Received March 24, 2020, accepted April 7, 2020, date of publication April 10, 2020, date of current version April 23, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2987078

A Meta-Heuristic-Based Approach for Qos-Aware Service Composition

CHENYANG LI, JUN LI, AND HUILING CHEN
College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China

Corresponding author: Jun Li (omama@wzu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61402337, and in part by the Zhejiang Provincial Natural Science Foundations of China under Grant LQ13F020011.

ABSTRACT Recently, with the rapid increase in the number of web services, QoS-aware Web Service Composition(QWSC) has become a popular topic in both industry and academia. Meta-heuristic algorithm, as an effective way to solve classical optimization problems, has been successfully applied to QWSC nowdays. However, such approach has intrinsic drawbacks and usually lack of good performance in large-scale scenarios. For example, some meta-heuristic algorithms are suitable for continuous search space, while the search space of QWSC is discrete. For solving those problems which were commonly faced when applying meta-heuristic algorithm on QWSC, in this research, we firstly introduce a preprocessing approach for constructing fuzzy continuous neighborhood relations of concrete services, which makes the local search strategy of meta-heuristic algorithms be as effective in discrete space as in continuous space, thus improving the optimization performance. Second, we combine Harris Hawks Optimization (HHO) algorithm and logical chaotic sequence to propose an improved meta-heuristic algorithm named CHHO for solving QWSC. The ergodic and chaotic characteristics of chaotic sequences are used to enhance the ability of the CHHO to jump out of the local optimum for further optimization. Experimental results show that the CHHO has better optimization performance by comparing with the existing mainstream algorithms when solving QWSC problems. Additionally, the preprocessing approach not only greatly improves the optimization performance of the CHHO but also can be freely utilized in other meta-heuristics based approaches.

INDEX TERMS Meta-heuristic algorithm, QoS-aware web service composition, Harris hawks optimization, fuzzy continuous, logical chaotic sequence.

I. INTRODUCTION

Service Oriented Computing (SOC) is a computing method that uses services as the basic unit to rapidly build distributed software systems and enterprise applications through service composition technology [1]. SOC mainly organizes software applications into a series of interactive services through a Service-Oriented Architecture (SOA) [2]. SOA is a new distributed software system architecture proposed to solve service sharing, service reusing and business integration in the Internet environment. In the SOA software architecture, services can be published, discovered, bounded and invoked through interfaces in a standard format [3].

Web services solve the problems of heterogeneous distributed computing and code reuse through web services description language, simple object access protocol, universal description discovery and itegration, and extensible markup language technologies [4]. Web services have become a recognized mainstream technology for implementing service-oriented architecture (SOA), and flexible aggregation of resources through dynamic composition of services has become a natural way of thinking for technological development [5]. Due to the diversity and complexity of the real world and the dynamic nature of user requirements, to guarantee the Quality of Service (QoS) of web service composition becomes a crucial and significant challenge [6], [7]. QoS-aware Web Service Composition(QWSC) is one of the core technologies to solve this problem.

QWSC is an optimization problem with global constraints, and it is an Non-deterministic Polynomial(NP)-hard problem [8]. This characteristic makes QWSC problem difficult to solve, especially with the explosive growth of the number of web services in recent years. Although various approaches such as deterministic algorithms, heuristic

The associate editor coordinating the review of this manuscript and approving it for publication was Halil Yetgin.

algorithms, meta-heuristic algorithms, and hybrid methods have been proposed to solve this problem, there are still many problems in QWSC that need to be further studied and solved to meet the increasing requirements of users in a realistic and complex network environment [9], [10]. Therefore, a new approach for solving QWSC problem is required.

As one of the commonly used methods to solve optimization problems, meta-heuristic algorithms have been demonstrated that they can obtain approximate optimal solutions in a reasonable time without considering the mathematical properties and characteristics of the problem itself [11]. Early well-known meta-heuristic algorithms include, but are not limited to: Genetic Algorithms(GA) [12], Particle Swarm Optimization(PSO) [13], Differential Evolution(DE) [14], Ant Colony Optimization(ACO) [15], Artificial Bee Colony (ABC) [16]. According to No Free Lunch(NFL) theorem [17], with the continuous research of meta-heuristic algorithms, many excellent new algorithms have been proposed, such as Whale Optimization Algorithm (WOA) [18], Grasshopper Optimization Algorithm(GOA) [19], Slime Mould Algorithm(SMA) [20], Harris Hawks Optimization(HHO) [21].

Among them, HHO is the latest meta-heuristic algorithm proposed in 2019, due to its simple and easy to implement, less parameters to be tuned, and excellent performance, it has attracted widespread attention once it was proposed [22]–[25]. Therefore, in this paper, we firstly utilize HHO as the basis for solving QWSC. However, one of the potential problems faced by all meta-heuristics is the possibility of early convergence or falling into local minima [25], in practice, it is usually required that the algorithm needs to be modified to take full advantage of the unique characteristics of the optimization problem to overcome this potential threat. Therefore, how to design a robust and high performance optimization technique for solving QWSC problem based on the HHO algorithm is still a challenging task. Meanwhile, the original HHO is just suitable for solving continuous optimization problems [21], but QWSC is a discrete combination problem [8]. If this meta-heuristic algorithm designed based on continuous optimization problems is simply applied to QWSC problem without modification, the local search strategy of this algorithm cannot guarantee to find similar or better solution in the local scope of the current optimal solution, resulting in poor optimization performance [26].

To address the issues discussed above, we propose an improved meta-heuristic algorithm based on HHO and a preprocessing approach for constructing fuzzy continuous neighborhood relations of concrete services. The contributions of this paper are shown below:

- A novel preprocessing method is proposed to establish concrete services into fuzzy continuous neighborhood relations.
- A Logistic Chaotic Single-Dimensional Perturbation(LCSDP) strategy is proposed to enhance the algorithm's later exploration performance based on the characteristics of logistic chaos and QWSC.
- A novel Chaos Harris Hawk Optimization (CHHO) algorithm is proposed by combining the LCSDP strategy and the original HHO. The computational complexity of the CHHO algorithm is analyzed.
- We conduct extensive experiments to prove the effectiveness of the proposed algorithm and new preprocessing method.

The rest of the paper is structured as follows: Section II discusses related works of QWSC problem. Section III establishes the service composition model and construct fuzzy continuous neighborhood relationships of concrete services. Section IV details our proposed CHHO algorithm. Section V demonstrates and analyzes the experimental results. Section VI provides concluding remarks and highlights some future work.

II. RELATED WORKS

In this section, we discuss some of the existing literature on solving QWSC problem. Many researchers have conducted in-depth researches on QWSC and contributed a lot for solving the problem. The existing technologies are mainly divided into three categories [9]: deterministic algorithms, heuristic algorithms, and meta-heuristic algorithms. Among them, meta-heuristic algorithms are more popular [10].

A. DETERMINISTIC ALGORITHMS

In [27], the author proposed an integer linear programming model with penalty coefficients. In this model, violations in global constraints are allowed, but will be punished accordingly. When user constraints are hard, this method performs superior to standard methods. In [28], The author considered the execution time and throughput criteria of service composition and proposed a new service composition optimization algorithm based on directed graph structure. In [29], the author proposes a method for efficiently selecting and combining services by trimming non-skyline services based on QoS value identification of the skyline service. Nevertheless, these methods are deterministic algorithms. When the candidate service increases, its time and space consumption will increase exponentially. Their computations are inefficient when dealing with large-scale data structures.

B. HEURISTIC ALGORITHMS

Heuristic algorithms can obtain high-quality approximate solutions in a reasonable time under large-scale data. However, heuristic algorithms are generally designed based on the specific experience of optimization problems [10]. So there will be many restrictions on algorithm extension. In [30], the author proposed an efficient heuristic algorithm for different service composition structures, which can solve multidimension multichoice 0-1 knapsack models and multi constraint optimal path models. For a large number of candidate services, the heuristic algorithm can find a feasible choice of service composition problems in linear time. However, when the aggregation function of the objective function is nonlinear, or any global QoS constraint cannot be described in a linear form, the aggregation function and the QoS constraint cannot be linearized. More explanation of heuristic algorithm can refer to the following literature [9], [10], [31].

C. META-HEURISTIC ALGORITHMS

Meta-heuristics is also an approximation algorithm, which is an advanced strategy for exploring search space using different methods [32]. It does not pay special attention to the mathematical nature and special experience of optimization problems [11]. In [33], used Genetic Algorithm (GA) to solve the problem of web service composition and proved that GA can be used as an effective method for web service composition optimization. Furthermore, In [34], the author proposed a multi-objective GA, which can obtain acceptable solutions in a short time. However, in the worst case, the complexity of the GA is exponential, which is not suitable for large-scale service-oriented applications.

