Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 9931689, 10 pages
https://doi.org/10.1155/2021/9931689

Research Article

Research on Edge Service Composition Method Based on BAS Algorithm

Zhoujie Du [1] and Huaikou Miao [1,2]

[1]*School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China*
[2]*Shanghai Key Laboratory of Computer Software Testing and Evaluating, Shanghai 201112, China*

Correspondence should be addressed to Zhoujie Du; duzhoujie@163.com

Received 16 March 2021; Revised 1 July 2021; Accepted 8 July 2021; Published 15 July 2021

Academic Editor: Ezequiel López-Rubio

Edge services are transferred data processing, application running, and implementation of some functional services from cloud central server to network edge server to provide services. Combined edge service can effectively reduce task computation in the cloud, shorten transmission distance of processing data, quickly decompose task of service request, and select the optimal edge service combination to provide service for users. BAS is an efficient intelligent optimization algorithm, which can achieve efficient optimization and neither need to know the specific form of function nor need gradient information. This paper designs an edge service composition model based on edge computing and proposes a method about edge service composition by BAS optimization algorithm. Our proposed method has obvious advantages in service composition efficiency compared with service composition method based on PSO or WPA heuristic algorithm. Compared with cloud service composition method, our proposed method has advantages of shorter service response time, low cost, and high quality of user experience.

1. Introduction

In recent years, researchers and scholars have made some achievements in research of QoS optimized service composition methods. Each task has multiple services that meet user's functional requirement in QoS optimized service composition execution workflow model. How to generate optimized composite services by using evaluation conditions of QoS solution. Considering QoS multidimensional attribute characteristics value in service composition, at present, there are main research methods including integer programming [1–3], mixed-integer programming [4–6], heuristic search algorithm [7–15], intelligent algorithm [16–23], etc.

In 2004, IBM T. J. Watson Research Institute Zeng et al. [1, 2] proposed an AgFlow service composition middleware platform, which modeled QoS-aware service composition problem as an integer programming problem driven by workflow, and used CPLEX solver to generate the best QoS composition service solution under business flow. Then, Deng et al. [3] proposed a QoS constraint-driven service composition optimization method in mobile cloud computing application environment. Ardagna and Pernici [4, 5] modeled service composition problem based on workflow as a mixed-integer programming problem and transformed cycle structure into a sequential structure, which reduced the problem of low performance caused by cycle. When user's global QoS constraints cannot be satisfied, the second optimization can reduce probability of service composition failure. Then, Wang et al. proposed an effective optimal service composition method [6] by using mixed-integer programming method from perspective of user trust and preference characteristics. Berbner et al. [7] proposed a heuristic search method considering constraints from QoS-aware service composition driven by workflow model. They used backtracking algorithm to generate effective service composition solution and selected optimal solution of service composition through branch limit method. Considering user end-to-end QoS constraints [8, 9], as well as shortcomings of global optimization and local optimization, Alrifai et al. [10]

proposed a comprehensive Web service composition method combined global optimization and local selection technology. It used mixed-integer programming method to search global QoS constrained optimal solution under local constraints by local selection generated local optimal Web service. Under predefined workflow model, Hwang et al. [11] proposed a reliability measurement selection method to generate composite service solution that meets requirement by using finite state machine to represent call process among services. In order to further improve the reliability of service, Haddad et al. [12] proposed a service composition algorithm of local optimal selection from two perspectives of service transaction attribute and QoS. Subsequently, some researchers put forward efficient QoS optimization service composition methods from the perspective of local optimization selection [13–15].

Considering multidimensional nonfunctional characteristic attribute of service QoS, some scholars modeled QoS-aware service composition problem based on workflow as a multiobjective optimization problem [16, 17], which uses intelligent algorithms to solve and generate optimal composite service. Wagner et al. [18] proposed a QoS service composition model based on multiobjective optimization problem, and it considered multiple possible workflow execution structure at the same time. Cremene et al. [19] analyzed and compared QoS-aware Web service composition problem of the most popular centralized multiobjective optimization algorithm in the moment. Yilmaz et al. [20] proposed an improved genetic algorithm for QoS-aware to realize dynamic service composition. Kim et al. [21] put forward a method to automatically realize service composition in IoT application environment, which is based on efficient resource allocation method of genetic algorithm. Hossain et al. [22] proposed a large-scale data-driven service composition method by using parallel clustering particle swarm optimization algorithm in mobile environment. Peng et al. [23] implemented an adaptive service composition method by using distribution estimation of constrained Boltzmann machine.

As a novel stochastic optimization algorithm, Beetle antenna search (BAS) algorithm [24] is proposed in 2017 by Jiang and Li, which has a more concise search strategy based on the foraging behavior of beetles. The beetle antenna search can achieve efficient optimization does not need to know the specific function form and gradient information. Compared with particle swarm optimization (PSO), BAS only needs one beetle, which can reduce computational burden greatly. It does not know where food is when beetle foraging, but it can find food based on concentration of the food smell. The beetle has two long antennas; if smell concentration detected by left antennas is higher than that on the right, then the beetle will fly to the left; otherwise, it will fly to the right. According to this simple principle, beetles can find food effectively. The smell of food is equivalent to a function. Each point in the three-dimensional space function has a different value. Two antennas of beetle can sense two points of odor value near itself. The purpose of beetle antenna search is to find the point (where the food is) with the largest odor value in global space. We can optimize service composition efficiently by imitating the behavior of beetle foraging.

In this paper, we propose an edge service composition model based on edge computing and propose an edge service composition method based on BAS optimization algorithm. Beetle antenna search (BAS) algorithm [24] is proposed in 2017 (Jiang and Li), which has a more concise search strategy based on the foraging behavior of beetles.

2. Edge Service Composition

2.1. Edge Service Model. Edge computing is a new large-scale computing processing model. It integrates network, computing, storage, and application to provide edge intelligence service nearby. It can satisfy the key requirements of industry digitalization in agile connection, real-time business, data optimization, application intelligence, and security and privacy protection [25–27]. Cognitive computing on the edge for healthcare service robots can be processed by the robot without frequent communications with data centers [28]. An Edge Traffic Flow Detection Scheme and model proposed in [29, 30] can take full advantage of the computing resources of the surrounding vehicles and greatly reduce the execution time of the computation tasks.

