Web Service Composition Optimization Based on Adaptive Mutant Beetle Swarm

Hongwei Yang1, Fucheng Xue1, Hongmei Zhu2, Leilei Zhu1 and Li Li1*

1School of computer science and technology, Changchun University of Science and Technology, Changchun City, Jilin Province, 130022, China
2School of electronics and information engineering, Changchun University of Science and Technology, Changchun City, Jilin Province, 130022, China
*Corresponding author’s e-mail: ll@cust.edu.cn

Abstract. With the continuous development of cloud computing technology and big data, web services are widely used in intelligent information processing in all walks of life, when facing a large number of services with similar functions and quality, web service composition technology is a crucial technology in web services, which enable common services to be combined and select the optimal service combination to meet the wide range of users' needs. Aiming at the problems of algorithm accuracy and stability of existing service composition technologies, the penalty function is used to construct the fitness function in this paper, which transforms the constrained optimization problem into an unconstrained optimization problem, and realizes the global optimization. Traditional web service composition models are aimed at QoS attribute perception. This paper establishes a QoE model that focuses on user experience, which can better meet the needs of users. An adaptive mutant beetle herd algorithm was proposed in this paper, the author introduced Beetle Antennae Searching Algorithm into the common particle swarm optimization algorithm, and added an adaptive mutation factor to enhance the searching ability of the algorithm; The use of dynamic learning factors allows the algorithm to have better population diversity and enhanced convergence. The experimental results that the algorithm proposed in this paper has higher algorithm accuracy, comparatively faster convergence speed and stability.

1. Introduction

Web services have been deeply applied to implement service-oriented computing (SOC) and service-oriented architecture (SOA) [1]. With the development of Web service technology, a single Web service is often difficult to meet people's growing needs. Web service composition combines multiple Web services into a composite service that meets user needs, making it more powerful. Quality of service (QoS) is used to distinguish the non-functional attributes of Web services with the same function. How to choose a combination of services that meets user needs and has optimal QoS is the current research hotspot. The flow frame of the service composition is shown as follows in Figure 1:
Figure 1. Flow Frame of the Service Composition.

Composing the existing web services in the cloud environment is of great significance for lowering cloud service costs and increasing cloud service quality. The five-dimensional QoS Model proposed by AgFlow[2] is usually adopted in web service composition. QoS is mainly used to indicate the non-functional attributes of web service and distinguish services of the same or similar functions. Ni Wancheng and his partners[3] classified typical web service composition methods and systematically discussed web service composition models. The QoS currently adopted can only reflect the quality of Web services in terms of performance, and fails to consider the user’s experience. However, QoE can accurately reflect the user’s satisfaction with the service. The existing service composition optimization studies all have the problems of too large a scale and poor composition efficiency. Their weak composition organization compatibility leads to its low performance, how to efficiently pick out high-quality service composition from huge composition schemes to meet users’ needs, and value the user’s quality of experience(QoE) at the same time has become a key problem in the service optimization field and it plays a leading role in enhancing the development of cloud computing.

2. Related Work
To solve the problem of web service composition optimization, researchers have proposed different intelligent algorithms. Researchers such as Yang Zhen proposed a cloud service dynamic combination method based on the service trust attribute as an evaluation standard. This method decomposes the cloud service trust attribute into a basic trust set and an empirical trust set, which effectively solves the service in a continuously changing cloud environment. Combination problem [4]. Karimi MB, Isazadeh A [5] aimed at QoS-aware service combination can be optimized in the shortest time, proposed association rules and service clustering and genetic algorithm to achieve service selection to reduce the search space, through QoS parameters such as response time, Cost, availability, and success rate find the global optimal solution. The genetic algorithm (GA) and PSO are two comparatively typical algorithms in the field of service composition optimization[6]. Seghir[7] proposed the hybrid GA (HGA). When HGA was combined with fruit fly optimization, computing time and complexity were decreased. Afshin[8] introduced a highly efficient service composition technique that can be applied to cloud computing. Based on the proxy method, this technique can compose services by identifying QoS parameters; based on the fitness function, it can select the best service by using PSO.