In [35], the author proposed a novel Eagle Strategy with Whale Optimization Algorithm (ESWOA) to solve the QWSC. ESWOA can better balance the exploitation and exploration stages, thereby speeding up convergence and avoiding premature convergence. Compared with other meta-heuristic algorithms, it can achieve global optimal solutions more seamlessly. In [36], proposed a modified Artificial Bee Colony(mABC) algorithm by incorporating chaotic-based opposition learning method and differential evolution strategies into the artificial bee colony. Compared with other meta-heuristic algorithms, mABC can find the feasible solution of QWSC faster and has a strong scalability. However, ESWOA and mABC did not pay attention to the problem that the discrete features of the QWSC problem cannot make the local search of the meta-heuristic algorithm maximize the performance. When confront with large-scale high-dimensional data, it is easy to fall into a local optimum and cannot continue to evolve.

In [37], the author proposed an enhanced Artificial Bee Colony(ABC) algorithm for solving the QWSC. This method improves the performance of ABC on QWSC problems through adaptive neighborhood selection and replacement strategies. So, this algorithm makes full use of the characteristics of the current optimal solution to generate the next solution at the later stage of the iteration. This method fully considers the characteristics of the QWSC problem and retains the historical optimal information, so it can find the best solution faster. However, as the dimension of the abstract service composition continues to increase, the role of historical optimal information decreases, and the algorithm can easily fall into a local optimal. In [26], proposed a definition for the optimal continuity concept. Meanwhile, three enhanced ABC algorithms are proposed, using different local search strategies to ensure that the ABC can be optimized in the discrete space of QWSC problem in a manner to searching in continuous space. This method can make full use of the local search strategy of ABC, and greatly enhance the optimization performance of ABC on QWSC. However, these strategies are customized according to the ABC algorithm. Hence, a new data preprocessing method needs to be designed to allow more meta-heuristic algorithms with the same properties to exert greater performance in discrete space, thereby promoting the further application of meta-heuristic algorithms on QWSC.

III. MODEL DESCRIPTION

In this section, we firstly model the QWSC problem as a mixed integer linear programming problem [38], then construct fuzzy continuous neighborhood relations of concrete services.

A. MODEL CONCEPTS

The key notations and the corresponding illustration of our model are listed in Table.1. Below are the concrete definitions.

TABLE 1. Model notations.

Notation	Illustration
S	A composite service
T_i	i-th task in S
C_i	Concrete services for i-th task
L	Global user requirements for S
n	Number of tasks in S
m	Number of concrete services per task
h	Number of quality attributes
$f_k(\cdot)$	Aggregate function of a composite service
$C_{i,j}$	Concrete service j in task i
$QC_{i,j}$	A vector of quality attributes for $C_{i,j}$
$UQos$	The value of the utility function

- Composite service is defined as $S = \{T_1, T_2, T_3, \ldots, T_n\}$, where T denotes a task and n is the number of tasks required to construct the composite service S.
- Abstract service. Task T_i represents the *ith* abstract service in the composite service S. Each abstract service represents a component in a composition service, which helps the composition service complete complex software functions.
- Concrete service is defined as $C = \{C_1, C_2, \ldots, C_n\}$, where $C_i = \{C_{i,1}, C_{i,2}, \ldots, C_{i,m}\}$ represents a set of concrete services of the abstract service T_i
- Quality of Service (QoS) of the *jth* concrete service of the *ith* abstract service is defined as $QC_{i,j} = \{q_{i,j,1}, q_{i,j,2}, \ldots, q_{i,j,h}\}$, where h represents the number of QoS attributes.
- Aggregate function is defined as $f_k(\cdot)$, where k is the *kth* QoS attribute of S. Different QoS attributes have different types of aggregation function based on different patterns of service composition such as parallel, sequence, loop, and conditional. Table.2 shows the aggregation function of four specific QoS attributes based on the sequence combination pattern. Because loop pattern, parallel pattern, and conditional pattern can all be transformed into sequential pattern in corresponding ways [39], so in this paper, we only consider sequential pattern.

TABLE 2. Aggregation function based on sequential model.

Attribute	Aggregation Function
Response Time	$q_{pr}(S) = \sum_{i=1}^{n}(q_{pr}, i)$
Throughput	$q_{th}(S) = min(q_{th}, i)$
Availability	$q_{av}(S) = \prod_{i=1}^{n}(q_{av}, i)$
Latency	$q_{la}(S) = \sum_{i=1}^{n}(q_{la}, i)$

- Global constraints is defined as a vector $L = (l_1, l_2, \ldots, l_h)$, where $l_r (1 \leq r \leq h)$ represents the user's global constraint on the corresponding QoS attribute.
- Utility function is defined as following:

$$UQos = \sum_{k=1}^{h} w_k f_k(\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij} q'_{ijk})$$

$$St. \sum_{k=1}^{h} w_k = 1, \quad x_{ij} \in \{0, 1\}$$

$$\sum_{j=1}^{m} x_{ij} = 1. \tag{1}$$

where $UQos$ represents the aggregation utility value of the composite service S. The higher the value of $UQos$ in the case of satisfying the global constraint, the better the composite service S is. w_k is the weight of the kth QoS attribute, and with $x_{ij} \in \{0, 1\}$ is the selecting variable. The q'_{ijk} represents the normalized QoS value, which are defined as follows.

$$q'_{ijk} = \begin{cases} \dfrac{q_{ijk} - q_{ijk}^{min}}{q_{ijk}^{max} - q_{ijk}^{min}} & \text{if } q_{ijk}^{max} - q_{ijk}^{min} \neq 0 \\ 1 & \text{if } q_{ijk}^{max} - q_{ijk}^{min} = 0 \end{cases} \tag{2}$$

$$q'_{ijk} = \begin{cases} \dfrac{q_{ijk}^{max} - q_{ijk}}{q_{ijk}^{max} - q_{ijk}^{min}} & \text{if } q_{ijk}^{max} - q_{ijk}^{min} \neq 0 \\ 1 & \text{if } q_{ijk}^{max} - q_{ijk}^{min} = 0 \end{cases} \tag{3}$$

For positive attributes (nomalized by equation.(2), e.g.., Availability), higher values represent better quality, whereas for negative attributes (nomalized by equation.(3), e.g.., Response time), lower values represent better quality.

B. PROBLEM FORMULATION

The goal of QWSC optimization is to find a set of abstract service combinations that satisfy the user's global constraints and maximize aggregate QoS performance. Therefore, we can model QWSC as mixed integer programming problems. The mathematical description is formulated as following:

$$Maximize\ csQos = 0.5UQos + 0.5(\frac{L}{h})$$
$$subject\ to\ L > 0, \quad L \in N. \tag{4}$$

where $UQos$ is the utility function calculated by equation.(1). h is the number of global constraints. L is the number of global constraints that the composite service S meets. The calculation formula of L is as follows:

$$f_k(\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij} q'_{ijk}) \leq l_k. \tag{5}$$

where l_k is the global constraints of the kth QoS attribute. The number of global constraints that the composite service satisfies can be calculated by equation.(5). Thereby an objective function with a penalty constraint is constructed.

C. INTEGER ENCODING AND FUZZY CONTINUOUS NEIGHBORHOOD RELATIONS

For solving QWSC, the concrete services of each abstract service can be stored in the form of integer encoding. For example, number #1 represents the first concrete service. Fig.1 shows an example of integer encoding for a composition service.

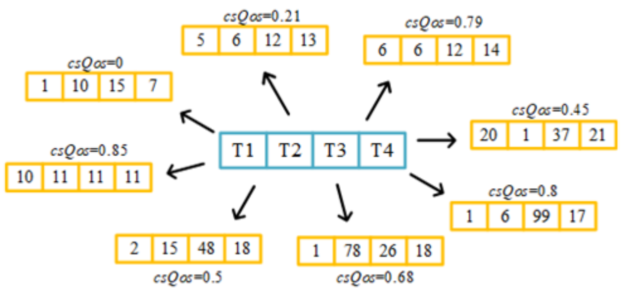


FIGURE 1. An example of integer encoding.

As shown in Fig.1, the number of abstract services ($\{T_1, T_2, T_3, T_4\}$) of the composition service is 4, and there are eight different combinations in the form of integer encoding. For example, if we select #1 concrete service for T_1, #10 concrete service for T_2, #15 concrete service for T_3 and #7 concrete service for T_4, then the value of $csQos$ is equal to 0, which indicates the composition service does not satisfy the global constraint.