Therefore, research on edge service composition method proposed in this paper based on edge computing mode uses superiority of edge computing and heuristic intelligent optimization algorithm under the environment of Internet. Edge service composition is generally divided into four main links: edge service request, edge computing, composite edge services, and feedback request processing result, reports the generated service logs to cloud service platform, reports confirmation, and so on. The most important difference between edge service composition and cloud service composition is that the edge service composition offloads work tasks (such as computing analysis) from cloud and loads it to edge server (that is, close to the end of service request). The advantage of edge service composition is that it can effectively reduce task computing scale in cloud, shorten transmission distance of service request, split service request quickly, and choose the best edge service composition to provide service for users, as well as improve quality of user experience greatly. Although the distance between edge and cloud is far, they are connected and dependent on each other. On the one hand, when a service request is sent out, the nearest edge server composites best service resource quickly to process request content and feeds back the execution result. On the other hand, edge server will report service status to be under the jurisdiction of cloud in the form of log, and cloud server will record service content, archive, and send confirmation feedback to edge server. Edge server will send a support request to cloud if composite edge service fails to meet a complex task request. The nearby cloud resources respond to request of edge server and accomplish request content together with edge server when cloud receive service request comes from edge server. In this paper, edge service composition model is proposed as shown in Figure 1.

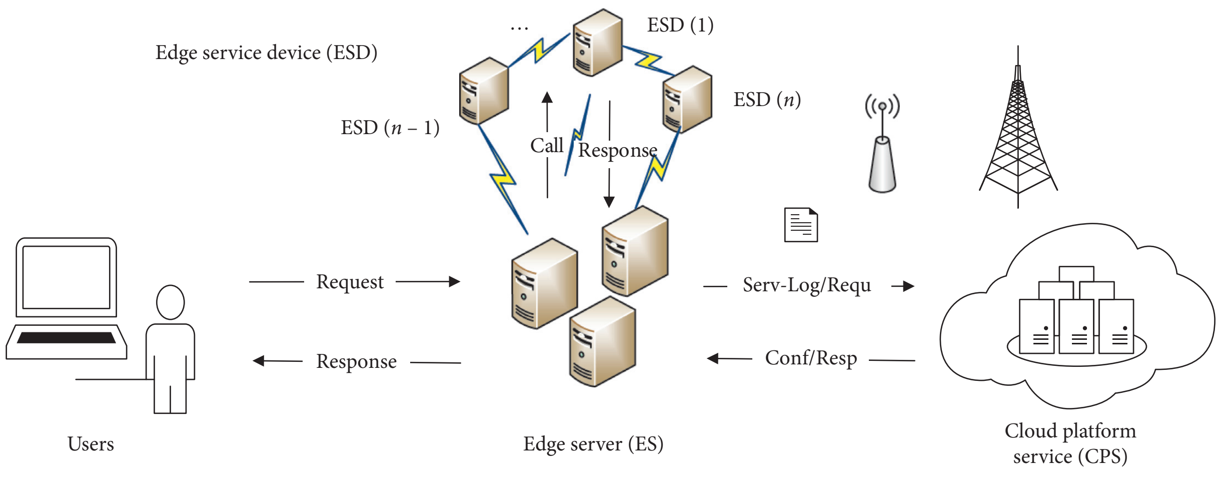


Figure 1: Edge service composition model.

Since QoS of service composition is evaluated from the perspective of nonfunctional attributes, so we use nonfunctional QoS attribute indicators to evaluate edge computing service composition in this paper. We evaluate and study QoS attributes such as service time (T), service cost (C), service availability (Av), service reliability (Rel), service reputation (Rep), and so on. The expression of QoS is shown in the following formula:

$$\mathrm{QoS} = \{T(\mathrm{ES}), C(\mathrm{ES}), \mathrm{Av}(\mathrm{ES}), \mathrm{Rel}(\mathrm{ES}), \mathrm{Rep}(\mathrm{ES})\}. \quad (1)$$

The attribute of QoS definition and quantitative expressions are as follows:

(1) Trefers to the total time expend from user submit service to accomplish service execution and return result. Generally, the shorter the service time is, the higher the service efficiency is.

(2) Crefers to all cost from user submit service request to accomplish service execution and return result.

(3) Avrefers to the probability of service can be accessed successfully, that is, the ratio of the number of successful accesses edge service to total number of accesses, expressed as $\mathrm{Av} = A_s/A_n$, and A_n is total number of accesses to the edge service over a period of time; A_s is the number of edge service successful responses.

(4) Relrefers to the probability that edge server can provide service normally, that is to say, the proportion of service instance occupied time to total service working time, expressed as $\mathrm{Rel} = T_r/T_n$, and T_n is the total running time of edge service; T_r is normal operation time in this period of time.

(5) Reprefers to measure trustworthiness of a service, that is, the evaluation made by users after using edge service, expressed as $\mathrm{Re}p = \sum_{i=1}^{n} R_i/n$, where R_i is the evaluation of edge service made by ith user and n is counts for users evaluate edge service.

To build an QoS evaluation model of edge service, we can select a group of optimal service compositions from a large number of edge services by comparing QoS attribute values. However, because the significance of each QoS attribute is different, the representation method and quantization standard are also different. In order to eliminate influence of different dimensions, we need to normalize QoS.

The service time, service cost, and other attribute values of edge service QoS are expressed in the following normalization formula:

$$q_i^- = \frac{q_{\max} - q_i}{q_{\max} - q_{\min}}. \quad (2)$$

The availability, reliability, reputation, and other attribute values of edge service QoS are expressed in the following normalization formula:

$$q_i^+ = \frac{q_i - q_{\min}}{q_{\max} - q_{\min}}, \quad (3)$$

where $q_{\max} - q_{\min} \neq 0$, q_i^- or q_i^+ represent normalized value of edge service ith attribute, q_i represents ith QoS attribute value of edge service, $q_{\max}$ and $q_{\min}$ represent QoS attribute maximum and minimum in candidate set of edge service. It can be seen from the above formula that the normalized value of QoS attribute increases along with the increase of availability, reliability, and reputation and decreases along with the increase of service time and service cost. The QoS attribute values of edge service are all in the range of [0, 1] after they are normalized. We can unify processed QoS attribute values directly in this way.

2.2. Edge Service Composition Processes

Definition 1 (edge service (ES)). Edge service is provided by edge node server (close to service request) instead of central server in network deal with data or application operation for users. The edge service is formalized into a seven-tuple shown in the following formula:

$$\mathrm{ES} = \left(\mathrm{ID}, \mathrm{CID}, \mathrm{Fun}, \mathrm{CN}, \frac{\mathrm{Int}}{\mathrm{Out}}, \mathrm{Info}, \mathrm{QoS}\right). \quad (4)$$

where ID is identifier of edge service in edge service system, CID is the identifier of cloud service corresponding to the edge service, Fun is functional description of edge service, CN is the functional classification number of edge service, Int/Out represent input or output of service, Info is description of edge service attribute by provider, and QoS is quantitative evaluation of edge service quality.

Definition 2. (edge service candidate set (ESCS/ES)). It is a set of several edge services with same or similar functions that can complete a certain task independently; that is, edge services in the same set have same CN. Edge service candidate set is formalized as a triple shown in the following formula:

$$\frac{\mathrm{ESCS}}{\mathrm{ES}} = (\mathrm{CN}, \mathrm{Fun}, \mathrm{Memb}), \tag{5}$$

where CN represents classification number of edge service candidate set, Fun is functional description of all edge services in candidate set, and Memb is a sequence of all edge services IDs.