Researchers such as Hayyolalam V [9] focused on service QoS parameters to check QoS parameters with centralized and distributed identification, systematically classify and evaluate QoS-aware cloud service portfolio research methods and strategies. To maximize fuzzy or clear QoS attributes, Xu J, Guo L et al. proposed the Triangular Fuzzy Genetic Algorithm (TGA), through which the optimal
combination was selected to meet user needs [10]. The QoS mostly used in the above literature can only reflect the quality of web services in terms of performance, and QoE can accurately reflect the user's satisfaction with the service. In response to the above problems, this article uses a model[11] based on user experience. Under the premise of guaranteeing QoS constraints, according to the QoE requested by the user, this model selects the appropriate resources and performs calculations to minimize the time spent on the service, thereby satisfying the user's requirements, and can essentially be reduced to a multi-objective optimization problem. First, user QoE is mapped to the QoS parameter index, and for each service request, the optimal service combination that satisfies its QoS is selected. According to this model, we propose an adaptive mutation beetle swarm optimization (MBSO) algorithm for cloud service composition optimization in this study. By expanding the beetle antennae search (BAS) algorithm to population, the PSO's problem of easily falling into the local best value is solved. By using the trigonometric controlling learning factor, global convergence capability and rate are increased. By using the dynamic learning factor, the stability of the algorithm was improved. By designing the mutation operator, the particle search range is expanded and the particle's ability to jump out of local optimization is enhanced, avoiding premature and falling into local optimum, the accuracy of the algorithm is improved. This paper proves the feasibility of the algorithm through experiments, and has good accuracy and algorithm stability for cloud service composition problems of higher dimensions.

3. Web service composition model based on QoE
To meet the requirements put forward by users, this paper abstracts the requirements into n sub-tasks with different functions $T=(t_1,t_2,...,t_n)$. In different subtasks, you can select the service $S_{ij}$ in the relevant candidate service set $S_i=(S_{i1},S_{i2},...,S_{in})$ to complete the subtask. The $S_{ij}$ service contains k different QoS attributes $Q_{ij}=(q_{ij1},q_{ij2},...,q_{ijk})$, and finally generates a combined service $CS=(s_{ij1},s_{ij2},...,s_{ijm})$. This article defines $Q_{ij}=Q(t)=(T,S,R,A)$. In this equation, $T$, $S$, $R$, and $A$ represent the execution time, success rate, reliability, and availability of a service, respectively. The QoS calculation formula of this paper is shown in Table 1.

| QoS Indicators | Sequence Structure | Parallel Structure | Selective Structure | Loop Structure |
|----------------|--------------------|--------------------|--------------------|----------------|
| Execution Time | $\sum_{i=1}^{n} T_i$ | $\sum_{i=1}^{n} T_i$ | $\sum_{i=1}^{n} (P_i T_i)$ | $c \sum_{i=1}^{n} T_i$ |
| Success Rate   | $\prod_{i=1}^{n} S_i$ | $\prod_{i=1}^{n} S_i$ | $\prod_{i=1}^{n} (P_i S_i)$ | $(\prod_{i=1}^{n} S_i)^k$ |
| Reliability    | $\prod_{i=1}^{n} R_i$ | $\prod_{i=1}^{n} R_i$ | $\prod_{i=1}^{n} (P_i R_i)$ | $(\prod_{i=1}^{n} R_i)^k$ |
| Availability   | $\prod_{i=1}^{n} A_i$ | $\prod_{i=1}^{n} A_i$ | $\prod_{i=1}^{n} (P_i A_i)$ | $(\prod_{i=1}^{n} A_i)^k$ |

The Workflow Management Alliance (WFMC) has proposed four workflow management model structures: Sequence, Circular, Selective, and Parallel, which can support Web service composition modeling, including parallel, selection, and circulation. All three basic models can be converted into sequence types. For ease of discussion, this article only discusses the sequential model.