For meta-heuristic algorithms, they are mainly to find the optimal solution through a combination of global and local search. Local search strateges performs local search near the location of the current optimal solution, which usually performs well on continuous functions but cannot guarantee the effectiveness while solving QWSC which is a discrete problem. This is because the QoS values of concrete services are independent to each other, even if the integer encoding is very close. For example, as shown in Fig.1, the [5, 6, 12, 13] code is close to the [6, 6, 12, 14] code, but the values of $csQos$ are quite different.

To address the problem discribed above, we use a clustering and sorting method to preprocess concrete services for establishing fuzzy continuous neighborhood relations between them. We call such preprocessing approach Fuzzy

Optimal Continuity Construction (FOCC). The specific steps are described as follows.

- Step 1: K-mean clustering algorithm [40] is used to group concrete services into classes based on the similarity of QoS values. The similarity is calculated by Manhattan distance [41], which is described below.

$$y = \sum_{i=1}^{h} \left| q_i^a - q_i^b \right| . \tag{6}$$

where q_i^a and q_i^b represent the *ith* QoS attribute of the concrete service a and b, respectively. h is the number of QoS attributes.

- Step 2: Clusters are sorted in ascending order by the *sumQoS* value of each cluster center. The *sumQoS* value is defined as following.

$$sumQoS_i = \sum_{j=1}^{h} q_j(i). \tag{7}$$

where $q_j(i)$ represents the *jth* QoS attribute of the *ith* cluster center. Larger integer encodings are assigned to the cluser with higher value of *sumQoS*.

- Step 3: Inside each cluster, each concrete service is sorted by its *sumQoS* value in ascending order.

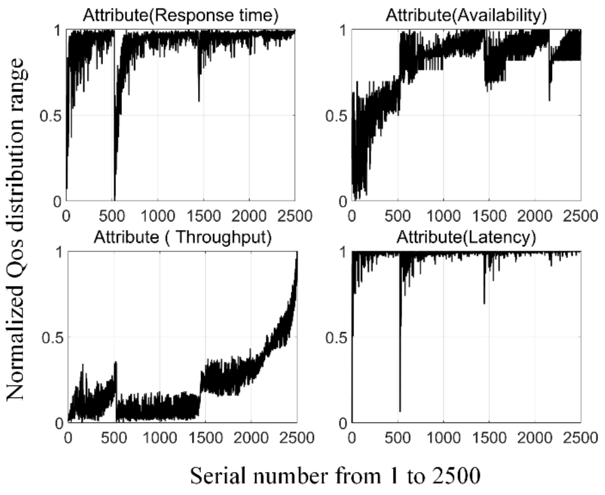


FIGURE 2. Distribution of QoS values after optimal continuity construction.

Fig.2 shows the distributions of four QoS attributes (response time, availability, throughput and latency) of 2500 concrete services randomly selected from the real world dataset QWS2.0 [42] after FOCC. The horizontal axis represents the encoding from 1 to 2500. For example, #1 represents the first concrete service. The vertical axis represents the normalized QoS value of each attribute.

From Fig.2, we can obviously observe that after preprocessing, as the serial number increasing, the QoS values of the corresponding service increase in general trend. Although this relation is not exactly accurate, it can provide a fuzzy standard to establish continuous neighborhood relations between concrete services, as a result, we can greatly improve the optimization accuracy and the convergence rate of meta-heuristic algorithms(demonstrated in SectionV).

IV. THE IMPROVED META-HEURISTIC ALGORITHM

In this section, we combined logical chaotic sequence with Harris Hawks Optimization (HHO) algorithm to propose a improved meta-heuristic algorithm named CHHO for improving the effectiveness while solving QWSC problem.

A. LOGISTIC CHAOTIC SINGLE-DIMENSIONAL PERTURBATION

Chaos [43] is a disordered state existing in nonlinear dynamical systems and is widely distributed in nature and social phenomena. Chaos has the characteristics of pseudo-randomness, ergodicity and sensitivity to initial values. Chaotic system can be divided into one-dimensional chaotic system and high-dimensional chaotic system. One of the very simple and widely studied one-dimensional chaotic dynamic system is logistic mapping [44], which can be defined as following:

$$C(t+1) = \mu C(t)(1 - C(t)). \tag{8}$$

where $C(t)$, $1 \leq t \leq (n-1)$ represents the $t-th$ chaotic sequence. $0 < \mu \leq 4$ is the parameter of the logistic chaotic sequence.

Fig.3 shows the effect of different values of μ on the chaotic distribution. We can observe that the closer μ is to 4, the stronger the chaos and ergodicity of the sequence is. In our model, to make better use of chaotic traversal features, we set $\mu = 4$, the initial value of chaos $C(1)$ to be a random value between (0,1) and $C(1) \neq 0.25, 0.5, 0.75, 1$.

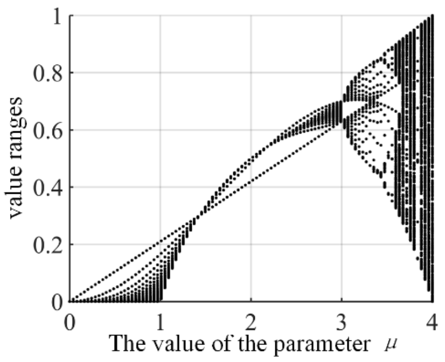


FIGURE 3. Influence of different values of parameter μ on logistic chaotic distribution.

Chaotic motion is non-repeating and has a higher search speed than random search [45]. Fig.4 shows a example of points(connected by lines) gernerated by one-dimensional Logistic chaotic sequence and uniformly distributed sequence with 2000 iterations, we can observe that the logistic chaotic sequence is more uniformly distributed in the range of (0,1) than the random distribution sequence. It can well traverse

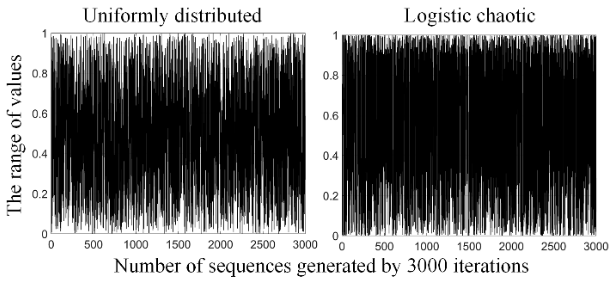


FIGURE 4. **Uniformly distributed and logistic chaotic sequences.**

all states in (0,1) space in sufficient time. Based on this characteristic, we can perform a single-dimensional chaotic traversal search on the location of the optimal solution. Better solutions can be found by traversing in a single dimension, while maintaining the most excellent dimensional information of the optimal solution. In this paper, we call such process Logistic Chaotic Single-Dimensional Perturbation (LCSDP) strategy. The LCSDP can be formulated as follows.

$$X(t+1) = X_{best}. \tag{9}$$

$$X(t+1)_i = lb_i + C(t)(ub_i - lb_i). \tag{10}$$

where lb_i and ub_i are the lower and upper bounds of the *ith* dimension, respectively. LCSDP firstly transfer the position of the current optimal solution to the new position $X(t+1)$ by equation.(9), then randomly select a dimension and let $X(t+1)$ use a logistic chaos to traverse the search space in a single dimension by using equation.(10).

LCSDP is specifically designed for QWSC problem. It can significantly enhance the exploration ability of the algorithm. On the one hand, most of the excellent dimensional information of the current optimal solution is retained in the service composition process, which ensures the convergence of the algorithm on the QWSC problem. On the other hand, this strategy uses Logistic chaos to continuously perturb a single dimension of the current optimal solution, thereby enhancing the algorithm's ability to jump out of the local optimum in the later iterations.

B. CHAOS HARRIS HAWK OPTIMIZATION ALGORITHM(CHHO)

Based on the original HHO algorithm, we combine LCSDP strategy with the HHO to propose an improved meta-huristic algorithm name Chaos Harris Hawk Optimizer (CHHO). Compared with the original HHO, the CHHO algorithm has two main changes. These changes are:

1) First, we deleted the two HHO's exploitation strategies which are soft besiege with progressive rapid dives and hard besiege with progressive rapid dives.
2) Second, we introduced the LCSDP strategy during the algorithm exploitation stage.

In this paper, we assume the location of the current optimal solution is a rabbit, and search agents are Harris hawks. Each Harris hawk adopts a different strategie to chase the rabbit according to the dynamic characteristics of different scenes and the rabbit escape pattern. The details of CHHO are described as follows. Algorithm.1 shows the detail pseudo code of the CHHO.