Definition 3 (composite edge service (CES)). CES is a logical service set composed of several sub-edge services with different functions, which can meet the user needs of handle complex tasks. The combined edge service can be expressed as a sequence shown in the following formula:

$$\mathrm{CES} = \left(\mathrm{ES}_1, \mathrm{ES}_2, \ldots, \mathrm{ES}_n\right), \tag{6}$$

where $\mathrm{ES}_i\,(i = 1, 2, \ldots, n)$ is subservice of edge service combination and n is the number of subservices in edge service combination.

Stable and efficient edge computing is the key to edge service complete various tasks, because single-function edge services cannot satisfy multifunctional needs of accomplished large-scale and complex tasks. It is necessary to combine multiple edge services according to certain logic to provide service to meet this requirement. Generally, we select a suitable edge service combination for a complex task through several main processes including task division, scheduling analysis, resource (or service) configuration, resource (or service) optimization, composite edge service, and general process as shown in Figure 2.

(1) Service division: the edge server divides service request into n subtasks that cannot be divided and can be executed by a single edge server after received service request. T represents the task of service request, and T_i represents subtask; then, $T = \sum_{i=1}^{n} T_i$, $(i \in [1, n])$.

(2) Scheduling analysis: edge server marks all edge service resources that can provide service for client requests through calculation and analysis, and edge service resources with the same or similar functions are marked as same type of resources.

(3) Resource (or service) configuration: ES_{T_i} represents edge services that can provide services for subtask (T_i); there are m (m is an integer) edge services resources that can provide services to each subtask; then T_i corresponding candidate service set can be expressed as shown in the following formula:

$$\mathrm{ES}_{T_i} = \left\{\mathrm{ES}_{T_i}^{1}, \mathrm{ES}_{T_i}^{2}, \ldots, \mathrm{ES}_{T_i}^{m-1}, \mathrm{ES}_{T_i}^{m}\right\}. \tag{7}$$

If subtask T_i finally selects edge service $\mathrm{ES}_{T_i}^{j}$ from candidate service set ES_{T_i} to provide services for it and enables the value of QoS (T_i) to be optimized as well, then $\mathrm{ES}_{T_i} = \mathrm{ES}_{T_i}^{j}$, and $i \in [1, m]$.

(4) Resource (or service) optimization: we can use edge computing advantages to find a suitable edge service $(\mathrm{ES}_{T_i}^{j})$ for each subtask (T_i) through certain service optimization rules (such as beetle whisker algorithm, particle swarm algorithm, ant colony algorithm, wolf swarm algorithm, etc.) and then combine all selected services together and satisfy QoS (T) value optimal.

(5) Composite edge service: according to the results of resource optimization, choose a group of services composition to provide services for user requests and make QoS value optimal. CES represents composite edge service, and $\mathrm{CES} = \sum_{i=1}^{n} \mathrm{ES}_{T_i}$($n$ is the number of subservices). Let us suppose that $\mathrm{ES}_{T_1} = \mathrm{ES}_{T_1}^{1}$, $\mathrm{ES}_{T_2} = \mathrm{ES}_{T_2}^{m-1}$,$\ldots\mathrm{ES}_{T_{n-1}} = \mathrm{ES}_{T_{n-1}}^{2}$, and $\mathrm{ES}_{T_n} = \mathrm{ES}_{T_n}^{m}$ after service selection optimization, and composite edge service set is expressed as follows:

$$\begin{aligned}\mathrm{CES} &= \left\{\mathrm{ES}_{T_1}, \mathrm{ES}_{T_2}, \ldots, \mathrm{ES}_{T_{n-1}}, \mathrm{ES}_{T_n}\right\} \\ &= \left\{\mathrm{ES}_{T_1}^{1}, \mathrm{ES}_{T_2}^{m-1}, \ldots, \mathrm{ES}_{T_{n-1}}^{2}, \mathrm{ES}_{T_n}^{m}\right\},\end{aligned} \tag{8}$$

where n is the number of subtasks and m is the number of candidate services for each subtask.

2.3. The Formal Model of Service Composition. The QoS attribute value of composite edge service is not only related to QoS attribute value of single edge service but also related to structure between edge services. There are four basic structures: sequential structure, selective structure, parallel structure, and cycle structure in edge service composition. The QoS attribute aggregation function models corresponding to four basic structures are shown in Table 1.

In summary, we assume that each service composition has n unique edge services and has time, cost, availability, reliability, and credibility five QoS attributes, and any edge service composition path is P; then, the QoS expression is as follows:

$$\mathrm{QoSP} = \{T(P), C(P), \mathrm{Av}(P), \mathrm{Rel}(P), \mathrm{Rep}(P)\}. \tag{9}$$

We can derive formula (10) by using formula (9), which is expressed as follows:

$$\begin{aligned}\mathrm{QoSP} = {} & F_{\mathrm{Seq}}(T, C, \mathrm{Av}, \mathrm{Rel}, \mathrm{Rep}) + F_{\mathrm{Sel}}(T, C, \mathrm{Av}, \mathrm{Rel}, \mathrm{Rep}) \\ & + F_{\mathrm{Par}}(T, C, \mathrm{Av}, \mathrm{Rel}, \mathrm{Rep}) + F_{\mathrm{Cyc}}(T, C, \mathrm{Av}, \mathrm{Rel}, \mathrm{Rep}).\end{aligned} \tag{10}$$

The values of F_{Seq}, F_{Sel}, F_{Par}, and F_{Cyc} depend on the structure of actual execution path of edge service, and

