Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 7051248, 16 pages
https://doi.org/10.1155/2019/7051248

Research Article

A Hybridization of Cuckoo Search and Differential Evolution for the Logistics Distribution Center Location Problem

Rui Chi,[1] **Yixin Su**,[2] **Zhijian Qu**,[1] **and Xuexin Chi**[1]

[1]*School of Electrical and Automation Engineering, East China Jiaotong University, Nanchang 30013, China*
[2]*School of Automation, Wuhan University of Technology, Wuhan 430070, China*

Correspondence should be addressed to Xuexin Chi; dr_chi@126.com

Received 8 September 2018; Revised 10 January 2019; Accepted 17 January 2019; Published 6 February 2019

Academic Editor: Erik Cuevas

Copyright © 2019 Rui Chi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The location selection of logistics distribution centers is a crucial issue in the modern urban logistics system. In order to achieve a more reasonable solution, an effective optimization algorithm is indispensable. In this paper, a new hybrid optimization algorithm named cuckoo search-differential evolution (CSDE) is proposed for logistics distribution center location problem. Differential evolution (DE) is incorporated into cuckoo search (CS) to improve the local searching ability of the algorithm. The CSDE evolves with a coevolutionary mechanism, which combines the Lévy flight of CS with the mutation operation of DE to generate solutions. In addition, the mutation operation of DE is modified dynamically. The mutation operation of DE varies under different searching stages. The proposed CSDE algorithm is tested on 10 benchmarking functions and applied in solving a logistics distribution center location problem. The performance of the CSDE is compared with several metaheuristic algorithms via the best solution, mean solution, and convergence speed. Experimental results show that CSDE performs better than or equal to CS, ICS, and some other metaheuristic algorithms, which reveals that the proposed CSDE is an effective and competitive algorithm for solving the logistics distribution center location problem.

1. Introduction

A green and well-developed city logistics system can reduce unnecessary transaction cost and improve economic efficiency. Gevaers Roel [1] has considered that the last mile of the city logistics is the most expensive, inefficient, and palliative part of the supply chain. The logistics centers are served as key nodes among the cities and are the main parts of the modern urban logistics system. Thus, the location planning of city logistics centers is a crucial issue which may affect the overall delivery efficiency. Meanwhile, the number of logistics centers needs to be considered to avoid too much operational cost. Taking into consideration the size, number, and locations of distribution centers, they all affect the input cost of city logistics system. Logistics distribution centers location problems are concerned with the optimal service or supply of a set of existing facilities whose location are fixed and known [2]. In general, the location of the distribution centers needs to be situated in relatively close proximity to the geographic area that it serves [3], and the ultimate goal is to determine where to locate facilities and how to move commodities so that the customers' demands are satisfied with minimized total cost [4].

In the literatures on location problems, many researchers have paid much attention to the optimal choice of distribution centers. Weber and Friedrich [5] first proposed the location theory in 1929. Hakim [6] further theorized the location problem based on [5] and gave procedures to find the optimum location of distribution centers. Since then, a systematic theoretical framework of the location problem has been established. In terms of location models of distribution centers, many researchers have developed the formulation of mathematical models for distribution center locations [7–9]. Aikens [7] proposed nine basic facility location models including the simple uncapacitated, the capacitated, and the dynamic, and stochastic capacitated location models. Klose and Drexl [8] presented the current state of facility location models for logistics distribution system. Sun and Gao et al. [9] presented a bilevel programming model to find the optimal location for logistics distribution centers.

In brief, most of the mathematical formulations belong to mixed integer programming models. They generally include two kinds of variables: discrete variables (integer variables) and continuous variables [9]. Many approaches in terms of solving these location models have been studied in literatures [10–17]. Chou and Chang [10] summarized several primary conventional methods which were frequently used to solve facility location selection problems, such as break-even analysis [11], center-of-gravity method [12], and so on. In recent years, the swarm intelligence optimization algorithms have shown their advantages for solving the complex combinational problems [13, 14], which provide new ideas and methods for location problem [4, 15–19]. Resende and Werneck [4] presented a multistart heuristic for solving the uncapacitated facility location problem. Li [15] presented a 1.488-approximation algorithm to solve the metric uncapacitated facility location problem. Elhedhli and Merrick [16] proposed a Lagrangian heuristic algorithm to decompose the design model by echelon and warehouse site. Rahmani and MirHassani [17] proposed a hybrid firefly genetic algorithm for the capacitated facility location problem. Ho Sin C [18] presented an iterated tabu search heuristic for solving the single source capacitated facility location problem, and the heuristic combined tabu search with perturbation operators to avoid getting stuck in local optima. Thongdee and Pitakaso [19] proposed a modified differential evolution algorithm to solve a multiobjective, source, and stage location-allocation problem, and this algorithm outperformed pervious results and used less computational time.

Although a few improvements for location problems have been achieved, it is worthy of further investigation due to the complexity of multiple constraints and optimal objectives. Therefore, new approaches to solve these problems have drawn a growing attention in recent years. Cuckoo search (CS) algorithm [20] is one of the latest proposed metaheuristic algorithms, which is inspired by the brood parasitism of cuckoo species in nature. CS algorithm has achieved satisfactory performance in solving benchmark unconstrained functions [20] and real-world problems, such as manufacturing scheduling [21], structural optimization [22], and so on. But it has been rarely applied in solving location problems. Another algorithm, differential evolution (DE) algorithm, is developed by Storn and Price [23], which is a population-based technique, and it is inherently parallel. DE has been introduced to solve many kinds of location problems [19, 24, 25]. However, both CS and DE are also limited in certain aspects to solve optimization problems [26–29]. CS has the advantage of identifying the promising area of the search space, but it is less good at fine-tuning the area which is close proximity to the optimum [27]. DE is good at exploring local search but it has no mechanism to memory the previous process and use the global information about the search space [28, 29].

To make up the disadvantages of these two algorithms, this paper proposes an effective hybrid algorithm based on cuckoo search algorithm and differential evolution algorithm (CSDE). To the best of our knowledge, the hybridization of CS and DE has not been attempted in application with location problems. To extend its applications range, the proposed CSDE is used to solve several standard benchmarking functions and a typical logistics distribution center location problem. The performance of the approach is analyzed and compared with other methods in the literatures.

This paper is organized as follows. In Section 2 the model of logistics distribution center location problem is described in formal terms. Section 3 presents the original CS algorithm and DE algorithm and then provides the framework of our proposed CSDE algorithm. In Section 4, experimental results and comparisons with other metaheuristic algorithms are presented. Section 5 summarizes the conclusions and outlines directions for the further investigation.

2. The Model of Logistics Distribution Center Location Problem

Many researchers generally describe a location problem under several simplified assumptions. The problem studied in this paper is based on the following assumptions [30].

(1) The distribution centers always have enough capacities which could satisfy the customer requirements of the total demand points, and the scale of capacity is dependent on the demands within the scope of distribution.

(2) Each demand point is only supplied by one distribution center.

(3) The transportation cost between factories and distribution centers is not considered.

