

Article An Improved Whale Optimization Algorithm for Web Service Composition

Fadl Dahan ^{1,2}

- ¹ Department of Management Information Systems, College of Business Administration—Hawtat Bani Tamim, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia; f.naji@psau.edu.sa
- ² Department of Computer Sciences, Faculty of Computing and Information Technology Alturbah, Taiz University, Taiz 9674, Yemen

Abstract: In the current circumstance, the Web Service Composition (WSC) was introduced to address complex user needs concerning the Quality of Services (QoS). In the WSC problem, the user needs are divided into a set of tasks. The corresponding web services are retrieved from the web services discovery according to the functionality of each task, and have different non-functional constraints, such as QoS. The WSC problem is a multi-objective optimization problem and is classified as an NP-hard problem. The whale optimization algorithm (WOA) is proven to solve complex multiobjective optimization problems, and it has the advantage of easy implementation with few control parameters. In this work, we contribute to improving the WOA algorithm, where different strategies are introduced to enhance its performance and address its shortcomings, namely its slow convergence speed, which produces low solution accuracy for the WSC problem. The proposed algorithm is named Improved Whale Optimization Algorithm (IWOA) and has three different strategies to enhance the performance of the WOA. Firstly, the Sine chaos theory is proposed to initiate the WOA's population and enhance the initialization diversity. Secondly, a Lévy flight mechanism is proposed to enhance the exploitation and exploration of WOA by maintaining the whales' diversity. Further, a neighborhood search mechanism is introduced to address the trade-off between exploration and exploitation searching mechanisms. Different experiments are conducted with datasets on 12 different scales (small, medium, and large), and the proposed algorithm is compared with standard WOA and five state-of-the-art swarm-based algorithms on 30 different independent runs. Furthermore, four evaluation criteria are used to validate the comparison: the average fitness value, best fitness values, standard deviation, and average execution time. The results show that the IWOA enhanced the WOA algorithm's performance, where it got the better average and best fitness values with a low variation on all datasets. However, it ranked second regarding average execution time after the WOA, and sometimes third after the WOA and OABC, which is reasonable because of the proposed strategies.

Keywords: web service composition; whale optimization algorithm; improved whale optimization algorithm

1. Introduction

The swarm-based algorithms are becoming more popular and widely used to solve different optimization problems using the concept of information sharing between search agents. They search for an optimal solution regarding the objective (fitness) function based on the trade-off between the searching mechanism (exploration and exploitation). Various proposed swarm-based algorithms were introduced to solve the web service composition (WSC) problem, such as the Bat Algorithm, Artificial Bee Colony, Ant Colony Optimization, Particle Swarm Optimization, Cuckoo Search, etc. The aim of the proposed improvements is to optimize convergence speed (execution time) and convergence rate (related to the fitness of the solutions) while searching for near-optimal solutions. In this work, we introduce an improvement for the whale optimization algorithm (WOA) [1] to address



Citation: Dahan, F. An Improved Whale Optimization Algorithm for Web Service Composition. *Axioms* 2022, 11, 725. https://doi.org/ 10.3390/axioms11120725

Academic Editor: Fevrier Valdez

Received: 2 November 2022 Accepted: 12 December 2022 Published: 13 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https:// creativecommons.org/licenses/by/ 4.0/). its shortcomings in terms of its slow convergence speed, which leads a low convergence rate (solution accuracy). Different strategies are introduced based on the Sine chaos theory, Lévy flight mechanism, and neighborhood search mechanism.

The service-oriented architecture aims to integrate Web Services (WSs) to accomplish complex user needs (business processes) by means of standard protocols in heterogeneous environments distributed systems. This architecture helps reuse the WSs through composition to accomplish a complex business process [2] with the reusability of its components. The utilization of service-oriented architecture is in rapid growth, which has led WS providers to publish as many services as possible with a similar function with different non-function constraints, such as Quality of Services (QoS). The WSC aims to satisfy user needs by constructing a WS combination of several existing services. This process could produce different compositions because of the availability of functionally similar WSs with different QoS associated; this process is called WSC.

The main challenge of WSC is the proliferation of WS providers that produce functionally similar WSs with different QoS constraints. The selection of the appropriate WSs to meet user needs with a specific level of QoS is a key process of the WSC problem. In reality, user needs are represented as a business process with different tasks, where each task includes a specific function of the user's needs. For each task, many WSs might satisfy their function with different QoS constraints; if we assume we have *n* tasks and m WSs for each task, then we have m^n possible solutions (WSs combination), where the WSC is proven to be NP-hard [3]. A representative model of the WSC problem is described in Figure 1. The four QoS constraints adopted in this work are Response Time (RT), Cost (C), Reliability (R), and Throughput (T). These QoS constraints can be aggregated based on the formulas illustrated in Table 1. In the table, *i* represents the *i*th task, *j* represents the *j*th web service in the same task, and *n* is the number of tasks.

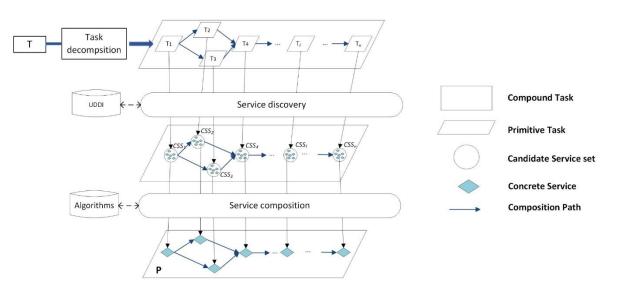


Figure 1. The WSC problem representative model.

