GA is not large under the same number of iterations; Three algorithms search for the optimal solution in the iteration times 5000, and convergence has begun; Although the time used by ISCO algorithm to search the optimal solution is a little inferior to that used by the GA algorithm; But more importantly, the solution searched by ISCO is the best and the solution searched by GA is the second and that searched by ACO algorithm is the worst. By comparing, it can be concluded that the convergence speed and optimization ability of ISCO algorithm are better than those of GA and ACO algorithm generally.

6. Conclusion

In order to solve the conflict of concurrent requests, this paper presents a Web service selection method for concurrent demand. Through the simulation, it can be concluded that ISCO algorithm constructed in this paper is an optimization algorithm with relatively strong searching capabilities and faster searching speed, is an efficient new evolutionary algorithm solving discrete optimization problems. Algorithm rules are simple, easy to implement, and the execution time is less in the same number of iterations, which prove the feasibility of the algorithm. The algorithm has a strong expanding capacity, can be extended to other applications. Therefore, ISCO algorithm can meet the actual application requirements and efficiency requirements that the Wed services selection for concurrent requests needed.

References

[1] T. Yu, KJ. Lin. Service selection algorithms for Web services with end-to-end QoSconstraints. *Information Systems and Ebusiness Management*. 2005; 3: 103-126.

[2] T. Yu, Y. Zhang, KJ. Lin. Efficient algorithms for Web services selection with end-to-end QoSconstraints. *ACM Transactions on the Web (TWEB).* 2007; 1: 1-6.
[3] L. Zeng, B. Benatallah, AHH. Ngu, M. Dumas, J. Kalagnanam, H. Chang. QoS-aware middleware for web services composition. *Software Engineering, IEEE Transactions*. 2004; 30: 311-327.
[4] Y. Liu, H. Anne, H. Ngu, L. Zeng. *QoS computation and policing in dynamic web service selection*. In Proceedings of the International World Wide Web Conference. 2004: 66-73.
[5] T. Yu, KJ. Lin. Service selection algorithms for Web services with end-to-end QoS constraints. *Information Systems and Ebusiness Management*. 2005; 3: 103-126.
[6] T. Yu, Y. Zhang, KJ.Lin. Efficient algorithms for Web services selection with end-to-end QoS QoS constraints. *ACM Transactions on the web (TWEB)*. 2007; 1: 1-6.
[7] EM. Maximilien, MP. Singh. Agent-based architecture for autonomic web service selection. *Journal on Web Semantics*. 2003: 261-279.
[8] EM. Maximilien, MP. Singh. Toward autonomic web services trust and selection. *ACM.* 2004: 212-221.
[9] D. Ardagna, B. Pernici. *Global and local QoS guarantee in web service selection.* Third Internation Conference on Business Process Management. 2006: 32-46.
[10] M. Alrifai, T. Risse. *Combining Global Optimization with Local Selection for Efficient QoS-aware Service Composition.* In Proceedings of the International World Wide Web Conference. 2009: 881-890.
[11] Z. Zheng, H. Ma, MR. Lyu, I. King. *WSRec: A Collaborative Filtering Based Web Service Recommender System.* IEEE International Conference on Web Services. 2009: 437-444.
[12] X. Chen, X. Liu, Z. Huang, H. Sun. *RegionKNN: A Scalable Hybrid Collaborative Filtering Algorithm for Personalized Web Service Recommendation.* IEEE International Conference on Web Services. 2009: 9-16.
[13] S. Shahand, SJ. Turner, W. Cai, H. Khademi. DynaSched: a dynamic Web service scheduling and deployment framework for data-intensive Grid workflows. *Procedia Computer Science*. 2010; 1: 593-602.
[14] D. Dyachuk, R.Deters. *Scheduling of composite web services.* On the Move to Meaningful Internet Systems 2006:OTM 2006 Workshops. 2006: 19-20.
[15] Zhi zhong, Liu. *Research on Key Technologies of Web Service Selection Based on QoS*. Hohai University. 2011.
[16] De Jong KA. *The analysis of the behavior of a class of genetic adaptive systems*. Ann Arbor: University of Michigan. 1975.