On the basis of these assumptions, a location/allocation model is formulated [30]. This model is proposed to select the optimized place of distribution centers among the total demand points and deliver products to each demand point. The objective is to find the min-sum of the product of demand and distance between each distribution center and each demand point, which is stated as

$$\min \quad F = \sum_{i \in N} \sum_{j \in M_i} \omega_i d_{ij} Z_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{j \in M_i} Z_{ij} = 1, \tag{2}$$

$$Z_{ij} \le h_j, \tag{3}$$

$$\sum_{j \in M_i} h_j = p, \tag{4}$$

$$d_{ij} \le s, \tag{5}$$

$$Z_{ij}, h_j \in \{0, 1\}, \tag{6}$$

$$i \in N, \ \ j \in M_i, \ \ M_i \subseteq N, \tag{7}$$

where N is a set of ordinal numbers of the total demand points; M_i is sets of candidate distribution centers, if the distance between these distribution centers and demand point i is smaller than s; ω_i is quantity demand of demand

point i; d_{ij} is distance from the demand point i to the nearest distribution center j; s is maximum distance between the new distribution centers and demand point it served; p is the number of selected distribution centers;

The decision variables are as follows:

$Z_{ij} = 1$, if demand point i is severed by distribution center j; $Z_{ij} = 0$ otherwise; $h_j = 1$, if demand point j is chosen as a distribution center; $h_j = 0$ otherwise.

The objective function (1) minimizes the transportation cost from the selected distribution center to its served demand points. Constraint set (2) guarantees that each demand point has one distribution center to provide service. Constraint set (3) guarantees that it is necessary that each demand point has a corresponding distribution center. Constraint set (4) states that the number of selected distribution center is p. Constraint set (5) is a standard inequality constraint. Constraint set (6) guarantees that the demand points are all in the area where the distribution centers could provide service.

3. The Hybrid CSDE Algorithm

3.1. The Original DE. Differential evolution (DE) is a stochastic algorithm based on swarm intelligence. It finds the optimal solution by cooperation and competition among individuals within the population. This section briefly presents three mainly operations of DE: mutation, crossover, and selection. DE utilizes N, D −dimensional parameter vectors which are also called individuals and encoded the candidate solutions. A parameter vector in the population is stated as $\vec{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, $i = 1, 2, \ldots, N$. The initial population $S = \{x_1, x_2, \ldots, x_N\}$ is randomly generated and it follows a uniform distribution. For each generation, all parameter vectors in S are regarded as a target for replacement. Thus, there are N competitions to determine the specific numbers of S in the next generation. This is achieved by a loop of operations as follows.

(1) Mutation. After initialization, for each target $\vec{x}_i(k)$, a mutant vector $\vec{w}_i(k+1)$ is generated by certain mutation strategy. The following are five most frequently used mutation strategies [31].

(a) "DE/rand/1"

$$\vec{w}_i(k+1) = \vec{x}_{r_1}(k) + F\left[\vec{x}_{r_2}(k) - \vec{x}_{r_3}(k)\right] \tag{8}$$

(b) "DE/best/1"

$$\vec{w}_i(k+1) = \vec{x}_{best}(k) + F\left[\vec{x}_{r_1}(k) - \vec{x}_{r_2}(k)\right] \tag{9}$$

(c) "DE/rand/2"

$$\begin{aligned}\vec{w}_i(k+1) &= \vec{x}_{r_1}(k) \\ &\quad + F\left[\left(\vec{x}_{r_2}(k) - \vec{x}_{r_3}(k)\right) + \left(\vec{x}_{r_4}(k) - \vec{x}_{r_5}(k)\right)\right]\end{aligned} \tag{10}$$

(d) "DE/best/2"

$$\begin{aligned}\vec{w}_i(k+1) &= \vec{x}_{best}(k) \\ &\quad + F\left[\left(\vec{x}_{r_1}(k) - \vec{x}_{r_2}(k)\right) + \left(\vec{x}_{r_3}(k) - \vec{x}_{r_4}(k)\right)\right]\end{aligned} \tag{11}$$

(e) "DE/rand-to-best/1"

$$\begin{aligned}\vec{w}_i(k+1) &= \vec{x}_i(k) \\ &\quad + F\left[\left(\vec{x}_{best}(k) - \vec{x}_i(k)\right) + \lambda\left(\vec{x}_{r_1}(k) - \vec{x}_{r_2}(k)\right)\right]\end{aligned} \tag{12}$$

where $i = 1, 2, \ldots, N$, $r_1, r_2, r_3, r_4, r_5 \in \{1, 2, \ldots, N\}$ are five different randomly generated indices and are different from the current index i; the scaling factor F is a real constant. $\vec{x}_{best}(k)$ is the best individual which has the best fitness. λ is a random number within the range $[0, 1]$. k is the current generation.

(2) Crossover. The target vector $\vec{x}_i(k)$ is mixed with the mutated vector $\vec{w}_i(k+1)$ to produce a trial vector $\vec{u}_i(k+1)$.

$$\vec{u}_{ij}(k+1) = \begin{cases}\vec{w}_{ij}(k+1), & if\ rand_j \le Cr\ or\ j = jrand, \\ \vec{x}_{ij}(k), & otherwise.\end{cases} \tag{13}$$

where $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, D$. $jrand$ is a random integer within $[1, D]$, $rand_j \in [0, 1]$ is a random number, and $Cr \in [0, 1]$ is a crossover constant.

(3) Selection. A greedy criterion is used to select the better vector from the target vector $\vec{x}_i(k)$ and the trial vector $\vec{u}_i(k+1)$ by comparing their fitness value. For example, if we aim at optimizing a set of minimization problem; the specific selection operation is given by

$$\vec{x}_i(k+1) = \begin{cases}\vec{u}_i(k+1), & if\ f\left(\vec{u}_i(k+1)\right) \le f\left(\vec{x}_i(k)\right), \\ \vec{x}_i(k), & otherwise.\end{cases} \tag{14}$$

If the fitness value of trial vector $\vec{u}_i(k+1)$ is lower than $\vec{x}_i(k)$, $\vec{u}_i(k+1)$ is set to $\vec{x}_i(k+1)$; otherwise, the old vector $\vec{x}_i(k)$ is retained.

Reproduction (mutation, crossover, and selection) continues until a reset condition is met.

3.2. The Improved Mutation Operation in DE. According to the different producing ways of the mutated vectors, a variety of differential evolution methods are formed. There are five most frequently used mutation strategies in [31]. The

"DE/best/1" "DE/best/2" and "DE/rand-to-best/1" strategies rely on the best solution found so far. They usually perform well and achieve the fast convergence speed when optimizing unimodal problems. But when solving multimodal problem, they are easy to be trapped into local optimization and thereby lead to a premature convergence. The "DE/rand/1" strategy can have stronger exploration capability and slow convergence speed. Therefore, it is usually more suitable for solving multimodal problems than the strategies relying on the best solution found so far [31]. The "DE/rand/2" strategy bears a better exploration capability due to the Gaussian-like perturbation [31]. The advantage of using two-difference-vectors-based strategies may result in better perturbation than one-difference-vector-based strategies [32]. The variant DE/rand/1 (8) is composed by three different random individuals, which is beneficial to maintain the diversity of population and improve the global search ability of DE. The variant DE/best/1 (9) uses the current best individual to direct the production of a new mutant vector, which is helpful for enhancing the local search ability of DE.