Algorithm 1 The Pseudo-Code of CHHO

```
1:  Initialize Harris hawks population $X_i (i = 1, 2, \ldots, N)$.
2:  Calculate the fitness of each Harris hawk.
3:  $X_{best}$ = rabbit(the current best solution).
4:  The population is divided into two equal sub-populations
    $S1$ and $S2$.
5:  While($t \leq$ Max number of iterations)
6:    For(Updated each Harris hawk position $(X_i)$)
7:    Update $E$, $r$ and $J$
8:      If1($|E| \geq 1$) //exploration stage
9:        If3($r \geq 0.5$)
10:         Use the first exploration strategy.
11:       Else If3($r \leq 0.5$)
12:         Use the second exploration strategy.
13:       End If3
14:     Else If1($|E| < 1$) //exploitation stage
15:       If2($X_i$ is a member of the subpopulation $S1$)
16:         Use LCSDP strategy to update position.
17:       Else If2($X_i$ is a member of the $S2$)
18:         If4($|E| \geq 0.5$)
19:           Use HHO's soft besiege strategy.
20:         Else If4($|E| < 0.5$)
21:           Use HHO's hard besiege strategy.
22:         End If4
23:       End If2
24:     End If1
25:   End for
26: Calculate the fitness of each Harris hawk.
27: Update $X_{best}$ if there is a better solution.
28: $t = t + 1$
29: End while
30: Return $X_{best}$
```

1) INITIALIZATION

N number of Harris hawks positions are randomly generated by using the following equation.

$$X_i = lb + \lfloor rand(0, 1)(ub - lb) \rfloor. \tag{11}$$

where N represents the number of Harris hawks populations. $rand(0, 1)$ generates a random number between (0,1). $\lfloor . \rfloor$ is a round-down operation. ub and lb are the upper and lower bounds of the search space, respectively. Line 2th and 3th of the Algorithm.1 are based on equation.(4) to calculate the $csQos$ value of each Harris hawk and the Harris hawk position with the best $csqos$ is chosen as the rabbit position.

2) CONTROLS ADAPTIVE PARAMETERS FOR EXPLOITATION AND EXPLORATION

Line 5th in Algorithm.1, CHHO enters iterative main program. The control parameter E of the 7th line in Algorithm.1

is used to adaptively adjust the CHHO exploration and exploitation stage. The mathematical description of E is shown below:

$$E = 2E_0 \left(1 - \frac{T}{t}\right). \tag{12}$$

where T represents the maximum number of iterations. t is the current number of iterations. E_0 is a random number between $(-1,1)$. Fig.5 shows the distribution of parameter E with 1000 iterations. From the Fig.5 we can observe that the parameter E gradually converges from -2 and 2 to 0 as the iteration increases. The parameter E reflects the movement state of the rabbit, and different movement states stimulate the Harris hawks to use different stages to update their position.

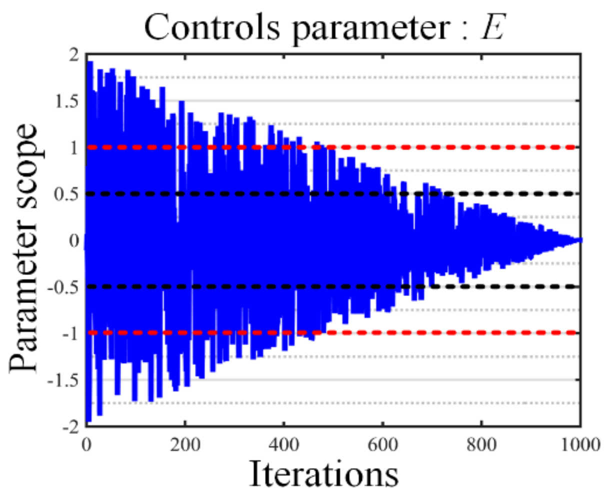


FIGURE 5. The fluctuation range of the control parameter *E* with iteration.

3) EXPLORATION STAGE

As shown in Fig.5, when the parameter E is outside the red line area, it means that the rabbit has not been locked, and the Harris hawk enters the exploration phase to search for the rabbit. From line 9th to 13th in Algorithm.1 are the exploration stages of the algorithm. The CHHO algorithm updates the positions by using two different exploration strategies.

The first strategy exploration strategy is co-evolution. The Harris hawk use its own position and the position of some other family member in the population to update the next position. The mathematical description is shown below.

$$X(t+1) = X_{rand} - \lfloor r_1 \left| X_{rand} - 2r_2 X(t) \right| \rfloor. \tag{13}$$

where X_{rand} is a randomly selected individual from the population. $X(t)$ is the current position of the Harris hawk. To increase the randomness and range of exploration, set r_1 and r_2 to random numbers between (0,1). $\lfloor . \rfloor$ is a round-down operation.

The second strategy is to update the current individual position based on the current rabbit position information, the average position information of the Harris hawks and random position information in the search space. Its mathematical description is as follows:

$$X(t+1) = \left\lfloor \left(X_{best} - \overline{X_m} \right) \right\rfloor - \lfloor r_1 \left(lb + r_2(\text{ub-l}b) \right] \tag{14}$$

where X_{best} is the position of the rabbit. $\overline{X_m}$ is the average position of the Harris hawks. r_1 and r_2 are random numbers between (0,1). ub and lb are the upper and lower bounds of the search space, respectively.

4) EXPLOITATION STAGE

Lines 15th to 23th in Algorithm.1 are the exploitation stages of CHHO. Compared to the original HHO algorithm, the LCSDP strategy of CHHO will be adopted during the exploitation stage. Meanwhile, for the web service composition problem, the exploitation stage will also be redesigned.

In the exploitation stage, the Harris hawk population is divided into two sub-populations of the same number. One of the sub-populations adopts the LCSDP strategy to improve the ability of CHHO in breaking away from the local optimum in the later iterations. Details are described in SectionIV-A.

Another part of the subpopulation adopts the following strategy to update the position.

Soft besiege strategy. As shown in Fig.5, When the parameter E appears in the area between the red and black lines, the rabbit still has a lot of energy. So, the Harris hawk used soft enveloping and approached the rabbit gently. The mathematical description is shown below.

$$\Delta X(t) = \lfloor X_{best} - X(t) \rfloor. \tag{15}$$

$$X(t+1) = \Delta X(t) - \lfloor E \left| J X_{best}(t) - X(t) \right| \rfloor. \tag{16}$$

where $X(t)$ is the current Harris hawk position. $X(t+1)$ is the position after $X(t)$ update. $\Delta X(t)$ is calculated by equation.(15). X_{best} is the current position of the rabbit. J is a random number between (0,2).

Hard besiege strategy. As shown in Fig.5, When the parameter E appears in the area between the black lines, the rabbit's energy starts to decrease.The Harris hawks found an opportunity to make a direct raid, calling it a hard besiege. The mathematical description is shown below:

$$X(t+1) = X_{best}(t) - \lfloor E \left| X_{best} - X(t) \right| \rfloor. \tag{17}$$

where X_{best} is the current rabbit position and $X(t)$ is the current Harris hawk position. $X(t+1)$ is the position of the Harris hawk to be updated next time.

5) COMPUTATIONAL COMPLEXITY

The proposed CHHO algorithm includes three main components: initialization, the updating of Harris hawks's position, and fitness evaluation. With population size P and maximum number of iterations T, the computational complexity of the initialization and the fitness evaluation step is $O(P)$ and $O(T \times P)$, respectively. In addition to, the computational complexity of the updating of Harris hawks's position is $O(T \times P \times D)$, where D represents the dimension of the problem. Therefore, the computational complexity of the CHHO is $O(P \times (T \times D + T + 1))$.

