Binary PSO for Web Service Location-Allocation

Hai Huang, Hui Ma, Mengjie Zhang

Victoria University of Wellington, New Zealand
simonhuang211@gmail.com | hui.ma@ecs.vuw.ac.nz |
mengjie.zhang@ecs.vuw.ac.nz

Abstract. Web services are independent programmable application components which scatter over the Internet. Due to the improvement on local computing power and development on high-speed Internet, network latency has a significant impact on determining the service response time. Thus, physical locations of web services and users should be taken into account for web service composition. In this paper, we propose a new solution based on the binary PSO-based approach to allocate the service locations. Although several heuristic approaches have been proposed for web service location-allocation, to our best knowledge, PSO-based approach is the first attempt to solve the problem. A simulated experiment is done using WS-DREAM data set with five different complexities. To compare with two GA-based approaches, the proposed BPSO approach has advantages in most situations.

1 Introduction

Since the sweeping wave of Web 2.0 over the world in recent years, much more attention has been put on Web services. Web services are considered as self-contained, self-describing, modular applications that can be published, located, and invoked across the Web [22]. Moreover, an increasing amount of enterprises and organizations implementing their core business and outsource other services over Internet. Thus, the ability to efficiently and effectively select and integrate inter-organizational and heterogeneous services on the Web at runtime is an important step towards the development of the Web service applications [22]. As a result, a single service is typically not sufficient to respond to the users' requests and services should be combined through service composition to achieve a specific goal. Thus it triggers single web services to integrate with each other in today's world. This naturally drives a considerable number of research efforts on the Web.

The criteria of evaluating service performance on the distributed system environment have been considered. It is acknowledged that Quality of Service (QoS) has been one of the important principles for many years [7, 28]. QoS, in brief, usually refers to the overall performance of a computer network served for its users. With the purpose of quantifying the term of QoS, several relevant aspects of the network service are measured, for example, response time, availability, cost and so forth [14]. But it invisibly increases the difficulty of measurement

if combining all these parameters. Service response time, one significant factor which consists of local processing time and network response time, therefore is often chosen [13]. Obviously, quicker replies mean the higher possibilities of users' satisfaction.

In spite of all these efforts, the task of Web service composition is complicated. First of all, although the service-oriented architecture gives us a new horizon and creates new opportunities to assemble distributed web services [21], it is still unclear that how to create robust service composition. Yet some other doubt remains. The paradox can mainly explained in terms of the balance between the service composition and service response time for client's request. The complexity of this problem rockets because of the following reasons. First, the number of services available over the Web increases dramatically during the recent years. Second, clients with increasing scale expect to use more categories and amounts of services with higher frequencies.

In addition, based on service composition, one extended problem about how to design the service location for the best service performance cannot be neglected either. This indispensable factor whereas draw little attraction in the majority of existing studies. The importance of service location, argued by many researches [20, 23], must be taken into account, because with the improvement of local computing and high-speed Internet, the only way to reduce network latency is to move the server closer to the client computer. From this standpoint, the issue centers on how to seek an optimal solution for web service deployment on the premise that the server response delay in the range of that can be tolerated by users. In other words, how to choose an appropriate algorithm for solving the optimizing problem is of importance. Some conventional optimization techniques are proposed in [19, 24, 25]. Also, a number of effective strategies based on Genetic Algorithm (GA) are explicitly elaborated in [2, 26]. GA is an iterative procedure that borrows the idea of Darwin in his research of biological evolution. Its main purpose is to eliminate those unsuitable solutions, so the solutions which are fittest are survived after a set number of generations.

Another powerful algorithm Particle Swarm optimization (PSO), however, does not achieve so much focus particularly in the topic of service location problem. PSO, which was introduced by Kennedy and Eberhart in 1995 [5, 15], is one significant branch of swarm intelligence paradigms. In recent decades, the development of PSO is considerable and contributes many researches in solving real-world optimization problems [17, 8, 12]. The idea of PSO originates from the swarm behavior of birds flocking and fish schooling to guide the particles to search for globally optimal solutions [4]. The basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions which are named particles. Particles move through the search space using a combination of an attraction to the best solution that they individually have found, and an attraction to the best solution that any particle in their neighborhood has found [1]. In PSO, a neighborhood is defined for each individual particle as the subset of particles which it is able to communicate with [1].