Combined with the characteristics of the two different mutation methods, we prefer to develop each advantage and mix the two variants. The mixed mutation equation [33] is stated as follows:

$$\begin{aligned}\vec{w}_i(k) = \lambda(k) \times \vec{x}_{r1}(k) + (1 - \lambda(k)) \times \vec{x}_{best}(k) \\ + F(k) \times \left(\vec{x}_{r2}(k) - \vec{x}_{r3}(k)\right)\end{aligned} \tag{15}$$

where $\vec{x}_{r1}$ is a random vector and $\vec{x}_{best}$ is the current best vector. $\lambda(k)(\lambda(k) \in [0,1])$ is a random number which is produced by (16). If $\lambda = 1$, $F(k) = 2F_0$, (15) degenerates into mutation equation DE/rand /1; if λ is approaching to 0, $F(k) = F_0$, (15) degenerates into mutation equation DE/best/1. For a good algorithm, its global search ability is usually strong in the initial stage and local search ability is often excellent at the end of the search stage. Therefore, in the whole searching process λ should gradually reduce from 1 to 0. The influence of $\vec{x}_{r1}$ is gradually reduced, and the influence of $\vec{x}_{best}$ is gradually increased.

In this algorithm, an adaptive scaling factor [34] is proposed to dynamically adjust the value of F according to the search progress of algorithm. The dynamic scaling factor is calculated as follows:

$$\begin{aligned}\lambda(k) &= e^{1-k_{\max}/(k_{\max}+1-k)}, \\ F(k) &= F_0 \times 2^{\lambda(k)}\end{aligned} \tag{16}$$

where k is the number of the current generation and $k_{\max}$ is the maximum number of iterations. When k increases, λ will decrease and $F(k)$ will also continue to decrease correspondingly. $F(k)$ is a constant within $[0, 2]$, $F(k) \in [F_0, 2F_0]$, and F_0 is the initial scaling factor. In the early generations, to preserve the diversity of the solution vectors and avoid premature convergence, the value of $F(k)$ is set as $2F_0$ which should be large enough. $F(k)$ is gradually reduced according to the continuous update of iterations. In the late generations, to keep the excellent population information and increase the probability of finding the global optimal solution, $F(k)$ is reduced and approximates to F_0.

3.3. The Original CS. Cuckoo search algorithm (CS) is also a population-based stochastic algorithm which is inspired by the breeding behavior of cuckoos. It simulates the behavior that female cuckoos lay their eggs in the nest of other birds and let the hosts hatch new cuckoo chicks. Cuckoos use a few strategies to decrease the probability of being abandoned by the hosts.

CS is introduced based on the following three ideal rules [20]: (1) each cuckoo lays only one egg (solution) at a time and randomly selects one nest; (2) the best nest which has the best eggs will be passed from the current generation onto the next generation; (3) the number of hosts is fixed and p_a ($p_a \in [0, 1]$) is a probability with which a host cuckoo discovers an alien egg.

For a cuckoo i, a new solution $\vec{x}_i(k+1)$ is generated using (17) by a Lévy flight which is carried out as in (18).

$$\vec{x}_i(k+1) = \vec{x}_i(k) + \alpha \oplus \text{Lévy}(s, \lambda) \tag{17}$$

$$\text{Lévy}(s, \lambda) \sim s^{-\lambda}, \quad (1 < \lambda < 3) \tag{18}$$

where $\vec{x}_i(k)$ is the current solution, k is the current generation, $\alpha(\alpha > 0)$ is the step size which is associated with the scale of the optimized problem, and $\oplus$ represents entry-wise multiplications. Equation (17) has an infinite variance with an infinite mean [20].

3.4. The Hybrid CSDE Procedure. In general, an effective trade-off between the global search and the local search in an algorithm plays a beneficial role on improving the performance of this algorithm [27]. CS is good at searching the promising area of the whole search space and but not good at fine-tinning at the end of the search stage [27]. DE has the advantage of local search but is not good at using the global information of the search space [28, 29]. In the present work, to take the advantage of these two algorithms, a hybrid algorithm (CSDE) by combining CS and the improved DE is proposed to solve the typical logistics distribution center location problem mentioned before.

An integration of two different ways is used to produce a new solution in CSDE. The first way is the Lévy flight in the original CS ((17)-(18)); the second way is the improved mutation operation in DE ((15)-(16)). The detailed hybrid operation is shown as follows:

$$\vec{y}_i(k+1) = d \times \vec{x}_i(k+1) + (1 - d) \times \vec{w}_i(k+1) \tag{19}$$

where $\vec{x}_i(k+1)$ is a new solution generated by Lévy flight in CS, $\vec{w}_i(k+1)$ is a new solution produced by the improved mutation operation in DE, $\vec{y}_i(k+1)$ is a new solution produced by the combination of $\vec{x}_i(k+1)$ and $\vec{w}_i(k+1)$, and d ($d \in [0, 1]$) is a random constant.

From the above discussion, the proposed CSDE can be summarized as given in Algorithm 1.

Initialize objective function $f(\vec{b})$, $\vec{b} = (b_1, \ldots, b_D)^T$ $\vec{x} = \vec{b}$, $\vec{y} = \vec{b}$, $\vec{w} = \vec{b}$
Initialize a population of N individuals $\vec{y}_i$ $(i = 1, 2, \ldots, N)$
For all $\vec{y}_i$ $(i = 1, 2, \ldots, N)$ do
  Evaluate the fitness F_i
  $\vec{y}_{best} = \arg\min f(\vec{b}_i)$
End for
While ($k <$ Max Generation) and (the stop criterion is not met) do
 For each individual $i \in 1 \longrightarrow N$ do
  Calculate the step size of CS by using (19)
  $\vec{x}_i\,(k+1)$ = new solutions updated using ((17)-(18))
  $\vec{w}_i\,(k+1)$ = new solutions updated using ((15)-(16))
   Generate a new solution $\vec{y}_i\,(k+1) = d \times \vec{x}_i\,(k+1) + (1-d) \times \vec{w}_i\,(k+1)$
   Evaluate its quality/fitness F_i
   If $(f(\vec{y}_i\,(k+1)) < f(\vec{y}_i(k)))$
    Replace i by the new solution
   End if
End for
Choose a nest among N (say j) randomly
Abandon worse nests with a fraction p_a and rebuild the corresponding new nests
$\vec{x}_i(k+1) = \vec{y}_i\,(k+1), \vec{w}_i\,(k+1) = \vec{y}_i\,(k+1)$
Update $\vec{y}_{best}\,(k+1)$
Keep the best solutions (or nests with quality solutions)
Rank the solutions and find the current best
End while
Post-process results and visualization

Algorithm 1: Pseudocode of the proposed algorithm. A hybridization of cuckoo search and differential evolution (CSDE).