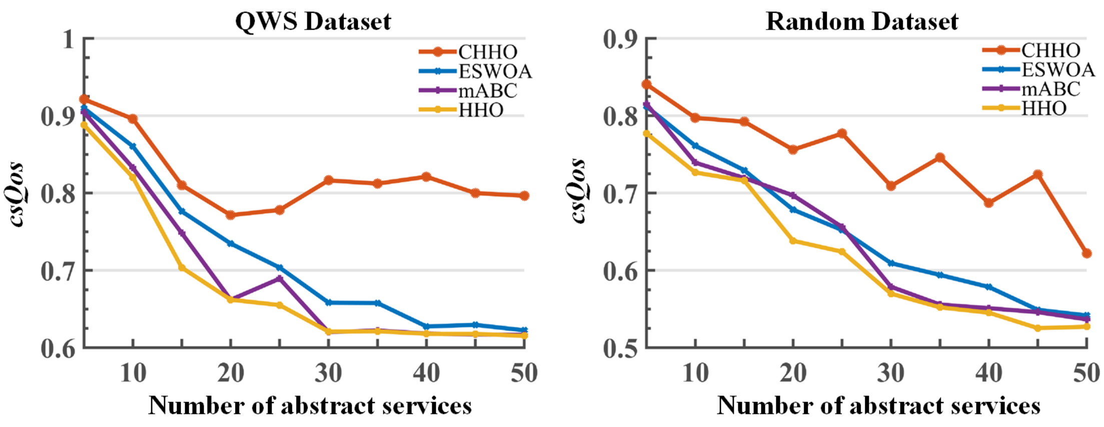


FIGURE 6. Comparison of different algorithms in different numbers of abstract services($M = 50$, N varies from 5 to 50).

V. EVALUATION

The experiments in this section mainly answer the following questions:

1) How does the proposed algorithm compare with other algorithms in solving QWSC problem?
2) How effective is our proposed FOCC method?
3) What is the effect of the new approach combining FOCC and CHHO?

A. EXPERIMENTAL SETUP

All experiments are performed on Windows Server 2012 R2 operating system, using Intel (R) Xeon (R) CPU E5-2660 v3 (2.60 GHz) and 8GB RAM. All algorithms are coded and run on MATLAB R2014b software. The meta-heuristic algorithms compared in this paper include: CHHO, HHO [21], ESWOA [35], mABC [36]. The population size of all algorithms is set to 30($P = 30$), and the maximum number of fitness function evaluation is set to 15000. Meanwhile, in order to reduce the impact of random noise, each algorithm run independently 30 times, and then the average result of these runs is obtained as the final result. On the other hand, in order to make a fair comparison, the initial parameters of all algorithms are set according to the original parameters of the corresponding paper. Table.3 shows the parameter values of these algorithms.

TABLE 3. Parameters setting for compared algorithms.

Algorithm	Parameters setting
CHHO	$\mu = 4, \lvert E\rvert \in (0, 2)$
HHO	$\beta = 1.5, \lvert E\rvert \in (0, 2)$
ESWOA	$P_e = 0.2$
mABC	$\mu = 4, K_{max} = 300, limit = \frac{T \times P}{2}$

The experiments are conducted on a real dataset QWS2.0 [42] containing 2507 samples and a standard normal distribution dataset containing 5000000 samples for ensuring impartiality. Four QoS attributes(response time, availability, throughput and latency) with different aggregate functions (shown in Table.2) are used in the evaluation.

We use the normalized *csQos* value which is defined in equation.(18) as the optimization criteria.

$$csQos' = \frac{csQos - csQos^{min}}{csQos^{max} - csQos^{min}}. \tag{18}$$

where $csQos^{max}$ is the largest value and $csQos^{min}$ is the smallest value of the *csQos*. Therefore, the closer *csQos* is to 1, the better the performance of the algorithm.

The global constraints are varied according to different test datasets. To be specifically, we firstly calculate the average QoS value of each attribute based on all concrete services in each abstract service, then obtain the global constraints by aggregating these values according to Table.2.

Additionally, Table.4 shows the notation and the corresponding description used in the experiments.

TABLE 4. Notations for the experiments.

Notation	Illustrate depict
N	Number of abstract services in composition
M	Number of concrete services per abstract service
D	Number of QoS attributes
$CT(s)$	Computation time for each algorithm
$csQos$	The optimal value of the objective function

B. COMPARISON OF DIFFERENT ALGORITHMS

In this section, we firstly compared the CHHO with ESWOA, mABC, and HHO without using the preprocessing method to verify the effectiveness of CHHO.

1) DIFFERENT NUMBER OF ABSTRACT SERVICES

Fig.6 shows the optimization performance of the algorithms mentioned above on both QWS2.0 dataset and random

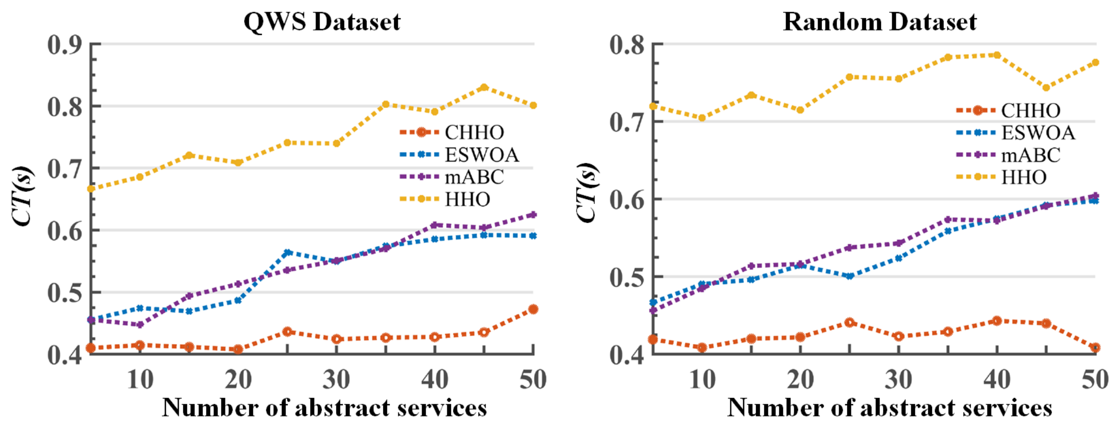


FIGURE 7. Computation time comparison on QWS dataset and random dataser($M = 50$, N varies from 5 to 50).

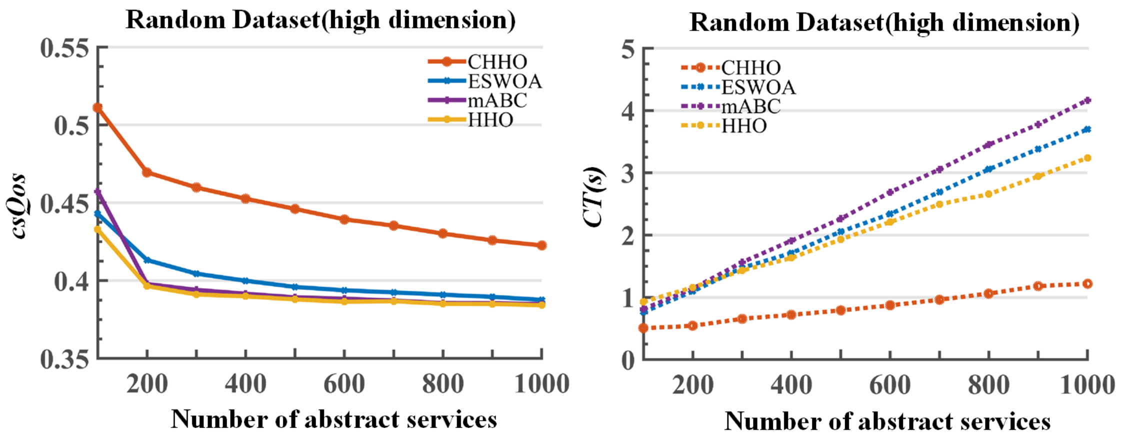


FIGURE 8. High-dimensional testing ($M = 100$, N varies from 100 to 1000) of different algorithms on random dataset.

dataset with N varing from 5 to 50, and M fixed to 50. We can obviously see that CHHO outperforms other algorithms (with the increasing of N, the optimal value obtained by other algorithms decline dramatically while CHHO still maintain a certain level). The results demonstrated that CHHO can significantly improve the optimization performance of meta-heuristic approaches especially in large-scale scenarios. This is because CHHO adopt the LSCDP strategy, which can continuously chaotic single-dimensional perturbation of the current optimal solution during the iteration process, thus jumping out of the local optimal and evolving to the global optimal. On the other hand, we can also observe that the optimization ability of the algorithms decreases with the increase of service scale. The reason is that with the exponential growth of the search space, it becomes more and more difficult for the algorithms to obtain the gloal optimal results.

In order to satisfy users' real time requirements, the computation time of the algorithms is another crucial fator we need to consider for the QWSC problems. Fig.7 shows the computation time of the algorithms according to different numbers of abstract services(varies from 5 to 50). We can observe that the computation time of the CHHO algorithm was greatly reduced compared to the original HHO algorithm. This is because we simplify the exploitation phase, and removes the parts of HHO that are not helpful for solving QWSC problem (see Algorithm.1). Moreover, we can still see that CHHO outperforms other algorithms in most of the cases.