In this paper, a binary PSO-based approach for solving the web service location allocation will be proposed. More precisely, three main objectives shown in below will be investigated.

- Whether the binary PSO can be used to solve the service location-allocation problem.
- Whether the proposed binary PSO-based approach are effective and efficiency in solving the service location-allocation problem, comparing with other existing approaches.

The remainder of this paper is organised as follows. Section 2 is a review of recently research on web service location-allocation and a binary PSO. Section 3 is a description of web service location-allocation and our proposed problem model. Section 4 presents binary PSO-based approach to the web service location-allocation. Section 5 and Section 6 present experimental evaluation results of our proposed binary PSO-based approach to service location-allocation problem. Finally, we draw our conclusions and future prospects in Section 7.

2 Related Work And Background

2.1 Related Work

The majority of recent studies on web service quality can basically be sorted into two categories, service selection [19, 21] and service composition [2, 20, 23–25]. In general, service selection is to seek the better service instances and service composition is to build better service work flow. The objective of these researches are similar, to improve the service quality in several dimensions, e.g. response time, service execution cost, service availability and service reliability. However, these studies are starting from the perspective of service consumers and ignoring service providers have more privileges to improve the service qualities.

Comparing with service selection and service composition, web service location-related research is a newborn in this domain. In study [20], Liu et al have proved that location and time have big impact on service quality, a location-related service composition framework has been proposed. However, this study does not contain any comparison with other existing approaches. In study [23], authors proposed using integer programming techniques to solve the web service location-allocation problem. However, the results show integer programming can not obtain satisfactory performance in large scale data set.

Although web service location-allocation still in its infant stage. Traditional location-allocation problem has been extensively investigated. Using PSO-based approaches to solve the traditional location-allocation problem also appear in recently years. Studies [9, 10] have propose using discrete particle swarm to solve the location-allocation problem, the results show PSO-based approaches can achieve a better performance than many meta-heuristic approaches, such as genetic algorithm (GA) and simulated annealing (SA). However, the nature of web service location-allocation problem is quite different. Traditional location-allocation problem usually is based on one kind of facility, such as fire station and

hospital. Hence, the location optimization just consider the distance from service consumer to the nearest facility. However for web service location-allocation we need to consider the distance from user to a set of services. In [11], an genetic algorithm based approach was used as heuristic method to optimized the service allocation matrix. In this paper, binary PSO was used as heuristic method to optimize the proposed problem.

2.2 Review of Binary PSO

The originated purpose of PSO is solving the continuous problem. It was also proven PSO had advancement than other meta-heuristic approaches on solving the continuous problem[6, 8, 12, 17]. To solve the binary problem, Kenndy and Eberhart [16] introduce binary PSO (BPSO). Service location-allocation problem is a binary PSO problem. Therefore we present a brief review of PSO and BPSO in this section. Suppose that our search space is d-dimensional, a particle i is a potential solution in this d-dimensional space. A d-dimensional vector is used to represent the particle position, say $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$. A swarm is a set of potential solutions in current optimization problem search space. The velocity vector represents the next movement and direction of a particle, which is defined as $V_i = (v_{i1}, v_{i2}, \ldots, v_{id})$. $P_{i,best}$ and $P_{g,best}$ respectively represent the personal best and global best position. Hence, they are also represented as a d-dimensional position vector. The position and velocity of each particle is updated by iteration according to $P_{i,best}$ and $P_{g,best}$. Due to the evolutionary nature, the position and velocity of each particle is updated by iteration according to $P_{i,best}$ and $P_{g,best}$. The position and velocity of particles are updated by the following formulae:

$$V_i^{t+1} = w \cdot V_i^t + c_1\varphi_1(P_{i,best} - X_i) + c_2\varphi_2(P_{g,best} - X_i) \tag{1}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \tag{2}$$

where c_1 and c_2 are positive constants, whereas φ_1 and φ_2 are two random variables with range between 0 and 1. w is the inertia weight which represents the impact of current velocity on the new velocity. The feature that drives PSO is social interaction. The behaviour of particles within the swarm is affected by each other. A limitation V_{max} is applied to new velocity updates as constraints. This limitation prevents the particles moving out from the problem search space. The initialization of V_{max} usually based on the problem search range. For example, an optimization problem $f(x)$ is searching the maximum value that x within the range $(-5, 5)$. Then, the V_{max} can configure as $(-5, 5)$ or $(-0.25, 0.25)$.