4. Experimental Studies on Benchmark Functions

4.1. Benchmark Functions. In this section, the proposed CSDE is tested through an array of benchmark functions. To get a fair result, all approaches are implemented on the same machine with an Intel core i5-4460 3.20GHz processor, 8.0GB memory, and windows 7 operating system with Matlab2013b.

The benchmark functions shown in Table 1 are standard test functions. Further information about these functions can be referred to [35, 36]. In Tables 1(a) and 1(b), the dimension of test functions F6, F7, F8, F11, and F12 are fixed and cannot be changed. The dimension number of other test functions F1-F5, F7, and F10 is unfixed, and we set their dimension functions as 10. To further demonstrate the performance of the proposed CSCD algorithm, I also implement unfixed dimension functions with D=30 and D=50. Two benchmark test functions F11 and F12 of which the global optimal is different to zero are also considered in Tables 1(a) and 1(b) to further verify the proposed CSDE. In each table, the best value obtained for each test function is shown in bold.

4.2. Simulation and Comparison. For the purpose of verifying the performance of CSDE, we compare its performance on ten benchmark functions (in Table 1) with six other optimization approaches, which are CS [20], DE [23], PSO [37], BA [38], HS [39], ICS [40], and CSPSO [41].

In all experiments, the same parameters for all algorithms are assigned as the population size $N = 20$; the maximum generation $k_{\max} = 2000$. To reduce the influence of the randomness, each experiment is repeated 50 times (see Tables 2–4). Other detailed parameters for all algorithms are given below.

(1) CSDE: $p_a = 0.25$ and $F_0 = 0.6$;

(2) CS [20]: $p_a = 0.25$ and $\alpha = 1$;

(3) DE [23]: $F = 0.6$ and $CR = 0.1$;

(4) PSO [37]: $c_1 = c_2 = 2$, $w_{\min} = 0.4$, and $w_{\max} = 0.9$;

(5) BA [38]: $A = 0.25$, $r = 1$, $Q_{\min} = 0$, and $Q_{\max} = 2$;

(6) HS [39]: $M = 30$, $HMCR = 0.95$, $PAR = 0.3$, and $BW = 0.06$;

(7) ICS [40]: $p_{\min} = 0.005$, $p_{\max} = 0.5$, $\alpha_{\min} = 0.01$, and $\alpha(\max) = 0.5$;

(8) CSPSO [41]: $p_a = 0.25$, $\alpha_{\min} = 0.01$, and $\alpha_{\max} = 0.5$.

In general, the value of parameter has great influence on the performance and convergence of the algorithm. To further evaluate its behavior regarding performance, in this paper, the CSDE algorithm is taken as an example. A sensitivity analysis is performed according to varying the values of the CSDE parameters p_a and F_0 in Figures 1 and 2. By varying $p_a = 0.05, 0.1, 0.15, \ldots, 0.5$, $F_0 = 0.2, 0.4, 0.6, \ldots, 2.0$, it has been found that the best parameters for all cases used in the CSDE is $p_a = 0.25$ and $F_0 = 0.6$. For the Sphere function as an example, from Figures 1 and 2 we can see that $p_a = 0.25$ and $F_0 = 0.6$ are the best parameters for the Sphere function in CSDE.

Table 1

(a) Benchmark functions

No.	Name	Formula
F1	Sphere	$F_1(x) = \sum_{i=1}^{D} x_i^2$
F2	Schwefel2.22	$F_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$
F3	Quadric	$F_3(x) = \sum_{i=1}^{D} i x_i^4 + \text{random}(0, 1)$
F4	Schwefel2.21	$F_4(x) = \max\{\lvert x_i \rvert, 1 \le i \le D\}$
F5	Zakharov	$F_5(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5 i x_i\right)^2 + \left(\sum_{i=1}^{D} 0.5 i x_i\right)^4$
F6	Matyas	$F_6(x) = 0.26(x_1^2 + x_2^2) - 0.48 x_1 x_2$
F7	Powell	$F_7(x) = \sum_{i=1}^{n/k} \left[(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4\right]$
F8	Beale	$F_8(x) = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2) + (2.625 - x_1 + x_1 x_2^3)^2$
F9	Rastrigin	$F_9(x) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$
F10	Griewank	$F_{10}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
F11	Trid6	$F_{11}(x) = \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$
F12	Trid10	$F_{14}(x) = \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$

(b) Benchmark functions

No.	D	Range	Optima
F1	10	[−100, 100]	0
F2	10	[−10, 10]	0
F3	10	[−1.28, 1.28]	0
F4	10	[−100, 100]	0
F5	10	[−5, 10]	0
F6	2	[−10, 10]	0
F7	24	[−4, 5]	0
F8	2	[−4.5, 4.5]	0
F9	10	[−5.12, 5.12]	0
F10	10	[−600, 600]	0
F11	6	[−D2, D2]	-50
F12	10	[−D2, D2]	-210

Table 2 compares the best results obtained by CSDE and the other considering algorithms with the global optimums. From Table 2, it can be clearly seen that CSDE has the strongest search ability of finding the minimum in eight functions: F4, F5, F6, F7, F8, F11, and F12. On functions F1, F2, and F3, CSPSO achieves the best "best" result, and CSDE gets a better "best" result in comparison with other six algorithms. On function F4, compared with PSO, CSDE obtains a similar result. On function F8, except for BA and HS, the result obtained by other six algorithms all approximate to the global optimum. On function F9, CSDE achieves the same results as DE, ICS, and CSPSO. On function F10, DE performs the highest performance and CSDE shows a moderate performance. On function F11 and F12, except for HS, the result obtained by other seven algorithms all approximate to the global optimum.

Table 3 shows the "mean" results achieved by all the eight algorithms. From Table 3, we can see that CSDE performs the best on nine of the twelve benchmark functions (F1, F3-F8, and F11-F12). On function F8, CSDE gets the same best "mean" result as CS and CSPSO and the value of this result approximate to the global optimum. On function F9, ICS achieves the best "mean" result, and CSDE gets a better "mean" result in comparison with other six algorithms. On function F10, the best "mean" result is obtained by BA. DE provides a better "mean" result. On function F11, except for HS, other seven algorithms all achieve the best "mean" result. On function F12, CSDE gets the same best "mean" result as

Table 2: Best optimization results in 12 benchmark functions.