Given that the evaluation of each attribute is not of the same order of magnitude, the formulas will be normalized as follows:
\[
Q_j = \begin{cases} 
\frac{Q_j^{\text{max}} - Q_j^{\text{min}}}{Q_j^{\text{max}} - Q_j^{\text{min}}} & Q_j^{\text{max}} \neq Q_j^{\text{min}} \\
1 & Q_j^{\text{max}} = Q_j^{\text{min}}
\end{cases} \quad \text{passive type,} \tag{1}
\]
\[
Q_j = \begin{cases} 
\frac{Q_j^{\text{max}} - Q_j^{\text{min}}}{Q_j^{\text{max}} - Q_j^{\text{min}}} & Q_j^{\text{max}} \neq Q_j^{\text{min}} \\
1 & Q_j^{\text{max}} = Q_j^{\text{min}}
\end{cases} \quad \text{positive type,} \tag{2}
\]

where $Q_j^{\text{max}}$ and $Q_j^{\text{min}}$ are the maximum and minimum values of the $j$-th QoS attribute. With increasing and decreasing cloud services, their values will change dynamically. Passive QoS attribute values are computed and transformed as benefit types in Equation (2). Meanwhile, positive QoS attribute values are computed and transformed as cost types in Equation (3).

Compared with the traditional cloud service combination focusing on quality of service (QoS), the cloud service portfolio model in this paper is more focused on the user’s quality of experience (QoE). Aiming at the problem of the QoS and QoE mapping method, this paper adopts the IQoE2QoS algorithm[12] and uses a multi-index fuzzy decision theory to realize the mapping of user’s quality of experience (QoE) to quality of service (QoS). This article uses the evaluation method of MOS (Mean Opinion Score) [13] to evaluate the user’s quality of experience. Based on IQoE2QoS algorithm, first use the indicator statistics chart to divide the range of the indicator, and discretize the indicator to get the learning set, then use this learning set to calculate the weight relationship between the indicators. Suppose the index set is \( I = \{I_1, I_2, I_3, \ldots, I_m\} \), and each index set is discretized to get \( I_i = \{a_{i1}, a_{i2}, a_{i3}, \ldots, a_{in}\} \), then the mutual information is as follows:

\[
I(S, I) = H(S) - H(S|I) \tag{3}
\]

Normalized weight:

\[
w_i = \frac{I(S, I_i)}{I(S, I_1) + \cdots + I(S, I_n)} \tag{4}
\]

Definition 2: Service composition is a five-tuple \( SP = \{n, m, QoS, w, F\} \), \( m = \{m_1, m_2, \ldots, m_t\} \) represents the number of tasks. The value \( n \) represents the number of candidate services.

The literature[14] uses penalty function to construct fitness function, which turn constrained optimization problem into an unconstrained optimization problem, and realize global optimization. The fitness function Fitness is:

\[
Q(x) = \sum_{k \in \{7, S, R, A\}} w_k Q_k \tag{5}
\]

\[
\text{Fitness} = Q(x) - \lambda \sum_{k \in \{7, S, R, A\}} \left( \frac{\Delta Q_k}{Q_k^{\text{max}} - Q_k^{\text{min}}} \right)^2 \tag{6}
\]

Where \( \lambda \) is Penalty function factor; \( Q_k^{\text{max}} \) is the maximum constraint value for the \( k \) attribute; \( Q_k^{\text{min}} \) is the minimum constraint value for the \( k \) attribute; The formula of \( \Delta Q_k \) is:

\[
\Delta Q_k = \begin{cases} 
Q_k^{\text{min}} - Q_k, Q_k \leq Q_k^{\text{min}} \\
0, Q_k^{\text{min}} < Q_k < Q_k^{\text{max}} \\
Q_k - Q_k^{\text{max}}, Q_k \geq Q_k^{\text{max}}
\end{cases} \tag{7}
\]
4. Web Service Composition Optimization Based on Adaptive Mutant Beetle Swarm Algorithm

PSO[15] is a heuristic algorithm that is derived from the intelligent behavior simulation of simplified social groups. The PSO algorithm has few parameters, fast convergence speed, strong searchability, and good effect on solving high-dimensional multi-objective optimization problems. According to the problems of the PSO algorithm, we merge the particle swarm optimization algorithm with the beetle antennae algorithm.