In binary PSO, the main change is that the velocity does not represents the next movement and direction for particles. Velocity is a probability that effects a bit (position) of particle to takes on 1 or 0. In the binary PSO, for updating velocities will be the same as Eq. 1. For updating particle the vector position is restricted to only 1 and 0. The updating equation in Eq. 2 is reformulate in

Eq. 3.

$$X_i^{t+1} = \begin{cases} 0 & \text{if } Rand \geq S(V_i^{t+1}) \\ 1 & \text{if } Rand < S(V_i^{t+1}) \end{cases} \tag{3}$$

where S is the sigmoid function that transform the particle velocity to the probability as the following Eq.4. *Rand* is the random number which range fro 0 to 1.

$$S(V_i^{t+1}) = \frac{1}{1 + e^{-V_i^{t+1}}} \tag{4}$$

3 Problem Modelling

In this section we first describe the service location-allocation problem in details then we will present models for the service location-allocation problem, which makes use of a set of matrix and the objective function of the problem.

3.1 Problem Description

Web service location-allocation problem is to find reasonable physical locations for web services. On one hand, optimizing services allocation can improve the users' experience when delivering the service bundles, such as improving the service performance of response time and throughput. On the other hand, reasonable services layout can decrease the established cost of services deployment. Hence, a reasonable services distribution layout should be found with different quality measurements standing in a satisfactory range.

3.2 Problem Objectives

In this paper, we consider service location-allocation as a problem with two objectives. The first objective is to reduce aggregate response time when a group of users invoke the services that are distributed at different locations. The second objective is to reduce the established cost for web service providers. The objective function of service location-allocation problem shown as below:

$$TimeObjective = MIN(\sum_{i=1}^{m}\sum_{s=1}^{l} r_{is}) \tag{5}$$

Where r_{is} is the response time measurement between specific user i and service s. Time objective is to minimise the aggregate time which sum up the response time from users with different locations to each service deployed by service providers.

$$CostObjective = MIN(\sum_{j=1}^{n}\sum_{s=1}^{l} c_{js}) \tag{6}$$

Where c_{js} is to minimise the established cost measurement between specific server j and service s. Cost objective is the aggregate cost which sum up the established cost from servers to services.

Combined with response time objective and established cost objective, a composite objective function is proposed in Eq. 7. Parameters w_1 and w_2 are the constant weights for the time objective and cost objective, respectively.

$$ObjectiveFunction = MIN(w_1 \sum_{i=1}^{m} \sum_{s=1}^{l} r_{is} + w_2 \sum_{j=1}^{n} \sum_{s=1}^{l} c_{js}) \tag{7}$$

3.3 Model

In [11] we established a model in terms of transferring a service allocation matrix to a user response time matrix. However, the established cost of service providers will be treated as a main consideration in this paper. In this section, a short review of problem model will be presented. Fig. 1 is a flow graph which shows the steps of how solve the problem in this paper. As we can see, input is a given response time matrix, a given estimated cost matrix and a potential service allocation matrix. Response time matrix and estimated cost matrix contain the corresponding data depending on different locations. Service allocation matrix is a binary matrix which only contains 0 and 1, the elements in the matrix represent whether service was deployed in a location. Based on service allocation matrix and two given data matrix, we can calculate the actual user response times and established cost. Then, a heuristic method can be used for optimizing the service allocation which ensure the user response time and cost standing in rational interval.