Table 1. The QoS value	s aggregation formulas.
------------------------	-------------------------

QoS Criteria	Aggregation Formula
Cost (C)	$\sum_{i=1}^{n} C(ws_{ij})$
Response Time (RT)	$\sum_{i=1}^{n} RT(ws_{ij})$
Throughput (A)	$\prod_{i=1}^{n} T(ws_{ij})$
Reliability (R)	$\prod_{i=1}^{i-n} R(ws_{ij})$

Recently, the world's complexity and dynamicity has motivated the search for a better WS combination, because a simple and primitive web service is inadequate to meet user needs. Therefore, the demand for combinations of multiple WSs is increasing.

The paper is organized as follows. Related works are explained in Section 2. A brief description of WOA algorithms is presented in Section 3. Section 4 discusses the proposed algorithm, Improved Whale Optimization Algorithm (IWOA). Section 5 describes the experiment and analysis. Finally, the work's conclusion is presented in Section 6.

2. Related Work

The swarm-based algorithms are a group of optimization algorithms classified into nine groups [4] based on the inspiration approach. Considerable swarm-based approaches have been proposed to obtain a near-optimal solution for the WSC problem and web service combination. However, the main problem of most of these approaches is that of falling into a local optimum.

To our knowledge, few researchers used the WOA to address the WSC problem. Ju et al. [5] introduced hybrid methods that enhanced the performance of the WOA to overcome its slow convergence using chaos initialization, mutation, and nonlinear convergence factor methods. Jin et al. [6] proposed an enhancement for the WOA using two different approaches: the uniform mutation for the Eagle Strategy and a modified WOA. Their proposed algorithm is named the Modified Whale Optimization Algorithm (MWOA). The combination of these approaches is used to maintain a balance between the exploration and exploitation abilities of Eagle Strategy and WOA. Teng et al. [7] introduced an enhancement for the WOA using aggregation logarithmic and potential energy convergence factors. The proposed algorithm is named Logarithmic Energy Whale Optimization Algorithm (LEWOA). In addition, the proposed algorithm utilized the chaotic strategy to initialize the population. Ye et al. [8] improved the performance of the WOA by confirming the fitness function parameters and initializing the population using a tent map.

Other swarm-based algorithms have been introduced to solve the WSC problem in recent years, such as Artificial Bee Colony (ABC), Bat Algorithm (BA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Cuckoo Search (CS). The following paragraphs review the proposed method based on these swarm-based algorithms over three recent years.

In [9], the authors used fuzzy distance and ranking methods to introduce the fuzzy artificial bee colony. The idea of their proposed method is to maintain the diversity of artificial bees while searching for the best solution. Zhang et al. [10] applied the neighborhood search for the ABC to enhance the bees' searching mechanism, and the initialization diversity was improved using opposition learning. The neighborhood approach was also proposed in [11] to support the exploration and exploitation balancing in an artificial bee colony, where the farthest nodes were selected at the early stages, while near nodes were selected. Furthermore, this algorithm was improved on [12], where the swapping method was added as an additional step. The neighborhood selection was also enhanced by Seghir et al. [13] using the interval-based method.

The integrated probability was added by Arunachalam and Amuthan [14] with rulebased acceptance to improve the searching mechanism of the artificial bee colony. Both approaches introduced better exploration and exploitation balancing. The work of enhancing the exploration and exploitation was also introduced by Chandra and Niyogi [15], where a search procedure and differential evolution were applied to enhance the exploration and exploitation, respectively. The genetic algorithm was added to the ABC by Li et al. [16] to enhance the searching abilities of the ABC algorithm. The genetic algorithm with ABC was also used in [17]. Another hybrid algorithm was proposed [18] based on the ABC and Cuckoo Search, where the cuckoo agent enhanced the poorer bee.

A new algorithm based on the BA proposed by Dahan [19] uses the neighborhood search mechanism, cooperative population initialization, and the elitist mechanism. Kouicem et al. [20] proposed a novel BA based on the self-adaptive local search strat-

egy and Doppler effect compensation to enhance the convergence of BA and avoid the stagnation problem.

An ACO-based algorithm was proposed by Allali et al. [21], where the mobile agents were added to ACO to improve the searching mechanism. The genetic algorithm with ACO was proposed by Wang [22]. Dahan et al. [23] proposed a neighboring selection process and a multi-pheromone system to improve the searching mechanism of the ACO. Dahan [24] proposed a multi-agent based on ACO.

Wang et al. [25] used prior knowledge to improve the performance of the PSO algorithm. Shirvani [26] proposed a novel PSO algorithm where the PSO's parameters were tuned using the elapsed time to achieve better exploration and exploitation balancing. The genetic algorithm with PSO was proposed by Dogani and Khunjush [27], where the PSO exploration and exploitation were enhanced based on the genetic algorithm.

The genetic algorithm with CS was proposed by Subbulakshmi et al. [28]. The distributed network with CS was proposed by Ghobaei-Arani et al. [29]. Kouchi and Nacer [30] proposed AN enhancement of CS to solve the WSC problem.

The aforementioned swarm-based algorithms have a limitation in guaranteeing the best web services path because of the stochastic behavior of the swarm-based algorithms. In addition, the No-Lunch-Free theorem (NLF) [31] states that the optimizers cannot find enough to address all optimization problems. Therefore, the abovementioned algorithms cannot efficiently solve large-scale datasets and suffer from degraded performance. This discussion contributes to the work of developing new optimization algorithms to address the WSC by introducing better evaluation factors for optimization algorithms, namely efficient performance and execution time [31].