The beetle antennae searching (BAS) algorithm[16-17] is an intelligent algorithm for searching for an optimal solution. The creation of this algorithm’s principle was inspired by the manner in which beetles find their food. However, when faced with the problem of high dimensions, the algorithm has poor performance, insufficient global search capabilities, and local optimal solutions are prone to occur. Therefore, in this paper, the BAS algorithm and the PSO algorithm are combined and adaptive. The mutation factor makes the algorithm performance have a better global search ability when facing the web service composition problem and avoids the situation of entering the local optimal solution. The fused BAS-PSO algorithm has the same calculation process as the PSO algorithm. First, an individual beetle is expanded to a population and its position update formula is as follows:

$$\mathbf{v}_i^{k+1} = \mathbf{v}_i^k + c_1 \cdot r_1 \cdot (\mathbf{P}_{best} - \mathbf{x}_i^k) + c_2 \cdot r_2 \cdot (\mathbf{G}_{best} - \mathbf{x}_i^k) + \mathbf{v}_i$$

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^{k+1}$$

where $\mathbf{v}_i$ is the $i$-th beetle particle’s velocity after $k$ times of iterations. $\mathbf{x}_i$ is the $i$-th beetle particle’s position after $k$ times of iterations. $c_1$ and $c_2$ are the update rate of the beetle population. $r_1, r_2,$ and $r_3$ are the random numbers evaluated among $[0,1]$. $\omega$ is the inertia weight. $\mathbf{P}_{best}$ and $\mathbf{G}_{best}$ are the personal best of a particle and the global best of a population, respectively.

The adaptive mutation factor has a good convergence performance in genetic algorithm, therefore, based on the mutation ideas of genetic algorithm and the literature[18], this paper integrates the adaptive mutation operator into the beetle particle swarm optimization algorithm to prevent the algorithm from premature convergence and enhance the diversity of the population, the improved formula (19) is as follows:

$$\mathbf{v}_i^{k+1} = \omega \cdot \mathbf{v}_i^k + c_1 \cdot r_1 \cdot (\mathbf{P}_{best} - \mathbf{x}_i^k) + c_2 \cdot r_2 \cdot (\mathbf{G}_{best} - \mathbf{x}_i^k) + \mathbf{v}_i$$

where $\mathbf{Random}$ is the random position in the solution space; $\rho$ is the unknown curiosity coefficient. After a lot of experiments, the algorithm works best at 1.5, $r_3$ is the random numbers among $[0,1]$. The mechanism of exploring unknown space is added, which not only expands the particle optimization range and enhances the population diversity, but also makes the algorithm easier to jump out of the local optimum, thereby preventing the premature convergence of the algorithm.

The learning factors that focus on a constant value will cause the velocity and position of the local particles to exhibit regular linear changes. Thus, the algorithm’s ability to search for the best solution becomes limited. Reference[6] proposed nonlinear dynamic learning factors based on a trigonometric function to control the particles’ expanding capacity and global convergence capability. $c_1$ and $c_2$ are defined as the following equations:

$$c_1 = \rho_t + \cos^2(\rho_t \times t / G_{max})$$

$$c_2 = \rho_t + \sin^2(\rho_t \times t / G_{max})$$

where $t$ is the current iteration times, $\rho_t$ is a constant, and $G_{max}$ is the maximum iteration times.