To further verify the effectiveness of CHHO, Fig.8 shows a high-dimensional ($M = 100$, N varies from 100 to 1000) version of the optimization result and computation time. We can obviously see the similar results by comparing to Fig.6 and Fig.7. CHHO obtains better optimal results by comparing with other algorithms in all of the cases. For example, the optimal value of CHHO is 0.51 while other algorithms' values are below 0.46. On the other hand, CHHO spents least computation time in all of the cases and sustain a steady and slow growth with the increase of N by comparing with other approaches. This result demonstrate that, due to the effevtiveness of LCSDP strategy, CHHO still works relatively well in high-dimensional scenorios, even though its performance decreases with the increase of search space.

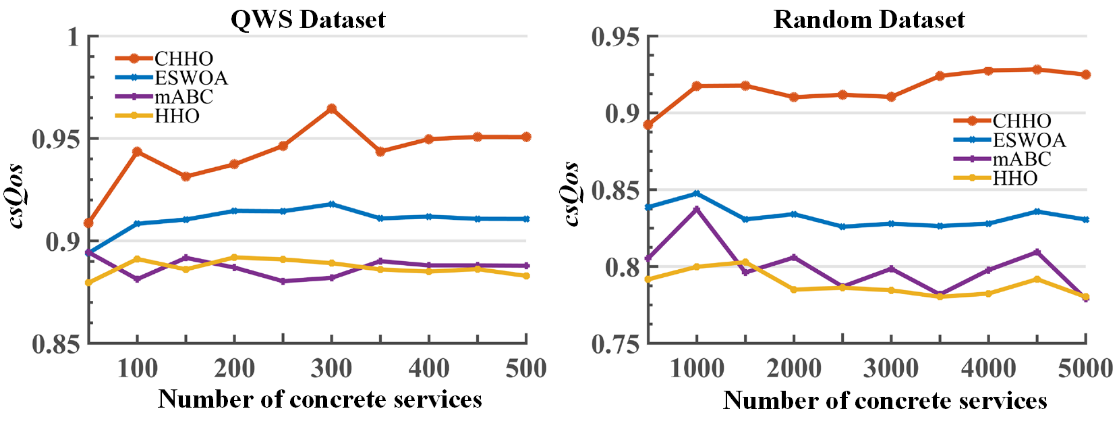


FIGURE 9. Comparison of algorithms in different numbers of concrete services under per abstract service. (*M* varies from 50 to 5000, *N* = 5).

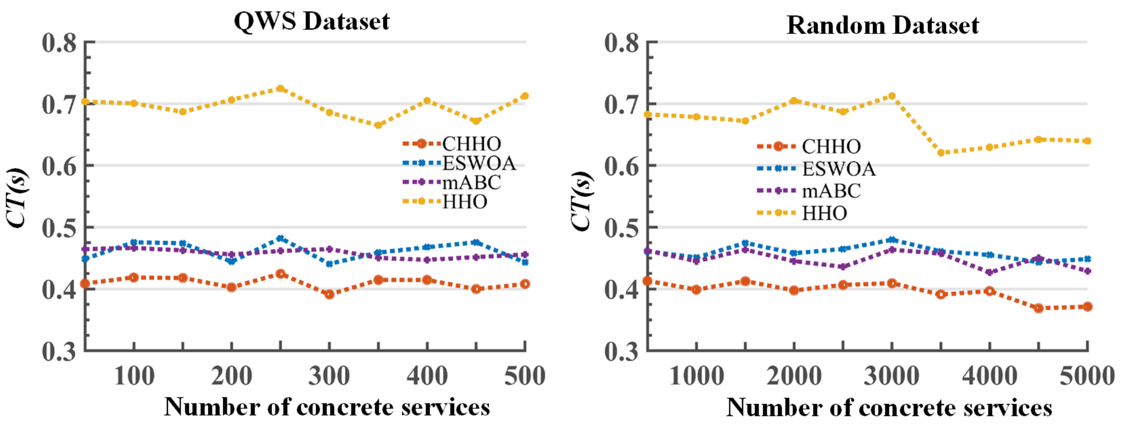


FIGURE 10. Computation time comparison on QWS dataset (*M* varies from 50 to 500, *N* = 5) and random dataser (*M* varies from 500 to 5000, *N* = 5).

TABLE 5. Comparison of algorithms under different number of QoS attributes.

Metric	Algorithm	$D=2$	$D=4$	$D=6$	$D=8$
$csQos$	CHHO	**0.968**	**0.923**↓	**0.891**↓	**0.874**↓
	ESWOA	0.877	0.842 ↓	0.823↓	0.815↓
	mABC	0.813	0.803↓	0.793↓	0.782↓
	HHO	0.817	0.803↓	0.791↓	0.786↓

2) DIFFERENT NUMBER OF CONCRETE SERVICES

In this setting, we fix the number of abstract services to 5 and vary M from 50 to 500 on the QWS dataset (due to the limited number of dataset) and vary M from 500 to 5000 on the random dataset (for testing large-scale scenario) to verify the effectiveness of CHHO.

From Fig.9, we can observe that CHHO maintains a optimal level of more than 0.91 on QWS dataset and more than 0.89 on random dataset in all of the cases. Similarly, although other algorithms perform worse than CHHO, they maintain a certain level of optimal value with the increase of M. This result demonstrate that the optimization performance of CHHO still outperforms other algorithms, and optimization performance of meta-heuristic algorithms is not significantly effected by M. This is because the distribution of concrete services in search space is disordered and the evolution times of meta- heuristic algorithm in this kind of unordered space are limited under certain number of fitness evaluation. Therefore, the number of concrete services will not have a great influence on the algorithm performance. Moreover, from Fig.10 we can observe that CHHO still has the least computation time in all of the cases. It is worth to mention that the computation time of these algorithms do not increase with the increase of M, which is a very useful characteristic of meta-heuristic algorithms when solving large-scale QSWC problem. The reason for this phenomenon is that meta-heuristic approach combines global search strategy and local search strategy, so as long as the number of fitness evaluation is fixed, the computation time will not change no matter how many M are in the search space.

TABLE 6. Comparison of the algorithms under different problem scales.

		CHHO		ESWOA		mABC		HHO	
N	M	$csQos$	$CT(s)$	$csQos$	$CT(s)$	$csQos$	$CT(s)$	$csQos$	$CT(s)$
10	50	0.792709	0.281028	0.771352	0.260561	0.737463	0.262590	0.723669	0.406234
	500	0.863422	**0.251318**	0.766245	0.291823	0.714425	0.298607	0.724080	0.400968
	5000	0.869658	**0.250061**	0.738386	0.304173	0.728054	0.262330	0.719557	0.400900
100	50	0.524644	0.263418	0.447769	0.379834	0.466763	0.383305	0.412681	0.497535
	500	0.545485	0.319343	0.458650	0.380284	0.513503	0.388794	0.452929	0.508188
	5000	0.556874	**0.267940**	0.462560	0.382767	0.504892	0.395373	0.458151	0.508520
1000	50	0.423593	0.658775	0.388798	1.413458	0.387433	1.605207	0.385108	1.430814
	500	0.422192	0.654355	0.386691	1.410386	0.383535	1.604522	0.383153	1.426340
	5000	0.422020	0.659617	0.383700	1.400270	0.382704	1.604350	0.382314	1.412181
		preCHHO		preESWOA		premABC		preHHO	
N	M	$csQos$	$CT(s)$	$csQos$	$CT(s)$	$csQos$	$CT(s)$	$csQos$	$CT(s)$
10	50	**0.806132**	0.289074	0.768372	0.261334	0.797728	**0.258017**	0.792934	0.394160
	500	**0.903948**	0.256998	0.880843	0.307546	0.901361	0.260200	0.897578	0.396434
	5000	**0.930000**	0.252304	0.904851	0.317575	0.926652	0.259243	0.910239	0.397486
100	50	**0.763135**	**0.258168**	0.743584	0.367513	0.588228	0.386553	0.745722	0.482776
	500	**0.814332**	**0.269263**	0.803383	0.375168	0.698974	0.397456	0.805180	0.495639
	5000	**0.841801**	0.270212	0.834364	0.380754	0.735885	0.401718	0.835640	0.507701
1000	50	**0.596884**	**0.634248**	0.587409	1.390842	0.411210	1.576964	0.590943	1.368921
	500	**0.794461**	**0.643471**	0.790440	1.391533	0.453513	1.589736	0.792202	1.392351
	5000	**0.825471**	**0.648198**	0.821631	1.392300	0.516719	1.589622	0.824140	1.402103

3) DIFFERENT NUMBER OF QOS ATTRIBUTE

For comprehensively evaluating the performance of CHHO, we consider different numbers of QoS attributes in this setting. We set $M = 500$, $N = 50$, and the value of $D \in \{2, 4, 6, 8\}$. in order to ensure fairness and facilitate extension, we generate random datasets with different D. Table.5 shows the results of optimization value. The symbol $\downarrow$ indicates that the optimization performance decrease with the increase of D. The best results with different size D are marked in bold font.