3. Whale Optimization Algorithm (WOA)

WOA is a swarm-based algorithm proposed in 2016 by Mirjalili and Lewis [1]. It mimics the humpback whales' hunting behavior and consists of three phases: encircling prey, bubble-net attacking, and searching randomly for prey. In WOA, the first two phases represent the exploitation mechanism, and the last phase represents exploration.

In the encircling prey phase, the individual considers the current best solution as target prey, and the other individuals update their positions to move closer to it. The following equations mathematically represent this encircling prey behavior:

$$\vec{D} = |\vec{C}.\vec{X^*}(t) - \vec{X}(t)|$$
(1)

$$\vec{X}(t+1) = \vec{X^*}(t) - \vec{A}.\vec{D}$$
⁽²⁾

where . is the element-by-element multiplication, $| \cdot |$ is the absolute value, *t* is the iteration number, $\vec{X^*}$ is the best solution vector, \vec{C} and \vec{A} are the coefficient vectors, and \vec{X} represents the current position vector that is updated in each iteration.

The coefficient vector A and C are calculated as follows:

$$\overrightarrow{A} = 2\overrightarrow{a}.\overrightarrow{r} - \overrightarrow{a}$$
(3)

$$\overrightarrow{C} = 2.\overrightarrow{r} \tag{4}$$

where \vec{r} is a random vector in [0, 1], and \vec{a} is decreased linearly from 2 to 0.

Two hunting strategies are described in the bubble-net attacking phase to model the bubble-net attacking behaviors, as follows.

In the shrink encircling strategy, the whales update their position based on Equation (3), where the value of \vec{a} is decreased linearly from 2 to -2, and \vec{A} is fluctuated and decreased in the interval [-a, a] randomly, and the value of a is decreased linearly from 2 to 0 with iteration. As a result, each whale updates its position according to the agent's current best and original position.

In the spiral updating position strategy, the whale calculates the distance of the prey position at (X^*, Y^*) and its position at (X, Y). Then, the humpback whale's helix-shaped movement is mimicked between the position of the whale and prey using a spiral equation as follows:

$$\vec{X}(t+1) = \vec{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X^*}(t)$$
 (5)

where $D' = |\vec{X}(t) - \vec{X}(t)|$ represents the distance between the prey position and *i*th whale, *l* is a random value between [-1, 1], and *b* is a constant that defines the spiral equation shape.

In searching randomly for a prey phase, a different variation of A is defined and utilized to search for prey. The value of \overline{A} is randomly generated in the interval [-1, 1] to encourage the whales to move far from the reference whale and support the search for space exploration. In the stages mentioned above, the whales updated their position based on the best search agent found, while in this phase, the position was updated based on randomly selected whales.

$$\vec{D} = |\vec{C}.\vec{X_{rand}} - \vec{X}|$$
(6)

$$\vec{X}(t+1) = \vec{X_{rand}} - \vec{A}.\vec{D}$$
(7)

where X_{rand} denotes the current population random position vector.

4. Improved Whale Optimization Algorithm (IWOA)

The following sections present the main contributions of this paper to improve an optimization algorithm based on WOA to solve the WSC problem.

4.1. Sine Mapping for Initialization

Population initialization diversity can greatly affect swarm-based algorithm performance and convergence speed [32]. The randomized method for initializing the population cannot guarantee these aspects. Sine mapping is a one-dimensional chaotic mapping system [33], and different maps can be used in optimization algorithms to generate chaotic numbers listed in [34]. The Sine map expression is represented as follows:

$$X_{n+1} = \sin\left(\frac{2}{X_n}\right), \ n = 0, 1, \dots, N \tag{8}$$

where the X_n represents the current solution and cannot be 0.

The Sine mapping formula was updated in this work to adopt the WSC problem. The proposed Sine mapping formula is shown as follows:

$$X_{i+1,j+1}^{w}(t+1) = \begin{cases} \left(\sin\left(\frac{2}{X_{i,j}}\right) * m\right) & if \ 0 < \left(\sin\left(\frac{2}{X_{i,j}}\right) * m\right) \le m \\ Randomised \ method \qquad Otherwise \end{cases}$$
(9)

where *w* represents the *w*th whale, *t* represents the iteration number, *i* represents the *i*th task, and *j* represents the *j*th web service in the same task. *m* is the number of WSs in each task.

The proposed Sine mapping formula accepts this value if the sin equation produces an index for WSs between 0 and *m*. Otherwise, the randomized method will be used to initiate the current position.

4.2. Lévy Flight Mechanism

Biologists have found that the Lévy flight is a preferred foraging strategy for many organisms [32]. Therefore, the Lévy flight has been introduced in many heuristic algorithms to achieve random search problems with better performance. It helps heuristic algorithms

to avoid the stagnation problem and increase the diversity by alternating the high-frequency short-range and low-frequency long-range exploration.

Lévy flight is a type of random walking strategy where the walking distribution is a power function distribution representing the heavy-tailed features probability distribution. The mathematical representation of Lévy flight is as follows:

$$L(s,\gamma,\mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}} \exp\left[-\frac{\gamma}{2(s-\mu)}\right] \frac{1}{(s-\mu)^{\frac{3}{3}}}, & 0 < \mu < s < \infty\\ 0 & otherwise \end{cases}$$
(10)

where *s* represents the step size, μ represents the minimum step size, γ represents the scale parameter.