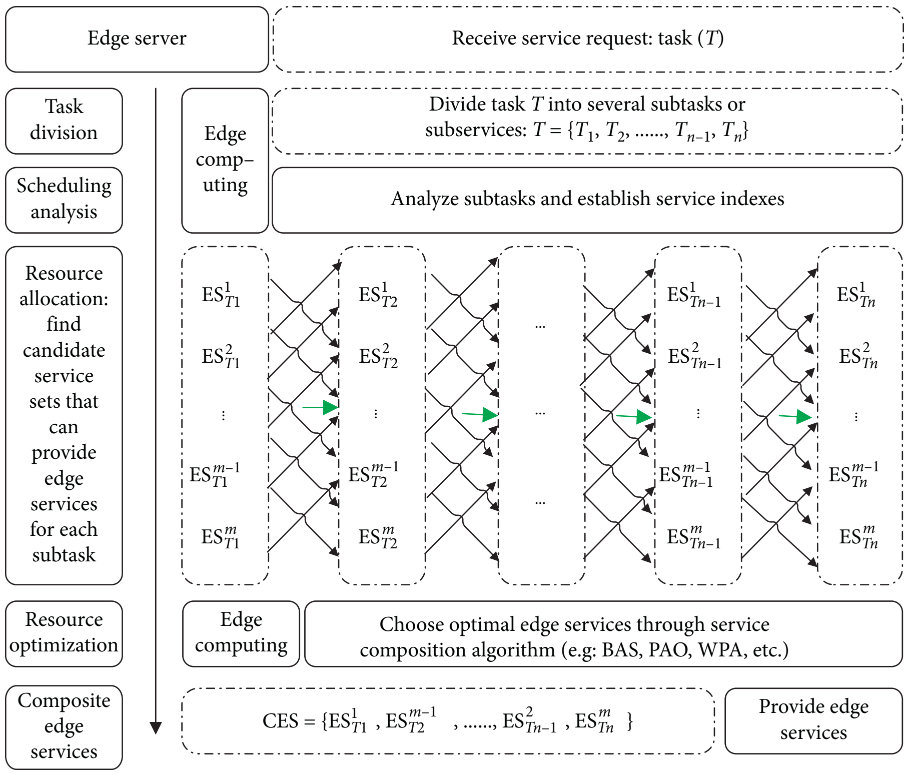
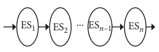
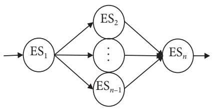
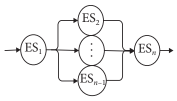
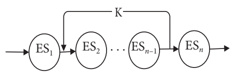

Edge server
Receive service request: task (T)
Task division
Scheduling analysis
Edge comp–uting
Divide task T into several subtasks or subservices: $T = \{T_1, T_2, \ldots\ldots, T_{n-1}, T_n\}$
Analyze subtasks and establish service indexes
Resource allocation: find candidate service sets that can provide edge services for each subtask
ES^1_{T1} ES^2_{T1} ES^{m-1}_{T1} ES^m_{T1}
ES^1_{T2} ES^2_{T2} ES^{m-1}_{T2} ES^m_{T2}
ES^1_{Tn-1} ES^2_{Tn-1} ES^{m-1}_{Tn-1} ES^m_{Tn-1}
ES^1_{Tn} ES^2_{Tn} ES^{m-1}_{Tn} ES^m_{Tn}
Resource optimization
Edge computing
Choose optimal edge services through service composition algorithm (e.g: BAS, PAO, WPA, etc.)
Composite edge services
$CES = \{ES^1_{T1}, ES^{m-1}_{T2}, \ldots\ldots, ES^2_{Tn-1}, ES^m_{Tn}\}$
Provide edge services

Figure 2: Edge service composition framework model.

Table 1: Aggregate function model of QoS attributes.

Attributes	Structure type			
	Sequence 	Select 	Parallel 	Cycle 
Service time (T)	$\sum_{i=1}^{n} T(\mathrm{ES}_i)$	$\sum_{i=1}^{n} (T(\mathrm{ES}_i) * p_i)$	$\mathrm{Max}(T(\mathrm{ES}_i))\ i \in [1, n]$	$K * \sum_{i=1}^{n} T(\mathrm{ES}_i)$ (note: K is the number of cycles)
Service cost (C)	$\sum_{i=1}^{n} C(\mathrm{ES}_i)$	$\sum_{i=1}^{n} (C(\mathrm{ES}_i) * p_i)$	$\sum_{i=1}^{n} C(\mathrm{ES}_i)$	$K * \sum_{i=1}^{n} C(\mathrm{ES}_i)$ (note: K is the number of cycles)
Availability (Av)	$\prod_{i=1}^{n} \mathrm{Av}(\mathrm{ES}_i)$	$\sum_{i=1}^{n} (Av(ES_i) * p_i)$	$\prod_{i=1}^{n} \mathrm{Av}(\mathrm{ES}_i)$	$\prod_{i=1}^{n} \mathrm{Av}(\mathrm{ES}_i)$
Reliability (Rel)	$\prod_{i=1}^{n} \mathrm{Rel}(\mathrm{ES}_i)$	$\sum_{i=1}^{n} (\mathrm{Rel}(\mathrm{ES}_i) * p_i)$	$\prod_{i=1}^{n} \mathrm{Rel}(\mathrm{ES}_i)$	$\prod_{i=1}^{n} \mathrm{Rel}(\mathrm{ES}_i)$
Reputation (Rep)	$\sum_{i=1}^{n} (\mathrm{Rep}(\mathrm{ES}_i)/n)$	$\sum_{i=1}^{n} (\mathrm{Rep}(\mathrm{ES}_i) * p_i)$	$\sum_{i=1}^{n} (\mathrm{Rep}(\mathrm{ES}_i)/n)$	$\sum_{i=1}^{n} (\mathrm{Rep}(\mathrm{ES}_i)/n)$

$F_{\mathrm{Seq}} + F_{\mathrm{Sel}} + F_{\mathrm{Par}} + F_{\mathrm{Cyc}} = 1$. We can derive formula (11) by using formula (9) and formula (10), which is expressed as follows:

$$\begin{aligned}\mathrm{QoSP} = &\left\{T_{\mathrm{Seq}}(P) + C_{\mathrm{Seq}}(P) + \mathrm{Av}_{\mathrm{Seq}}(P) + \mathrm{Rel}_{\mathrm{Seq}}(P) + \mathrm{Rep}_{\mathrm{Seq}}(P)\right\}\\ &+\{T_{\mathrm{Sel}}(P) + C_{\mathrm{Sel}}(P) + \mathrm{Av}_{\mathrm{Sel}}(P) + \mathrm{Rel}_{\mathrm{Sel}}(P) + \mathrm{Rep}_{\mathrm{Sel}}(P)\}\\ &+\{T_{\mathrm{Par}}(P) + C_{\mathrm{Par}}(P) + \mathrm{Av}_{\mathrm{Par}}(P) + \mathrm{Rel}_{\mathrm{Par}}(P) + \mathrm{Rep}_{\mathrm{Par}}(P)\}\\ &+\left\{T_{\mathrm{Cyc}}(P) + C_{\mathrm{Cyc}}(P) + \mathrm{Av}_{\mathrm{Cyc}}(P) + \mathrm{Rel}_{\mathrm{Cyc}}(P) + \mathrm{Rep}_{\mathrm{Cyc}}(P)\right\}.\end{aligned} \tag{11}$$