Function	CS	DE	PSO	BA	HS	ICS	CSPSO	CSDE
F1	1.04E-10	1.00E-79	5.14E-06	3.27E-05	5.90E-03	1.00E-48	**2.07E-124**	1.08E-110
F2	1.70E-17	7.80E-44	4.50E-03	1.39E-02	3.09E-02	1.25E-27	**7.04E-76**	4.00E-64
F3	4.00E-04	2.50E-03	6.70E-03	1.64E-02	1.00E-04	9.00E-04	**3.00E-04**	3.88E-04
F4	1.32E-55	4.49E-89	**0**	1.70E-08	8.00E-04	2.57E-76	3.47E-247	**0**
F5	1.60E-22	5.70E-05	1.94E-05	7.43E-05	2.12E-02	1.00E-04	2.30E-35	**1.14E-50**
F6	1.98E-90	7.79E-93	2.37E-177	3.10E-11	4.56E-07	3.33E-48	1.13E-304	**0**
F7	3.00E-04	7.99E-02	2.28E-01	1.88E-02	8.00E-04	2.30E-03	1.60E-03	**4.0E-04**
F8	**0**	**0**	**0**	3.40E-10	7.07E-06	**0**	**0**	**0**
F9	1.73E-02	**0**	1.67E-02	2.01E+00	8.20E-03	**0**	**0**	**0**
F10	3.20E-03	**0**	3.82E-02	3.32E-06	3.51E-02	1.85E-07	1.10E-14	5.55E-16
F11	**-5.00E+01**	**-5.00E+01**	**-5.00E+01**	**-5.00E+01**	-4.99E+01	**-5.00E+01**	**-5.00E+01**	**-5.00E+01**
F12	**-2.10E+02**	**-2.10E+02**	**-2.10E+02**	**-2.10E+02**	-2.09E+02	**-2.10E+02**	**-2.10E+02**	**-2.10E+02**

Table 3: Mean optimization results in 12 benchmark functions.

Function	CS	DE	PSO	BA	HS	ICS	CSPSO	CSDE
F1	1.13E-09	4.04E-77	1.50E-03	8.11E-05	1.09E-01	1.23E-46	5.30E-105	**2.04E-109**
F2	3.68E-16	9.61E-43	2.87E-02	2.33E-02	7.78E-02	1.03E-26	**2.40E-70**	1.33E-61
F3	2.90E-03	1.23E-02	2.48E-02	1.06E-01	3.90E-03	2.80E-03	1.00E-03	**4.49E-04**
F4	1.01E-48	2.50E-58	2.13E-94	7.98E-07	3.21E-02	2.82E-68	4.70E-238	**0**
F5	6.27E-21	2.00E-03	6.74E-01	1.41E-04	7.82E-01	7.00E-03	7.60E-23	**1.21E-48**
F6	3.30E-72	4.80E-80	1.50E-164	8.97E-10	1.50E-03	1.31E-42	1.42E-288	**0**
F7	4.20E-03	2.66E-01	5.21E+00	9.93E-02	1.72E-01	9.10E-03	6.90E-03	**1.10E-03**
F8	**0**	2.80E-11	1.07E-01	1.22E-01	1.23E-02	1.36E-31	**0**	**0**
F9	1.87E+00	2.79E-01	6.70E+00	9.29E+00	7.14E-02	**0**	5.14E-02	4.04E-13
F10	2.86E-02	1.70E-03	1.54E-01	**9.69E-06**	2.25E-01	1.26E-02	2.00E-02	1.63E-02
F11	**-5.00E+01**	**-5.00E+01**	**-5.00E+01**	**-5.00E+01**	-4.96E+01	**-5.00E+01**	**-5.00E+01**	**-5.00E+01**
F12	**-2.10E+02**	-2.09E+02	**-2.10E+02**	-2.09E+02	-1.76E+02	**-2.10E+02**	-2.09E+02	**-2.10E+02**

Table 4: Standard deviations of optimization results in 12 benchmark functions.

Function	CS	DE	PSO	BA	HS	ICS	CSPSO	CSDE
F1	1.38E-09	8.03E-77	2.20E-03	1.85E-05	6.35E-02	3.64E-46	3.78E-105	**2.73E-109**
F2	3.48E-16	1.44E-42	2.52E-02	2.70E-03	2.85E-02	8.45E-27	**1.31E-69**	1.88E-61
F3	1.40E-03	4.80E-03	8.60E-03	4.93E-02	4.30E-03	1.10E-03	8.00E-04	**8.59E-05**
F4	3.93E-48	1.77E-57	7.77E-94	8.87E-07	3.37E-02	1.39E-67	**0**	**0**
F5	1.19E-20	2.70E-03	4.76E+00	3.87E-05	8.41E-01	9.30E-03	4.44E-22	**1.76E-48**
F6	1.63E-71	2.71E-79	**0**	1.01E-09	1.90E-03	7.35E-42	**0**	**0**
F7	4.00E-03	1.61E-01	2.19E+01	5.07E-02	1.73E-01	4.50E-03	2.30E-03	**9.00E-04**
F8	**0**	1.99E-10	2.67E-01	2.82E-01	1.48E-02	4.73E-31	**0**	**0**
F9	1.15E+00	5.33E-01	2.08E+00	3.81E+00	5.55E-02	**0**	2.09E-01	**0**
F10	1.56E-02	3.80E-03	8.49E-02	**2.62E-06**	8.47E-02	7.70E-03	1.64E-02	1.17E-02
F11	**0**	**0**	**0**	**0**	2.67E-01	**0**	**0**	**0**
F12	**0**	1.20E+00	1.51E-01	2.55E+00	3.61E+01	1.37E-05	1.90E-03	**0**

CS, ICS, and PSO, and the value of this result approximates to the global optimum.

Table 4 shows the "standard deviations" results achieved by all the eight algorithms. From Table 3, except for F2 and F10, it can be clearly seen that CSDE performs best. On function F2, CSPSO gets the best "standard deviations" result, and CSDE achieves a better "standard deviations" result. On function F4, CSDE gets the same best "standard deviations" result as CSPSO. On function F6, CSDE, PSO, and CSPSO all get the best "standard deviations" result. On function F8, "standard deviations" results achieved by CSDE, CS, and CSPSO all approximate to "0". On function F9, CSDE and ICS achieve the best "standard deviations" result. On function F11, except for HS, other seven algorithms all achieve the best "standard deviations" result. On function F12, CSDE and CS get the same best "standard deviations" result.

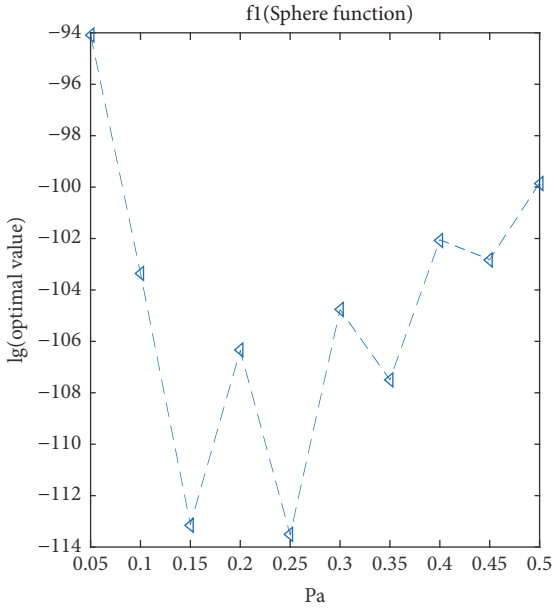


Figure 1: Curves of parameter p_a effect on optimal value.