When $s \rightarrow \infty$ the Equation (10) can be written as:

$$L(s,\gamma,\mu) \approx \sqrt{\frac{\gamma}{2\pi}} \frac{1}{s^{\frac{3}{3}}},\tag{11}$$

The value of the Lévy flight step size can be calculated using the following:

$$s = \frac{u}{|v|^{1/\beta}} \tag{12}$$

where β is a random number between 0 and 2, and *u* and *v* are normally distributed random numbers that are defined as:

$$\begin{cases} u \sim N(0, \sigma_u^2) \\ v \sim N(0, \sigma_v^2) \end{cases}$$
(13)

where σ_u and σ_v are defined as:

$$\begin{cases} \sigma_u = \left\{ \frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma[(1+\beta)/2]\beta^{2(\beta-1)/2}} \right\}^{\frac{1}{\beta}} \\ \sigma_v = 1 \end{cases}$$
(14)

where Γ is the gamma function.

In this work, the Lévy-flight model is added for WOA to update the whales' locations and maximize the efficiency of searching the targets.

$$X_{i+1,j+1}^{w}(t+1) = X_{i,j}^{w}(t) + (2r-1) \oplus (X_{i,j}^{w}(t) + r \oplus s)$$
(15)

where *r* denotes a random number between 0 and 1, \oplus is the element's dot product, *w* represents the *w*th whale, *t* represents the iteration number, *i* represents the *i*th task, and *j* represents the *j*th web service in the same task.

The Lévy-flight step with random search in different ranges helps the WOA to jump out of the stagnation problem. At the same time, it guarantees a good balance between the exploration and exploitation mechanisms to improve the proposed algorithm's performance.

4.3. Neighborhood Search Strategy

This strategy introduces a variant of the neighborhood search proposed in our previous works [18,19]. The proposed adaptive search balanced the diversity and concentration of swarm-based algorithms [12]. At the same time, it is an effective strategy to jump out of the local optima [35]. In [18,19], the neighborhood strategy searched around the best WSs neighborhood. However, this process increases the time complexity according to the neighborhood WS number. In IWOA, this process is enhanced to overcome the time shortcoming, as follows:

First: in each iteration, the quality of the local best solution is checked for improvement; if the solution quality is improved, then the neighborhood search is started; otherwise, the IWOA process continues.

Second: a random number (*RN*) is enhanced, where $1 \le RN \le W - 1$ (*W* is the population size) of neighborhood whales.

Third: for each neighborhood whale, a task is selected randomly, and its WSs are mutated in the best iteration solution.

5. Experiments and Comparative Analysis

The IWOA algorithm is compared with the standard WOA and the state-of-theart swarm-based algorithms to evaluate the performance effectiveness and verify the superiority of the IWOA using several experimental aspects. The following subsections describe the settings and experimental results of the experiments.

5.1. Experimental Settings

In this work, two different groups of datasets were utilized to validate the performance competitiveness and verify the superiority of IWOA compared to the competitors. The first group was generated from real-world datasets called QWS 2.0 [36]. QWS 2.0 is a well-known dataset comprising 2507 WSs with real measurements for their QoS. The first group is classified as small-size datasets keeping a constant number of task sizes and a varied number of WSs in a total of 23,000 WSs selected randomly with replacement. The second group was generated randomly based on the tool introduced in [37], and the values of QoS constraints based on the tool were generated between 1 and 1000. The second group is classified as medium-size and large-size datasets, keeping a constant number of WSs and various task services in a total of 52,000 WSs. Table 2 depicts the datasets describing the number of tasks and WSs/tasks. It should be remembered that the four QoS constraints adopted in this work are Response Time (*RT*), Cost (*C*), Reliability (*R*), and Throughput (*T*), where the fitness of each solution according to these constraints is mathematically calculated using Equation (16).

$$F_{i} = \left(\prod_{j=1}^{n} T_{jb} + \prod_{j=1}^{n} R_{jb} - \sum_{j=1}^{n} C_{jb} - \sum_{j=1}^{n} RT_{jb}\right)$$
(16)

where F_i represents the solution fitness of *i*th whales, *n* represents the task number, and *b* denotes the web services.

Dataset	Dataset Size		No. WSs/Task	
DS1		10	100	
DS2	0 11	10	400	
DS3	Small	10	800	
DS4		10	1000	
DS5		30	100	
DS6	M. P.	40	100	
DS7	Medium	50	100	
DS8		60	100	
DS9		70	100	
DS10	Lanas	80	100	
DS11	Large	90	100	
DS12	DS12	100	100	

Table 2. Dataset description.

To ensure unbiased and fair experiments, the Java working platform was the experimental environment of all algorithms, with the system configuration of Intel(R) Core(TM) i7-7500U CPU @ 2.70 GHz and 8.0 GB RAM of memory with Windows 10. The performance competitiveness of the IWOA was compared with standard WOA and five of the state-of-the-art swarm-based algorithms (OABC [10], SABC [12], MWOA [6], LEWOA [7], and ABC_CS [18]). To ensure an unbiased experiment level, the parameter setting of IWOA was set as WOA, while OABC, SABC, LEWOA, MWOA, and ABC_CS were set based on their work preferences; however, the common parameters of all algorithms were set as the population size (P) to 100, and the maximum iterations (Z) to 500. Furthermore, the evaluation metrics were the average fitness value, best fitness values, standard deviation, and average execution time for each algorithm, with 30 times independently run on each dataset. The average fitness value was counted in each run, and the execution time for finding the best fitness values was recorded.

5.2. Experimental Results and Analysis

The results of the IWOA compared to its competitors are shown in Figures 2–4 and in Table 3. In the table, the bolded values are the best performance values. In Figures 2–4, the notched boxplots represent the distribution of the average fitness value obtained by all algorithms in 30 independent runs. In contrast, the best fitness values, standard deviation, and average execution time of all algorithms on the 12 datasets are shown in Table 2.