The Steps Of Algorithm:

Input: $G_{max}$, $c_1$, $c_2$, pop, d0;
Output: $G_{best}$;
Initialization parameters, $k=0$;
For \(i = 1: \text{pop} \)
\[
P_{\text{best}} = x_i;
\]
end
\[
G_{\text{best}} = \text{maxfitness}(x_i);
\]
While \((k < G_{\text{max}})\)
Update \(c_1, c_2\) as (12) and (13);
For \(i = 1: \text{pop} \)
Update \(v_i\) and \(x_i\) as (8), (11) and (10);
If \(\text{fitness}(x_i) > \text{fitness}(P_{\text{best}})\)
\[
P_{\text{best}} = x_i;
\]
end
If \(\text{fitness}(x_i) > \text{fitness}(P_{\text{best}})\)
end
end
\[
G_{\text{best}} = \text{maxfitness}(P_{\text{best}});
\]
End

5. Experimental simulation and result analysis
To verify the effectiveness of the MBSO algorithm proposed in this paper to solve the problem of web service composition, a large number of experiments have been conducted in this paper, owing to the workflow structure of other service composition can be converted into a sequential structure, the author of this study conducted multiple sets of experiments in sequential structure using QWS real data sets. QWS data sets are web service data sets compiled by Professor Eyhab Al-Masri of Guelph University. All the data sets in QWS are real data sets collected from various service websites.

To prove the advantages of the MBSO algorithm proposed in this paper, this paper selected the CGA algorithm in[19] and the DPSO algorithm in[20] for comparison. The experiment mainly compares the feasibility, convergence, and stability of the three algorithms. The experiments use the same QWS data set, to improve the accuracy of the experiment, the data set is standardized by formula (2). The experimental environment was as follows: Intel(R) Xeon(R) CPU E3-1220 V2 @ 3.10GHz 3.50GHz, 12 GB ddr3 1333MHz memory, 64 bit Windows 10 OS, and MATLAB R2015b.

The experimental parameter settings used in this study are provided in Table 2.

| Parameters | Value |
|------------|-------|
| \(V_{\text{max}}\) | 3 |
| \(V_{\text{min}}\) | 0.5 |
| \(d_0\) | 0.8 |
| \(\rho\) | 1.5 |
| step | 3 |
| \(\omega_{\text{max}}\) | 0.9 |
| \(\omega_{\text{min}}\) | 0.4 |
| MaxIteration | 100 |

\(V_{\text{max}}\) and \(V_{\text{min}}\) are the maximum and minimum values of velocity, \(d_0\) is the distance between two antennas, step is the initial step, \(\omega_{\text{max}}\) and \(\omega_{\text{min}}\) are the inertia weights of the PSO algorithm, \(\rho\) is adaptive mutation factor.

According to the literature[21], the initial step was 3. For the distance between two antennae \((d_0)\) of MBSO, the initial value was 0.8. According to the literature[6], the inertia weight values are \(\omega_{\text{max}} = 0.9\)
and $\omega_{\text{min}} = 0.4$. The four QoS attributes, namely, execution time, reliability, availability, and success rate, were adopted in this study. Their attribute weights were set as 0.25, 0.25, 0.25, and 0.25, respectively.

5.1. Feasibility

According to the fitness function proposed in this paper, the three performances are compared. The higher the fitness, the higher the quality of service combination found; otherwise, the worse. To verify the feasibility of the algorithm proposed in this paper, this paper conducts experiments on the scale of $n \times m$, where $n$ is the number of tasks and $m$ is the number of candidate services. To reduce the chance in the experiment, this paper conducted 50 experiments to take the average.

| Table 3. $n \times m$ scale average fitness value |
|-----------------------------------------------|
| **n** | **m** | **Average Fitness** |
| **10** |       | **MBSO** | **CGA** | **DPSO** |
| 6    | 10    | 0.8369  | 0.8425 | 0.7135 |
| 50   | 0.8902 | 0.8845 | 0.7355 |
| 100  | 0.9111 | 0.9081 | 0.8579 |
| 200  | 0.9598 | 0.9331 | 0.8854 |
| 15    | 10    | 0.8340  | 0.8414 | 0.7147 |
| 50   | 0.8939 | 0.8829 | 0.7906 |
| 100  | 0.9055 | 0.8927 | 0.8508 |
| 200  | 0.9633 | 0.9484 | 0.9115 |
| 20    | 10    | 0.7954  | 0.8023 | 0.7656 |
| 50   | 0.9158 | 0.8868 | 0.8591 |
| 100  | 0.9407 | 0.9139 | 0.8848 |
| 200  | 0.9544 | 0.9227 | 0.9068 |
| 25    | 10    | 0.8619  | 0.8006 | 0.7945 |
| 50   | 0.8918 | 0.8462 | 0.8123 |
| 100  | 0.9536 | 0.8979 | 0.8728 |
| 200  | 0.9732 | 0.9332 | 0.8966 |