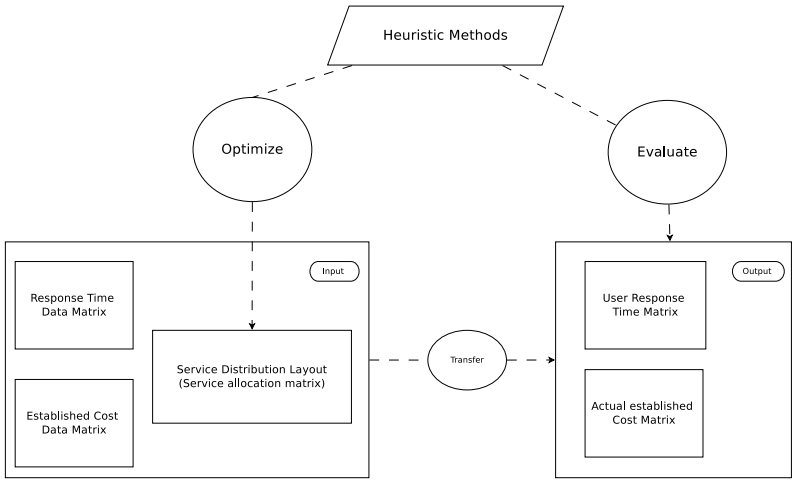

Fig. 1. Service Demand Calculation Example

4 BPSO for Web Service location-allocation

4.1 Encoding Scheme

In BPSO, a particle is a potential solution which represented by a fix-length binary vector. However, a potential service allocation solution are often represented by a binary matrix. In order to use BPSO we need to transfer service location-allocation design represented in a binary matrix into PSO particle representation. As seen in Fig. 2, a matrix used for represent service allocation and relevant calculation is transferred into a vector which can then be used for the BPSO evolutionary progress. In a binary service location allocation matrix, an element S_{ij} denote whether service i is allocated at location j. For example, the value of $S11$ means whether service one has a instance in location one.

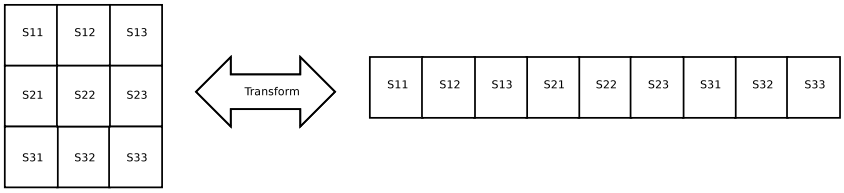

Fig. 2. Encoding Scheme Example

4.2 Algorithm

As shown in Algorithm 1, the particles are generated in random position with corresponding velocity randomly. Then, the swarm will get into evolutionary process. First, calculating the fitness value for each particle and replace the personal best if current fitness value is better. After this step, searching the global best individual in the whole population. Last, updating the velocity and particle position according to Eq. 1, Eq. 3 and Eq. 4. Repeat the evolutionary process until termination condition is met.

4.3 Parameters

As we discussed on Section 3, choosing parameters are difficulties for BPSO because their underlying effects are different than the original one. In BPSO,the parameters includes c_1, c_2 and w. Both c_1 and c_2 are acceleration coefficients, which determine the influences of personal best and global best solution on particle's current velocity. While w is the inertia weight which control the influence of previous velocity [18]. Also, inertia weight w can smooth the convergence for the particle trajectories [6]. As we can see in velocity update Eq. 1, term $c_1\varphi_1(P_{i,best} - X_i)$ is represented the influence of the personal best position vector. Term $c_2\varphi_2(P_{g,best} - X_i)$ is represented the influence of global best position

Algorithm 1 Binary Particle Swarm Optimization for Web Service Location-allocation

```
Initialize particles(swarm)P
Initialize velocity V_i^0 For each particle X_i
while Termination Condition is not met do
  for Each particle X_i do
    Calculate the fitness value
    if New fitness value is better than personal best P_{i,best} then
      Set new fitness value as P_{i,best}
    end if
  end for
  Select the best particle as global best P_{g,best}
  for Each particle X_i do
    Update V_i^t according to Eq. 1
    Update X_i according to Eq. 3 and Eq. 4
  end for
end while
```

to current particle position. If term $c_1\varphi_1(P_{i,best} - X_i)$ has a relatively large value, the flight of the particle will be driven to the personal best position, otherwise particle will be driven more to the global best position. Existing studies proposed many method to optimized these parameters. To ensure a good convergence, in this paper we employ a simple and effective way proposed in [3] to decide the parameters c_1, c_2 and w. The formula below shows the dependencies of c_1, c_2 and w.

$$\begin{cases} w = \frac{1}{\alpha - 1 + \sqrt{\alpha^2 - 2\alpha}} \\ c_1 = c_2 = \alpha w \end{cases} \tag{8}$$

where α is a real number which should be greater than 2, and c_1 and c_2 should be greater than 1.