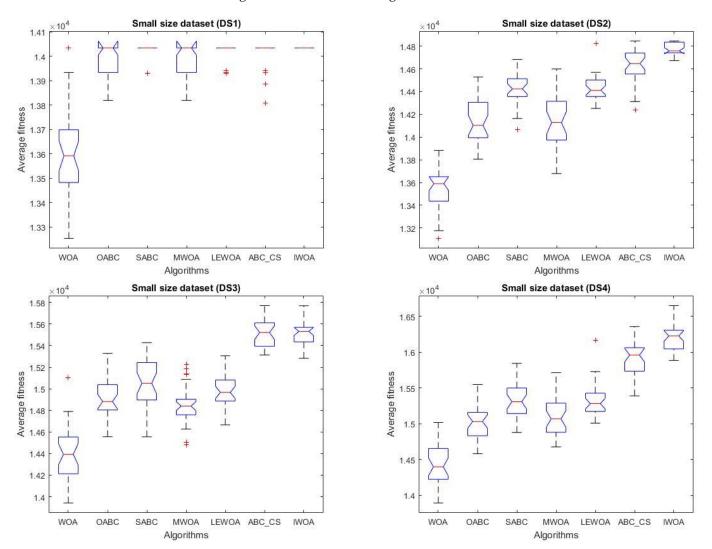


Figure 2. Boxplots for the average fitness value obtained using the compared algorithms for the small-sized datasets.

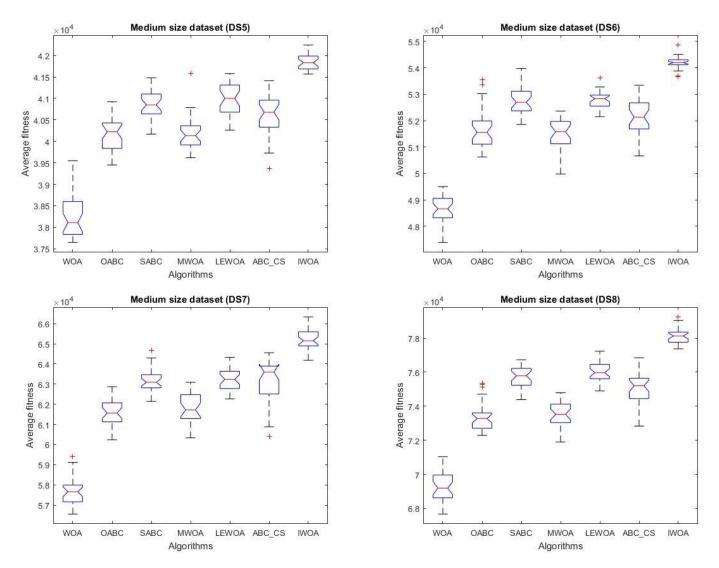


Figure 3. Boxplots for the average fitness value obtained using the compared algorithms for the medium size dataset.

Compared to other algorithms, two types of experiments were conducted to verify the IWOA capabilities on local exploitation and global search. The first type of experiment set was applied to the first group of datasets (small-sized datasets), and aimed to depict the capabilities of the IWOA on local exploitation [38] compared to the competitors, because the first group of datasets had a small-dimensional search space when keeping a constant number of task sizes. The second type of experiment set was applied to the second group of datasets (medium-size and large-size datasets), and aimed to depict the capabilities of the IWOA on global search [38] compared to the competitors, because the second group had a high-dimensional search space when maintaining various task services.

5.2.1. Local Exploitation Validation Experiments

These experiments were designed to verify the capabilities of the IWOA on local exploitation when running IWOA on small-size datasets. Figure 2 presents the average fitness value obtained by all algorithms in 30 independent runs. From the figure, the IWOA had a larger median value than other algorithms except for DS1, where the performance of IWOA did not differ from some competitors (SABC, LEWOA, and ABC_CS), and DS3, where the performance of IWOA was close to ABC_CS. The algorithm results in the average fitness value were also relatively concentrated. The figure also shows that the IWOA had no outlier value with better stability.

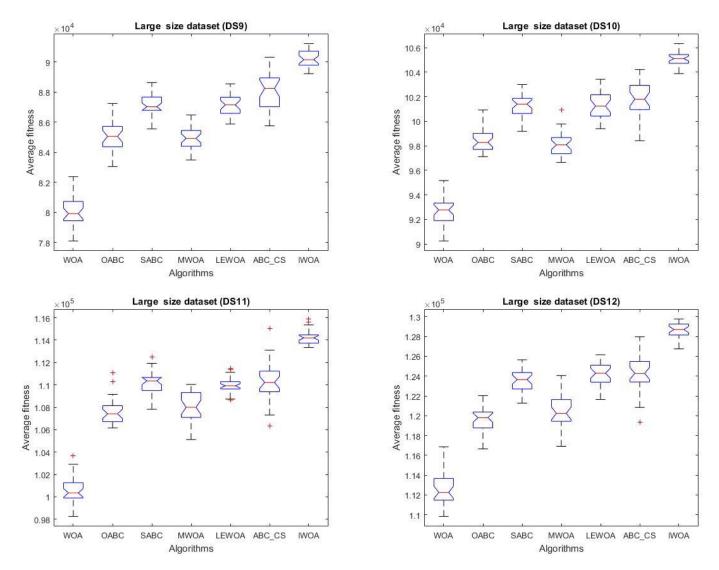


Figure 4. Boxplots for the average fitness value obtained using the compared algorithms for the large size dataset.

