ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 9, No. 3, 2014, *pp*. 163-168

An Effective QoS-aware Web Service Selection Approach

Tongguang Zhang

Department of Computer and Information Engineering
Xinxiang College, Xinxiang, Henan 453000, China
jsjoscpu@163.com
(*Received November 10, 2012, accepted January 3, 2014*)

Abstract. Service computing provides the capability of binding each service invocation in a composition to a service selected among the same function attributes but different QoS attributes to achieve a QoS goal. Hence, service selection approach plays a very important role in service composition. However existing approaches little considered the computation time. In this study, we propose a effective QoS-aware web service selection approach. This approach adopts a particle swarm optimization algorithm select the most service with users' QoS requirements Experimental results show that our approach can find best suitable services with low time cost in web service selection.

Keywords: Service computing, Web service selection, QoS

1. Introduction

At present, there has been a more than 130% growth in the number of published web services in the period from October 2006 to October 2007 [5-6]. Therefore, service requesters will be faced with a huge number of variations of the same services offered at different QoS. In addition, the QoS requirements dynamic changes can occur at run-time, which means that a quick response to adapt to the requests is important in web service selection [5-6]. Hence, the need of an effective and efficient service selection approach will increase.

The problem of QoS-aware web service selection has received a lot of considerable attention. In [7], the authors proposed two heuristic algorithms based on linear programming to find near-optimal solutions that may be more efficiently than exact solutions suitable for making runtime decisions. The improvement of the two algorithms is significant compared with exact solutions, but both algorithms do not scale when web services increase, which remains out of real time requirements. The authors of [8] adopted the global optimization approach to find the best service components for the composition by linear programming. Soon afterwards the authors of [9] also used linear programming to optimize user's end-to-end QoS constraints, but it differs in the solution of the optimization problem. Furthermore, the work of [10] extends the linear programming model to include local constraints. These linear programming approaches are effective when the size of the problem is not very large. However, their scalability is very poor due to the exponential time complexity of the applied search algorithms with the increasing size of the problem [11].

Although efforts above have been made and obtained in web service composition area, existing technologies on web service selection little considered the computation time or time complexity. They are still not mature yet and require significant efforts. In this paper we propose an approach to select web services for composition with particle swarm optimization. We evaluate proposed approach experimentally on real QoS data. Experimental results show that our approach significantly improves the time performance of web service selection process in service composition system.

The remainder of this paper is organized as follows. Section II introduces web service selection. Our research approach is proposed in Section III. Experiments are presented in Section IV. Finally, Section V is our conclusions.

2. Web Service Selection

In service composition, service candidates have a number of different QoS attributes, leading to large variation in their QoS attribute values or scopes. Considering global QoS constraints, computing or evaluating QoS is different for each service candidate. So QoS utility function was proposed in [7]. The

function maps the quality vector Qs into a single real value, enabling sorting and ranking of service candidates and simplifying choosing to satisfy QoS constraints of the service components. The QoS utility function in this paper is similar to [8]. For example, in the sequential composition model, the overall utility of a composite service S is computed as

$$U(s)=\sum_{k=1}^{r}\frac{Q_{j,k}^{\max}-q_k(s)}{Q_{j,k}^{\max}-Q_{j,k}^{\min}}.w_k \tag{1}$$

$$U(\mathrm{S})=\sum_{k=1}^{r}\frac{Q_{k}^{\max}-q_k(\mathrm{S})}{Q_{k}^{\max}-Q_{k}^{\min}}.w_k \tag{2}$$

with

$$Q_{j,k}^{\max}=\max_{\forall s_{ji}\in S_j} q_k(s_{ji}), Q_k^{\max}=\sum_{j=1}^{n}Q_{j,k}^{\max}$$

$$Q_{j,k}^{\min}=\min_{\forall s_{ji}\in S_j} q_k(s_{ji}), Q_k^{\min}=\sum_{j=1}^{n}Q_{j,k}^{\min}$$

where $w_k\in R^+(\sum_{k=1}^{r}w_k=1)$ represents users preferences, $Q_{j,k}^{\min}$ is the minimum value of the k-th attribute in all service candidates of the service class S_j and similarly, $Q_{j,k}^{\max}$ is the maximum value, $Q_k^{\min}$ is the minimum value of the k-th attribute of S and similarly, $Q_k^{\max}$ is the maximum value. Due to space limited, some concepts about service selection are omitted here. More information is in [7].