The results show that with the increase of D, the performance of each algorithm decrease. This is reasonable that algorithm becomes more and more difficult to find a composite service that meets all constraints when D is large. However, CHHO still performs better than other algorithms in all of the cases.

C. EFFECTIVENESS OF FOCC

In this section, to verify the effectiveness of FOCC, preCHHO, preHHO, preESWOA, premABC are compared to CHHO, HHO, ESWOA, mABC, respectively. We prefix the meta-heuristic algorithm with "pre" to represent the algorithm performs after FOCC (discussed in Section.III-C). For example, preCHHO represents CHHO performs after FOCC. The number of clustering category in our preprocessing method is set to 4.

Table.6 shows the optimization results of the above eight algorithms on random dataset according to different N and M. Excellent results are marked in bold font. From Table.6, we can see that all the algorithms with "pre" as prefix achieve better performance than the original ones in all of the cases. Especially, when $N = 1000$ and $M = 5000$, the optimal result obtained by preCHHO is 0.825471, which is greatly improved by comparing to CHHO whose result is 0.422020. These results are understandable because FOCC aggregates high-quality QoS services in advance to faciliate and speed up the local search of meta-heuristic algorithms. Moreover, FOCC does not filter any concrete service, which can avoid removing appropriate ones in advance. On the other hand, because FOCC can be done offline, it has no effect of the computation time of algorithms.

Therefore, we can conclude that FOCC can greatly improve the optimization performance of meta-heuristic algorithms(on only CHHO) without lossing computational complexity.

D. STATISTICAL ANALYSIS

In this section, we use Friedman test [46] and Wilcoxon signed rank test [47] for statistical analysis to ensure that our proposed methods are statistically significant. Table.7, Table.8 and Table.9 show the statistical analysis from different perspective, where p-value < 0.05 indicates that the two methods have significant differences, "$+$" indicates that the performance of the current method is better than the other one. Similarly, "$-$" / "$=$" indicate that the performance of current method is lower than / equal to the other one.

Table.7 shows the statistical analysis of the CHHO comparing with ESWOA, mABC, and HHO. We can see that the p-values of CHHO are extremely low in all of the cases(N varies from 10 to 1000, M varies from 50 to 5000), which

TABLE 7. Statistical analysis results of the proposed new algorithm (CHHO) and comparison algorithms.

		CHHO/ESWOA		CHHO/mABC		CHHO/HHO	
N	M	p-value	performance	p-value	performance	p-value	performance
10	50	2.37E-05	"+"	1.73E-06	"+"	1.73E-06	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
100	50	4.73E-06	"+"	2.61E-04	"+"	2.35E-06	"+"
	500	5.79E-05	"+"	4.72E-02	"="	1.15E-04	"+"
	5000	7.69E-06	"+"	6.16E-04	"+"	2.60E-05	"+"
1000	50	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"

TABLE 8. Statistical analysis results of the proposed new preprocessing method(FOCC).

		preCHHO/CHHO		preESWOA/ESWOA		premABC/mABC		preHHO/HHO	
N	M	p-value	performance	p-value	performance	p-value	performance	p-value	performance
10	50	2.16E-05	"+"	4.17E-01	"="	1.73E-06	"+"	1.73E-06	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
100	50	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
1000	50	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"

TABLE 9. Statistical analysis results of the new method combining new preprocessing method (FOCC) and new algorithm (CHHO).

		preCHHO/preESWOA		preCHHO/premABC		preCHHO/preHHO	
N	M	p-value	performance	p-value	performance	p-value	performance
10	50	3.84E-06	"+"	2.61E-04	"+"	1.60E-04	"+"
	500	7.24E-08	"+"	1.22E-04	"+"	1.67E-06	"+"
	5000	2.33E-06	"+"	3.59E-04	"+"	1.73E-06	"+"
100	50	3.11E-05	"+"	1.73E-06	"+"	3.11E-05	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	1.73E-06	"+"
1000	50	1.73E-06	"+"	1.73E-06	"+"	2.13E-06	"+"
	500	1.73E-06	"+"	1.73E-06	"+"	1.06E-04	"+"
	5000	1.73E-06	"+"	1.73E-06	"+"	2.41E-03	"+"

indicate that CHHO have significant differences to ESWOA, mABC, and HHO. Moreover, it can be observed from the ‘‘performance’’ index that the CHHO algorithm has the best performance among these algorithms. Statistically speaking, our proposed algorithm CHHO is highly competitive.

The results in Table.8 provide statistical analysis for whether FOCC is helpful for the meta-heuristic algorithms. We can observe that there is a significant difference between preCHHO and CHHO, and the performance of preCHHO is better. Similar results can also be observed between preESWOA and ESWOA, premABC and mABC. Therefore, from the perspective of statistical analysis, our preprocessing method FOCC can significantly improve the performance of meta-heuristic algorithms for solving QWSC problems.

Table.9 shows the statistical analysis of the preCHHO comparing with preESWOA, premABC, and preHHO. We can obviously see that our method preCHHO outperforms other algorithms in most of the cases.

Therefore, based on the results discussed above, we can conclude that our proposed approaches (CHHO and FOCC) are statistically significant.

VI. CONCLUSION AND FUTURE WORK

In this work, we have proposed an approximate optimization approach based on the HHO algorithm to address the

QoS-aware web service composition problem. We firstly proposed a preprocessing approach for concrete service datasets called FOCC, which makes the local search strategy of meta-heuristic algorithms be as effective in discrete space as in continuous space, thereby making some meta-heuristic algorithms designed for continuous problems suitable for solving QWSC(discrete problem). Second, we simplified the structure of the HHO based on the characteristics of the QWSC and combine the LSCDP strategy, a novel exploration strategy(committed to improving the ability of the algorithm to jump out of local optimization), to propose a novel algorithm CHHO. Experimental results based on real world and random datasets show that our approach(FOCC plus CHHO) has better performance by comparing with other mainstream methods when sovling QWSC problem, especially in large-scale scenarios. It is worth to mention that the FOCC approach can also be freely utilized to other meta-heuristic algorithms similar to CHHO.

However, the performance of CHHO is not good when QoS correlations [48] are exist between services. Qos correlation means that some QoS attributes of a service may depend on some QoS attributes of other services. In this way, the LCSDP strategy of the CHHO has to consider such correlation when performing single-dimensional disturbances, which leads to the decline of algorithm performance. Therefore, in the future work, we will consider the QoS correlations between services to further improve the proposed approach.

REFERENCES

[1] M. P. Papazoglou and D. Georgakopoulos, "Serviceoriented computing," *Commun. ACM*, vol. 46, no. 10, pp. 25–28, 2003.

[2] M. P. Papazoglou, "Service-oriented computing: Concepts, characteristics and directions," in *Proc. 7th Int. Conf. Properties Appl. Dielectric Mater.*, 2003, pp. 3–12.

[3] A. Arsanjani, "Service-oriented modeling and architecture," *IBM Developer Works*, vol. 1, p. 15, Nov. 2004.

[4] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the Web services Web: An introduction to SOAP, WSDL, and UDDI," *IEEE Internet Comput.*, vol. 6, no. 2, pp. 86–93, Mar. 2002.

[5] K. Gottschalk, S. Graham, H. Kreger, and J. Snell, "Introduction to Web services architecture," *IBM Syst. J.*, vol. 41, no. 2, pp. 170–177, 2002.

[6] J. Li and J. Lin, "Research on the influence of sampling methods for the accuracy of Web services QoS prediction," *IEEE Access*, vol. 7, pp. 39990–39999, 2019.

[7] J. Li and J. Lin, "A probability distribution detection based hybrid ensemble QoS prediction approach," *Inf. Sci.*, vol. 519, pp. 289–305, May 2020.

[8] X. Tang and J. Xu, "QoS-aware replica placement for content distribution," *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 10, pp. 921–932, Oct. 2005.