Figure 2 shows no overlapping between the notches of the IWOA boxplot and the competitors' boxplots over all datasets, except for DS1 and DS3; this indicates strong evidence that the results obtained by the IWOA significantly differed over all the datasets except for DS1 and DS3. The figure also shows that the IWOA obtained significantly higher boxplot positions over all the datasets except for DS1 and DS3. It can be seen from Table 3 that the IWOA algorithm had significant competitiveness and stability compared with other algorithms on all small-size datasets, and both its best fitness values and standard deviation performed better than other competitors. These results indicate that the IWOA performs well with significant competitiveness and stability. Regarding average execution time, WOA had the lowest values, while IWOA was third after ABC_CS and WOA.

5.2.2. Global Search Validation Experiments

These experiments were designed to verify the capabilities of the IWOA on global search when running IWOA on medium-size and large-size datasets, as shown in Figures 3 and 4, respectively. Figures 3 and 4 present the average fitness value obtained by all algorithms in 30 independent runs on medium-size and large-size datasets, respectively. From the figures, the IWOA has a larger median value than other algorithms, and the algorithm's results in terms of the average fitness value were also relatively concentrated. The figure also shows that the IWOA did not have many outlier values with better stability.

Size	Dataset	Evaluation	WOA	OABC	SABC	MWOA	LEWOA	ABC_CS	IWOA
		BFV	14,034.44	14,034.44	14,034.44	14,034.44	14,034.44	14,034.44	14,034.44
	DS1	STD	187.97	65.18	18.93	65.18	34.49	54.90	0.00
		AET	70	95	94	172	154	87	92
		BFV	13,881.90	14,528.76	14,683.33	14,598.99	14,822.91	14,844.86	14,844.86
	DS2	STD	195.56	194.24	137.85	218.07	114.44	151.31	54.04
		AET	100	416	129	380	237	120	129
		BFV	15,102.50	15,327.59	15,427.52	15,225.15	15,306.70	15,771.44	15,769.43
	DS3	STD	248.21	189.18	221.66	182.52	168.31	124.38	100.69
		AET	115	721	141	1380	211	138	145
		BFV	15,015.99	15,546.62	15,843.82	15,714.46	16,167.67	16,359.07	16,652.91
	DS4	STD	305.47	253.32	254.81	282.40	239.66	257.85	182.23
		AET	133	857	155	632	225	144	161
		BFV	39,549.11	40,925.86	41,480.35	41,579.97	41,580.27	41,303.38	42,244.81
	DS5	STD	466.95	387.99	304.97	415.87	380.29	499.45	189.24
		AET	130	211	295	383	320	163	159
		BFV	49,504.53	53,542.62	53,978.77	52,359.56	53,629.34	53,339.69	54,863.24
	DS6	STD	567.70	769.34	499.94	600.86	323.96	703.61	247.38
Medium –		AET	175	268	430	484	450	224	220
	DS7	BFV	59,404.74	62,868.06	64,666.89	63,090.82	64,332.29	64,557.35	66,334.41
		STD	758.02	683.73	569.30	802.93	531.55	1218.42	460.52
		AET	293	344	508	652	520	366	352
	DS8	BFV	71,031.27	75,333.25	76,726.71	74,787.07	77,232.77	76,847.28	79,246.31
		STD	905.19	830.80	612.59	757.43	592.98	985.48	496.85
		AET	362	371	589	694	602	442	411
	DS9	BFV	82,385.25	87,238.51	88,628.78	86,483.38	88,552.77	90,320.09	91,230.12
- Large -		STD	1086.45	1076.90	732.32	832.21	728.38	1198.45	545.88
		AET	442	466	787	808	764	552	508
		BFV	95,166.52	100,921.44	102,985.17	100,920.84	103,414.77	104,215.63	106,334.80
	DS10	STD	1200.26	977.53	921.33	1020.61	982.57	1490.25	563.01
		AET	494	560	954	867	798	659	560
		BFV	103,702.74	111,073.19	112,471.75	110,045.35	111,488.06	115,029.29	115,887.13
	DS11	STD	1256.80	1156.91	996.85	1421.54	687.06	1791.95	649.83
		AET	582	632	890	834	1647	798	678
	DS12	BFV	116,861.69	122,028.88	125,650.03	124,063.88	126,162.04	127,963.65	129,768.36
		STD	1744.84	1181.03	1141.18	1846.44	1228.71	1952.69	715.02
		AET	610	641	1141	921	895	930	790

Table 3. Best fitness values (BFV), standard deviation (STD), and average execution time (AET) values for IWOA compared to other algorithms.

Figures 3 and 4 show no overlapping between the notches of the IWOA boxplot and competitors' boxplots over all datasets; this indicates strong evidence that the results obtained by the IWOA differed significantly over all the datasets. The figure also shows that the IWOA obtained significantly higher boxplot positions over all the datasets.

Table 3 also shows that the IWOA algorithm had obvious best fitness values and standard deviation compared with other algorithms on medium-size and large-size datasets, and the obtained best fitness and standard deviation values were better than those of its competitors. These results indicate that the IWOA had a good performance and strong competitiveness and stability. In terms of average execution time, the table shows that the average execution time of some of the competitors was better than that of IWOA. However,

it ranked second regarding average execution time after the WOA, and sometimes third after the WOA and OABC, which was reasonable because of the proposed strategies.

5.2.3. Wilcoxon's Rank-Sum Test Analysis

In optimization, algorithm performance can be affected by the randomness factor. Thus, the Wilcoxon signed-rank test presented statistically significant results and evaluated the performance difference between IWOA and its competitors. For 30 independent runs on each dataset, the Wilcoxon two-by-two rank-sum test analysis with its significance evaluation index at 5% of IWOA and competitors was used to obtain the *p*-values. The average fitness value of IWOA statistically significantly differed from the algorithm if the *p*-value was less than 5%.