As it is well known that the service selection with global QoS constraints is an optimization process. The optimal selection for a given service composition S must meet the following two conditions:

1) An given vector of global QoS constraints $\mathrm{CS}=\{C_1,\cdots,C_m\},0\le m\le r$, $q(\mathrm{S})\le C,\forall C_k\in\mathrm{CS}$ ($q(\mathrm{S})$ is the aggregated QoS value of the composition service).

2) The maximum overall utility value $U(\mathrm{S})$ in the composition service.

However, finding the optimal composition requires enumerating all possible combinations of service candidates, which can be very expensive in terms of computation time. Therefore, we propose an approach to solve this problem.

3. PROPOSED SERVICE SELECTION APPROACH

Particle swarm optimization (PSO) as a parallel optimization algorithm can be used to solve a large number of complex and non-linear problems, and has been widely used to science and engineering for example function optimization, pattern classification and resource allocation fields [8]. PSO is most frequently applied to solve continuous optimization problems. At present, how to apply PSO to discrete optimization problems, especially combinatorial optimization problems, is an important research direction. In recent years, many researchers have proposed improved PSO for example in [12] for combinatorial optimization problems and obtained many good optimization solutions. Therefore, we also use PSO to find optimal quality control lines (local constraints).

For a canonical PSO, the velocity of each particle is modified iteratively by its best personal position, and the position of best particle from the entire swarm. As a result, each particle searches around a region defined by its best personal position and the best position of the population. For instance, a swarm consists of Np particles moving around in a D-dimensional search space. Every particle in the swarm has a position $x_{id}(i\in Np)$, a velocity v_{id} and a memory p_{best}, i.e., its best personal position (local optimization) for its best found position, which are all updated in every iteration step of PSO. The best solution p_{gbest} (the best position it has found so far, i.e., global optimization) depends on the iteration number (i.e. $it\max$). At the beginning, Np particles are initialized with a random position. The fitness of all initial position is evaluated by the fitness function, leading to an initial p_{gbest}. During every iteration step of PSO, The position of the i-th particle at the next iteration will be calculated according to the following equations[13]:

$$\begin{aligned}v_{id}^{t+1}=&w\times v_{id}^{t}+c_1\times rand()\times(p_{best}(t)-x_{id}^{t})\\&+c_2\times rand()\times(p_{gbest}(t)-x_{id}^{t})\end{aligned} \tag{3}$$

$$x_{id}^{t+1}=x_{id}^{t}+v_{id}^{t+1} \tag{4}$$

where c_1 and c_2 are two positive constants, called cognitive learning rate and social learning rate respectively, $rand()$ is a random function in the range [0,1], and w is an inertia factor which linearly decreases from 0.9 to 0.4 through the search process. In addition, the velocities of the particles are confined within $[V_{\min}, V_{\max}]^D$. If an element of velocities exceeds the threshold $V_{\min}$ or $V_{\max}$, it is set equal to the corresponding threshold.

In this paper, we use PSO to find the best service candidate for each service class (i.e. the optimal particle). First, a suitable coding scheme should be designed for searching the optimal quality control line. We use integer array coding scheme on the basis of the characters of the best service. The number of items in the array denotes the number of service classes, and each element of the array denotes the index of service candidates. The maximum of the element is the number of service candidates, and the minimum is 1. The velocities of the particles are also encoded as n-dimensional arrays, and each element value of the velocity is an integer that is within $[V_{\min}, V_{\max}]$.

In PSO, Every particle uses its own information, local best information, and global optimum to generate a new particle for better fitness, through parallel global search. When the algorithm stops, it obtains the best scheme.

4. EXPERIMENTS

In order to evaluate our proposed approach, we present an experimental evaluation of our approach, measuring computation time, in terms of the execution time required to find a solution.