It can be seen in Table 3 that the average fitness value of the three algorithms under different sizes of $n \times m$, MBSO is higher than the other two algorithms, proving that the MBSO algorithm proposed in this paper has good algorithm accuracy and better performance.

5.2. Convergence Analysis

To analyze the convergence of MBSO, this paper compares the convergence of three algorithms when the candidate service tasks were selected as 200, 10, 15 and 20. To analyze the convergence more clearly, this article sets the number of iterations to 100. A higher average fitness value is obtained at a lower number of iterations, indicating that the algorithm has good convergence. The average fitness value of this paper was averaged after 20 experiments under the same experimental environment.
It can be seen from Figures 2, 3, and 4 that as the number of tasks increases, the dimension of the problem is also increasing, and the convergence of the three algorithms is continuously slowing down, but the accuracy of the MBSO algorithm is significantly higher than the other two algorithms when the number of iterations is 50, the algorithm basically converges and the average fitness value is the highest. The experiment in this section proves that the MBSO algorithm has good algorithm accuracy and convergence.

5.3. The Stability Of The Algorithm

Robustness is the stability of the algorithm, which indicates the fault tolerance of the algorithm. The swarm intelligence algorithm has a certain degree of randomness. In the case of a large difference in the complexity of the problem, the gap between the solutions is more obvious. In this section, in order to verify the stability of the MBSO algorithm to solve the web service composition of different sizes, the MBSO, CGA and DPSO algorithm was run 50 times under the three service scales of 10*50, 15*100, and 20*200 respectively, and record the fitness value obtained by each algorithm and calculate its standard deviation. In this section, the standard deviation of the fitness value is used as the evaluation criterion of the robustness of the algorithm. At the same scale, the smaller the standard deviation of the algorithm fitness value, the stronger the robustness of the algorithm, and the more stable the performance of the algorithm.

It can be seen from Figure 5 that the standard deviations of fitness values of MBSO and CGA algorithms are not much different, however, the standard deviation of MBSO is smaller than that of the other two algorithms. This is because MBSO introduced adaptive mutation factors and dynamic learning...
factors in the BSO algorithm, which ensured the diversity of the population in the initial and iterative processes of the algorithm, and at the same time ensured the robustness of the algorithm. When solving sequential Web service composition problems with different numbers of services, the MBSO algorithm is more stable and robust than the other four algorithms.

6. Conclusion
In the face of QoS-aware web service composition optimization problems, this paper proposes an improved adaptive mutant beetle swarm algorithm. Combining the beetle search algorithm with the traditional PSO algorithm improves the convergence ability and accuracy of the PSO algorithm; the addition of adaptive mutation factors and dynamic learning factors enhance the diversity of algorithm populations and the stability of the algorithm.

This paper mainly studies the web service composition on the algorithm model and verifies the experimental research on the common web service composition model, in the future research, the improvement of web service composition model needs further study.

Acknowledgments
This research was funded by the Project of education department of Jilin province, the project name is: Remote Monitoring System of No-Tillage Planting Quality in Cloud Environment, grant number JJKH20200801KJ. The authors express their sincere appreciation to the editors and the anonymous reviewers for their helpful comments.