Table 4 shows the Wilcoxon signed-rank test summarization obtained for small, medium, and large-size datasets of IWOA and competitors. In the table, the "-" indicates that IWAO statistically significantly differed from the algorithm, the "+" indicates that competitors statistically significantly differed from the IWAO, and the "=" indicates that it was not possible to calculate an accurate *p*-value because the convergence of the IWOA and the current algorithm was so close.

Table 4. p-values of IWOA compared to other algorithms obtained from Wilcoxon's rank-sum test.

		WOA	OABC	SABC	MWOA	LEWOA	ABC_CS
Small datasets	—	4	4	3	4	3	2
	+	0	0	0	0	0	1
	=	0	0	1	0	1	1
Medium datasets	_	4	4	4	4	4	4
	+	0	0	0	0	0	0
	=	0	0	0	0	0	0
Large datasets	_	4	4	4	4	4	4
	+	0	0	0	0	0	0
	=	0	0	0	0	0	0

From Table 4 it can be observed that among the small-size datasets, there was no significant difference between IWOA, SABC, LEWO, and ABC_CS on one dataset (from Figure 2, this dataset is DS1); additionally, there was no significant difference between IWOA and ABC_CS on one dataset (from Figure 2 this dataset is DS3); on all other datasets, IWOA had a significant difference from its competitors.

6. Conclusions and Future Research

Although the WOA has been proposed to solve complex multi-objective optimization problems with the advantage of excellent global searchability, it can converge prematurely and fall into local optimum. In this paper, IWOA was proposed to solve the WSC problem and eliminate the WOA shortcomings that lead to low solution accuracy. IWOA was based on the standard WOA with three strategies proposed. First, the Sine chaos theory was proposed to initiate the WOA's population and enhance the initialization diversity. Second, a Lévy flight mechanism was proposed to enhance the exploitation and exploration of WOA by maintaining the whales' diversity. Third, a neighborhood search mechanism was introduced to enhance the exploration and exploitation of searching mechanisms.

The performances of the IWOA were compared within 12 datasets, and the IWOA results were compared with standard WOA and five state-of-the-art optimization algorithms using the same initial conditions. The obtained results from IWOA are promising and show that the IWOA achieved better optimization capability and stability than other optimization algorithms. The IWOA algorithm also showed obvious competitiveness and excellent performance.

In general, the proposed algorithm is expected to be used effectively in different optimization problems in the future, where it should yield promising results with the WSC problem. Furthermore, as a further research direction, the proposed strategies can also be used for different swarm-based algorithms that suffer from premature convergence.

Funding: The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number (IF2/PSAU/2022/01/21050).

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number (IF2/PSAU/2022/01/21050).

Conflicts of Interest: The authors declare that they have no competing interest.

References

- 1. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
- Podili, P.; Pattanaik, K.K.; Rana, P.S. BAT and hybrid BAT meta-heuristic for quality of service-based web service selection. *J. Intell.* Syst. 2017, 26, 123–137. [CrossRef]
- Canfora, G.; di Penta, M.; Esposito, R.; Villani, M.L. A lightweight approach for QoS-aware service composition. In Proceedings of the 2nd International Conference on Service Oriented Computing (ICSOC 04), New York, NY, USA, 15–19 November 2004; pp. 1–2.
- 4. Akyol, S.; Alatas, B. Plant intelligence based metaheuristic optimization algorithms. Artif. Intell. Rev. 2017, 47, 417–462. [CrossRef]
- 5. Ju, C.; Ding, H.; Hu, B. A Hybrid Strategy Improved Whale Optimization Algorithm for Web Service Composition. *Comput. J.* **2021**, bxab187. [CrossRef]
- 6. Jin, H.; Lv, S.; Yang, Z.; Liu, Y. Eagle strategy using uniform mutation and modified whale optimization algorithm for QoS-aware cloud service composition. *Appl. Soft Comput.* **2022**, *114*, 108053. [CrossRef]
- Teng, X.; Luo, Y.; Zheng, T.; Zhang, X. An improved whale optimization algorithm based on aggregation potential energy for qos-driven web service composition. *Wirel. Commun. Mob. Comput.* 2022, 2022, 9741278. [CrossRef]
- Ye, Y.; Chen, S.; Cheng, K.; Zhang, H. A Web Service composition Method Based on Improved Whale Optimization Algorithm. In Proceedings of the 2022 IEEE 12th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 15–17 July 2022; pp. 85–88.
- 9. Seghir, F. FDMOABC: Fuzzy discrete multi-objective artificial bee colony approach for solving the non-deterministic QoS-driven web service composition problem. *Expert Syst. Appl.* **2021**, *167*, 114413. [CrossRef]
- 10. Zhang, S.; Shao, Y.; Zhou, L. Optimized artificial bee colony algorithm for web service composition problem. *Int. J. Mach. Learn. Comput.* **2021**, *11*, 327–332. [CrossRef]
- 11. Dahan, F.; El Hindi, K.; Ghoneim, A. Enhanced artificial bee colony algorithm for QoS-aware web service selection problem. *Computing* **2017**, *99*, 507–517. [CrossRef]
- 12. Dahan, F.; Mathkour, H.; Arafah, M. Two-step artificial bee colony algorithm enhancement for QoS-aware Web service selection problem. *IEEE Access* **2019**, *7*, 21787–21794. [CrossRef]
- 13. Seghir, F.; Khababa, A.; Semchedine, F. An interval-based multi-objective artificial bee colony algorithm for solving the web service composition under uncertain QoS. *J. Supercomput.* **2019**, *75*, 5622–5666. [CrossRef]
- 14. Arunachalam, N.; Amuthan, A. Integrated probability multi-search and solution acceptance rule-based artificial bee colony optimization scheme for web service composition. *Nat. Comput.* **2021**, *20*, 23–38. [CrossRef]
- 15. Chandra, M.; Niyogi, R. Web service selection using modified artificial bee colony algorithm. *IEEE Access* **2019**, *7*, 88673–88684. [CrossRef]
- 16. Li, T.; Yin, Y.; Yang, B.; Hou, J.; Zhou, K. A self-learning bee colony and genetic algorithm hybrid for cloud manufacturing services. *Computing* **2022**, *104*, 1977–2003. [CrossRef]
- 17. Karthikeyan, P.; Preethi, G. Artificial bee colony and genetic algorithms in selecting and combining web services for enhancing QoS. *Des. Eng.* **2021**, 2021, 6009–6021.
- Dahan, F.; Alwabel, A. Artificial bee colony with cuckoo search for solving service composition. *Intell. Autom. Soft Comput.* 2023, 35, 3385–3402. [CrossRef]
- 19. Dahan, F. Neighborhood search based improved bat algorithm for web service composition. *Comput. Syst. Sci. Eng.* **2023**, 45, 1343–1356. [CrossRef]
- Kouicem, A.; Khanouche, M.E.; Tari, A. Novel bat algorithm for QoS-aware services composition in large scale internet of things. *Clust. Comput.* 2022, 25, 3683–3697. [CrossRef]