A. Experiments

In this study, we conduct our experiments using QWS real datasets. It comprises measurements of 9 QoS attributes as shown in Table I for 2500 real-world web services. These services were collected from public sources on the Web, including UDDI registries, search engines and service portals, and their QoS values were measured using commercial benchmark tools. More details about this dataset can be found in [14]. QoS attributes in the QWS dataset

The parameters are set as follows. $w = 0.7$, $c_1 = 2.0$, $c_2 = 2.0$, $it\max = 30$, $Np = 15$. All the experiments are taken on the same software and hardware, which were Pentium 2.8GHz processor, 2.0GB of RAM, Windows XP, and Java 1.4. All approaches run for 10 times and all results are collected on average.

B. Experimental Results

In this section, we evaluate the computation time of proposed approach. In Fig. 1, we evaluate the computation time of our approach with respect to the number of service candidates. In this experiment, the number of service classes is 10. In Fig. 2, we evaluate the computation time with respect to the number of service classes. In this experiment, the number of service candidates is 50.

From the Fig. 1, regardless of the number of service candidates, the computation time of our proposed approach is obviously short. In Fig. 1, when the number of service candidates is 50, the computation time is only 300.2 millisecond (msec). It is far shorter than 1 second. When the number of service candidates is 500, the computation time is 430.1 msec. The time cost is very short and also shorter than 1 second. Hence, according to its average computation time (351.1 msec), our proposed approach is very suitable to service selection for users with real-time requirement.

From the Fig. 2, regardless of the number of service classes, the computation time of our proposed approach is obviously short. In Fig. 2, when the number of service classes is 5, the computation time is only 296.3 millisecond (msec). It is far shorter than 0.5 second. When the number of service candidates is 50, the computation time is 549.7 msec. The time cost is very short and also shorter than 1 second. Hence, according to its average computation time (415.6 msec), our proposed approach is very suitable to service selection for users with real-time requirement.

Hence, according to the two experimental results of Figs. 1 and 2, our proposed approach is very effective for service selection.

TABLE I
QWS DATASET

QoS Attribute	QoS Description	Units
Response Time	Time taken to send a request and receive a response	millisecond
Availability	Number of successful invocations/total invocations	percent
Throughput	Total number of invocations for a given period of time	Invocations/second
Likelihood of success	Number of response/number of request messages	percent
Reliability	Ratio of the number of error messages to total messages	percent
Compliance	To which extent a WSDL document follows the WSDL spec.	percent
Best Practices	To which extent a web service follows the Web Services Interoperability (WS-I) Basic Profile	percent
Latency	Time the server takes to process a given request	millisecond
Documentation	Measure of documentation (i.e. description tags) in WSDL	percent

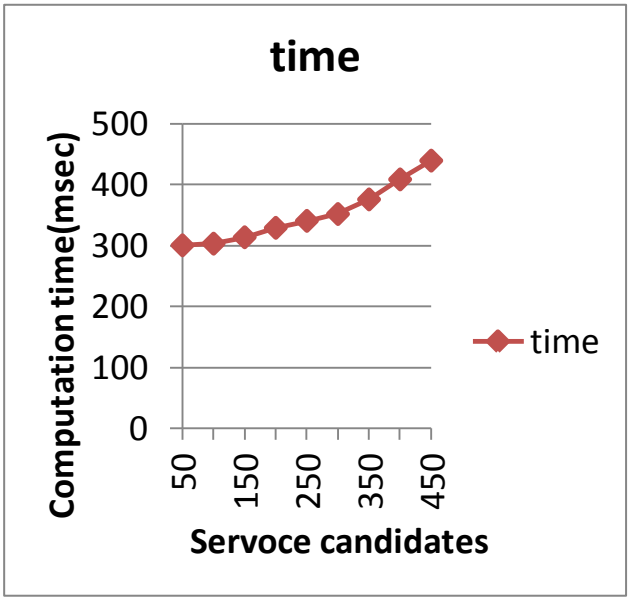

Fig. 1 Computation time with the number of service candidates.