5 Design of Experiments

In this section, experiments are designed for comparing the performance of BPSO and two versions of GA to web service location-allocation problem. Two measurements are taken into account, execution time and fitness value. Execution time is the searching time which excluded the file IO operations. Objective values are the calculation results based on the objective function Eq. 7.

5.1 Dataset

The datasets used for the experiments were generated from the WS-DREAM dataset [27]. The WS-DREAM dataset is a collection of historical data from 339 users and 5824 web services located in different locations. It records several non-functional attributes about the web services, including response time,

throughput, availability, etc. However, it does not contains the information about service established cost. Hence, service established cost were random generated according to server locations. Furthermore, all the input data involved in this experiments were normalized into interval $[0, 1]$. The response time is regarding as decrease measurement and established cost is increase measurement, which means value 1 represents fastest (smallest) response time and highest established cost, respectively. Five problems of different complexity levels of data were extracted from WS-DREAM. Table 1 outlines the extracted data and it corresponding attributes.

Table 1. Hypothetical Web Service Location-allocation Problems

Problem ID	User Location	Potential Server Location	Composite Services	Atomic Services
1	20	100	1	5
2	40	250	10	30
3	80	500	20	100
4	160	1000	40	200
5	200	2000	80	400

5.2 Environment and Parameters

The experiments were conducted on a personal laptop with 2.3GHz CPU and 4.0 GB RAM. For each approaches, 30 independent are performed for each problems with constant population size 100. The maximum number of iteration is 250, but it will termination earlier if some condition is met, such as objective value is not changed during 10 iterations. The weight of response time and established cost were set to 0.5 and 0.5, which means the importance of response time and established cost were equvenlent in this model. It is also can be adjusted when service providers have different preferences. For the GA-based approaches, the initial crossover and mutation possibility are 0.6 and 0.2, which is the configuration in our previous study [11]. For BPSO, parameter α and w were set to 2.07 and 0.689 according to study [3]. And the maximum velocity V_{max} was set to 4 in this experiments.

6 Results and Analysis

This section shows the experimental results of BPSO and two version of GA approaches to solving the service location-allocation problems with our predefined complexity levels. Table 2 demonstrates the comparison of BPSO and two version of GA based approaches in solving the problems designed for the test case. Each row represents the comparison among three method on a predefined problem. The first column represents the problem ID. The second column indicates the objective value comparison between three proposed approaches. The third column is the execution time. The forth column represents the average

Table 2. Experimental Results for the MFGA and GA

Problem ID	Fitness Value			Execution Time(s)			Average Generation			Find BKS times(s)		
	GA	MFGA	BPSO	GA	MFGA	BPSO	GA	MFGA	BPSO	GA	MFGA	BPSO
1	6.9	6.9	6.9	3.28	3.34	3.10	36	36	42	26	27	27
2	43.4	46.0	44.9	6.82	6.01	5.99	72	62	68	19	21	22
3	140.1	144.4	145.7	12.83	10.77	10.20	128	109	116	12	16	18
4	210.45	220.61	230.84	20.33	15.89	16.34	189	168	182	8	12	13
5	594.72	623.84	621.69	28.94	24.33	23.43	250	214	238	4	8	10

[a] BKS means Best Known Solution

convergence generation of the two approaches. The last column represents the best known value found by the algorithm.

As illustrated in Table 2, standard GA has the worst performance in solving all problems of different complexities. However, MFGA and BPSO have similar performance for the different problems. For problem 1, three approaches have equivalent performance due to its simplicity. For problem 2 to 5, MFGA and BPSO perform better than standard GA with respect to both fitness value and execution time. For problem 5, standard GA can not convergence during the maximum 250 iterations. For complex problems, problem 4 and 5, BPSO and MFGA performs similar, taking similar execution time to generate solutions of similar fitness values.

7 Conclusion and Future Work

In this paper we propose a BPSO-based approach to web service location-allocation problem. We have shown that BPSO can be used to solve service location-allocation problem. It is shown that our proposed BPSO based approach can be used to solve the service location allocation problem in a relative efficient and effective way, similar to our previous proposed MFGA based approach.

Future work probably includes more constraints into the problem model (i.e. throughput, availability and reliability). We will also investigate the sittings of the parameters used in the BPSO approaches.