References
[1] WANG Peng-wei, DING Zhi-jun, JIANG Chang-jun, et al. (2016) Automatic web service composition based on uncertainty execution effects. IEEE Transactions on Services Computing, 9(4): 551-565.
[2] Zeng Liangzhao, Benatallah B, Ng A, et al. (2004) QoS-aware Middleware for Web Services Composition. IEEE Trans. on Software Engineering, 30(5): 311-327.
[3] Samia Chibani Sadouki, Abdelkamel, Tari. (2019) Multi-objective and discrete elephants herding optimization algorithm for QoS aware web service composition. RAIRO-Oper. Res, 53: 445-459.
[4] Yang Zhen, Yang Tiantian, Fan Kefeng, Wang Yong. (2018) Cloud Service Composition Based on Trust Combination. Acta Electronica Sinica, 46(3): 614-620.
[5] Karimi M B, Isazadeh A, Rahmani A M. (2017) QoS-aware service composition in cloud computing using data mining techniques and genetic algorithm. The Journal of Supercomputing, 73(4): 1387-1415.
[6] WEN Tao, SHENG Guo Jun, GUO Quan, LI Ying Qiu. (2013) Web Service Composition Based on Modified Particle Swarm Optimization. CHINESE JOURNAL OF COMPUTERS, 36(5): 1031-1046.
[7] Seghir F, Khababa A. (2018) A hybrid approach using genetic and fruit fly optimization algorithms for QoS-aware cloud service composition. Journal of Intelligent Manufacturing, 6: 1-20.
[8] Afshin Naseri, Nima Jafari Navimipour. (2018) A new agent-based method for QoS-aware cloud service composition using particle swarm optimization algorithm. Journal of Ambient Intelligence and Humanized Computing, doi:10.1007/s12652-018-0773-8.
[9] Hayyolalam V, Kazem A A P. (2018) A systematic literature review on QoS-aware service composition and selection in cloud environment. Journal of Network & Computer Applications, 110: 52-74.
[10] Xu J, Guo L, Zhang R, et al. (2018) QoS-aware Service Composition Using Fuzzy Set Theory and Genetic Algorithm. Wireless Personal Communications, doi: 10.1007/s11277-017-5129-8.
[11] Jian C F, Wang Y. (2014) Batch task scheduling-oriented optimization modelling and simulation in cloud manufacturing. International Journal of Simulation Modelling, 13(1): 93101.
[12] Ni Ping, Liao Jian-xin, Zhu Xiao-min, et al. (2010) A method of QoE mapped to SLA based on QoS. Journal of Electronics & Information Technology, 32(6): 1463-1468.

[13] Tasaka S, Ishibashi Y. (2002) Mutually compensatory property of multimedia QoS. In: IEEE International Conference on Communications (ICC), IEEE. 11051111.

[14] ZHANG C, LIN X. (2009) A genetic algorithm with improved convergence capability for QoS-aware Web service selection. In: Proceedings of the International Conference on Management and service Science. Washington, D.C, USA: IEEE. 1-4.

[15] Eberhart R, Kennedy J. (1995) A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya, Japan. 39–43.

[16] Jiang, X. and S. Li. (2017) BAS: Beetle Antennae Search Algorithm for Optimization Problems. International Journal of Robotics and Control, 1(1). doi:10.5430/ijrc.v1n1p1.

[17] Jiang, X.and S. Li. (2017) Beetle Antennae Search without Parameter Tuning (BAS-WPT) for Multi-objective Optimization. arXive preprint arXiv:1710.10724, 2017.

[18] Xiao Liang, Liu Sitong. (2017) Asynchronous collaborative chicken swarm optimization with mutation based on cognitive diversity. Computer Science, 44( s1):99104.

[19] Yuanzhang Li, Jingjing Hu, Zhuozhuo Wu, Chen Liu1, Feifei Peng, Yu Zhang. (2018) Research on QoS service composition based on coevolutionary genetic algorithm. Soft Computing, doi: 10.1007/s00500-018-3510-5.

[20] Ali Husseinzadeh Kashan, Behrooz Karimi. (2009) A discrete particle swarm optimization algorithm for scheduling parallel machines. Computers & Industrial Engineering, 56: 216-223.

[21] JIANG Li, YE Runzhou, LIANG Changyong, LU Wenxing. (2019) Improved Second-Order Oscillatory Particle Swarm Optimization. Computer Engineering and Applications, 55(9): 130-138.