We can derive formula (12) by using formulas (9)–(11), which is expressed as follows:

$$\begin{cases} T(P) = T_{\text{Seq}}(P) + T_{\text{Sel}}(P) + T_{\text{Par}}(P) + T_{\text{Cyc}}(P), \\ C(P) = C_{\text{Seq}}(P) + C_{\text{Sel}}(P) + C_{\text{Par}}(P) + C_{\text{Cyc}}(P), \\ \text{Av}(P) = \text{Av}_{\text{Seq}}(P) + \text{Av}_{\text{Sel}}(P) + \text{Av}T_{\text{Par}}(P) + \text{Av}_{\text{Cyc}}(P), \\ \text{Rel}(P) = \text{Rel}_{\text{Seq}}(P) + \text{Rel}_{\text{Sel}}(P) + \text{Rel}_{\text{Par}}(P) + \text{Rel}_{\text{Cyc}}(P), \\ \text{Rep}(P) = \text{Rep}_{\text{Seq}}(P) + \text{Rep}_{\text{Sel}}(P) + \text{Rep}_{\text{Par}}(P) + \text{Rep}_{\text{Cyc}}(P). \end{cases} \tag{12}$$

Since the parallel, selection, and cycle modes can be converted to sequence mode by using related technologies, we only discuss serial workflow mode in this paper. The goal of service composition is to minimize time and cost and to maximize availability, reliability, and reputation, so the expression of edge service composition QoS model is shown in the following formula:

$$\begin{cases} \text{QoS}(P) = \text{Min}\left(\varphi_1 T(P) + \varphi_2 C(P) + \dfrac{\varphi_3}{\text{Av}(P)} + \dfrac{\varphi_4}{\text{Rel}(P)} + \dfrac{\varphi_5}{\text{Rep}(P)}\right), \\ \varphi_1 + \varphi_2 + \varphi_3 + \varphi_4 + \varphi_5 = 1. \end{cases} \tag{13}$$

3. Service Composition Method Based on BAS Optimization Algorithm

3.1. BAS Algorithm Analysis. We transform beetle antenna search into an optimization problem in n-dimensional space, using x_l as left antenna coordinate, x_r as right antenna coordinate, x as mass center coordinate, and d as distance between two antennas. Since the beetle's head orientation is arbitrary, as well as a standardized random vector can be generated from beetle's right antenna pointing to its left antenna. The standardized random vector is shown in the following formula:

$$\overrightarrow{b} = \frac{\text{rands}(n,1)}{\|\text{rands}(n,1)\|}. \tag{14}$$

The generated random vector (beetle's right antenna pointing to its left antenna) is shown as follows:

$$x_l - x_r = d * \overrightarrow{b}. \tag{15}$$

At t moment, if position of beetle is x^t, the coordinates of left and right antenna are shown as follows:

$$x_l^t = x^t + \frac{d}{2} * \overrightarrow{b}, x_r^t = x^t - \frac{d}{2} * \overrightarrow{b}. \tag{16}$$

If odor function is f (x), the values of the left and right antennas are shown as follows:

$$f_{\text{left}} = f(x_l), f_{\text{right}} = f(x_r). \tag{17}$$

At $t-1$ moment, if $f_{\text{left}} > f_{\text{right}}$, then the beetle moves left, and beetle position in next moment is $x^t = x^{t-1} +$ step $* \overrightarrow{b}$. If $f_{\text{left}} < f_{\text{right}}$, then beetle moving right, and beetle position in next moment is $x^t = x^{t-1} -$ step $* \overrightarrow{b}$. According to this rule, we use the following formula to express beetle moving position in next moment:

$$x^t = x^{t-1} - \text{step} * \overrightarrow{b} * \text{sign}\left(f_{\text{left}} - f_{\text{right}}\right). \tag{18}$$

Considering that beetle search step size will decay with increase of time, and a single beetle easy to lose in search process, we use n beetles search measures to circumvent these problems. Randomly select n individuals with different step lengths to spatial search, and keep each beetle step length unchanged throughout search period, and step length of each beetle is determined by the size of beetle itself; then, step expression can be expressed as follows:

$$\text{step}_i = \text{Cd}_i, \ (i = 1, 2, 3, \ldots, n; C \text{ is a constant}). \tag{19}$$

The search path, by ith beetle produce is expressed as P_i, calculate and record value of QoSP$_i$. We can calculate the probability of beetle reaching end point ρ and the variance S^2 of evaluation function QoSP ; if $\rho > \rho_{\text{limit}}$ and $S^2 < S^2_{\text{limit}}$, the value of can be calculated by the following formula; otherwise, reexecute the algorithm:

$$\begin{aligned} \text{QoSP} &= \text{MinQoSP}_1, \text{QoSP}_2, \ldots, \text{QoSP}_n \\ &= \text{Min}\left(\varphi_1 T(P_i) + \varphi_2 C(P_i) + \frac{\varphi_3}{\text{Av}(P_i)} + \frac{\varphi_4}{\text{Rel}(P_i)} + \frac{\varphi_5}{\text{Rep}(P_i)}\right), \end{aligned} \tag{20}$$

where $i = 1, 2, 3, \ldots, n$, $\varphi_1 + \varphi_2 + \varphi_3 + \varphi_4 + \varphi_5 = 1$ and $P = P_i$ is the optimal path for beetle foraging, which is the optimal service composition when it is applied in composite edge service model.

3.2. Edge Service Composition Algorithm. According to the above algorithm analysis, the edge service composition specific workflow is as follows:

Step 1. Initialize algorithm configuration and parameters; C is the ratio of beetle step length to distance between two antennas; then, step$/d = C$.

Step 2. Place n beetles with different step lengths (randomly generated) at the starting position; let $i = 0, k = 0$, i represents ith $(i = 0, 1, 2, 3, \ldots, n - 1)$ beetle in this cycle, and record the initial position, k ($k \in (0, n]$) represents the number of beetles reach target position successfully.

Step 3. $i = i + 1$ and $i < n$, go to next step; otherwise, execute step 8.

Step 4. If beetle is lost, return to step 3; else, go to next step.

Step 5. Initialize $j = 0$, the number of cycles N (N is a constant).

Step 6. If $j < N$, then $\mathrm{QoS}(P_i) = \mathrm{QoSP}_i^j$, output and record $\mathrm{QoS}(P_i^j)$ and P_i^j, j++, repeat step 6; else, go to next step.

Step 7. Calculate QoSP_i variance S_i^2 as follows:

$$\begin{aligned} S_i^2 &= \frac{1}{N}\left\{\left[\mathrm{QoS}\left(P_i^0\right) - \overline{\mathrm{QoSP}_i}\right]^2 + \left[\mathrm{QoS}\left(P_i^1\right) - \overline{\mathrm{QoSP}_i}\right]^2 \right. \\ &\quad \left. + \cdots + \left[\mathrm{QoS}\left(P_i^j\right) - \overline{\mathrm{QoSP}_i}\right]^2\right\}\overline{\mathrm{QoSP}_i} \\ &= \frac{1}{N}\sum_{j=1}^{N}\left(\mathrm{QoS}\left(P_i^j\right)\right). \end{aligned} \tag{21}$$

If $S_i^2 < S_{\mathrm{limit}}^2$, then $\mathrm{QoS}(P_i) = \min(\mathrm{QoSP}_i^0, \mathrm{QoSP}_i^1, \ldots, \mathrm{QoSP}_i^N)$, return the value of QoSP_i and P_i ($P_i = P_i^j$) and record them, then $k = k + 1$, return step 3; else, execute step 5.