[9] C. Jatoth, G. R. Gangadharan, and R. Buyya, "Computational intelligence based QoS-aware Web service composition: A systematic literature review," *IEEE Trans. Services Comput.*, vol. 10, no. 3, pp. 475–492, May 2017.

[10] Q. She, X. Wei, G. Nie, and D. Chen, "QoS-aware cloud service composition: A systematic mapping study from the perspective of computational intelligence," *Expert Syst. Appl.*, vol. 138, Dec. 2019, Art. no. 112804.

[11] Z. Beheshti and S. M. H. Shamsuddin, "A review of population-based meta-heuristic algorithms," *Int. J. Adv. Soft Comput. Appl*, vol. 5, no. 1, pp. 1–35, 2013.

[12] L. Chambers, *The Practical Handbook of Genetic Algorithms: Applications*, 2nd ed. Boca Raton, FL, USA: CRC Press, 2000.

[13] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intell.*, vol. 1, no. 1, pp. 33–57, Jun. 2007.

[14] R. Storn and K. Price, "Differential evolution simple and efficient heuristic for global optimization over continuous spaces," *J. global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[15] M. Dorigo, M. Birattari, and T. Stützle, "Ant colony optimization: Artificial ants as a computational intelligence technique," *IEEE Comput. Intell. Mag.*, vol. 1, no. 4, pp. 28–39, Jan. 2006.

[16] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optim.*, vol. 39, no. 3, pp. 459–471, Oct. 2007.

[17] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[18] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.

[19] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017.

[20] S. Li *et al.*, "Slime mould algorithm: A new method for stochastic optimization," *Future Gener. Comput. Syst.*, 2020, doi: 10.1016/j.future.2020.03.055.

[21] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019.

[22] H. Chen, S. Jiao, M. Wang, A. A. Heidari, and X. Zhao, "Parameters identification of photovoltaic cells and modules using diversification-enriched harris hawks optimization with chaotic drifts," *J. Cleaner Prod.*, vol. 244, Jan. 2020, Art. no. 118778.

[23] H. Zhang, A. A. Heidari, M. Wang, L. Zhang, H. Chen, and C. Li, "Orthogonal nelder-mead moth flame method for parameters identification of photovoltaic modules," *Energy Convers. Manage.*, vol. 211, May 2020, Art. no. 112764.

[24] H. M. Ridha, A. A. Heidarib, M. Wang, and H. Chen, "Boosted mutation-based Harris hawks optimizer for parameters identification of single-diode solar cell models," *Energy Convers. Manage.*, vol. 209, Apr. 2020.

[25] S. Birogul, "Hybrid harris hawk optimization based on differential evolution (HHODE) algorithm for optimal power flow problem," *IEEE Access*, vol. 7, pp. 184468–184488, 2019.

[26] X. Wang, X. Xu, Q. Z. Sheng, Z. Wang, and L. Yao, "Novel artificial bee colony algorithms for QoS-aware service selection," *IEEE Trans. Services Comput.*, vol. 12, no. 2, pp. 247–261, Mar. 2019.

[27] S. Bharathan, C. Rajendran, and R. P. Sundarraj, "Penalty based mathematical models for Web service composition in a geo-distributed cloud environment," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jun. 2017, pp. 886–889.

[28] V. Gabrel, M. Manouvrier, K. Moreau, and C. Murat, "QoS-aware automatic syntactic service composition problem: Complexity and resolution," *Future Gener. Comput. Syst.*, vol. 80, pp. 311–321, Mar. 2018.

[29] M. Alrifai, D. Skoutas, and T. Risse, "Selecting skyline services for QoS-based Web service composition," in *Proc. 19th Int. Conf. World Wide Web (WWW)*. New York, NY, USA: ACM, 2010, pp. 11–20.

[30] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," *ACM Trans. Web*, vol. 1, no. 1, p. 6, May 2007.

[31] J. Li, X.-L. Zheng, S.-T. Chen, W.-W. Song, and D.-R. Chen, "An efficient and reliable approach for quality-of-service-aware service composition," *Inf. Sci.*, vol. 269, pp. 238–254, Jun. 2014.

[32] M. Gendreau and J.-Y. Potvin, "Metaheuristics in combinatorial optimization," *Ann. Oper. Res.*, vol. 140, no. 1, pp. 189–213, 2005.

[33] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani, "An approach for QoS-aware service composition based on genetic algorithms," in *Proc. Conf. Genetic Evol. Comput. (GECCO)*. New York, NY, USA: ACM, 2005, pp. 1069–1075.

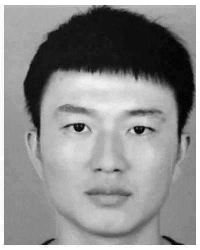
[34] H. Wada, J. Suzuki, Y. Yamano, and K. Oba, "E^3: A multiobjective optimization framework for SLA-aware service composition," *IEEE Trans. Services Comput.*, vol. 5, no. 3, pp. 358–372, 3rd Quart., 2011.

[35] S. K. Gavvala, C. Jatoth, G. R. Gangadharan, and R. Buyya, "QoS-aware cloud service composition using eagle strategy," *Future Gener. Comput. Syst.*, vol. 90, pp. 273–290, Jan. 2019.

[36] M. Chandra and R. Niyogi, "Web service selection using modified artificial bee colony algorithm," *IEEE Access*, vol. 7, pp. 88673–88684, 2019.

[37] F. Dahan, H. Mathkour, and M. Arafah, "Two-step artificial bee colony algorithm enhancement for QoS-aware Web service selection problem," *IEEE Access*, vol. 7, pp. 21787–21794, 2019.

[38] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for Web services composition," *IEEE Trans. Softw. Eng.*, vol. 30, no. 5, pp. 311–327, May 2004.
[39] J. Cardoso, A. Sheth, J. Miller, J. Arnold, and K. Kochut, "Quality of service for workflows and Web service processes," *J. Web Semantics*, vol. 1, no. 3, pp. 281–308, Apr. 2004.
[40] J. Yadav and M. Sharma, "A review of k-mean algorithm," *Int. J. Eng. Trends Technol.*, vol. 4, no. 7, pp. 2972–2976, 2013.
[41] A. Singh, A. Yadav, and A. Rana, "K-means with three different distance metrics," *Int. J. Comput. Appl.*, vol. 67, no. 10, pp. 13–17, 2013.
[42] E. Al-Masri and Q. H. Mahmoud, "QoS-based discovery and ranking of Web services," in *Proc. 16th Int. Conf. Comput. Commun. Netw.*, Aug. 2007, pp. 529–534.
[43] M. Tabor, *Chaos and Integrability in Nonlinear Dynamics: An Introduction*, vol. 3. New York, NY, USA: Wiley, 1989.
[44] C. Bonanno and G. Menconi, "Computational information for the logistic map at the chaos threshold," 2001, *arXiv:nlin/0102034*. [Online]. Available: https://arxiv.org/abs/nlin/0102034
[45] L. D. S. Coelho and V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1905–1913, Apr. 2008.
[46] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Amer. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, Dec. 1937.
[47] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, May 2010.
[48] S. Deng, H. Wu, D. Hu, and J. Leon Zhao, "Service selection for composition with QoS correlations," *IEEE Trans. Services Comput.*, vol. 9, no. 2, pp. 291–303, Mar. 2016.



CHENYANG LI is currently pursuing the degree in computer software and theory with Wenzhou University, China. His research interests include machine learning, evolutionary computation, and service computing.

JUN LI received the Ph.D. degree from the Department of Computer Science and Technology, Zhejiang University, China. He is currently an Associate Professor with the Department of Computer Science, Wenzhou University, China. He has published more than ten articles in international journals and conference proceedings, including *Information Sciences*, *Knowledge Based Systems*, the *International Journal of Web Service Research*, IEEE Access, and among others. His current research interests include service computing and big data mining.

HUILING CHEN received the Ph.D. degree from the Department of Computer Science and Technology, Jilin University, China. He is currently an Associate Professor with the College of Computer Science and Artificial Intelligence, Wenzhou University, China. He has published more than 100 articles in international journals and conference proceedings, including *Information Sciences*, *Pattern Recognition*, *Future Generation Computer System*, *Expert Systems with Applications*, *Knowledge-based Systems*, *Neurocomputing*, PAKDD, and among others. His current research interests include machine learning and data mining, as well as their applications to medical diagnosis and bankruptcy prediction. He is also a Reviewer for many journals, such as *Applied Soft Computing*, *Artificial Intelligence in Medicine*, *Knowledge-based Systems*, and *Future Generation Computer System*. He is currently serving as an Associate Editor for IEEE Access.