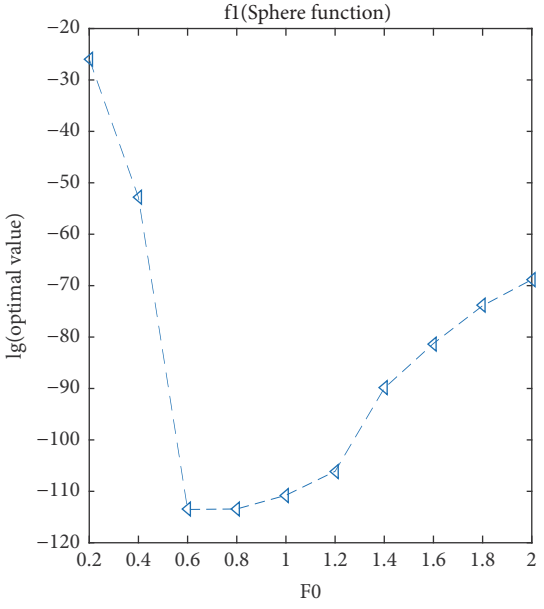


Figure 2: Curves of parameter F_0 effect on optimal value.

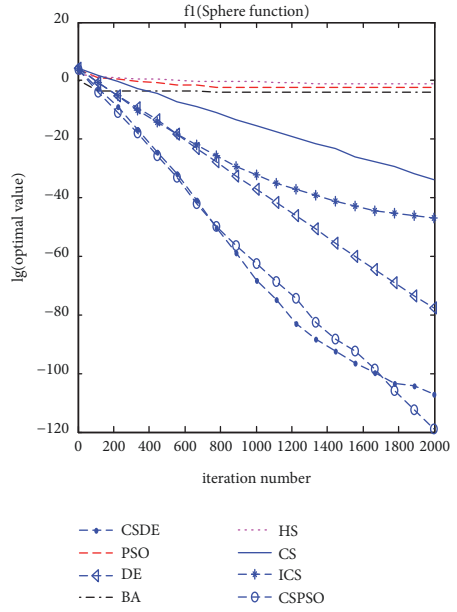


Figure 3: The convergence curves of F1.

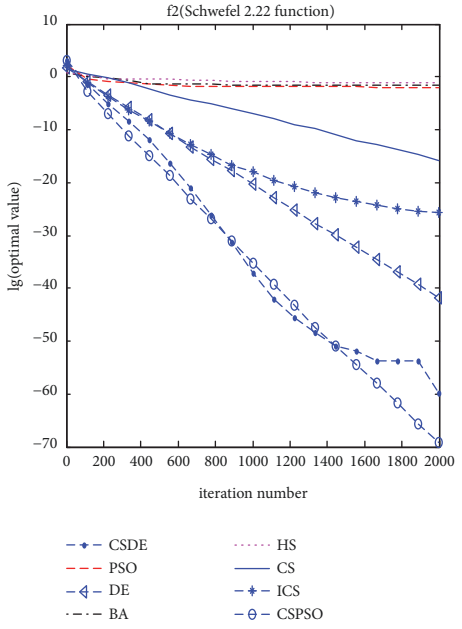


Figure 4: The convergence curves of F2.

In addition, to give a more visualized and detailed comparison, Figures 3–10 provide the convergence curves of the proposed CSDE and other considering algorithms. This paper only presents eight representative convergence curves on eight functions (Sphere, Schwefel 2.22, Quadric, Schwefel 2.21, Zakharov, Matyas, Powell, and Griewank). For the sake of contrastive analysis, the log-based 10 of the achieved results is made in plotting figures, where the label of X-axis represents the number of iteration and the label of Y-axis represents the objective best value.

From Figures 3 and 4, we can see that CSPSO has the fastest convergence speed compared with other seven algorithms on the functions F1-F2. From Figures 5–8, it can be clearly seen that CSDE converges faster than other seven algorithms on the functions F3-F6. From Figure 9, for function F7, CSDE is apparently the fastest algorithm at finding the best result. By carefully looking at Figure 9, at the beginning of the optimization process, BA converges faster than CSDE, while CSDE is more capable of improving its solution and performs steadily at later stage of the optimizing process.

In Figure 10, for function F10, very clearly, BA converges faster than all the other algorithms at the beginning of the

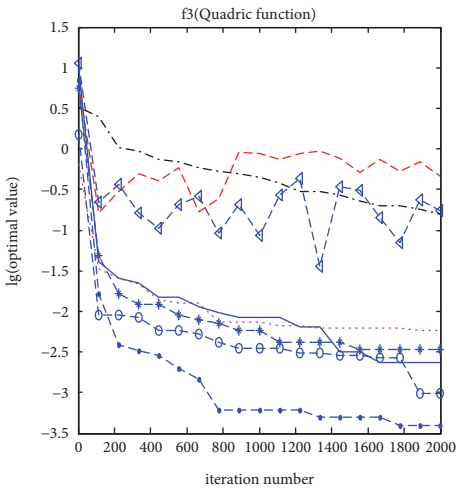


Figure 5: The convergence curves of F3.

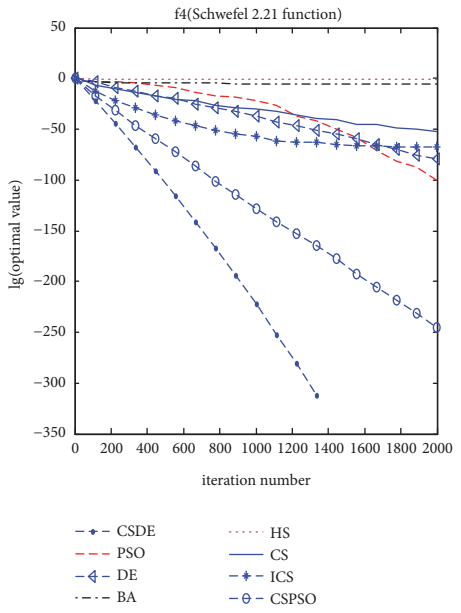


Figure 6: The convergence curves of F4.

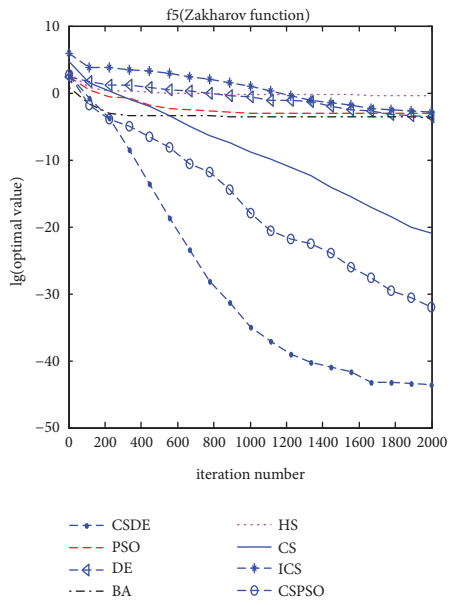


Figure 7: The convergence curves of F5.