Step 8. Calculate the value of QoSP and path P as follows:

$$\begin{aligned} \mathrm{QoSP} &= \mathrm{Min}\mathrm{QoSP}_1, \mathrm{QoSP}_2, \ldots, \mathrm{QoSP}_n, \\ P &= P_i = P_i^j. \end{aligned} \tag{22}$$

Finally, return the value of QoSP and path P.

4. Experiment and Analysis

4.1. Simulation Experiment Environment and Solution Target. The type of experimental computer is HP880G1, ACPI ×64-based PC. Processor is Intel® Core™ i5-4590 CPU @ 3.30 GHz. Random access memory (RAM) is 4.0 GB. System type is Windows 8 64-bit operation system. Simulation software is MATLAB-R2018b. The objective function is expressed as follows:

$$\begin{aligned} \mathrm{FX}_i = \mathrm{QoSP} = \mathrm{Min}\Bigg(&\varphi_1 T(P_i) + \varphi_2 C(P_i) + \frac{\varphi_3}{\mathrm{Av}(P_i)} \\ &+ \frac{\varphi_4}{\mathrm{Rel}(P_i)} + \frac{\varphi_5}{\mathrm{Rep}(P_i)}\Bigg). \end{aligned} \tag{23}$$

Due to $\varphi_1 + \varphi_2 + \varphi_3 + \varphi_4 + \varphi_5 = 1$ in formula (13), here we assign the parameters as follows: $\varphi_1 = 0.2$, $\varphi_2 = 0.2$, $\varphi_3 = 0.2$, $\varphi_4 = 0.2$, and $\varphi_5 = 0.2$.

4.2. Global Optimization Ability of BAS Algorithm. Because a single beetle with variable step length is easy to get lost in random direction in optimization process, in this paper, we use several beetles (a group of individuals with different step lengths) to perform a global optimization test, and the step size of each beetle is fixed during optimization process, and different beetles have different step lengths. In this experiment, different quantity beetles were used to test optimization ability of algorithm. The experimental results show that more than 90% of beetle can reach global optimal position successfully with a small quantity beetles. With the quantity of beetles continuous increase, the stronger the global optimization ability, as shown in Figure 3(a). The optimization state diagram of beetles at time t and $T + 5$ is shown in Figure 3(b).

4.3. The Efficiency of Edge Service Composition Method. In the case of equivalent quantity edge services, the relationship between beetles quantity and time T is shown in Figure 4(a), and our method is significantly better than PSO, WPA, and other optimization algorithms compared with the time performance, as shown in Figure 4(b).

In the case of swarm scale (or beetle cycles), we experiment the use of different candidate edge service sets to solve the same service composition problem; the result shows that the larger we used candidate service set, the longer solution time-consuming is, and the faster the growth rate of time-consuming is. The experimental result is shown in Figure 5(a). We simulate service composition algorithms based on PSO and WPA under the same conditions, and comparison showed that the BAS edge service composition algorithm used in this paper was significantly better than PSO and WPA methods in solving service composition in time performance. The comparison results are shown in Figure 5(b).

4.4. Comparison between Edge Service Composition and Cloud Service Composition. This experiment builds and designs an experimental platform according to the cloud service composition model and edge service composition model (as shown in Figure 1). The platform deploys a cloud cluster and four edge servers on the edge of cloud cluster. The cloud cluster consists of a master node server and three slave node servers. Each node server or edge server is deployed in different virtual machine.

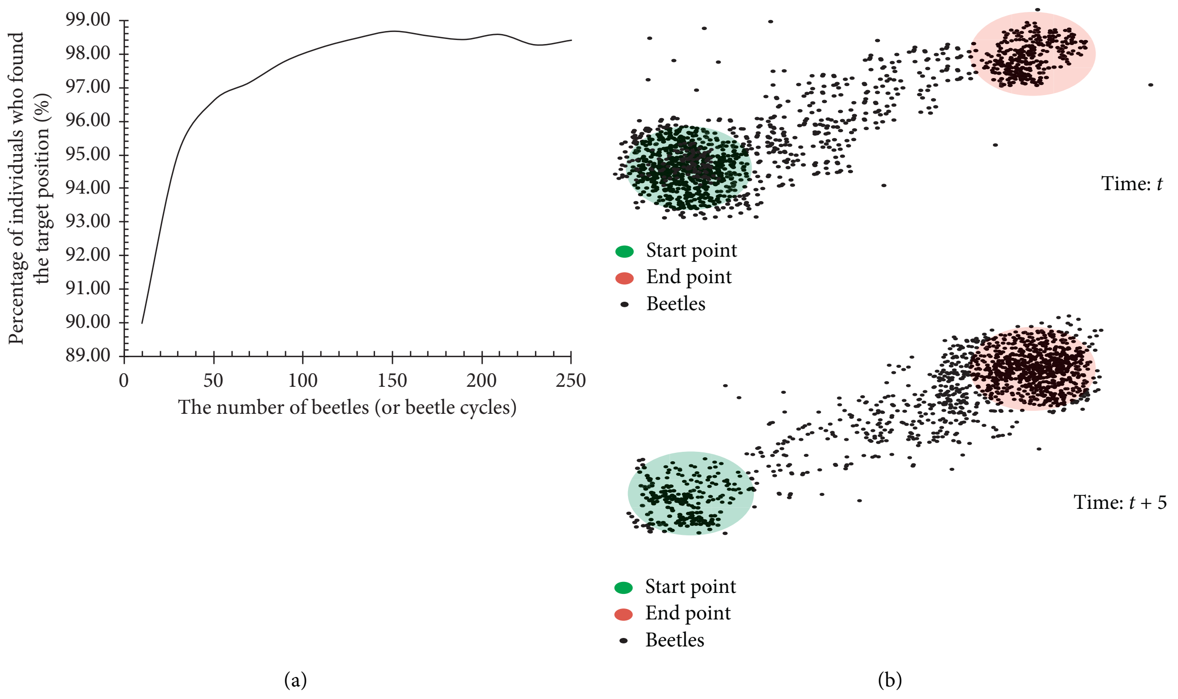


FIGURE 3: The optimization ability and state diagram of beetles. (a) The probability of reached target position. (b) State diagram of beetles at T and $T + 5$.