- 21. El Allali, N.; Fariss, M.; Asaidi, H.; Bellouki, M. A web service composition framework in a heterogeneous environment. *J. Ambient. Intell. Humaniz. Comput.* **2022**, *13*, 1–25. [CrossRef]
- 22. Wang, Z. Optimization of resource service composition in cloud manufacture based on improved genetic and ant colony algorithm. *Smart Innov. Syst. Technol.* **2022**, *268*, 183–198.
- 23. Dahan, F.; El Hindi, K.; Ghoneim, A.; Alsalman, H. An enhanced ant colony optimization based algorithm to solve QoS-aware web service composition. *IEEE Access* 2021, *9*, 34098–34111. [CrossRef]
- 24. Dahan, F. An effective multi-agent ant colony optimization algorithm for QoS-aware cloud service composition. *IEEE Access* 2021, 9, 17196–17207. [CrossRef]
- Wang, H.; Ding, Y.; Xu, H. Particle swarm optimization service composition algorithm based on prior knowledge. *J. Intell. Manuf.* 2022, 1–19. [CrossRef]
- Shirvani, M.H. Bi-objective web service composition problem in multi-cloud environment: A bi-objective time-varying particle swarm optimisation algorithm. *J. Exp. Theor. Artif. Intell.* 2021, 33, 179–202. [CrossRef]
- Dogani, J.; Khunjush, F. Cloud service composition using genetic algorithm and particle swarm optimization. In Proceedings of the 2021 11th International Conference on Computer Engineering and Knowledge (ICCKE), Mashhad, Iran, 28–29 October 2021; pp. 98–104.
- Subbulakshmi, S.; Ramar, K.; Saji, A.E.; Chandran, G. Optimized web service composition using evolutionary computation techniques. In Proceedings of the Intelligent Data Communication Technologies and Internet of Things: Proceedings of ICICI 2020, Coimbatore, India, 27–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2021; pp. 457–470.
- 29. Ghobaei-Arani, M.; Rahmanian, A.A.; Aslanpour, M.S.; Dashti, S.E. CSA-WSC: Cuckoo search algorithm for web service composition in cloud environments. *Soft Comput.* **2018**, *22*, 8353–8378. [CrossRef]
- Kouchi, S.; Nacer, H. Service selection in cloud computing environment by using cuckoo search. In Proceedings of the International Conference on Information, Communication & Cybersecurity, Khourigba, Morocco, 10–11 November 2021; Springer: Cham, Switzerland, 2021; pp. 219–228.
- 31. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. IEEE Trans. Evol. Comput. 1997, 1, 67–82. [CrossRef]
- 32. Yang, W.; Xia, K.; Fan, S.; Wang, L.; Li, T.; Zhang, J.; Feng, Y. A multi-strategy Whale optimization algorithm and its application. *Eng. Appl. Artif. Intell.* **2022**, *108*, 104558. [CrossRef]
- 33. Liu, L.; Zhang, R. Multistrategy Improved Whale Optimization Algorithm and Its Application. *Comput. Intell. Neurosci.* 2022, 2022. [CrossRef]
- 34. Altay, E.V.; Alatas, B. Bird swarm algorithms with chaotic mapping. Artif. Intell. Rev. 2020, 53, 1373–1414. [CrossRef]
- 35. Kaur, A.; Kumar, Y. Neighborhood search based improved bat algorithm for data clustering. *Appl. Intell.* **2022**, *52*, 10541–10575. [CrossRef]
- Al-Masri, E.; Mahmoud, Q.H. Discovering the best web service. In Proceedings of the 16th International Conference on World Wide Web, Banff, AB, Canada, 8–12 May 2007; pp. 1257–1258.
- Wang, X.; Wang, Z.; Xu, X. An improved artificial bee colony approach to QoS-aware service selection. In Proceedings of the IEEE 20th International Conference on Web Services ICWS, Santa Clara, CA, USA, 28 June–3 July 2013; pp. 395–402.
- Li, J.; Ren, H.; Li, C.; Chen, H. A novel and efficient salp swarm algorithm for large-scale QoS-aware service composition selection. Computing 2022, 104, 2031–2051. [CrossRef]