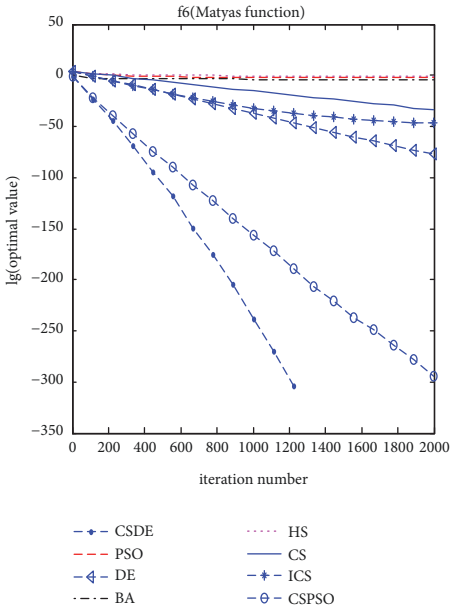


Figure 8: The convergence curves of F6.

iteration process, and CSDE converges faster than all the other algorithms at the later stage of the optimization process. Moreover, it should be noted that, in Figures 6, 8, and 10, the CSDE and DE stop iterate before the 2000 generations, because they have found their global optimum before the 2000 generations.

To further verify the optimization performance of the proposed CSDE algorithm, the dimensions of seven unfixed dimension functions (F1-F5, F9, and F10) in Tables 1(a) and 1(b) are also set to 30 and 50. Figures 11–17 show the convergence curves of these test functions with different dimensions.

From Figures 11–17, it can be clearly seen that for F1, F2, F3, F5, and F9 when the dimension of these functions is lower, CSDE algorithm has better convergence speed and accuracy. With the increase of dimension, the convergence speed and accuracy of CSDE algorithm reduce significantly. In Figure 14, it can be shown that the CSDE algorithm almost has the same convergence speed and accuracy with three different dimension numbers 10, 30, and 50 for F4, and thereby it can be concluded that the dimension of F4 has little effect on the optimization performance of CSDE. In Figure 17, for F10, we can see that when the dimension number is 10, the CSDE algorithm has the worst convergence speed and accuracy. When the dimension number is 30, the convergence speed and accuracy of CSDE algorithm is the highest. When the dimension number is 50, the convergence speed and accuracy of CSDE decrease obviously.

Through above analysis and discussion about the Tables 2–4 and Figures 3–17, we can draw the conclusion that the

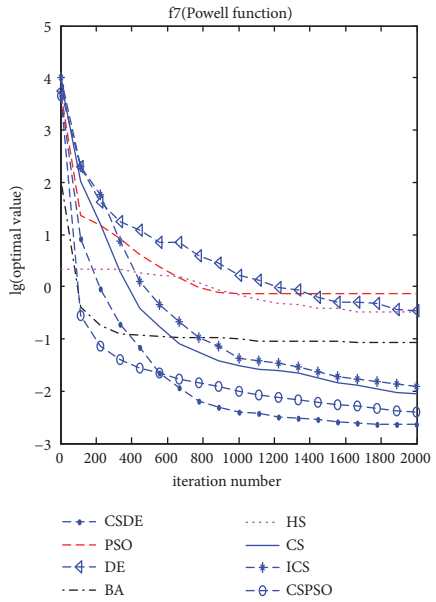


Figure 9: The convergence curves of F7.

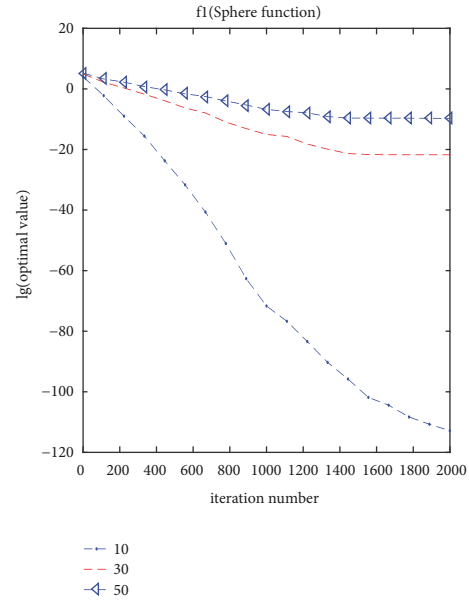


Figure 11: Curves of F1's dimension effect on optimal value.

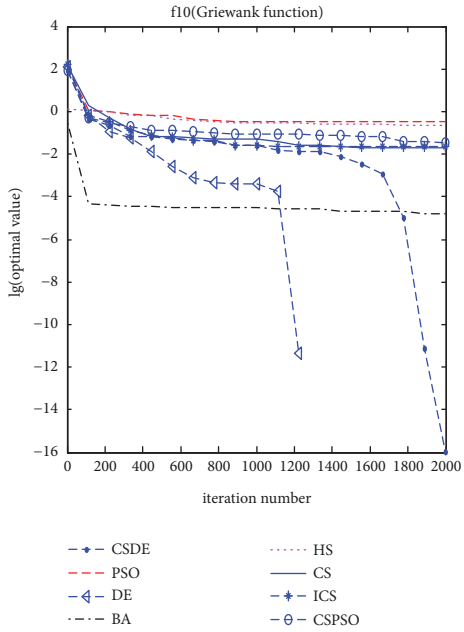


Figure 10: The convergence curves of F10.

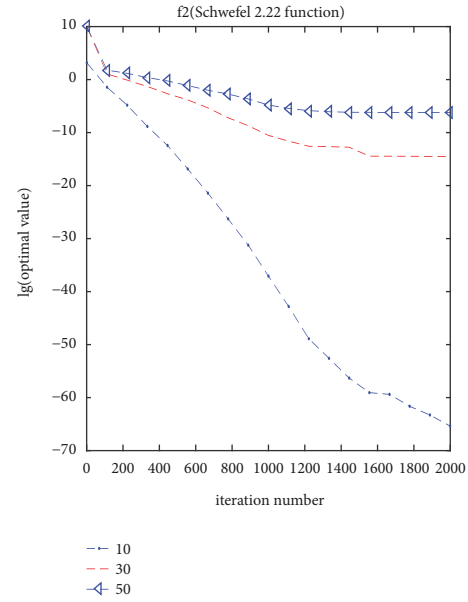


Figure 12: Curves of F2's dimension effect on optimal value.

proposed CSDE performs well for most of the test functions in comparison with other seven algorithms and it is efficient for solving these benchmark functions.

5. Application Studies on the Logistics Distribution Center Location Problem

In this section, the proposed CSDE algorithm is applied to solve the logistics distribution center location problem which has been described in Section 2. For this problem, it is assumed that there are 31 demand points, and 6 of them need to be selected as logistics distribution centers. The coordinates and capacity of each demand point are given in Table 5. The simulation results are compared with six different algorithms (CS [20], PSO [37], BA [38], ICS [40], CSPSO [41], and IA [42]). The parameters of these considering algorithms for the problem are set as the same as those described in Section 4.2. The population size and the maximum generation of IA are set as the same as other algorithms, and the other parameters of IA are set as follows: the memory storage capacity is 10, the mutation probability is 0.4, the crossover probability is 0.5, and the diversity evaluation parameter is 0.95. It must be noticed that the value of quantity demand has been standardized in Table 5.