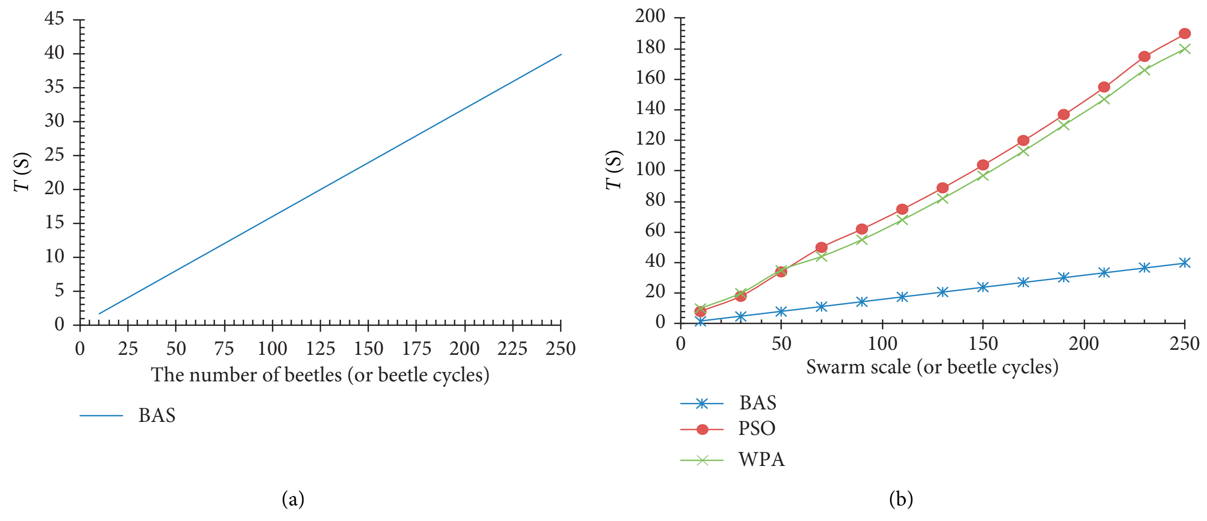
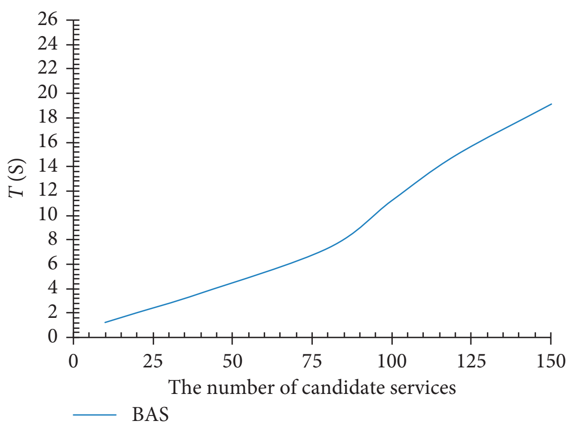


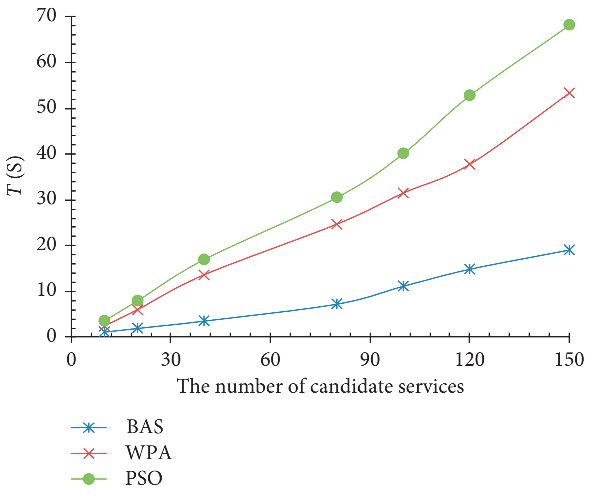
FIGURE 4: Edge service composition comparison of PSO, WPA, and BAS. (a) Relationship between time T and beetle cycles. (b) Comparison with PSO and WPA.

Experiment Description. The same service request is served by cloud service composition and edge service composition close to the service request, records and analyzes time performance from service request to complete service content.

The experiment results show that the closer service resource is to service request, the shorter response time to service request, the higher service efficiency, and the lower service time and cost. On the contrary, if service resource is in cloud far away from service request, the longer response time to service request, the lower service efficiency, and the service cost such as time and expense is higher relatively. As shown in Figure 6, it shows that the efficiency of edge service composition in providing services is significantly better than cloud service composition by time comparison between edge service composition and cloud service composition in providing service for service requests.



(a)



(b)

Figure 5: Relationship between edge service composition efficiency (time T and candidate service sets, comparison with other service composition methods (such as PSO and WPA)). (a) Relationship of time T and candidate service sets. (b) Comparison with PSO and WPA.

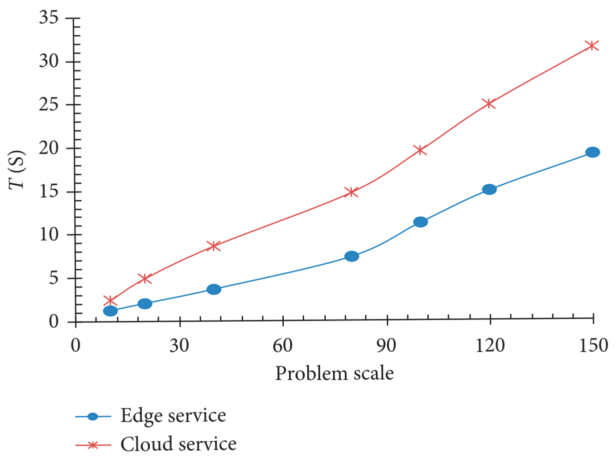


Figure 6: Comparison of edge service composition and cloud service composition.

5. Conclusion

In this paper, an edge service composition model based on edge computing is designed, and an optimization method for edge service composition based on beetle antenna search algorithm is proposed. Compared with cloud service composition model, the edge service composition model has advantage that a large amount of computing tasks is handed by edge service equipment that is closer to service request. It has fast data processing speed, no delay in transmission, fast response, and quality of user experience and satisfaction highly. Although edge service composition method based on BAS algorithm in this paper has obvious advantages in efficiency of service composition compared with the methods based on PSO and WPA, there are still some deficiencies and needs to be improved. BAS algorithm was first proposed by Jiang and Li in 2017 [24]; there are few cases for service composition, and it has many uncertainties and possibilities. We will combine particle swarm optimization, wolf swarm optimization, or ant colony algorithm to further improve edge service composition method based on BAS from the aspect of variable step length.

Data Availability

The experimental result data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This paper was supported by the National Natural Science Foundation of China (NSFC) under Grant no. 61572306.