References

1. D. Bratton and J. Kennedy. Defining a standard for particle swarm optimization. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 120–127. IEEE, 2007.
2. G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. A lightweight approach for qos-aware service composition. In *Proceedings of 2nd international conference on service oriented computing (ICSOC04)*, 2004.
3. M. Clerc and J. Kennedy. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on*, 6(1):58–73, 2002.
4. F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana. The next step in web services. *Communications of the ACM*, 46(10):29–34, 2003.

5. R. C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science*, volume 1, pages 39–43. New York, NY, 1995.
6. R. C. Eberhart and Y. Shi. Particle swarm optimization: developments, applications and resources. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 81–86. IEEE, 2001.
7. I. Foster and C. Kesselman. *The Grid 2: Blueprint for a new computing infrastructure*. Elsevier, 2003.
8. N. Franken and A. P. Engelbrecht. Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma. *Evolutionary Computation, IEEE Transactions on*, 9(6):562–579, 2005.
9. A. Ghaderi, M. Jabalameli, F. Barzinpour, and R. Rahmaniani. An efficient hybrid particle swarm optimization algorithm for solving the uncapacitated continuous location-allocation problem. *Networks and Spatial Economics*, 12(3):421–439, 2012.
10. A. R. Guner and M. Sevkli. A discrete particle swarm optimization algorithm for uncapacitated facility location problem. *Journal of Artificial Evolution and Applications*, 2008:10, 2008.
11. H. Hai, M. Hui, and Z. Mengjie. An enhanced genetic algorithm based approach for web service location-allocation. In *DEXA 2014 : 25th International Conference on Database and Expert*, page inprocess. Dexa, 2014.
12. S.-Y. Ho, H.-S. Lin, W.-H. Liauh, and S.-J. Ho. Opso: Orthogonal particle swarm optimization and its application to task assignment problems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(2):288–298, 2008.
13. J. M. Johansson. On the impact of network latency on distributed systems design. *Information Technology and Management*, 1(3):183–194, 2000.
14. A. Kaur. An overview of quality of service computer network. *Indian Journal of Computer Science and Engineering (IJCSE)*, 2011.
15. J. Kennedy, R. Eberhart, et al. Particle swarm optimization. In *Proceedings of IEEE international conference on neural networks*, volume 4, pages 1942–1948. Perth, Australia, 1995.
16. J. Kennedy and R. C. Eberhart. A discrete binary version of the particle swarm algorithm. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, volume 5, pages 4104–4108. IEEE, 1997.
17. R. A. Krohling and L. dos Santos Coelho. Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 36(6):1407–1416, 2006.
18. S. Lee, S. Soak, S. Oh, W. Pedrycz, and M. Jeon. Modified binary particle swarm optimization. *Progress in Natural Science*, 18(9):1161–1166, 2008.
19. Y. Liu, A. H. Ngu, and L. Z. Zeng. Qos computation and policing in dynamic web service selection. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 66–73. ACM, 2004.
20. Z. Liu and T. Lu. A location & time related web service distributed selection approach for composition. In *Grid and Cooperative Computing (GCC), 2010 9th International Conference on*, pages 296–301. IEEE, 2010.
21. Y. Ma and C. Zhang. Quick convergence of genetic algorithm for qos-driven web service selection. *Computer Networks*, 52(5):1093–1104, 2008.

22. J. Rao and X. Su. A survey of automated web service composition methods. In *Semantic Web Services and Web Process Composition*, pages 43–54. Springer, 2005.
23. Y. Sun and G. J. Koehler. A location model for a web service intermediary. *Decision support systems*, 42(1):221–236, 2006.
24. 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*, pages 411–421. ACM, 2003.
25. L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. *Software Engineering, IEEE Transactions on*, 30(5):311–327, 2004.
26. L.-J. Zhang, B. Li, T. Chao, and H. Chang. On demand web services-based business process composition. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, volume 4, pages 4057–4064. IEEE, 2003.
27. Y. Zhang, Z. Zheng, and M. R. Lyu. Wsexpress: A qos-aware search engine for web services. In *Proc. IEEE Int'l Conf. Web Services (ICWS'10)*, pages 83–90, 2010.
28. W. Zhao, D. Olshefski, and H. G. Schulzrinne. Internet quality of service: An overview. *IEEE*, 2000.