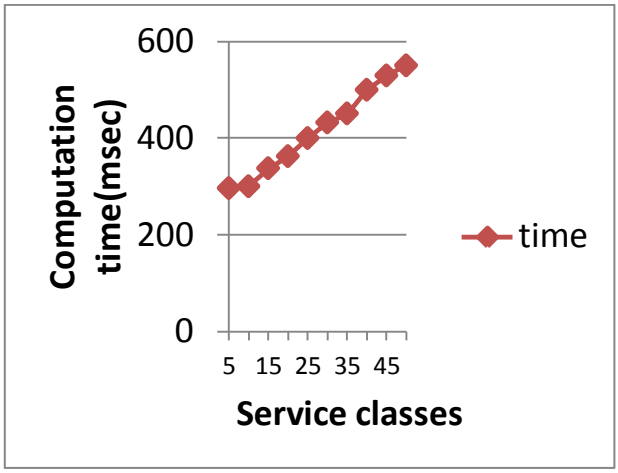

Fig. 2 Computation time with the number of service classes.

5. CONCLUSIONS

In the open and dynamic web service environment, the user's demand, supporting rapid web service composition, will radically increase. While the current studies often cause high time complexity and are out of the real-time requirements, in the paper, we propose an approach with particle swarm optimization. The comparison results with a existing scheme show that our proposed approach is efficient and can satisfy the real-time requirement for the fast selection of web services.

However, there are some limitations in this scheme. For instance, there are no clear-cut guidelines on what would be the best choice to some parameters of particle swarm optimization such as w and β. Therefore, in order to promote and deepen continuing researches in the future, it is worthwhile to investigate more studies to uncover the valuable new study issues.

6. References

[1] H.-C. Wang, C.-S. Lee, and T.-H. Ho, "Combining subjective and objective QoS factors for personalized web service selection," Expert Systems with Applications, vol. 32, pp. 571-584, 2007.

[2] S. G. Wang, Q. B. Sun, and F. C. Yang, "Towards Web Service selection based on QoS estimation," International Journal of Web and Grid Services, 4, vol. 6, pp. 424-443, 2010.

[3] P. Wang, K.-M. Chao, and C.-C. Lo, "On optimal decision for QoS-aware composite service selection," Expert Systems with Applications, vol. 37, pp. 440-449, 2009.

[4] S.-N. Chuang and A. T. S. Chan, "Dynamic QoS adaptation for mobile middleware," IEEE Transactions on Software Engineering, vol. 34, pp. 738-752, 2008.

[5] E. Al-Masri and Q. H. Mahmoud, "Investigating Web Services on the World Wide Web," in Proceedings of the 17th International Conference on World Wide Web (WWW2008), 2008, pp. 795-804.

[6] M. Alrifai, D. Skoutas, and T. Risse, "Selecting skyline services for QoS-based web service composition," in Proceedings of the 19th international conference on World Wide Web (WWW 2010), 2010, pp. 11-20.

[7] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," in Proceedings of the 18th international conference on World Wide Web (WWW2009), 2009, pp. 881-890.

[8] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," ACM Transactions on the Web, vol. 1, pp. 1-26, 2007.

[9] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality driven web services composition," in Proceedings of the 12th international conference on World Wide Web (WWW 2003), 2003, pp. 411-421.

[10] V. Cardellini, E. Casalicchio, V. Grassi, and F. Lo Presti, "Flow-based service selection for web service composition supporting multiple QoS classes," in Proceedings of IEEE International Conference on Web Services (ICWS 2007), 2007, pp. 743-750.

[11] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," IEEE Transactions on Software Engineering, vol. 33, pp. 369-384, 2007.

[12] K. RA and C. LD, "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems " IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS PART B-CYBERNETICS, vol. 36, pp. 1407-1416, 2006.

[13] P. Bajpai and S. N. Singh, "Fuzzy Adaptive Particle Swarm Optimization for Bidding Strategy in Uniform Price Spot Market," IEEE Transactions on Power Systems, vol. 22, pp. 2152-2160, 2007.

[14] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," IEEE Transactions on Evolutionary Computation, vol. 6, pp. 58-73, 2002.