References

[1] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality driven web services composition," in *Proceedings of the International Conference on World Wide Web*, pp. 411–421, Perth, Australia, April 2003.

[2] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for web services composition," *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 311–327, 2004.

[3] S. Deng, L. Huang, H. Wu, and Z. Wu, "Constraints-driven service composition in mobile cloud computing," in *Proceedings of the International Conference on Web Services*, pp. 228–235, San Francisco, CA, USA, July 2016.

[4] D. Ardagna and B. Pernici, "Global and local QoS constraints guarantee in web service selection," in *Proceedings of the International Conference on Web Services*, pp. 32–46, Sydney, Australia, July 2005.

[5] D. Ardagna and B. Pernici, "Adaptive service composition in flexible processes," *IEEE Transactions on Software Engineering*, vol. 33, no. 6, pp. 369–384, 2007.

[6] H. Wang, B. Zou, G. Guo, J. Zhang, and Z. Yang, "Optimal and effective web service composition with trust and user preference," in *Proceedings of the International Conference on Web Services*, pp. 329–336, New York, NY, USA, July 2015.

[7] R. Berbner, M. Spahn, N. Repp, O. Heckmann, and R. Steinmetz, "Heuristics for QoS-aware web service composition," in *Proceedings of the International Conference on Web Services*, pp. 72–82, Chicago, IL, USA, September 2006.

[8] I. Guidara, I. A. Jaouhari, and N. Guermouche, "Dynamic selection for service composition based on temporal and QoS constraints," in *Proceedings of the International Conference on Services Computing*, pp. 267–274, San Francisco, CA, USA, July 2016.

[9] T. Laleh, J. Paquet, S. Mokhov, and Y. Yan, "Constraint adaptation in web service composition," in *Proceedings of the International Conference on Services Computing*, pp. 156–163, Honolulu, HI, USA, June 2017.

[10] M. Alrifai, T. Risse, and W. Nejdl, "A hybrid approach for efficient web service composition with end-to-end QoS constraints," *ACM Transactions on the Web*, vol. 6, no. 2, pp. 1–31, 2012.

[11] S. Y. Hwang, E. P. Lim, C. H. Lee, and C. H. Chen, "Dynamic web service selection for reliable web service composition," *IEEE Transactions on Services Computing*, vol. 1, no. 2, pp. 104–116, 2009.

[12] J. E. Hadad, M. Manouvrier, and M. Rukoz, "TQoS: transactional and QoS-aware selection algorithm for automatic web service composition," *IEEE Transactions on Services Computing*, vol. 3, no. 1, pp. 73–85, 2010.

[13] Y. Chen, J. Huang, and C. Lin, "Partial selection: an efficient approach for QoS-aware web service composition," in *Proceedings of the International Conference on Web Services*, pp. 1–8, Anchorage, AK, USA, July 2014.

[14] Y. Chen, J. Huang, C. Lin, and J. Hu, "A partial selection methodology for efficient QoS-aware service composition," *IEEE Transactions on Services Computing*, vol. 8, no. 3, pp. 384–397, 2015.

[15] S. Deng, H. Wu, D. Hu, and J. L. Zhao, "Service selection for composition with QoS correlations," *IEEE Transactions on Services Computing*, vol. 9, no. 2, pp. 291–303, 2016.

[16] C. Lin, Y. Chen, and J.-W. Huang, "A survey on models and solutions of multi-objective optimization for QoS in services computing," *Chinese Journal of Computers*, vol. 38, no. 10, pp. 1907–1923, 2015.

[17] Y. Wang, I. R. Chen, J. H. Cho, A. Swami, and K. Chan, "Trust-based service composition and binding with multiple objective optimization in service-oriented mobile ad hoc networks," *IEEE Transactions on Services Computing*, vol. 10, no. 4, pp. 660–672, 2017.

[18] F. Wagner, A. Klein, B. Klopper, F. Ishikawa, and S. Honiden, "Multi-objective service composition with time- and input-dependent QoS," in *Proceedings of the International Conference on Web Services*, pp. 234–241, Honolulu, HI, USA, June 2012.

[19] M. Cremene, M. Suciu, D. Pallez, and D. Dumitrescu, "Comparative analysis of multi-objective evolutionary algorithms for QoS-aware web service composition," *Applied Soft Computing*, vol. 39, no. C, pp. 124–139, 2015.

[20] A. E. Yilmaz and P. Karagoz, "Improved genetic algorithm based approach for QoS aware web service composition," in *Proceedings of the International Conference on Web Services*, pp. 463–470, Anchorage, AK, USA, July 2014.

[21] M. Kim and I. Y. Ko, "An efficient resource allocation approach based on a genetic algorithm for composite services in IoT environments," in *Proceedings of the International Conference on Web Services*, pp. 543–550, New York, NY, USA, June 2015.

[22] M. S. Hossain, M. Moniruzzaman, G. Muhammad, A. Ghoneim, and A. Alamri, "Big data-driven service composition using parallel clustered particle swarm optimization in mobile environment," *IEEE Transactions on Services Computing*, vol. 9, no. 5, pp. 806–817, 2016.

[23] S. Peng, H. Wang, and Q. Yu, "Estimation of distribution with restricted Boltzmann machine for adaptive service composition," in *Proceedings of the International Conference on Web Services*, pp. 114–121, Honolulu, HI, USA, June 2017.

[24] X. Jiang and S. Li, "BAS: beetle antennae search algorithm for optimization problems," *International Journal of Robotics and Control*, vol. 1, no. 1, pp. 1–5, 2018.

[25] J. Yang, J. Wen, B. Jiang, Z. Lv, and A. K. Sangaiah, "Marine depth mapping algorithm based on the edge computing in internet of things," *Journal of Parallel and Distributed Computing*, vol. 114, pp. 95–103, 2018.

[26] S. Taherizadeh, A. C. Jones, I. Taylor, Z. Zhao, and V. Stankovski, "Monitoring self-adaptive applications within edge computing frameworks: a state-of-the-art review," *Journal of Systems and Software*, vol. 136, pp. 19–38, 2018.

[27] M. Ryden, K. Oh, A. Chandra, and J. Weissman, "Nebula: distributed edge cloud for data intensive computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3229–3242, 2017.

[28] S. Wan, Z. Gu, and Q. Ni, "Cognitive computing and wireless communications on the edge for healthcare service robots," *Computer Communications*, vol. 149, pp. 99–106, 2020.

[29] C. Chen, B. Liu, S. Wan, P. Qiao, and Q. Pei, "An edge traffic flow detection scheme based on deep learning in an intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1840–1852, 2020.

[30] C. Chen, Y. Zhang, Z. Wang, S. Wan, and Q. Pei, "Distributed computation offloading method based on deep reinforcement learning in ICV," *Applied Soft Computing*, vol. 103, Article ID 107108, 2021.