Table 5

(a) The coordinates and capacity of each demand point

j	(U_j, V_j)	c_j	j	(U_j, V_j)	c_j
1	(1304, 2312)	20	12	(2562, 1756)	40
2	(3639, 1315)	90	13	(2788, 1491)	40
3	(4177, 2244)	90	14	(2381, 1676)	40
4	(3712, 1399)	60	15	(1332, 695)	20
5	(3488, 1535)	70	16	(3715, 1678)	80
6	(3326, 1556)	70	17	(3918, 2179)	90
7	(3238,1229)	40	18	(4061, 2370)	70
8	(4196, 1044)	90	19	(3780, 2212)	100
9	(4312, 790)	90	20	(3676, 2578)	50
10	(4386, 570)	70	21	(4029, 2838)	50
11	(3007, 1970)	60	22	(4263, 2934)	50

(b) The coordinates and capacity of each demand point

j	(U_j, V_j)	c_j
23	(3429, 1908)	80
24	(3507, 2376)	70
25	(3394, 2643)	80
26	(3439, 3201)	40
27	(2935, 3240)	40
28	(3140, 3550)	60
29	(2545, 2357)	70
30	(2778, 2826)	50
31	(2370, 2975)	30

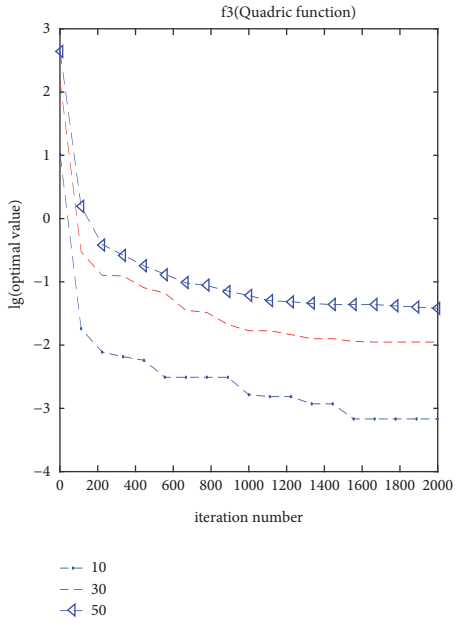


Figure 13: Curves of F3's dimension effect on optimal value.

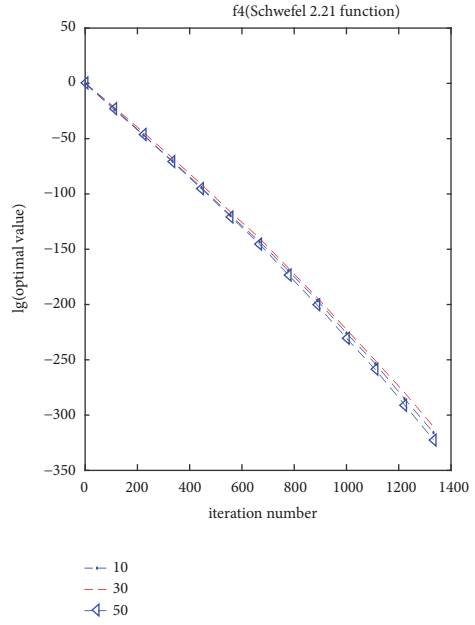


Figure 14: Curves of F4's dimension effect on optimal value.

The optimal results of the logistics distribution center location problem are presented in Table 6 where a comparison among CS, PSO, BA, ICS, CSPSO, IA, and CSDE is depicted. From Table 6, we can see that CSDE achieves the best cost. The optimal solution is obtained at $\vec{x}^* = (27, 19, 12, 20, 5, 9)$ with corresponding function value $F^*(\vec{x}) = 5.65E + 05$. ICS ranks two and it achieves the function value $F^*(\vec{x}) = 5.69E + 05$ at $\vec{x}^* = (14, 24, 27, 5, 9, 18)$.

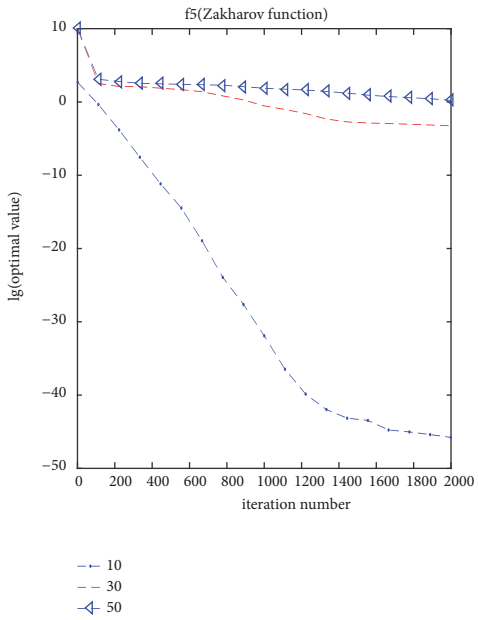


Figure 15: Curves of F5's dimension effect on optimal value.

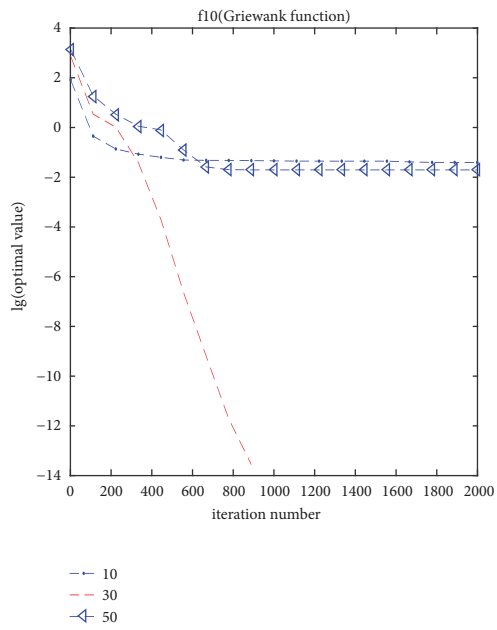


Figure 17: Curves of F10's dimension effect on optimal value.

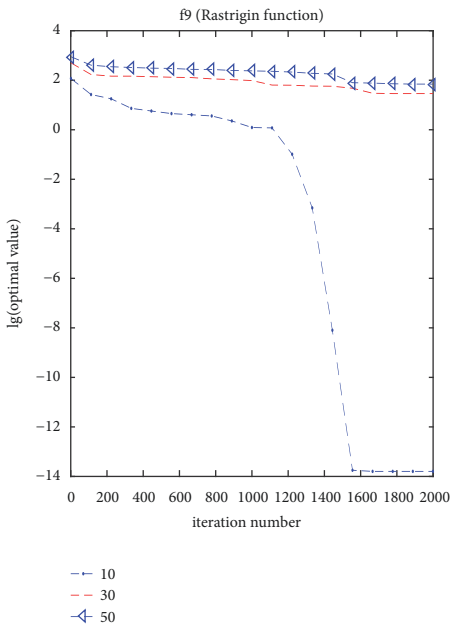


Figure 16: Curves of F9's dimension effect on optimal value.

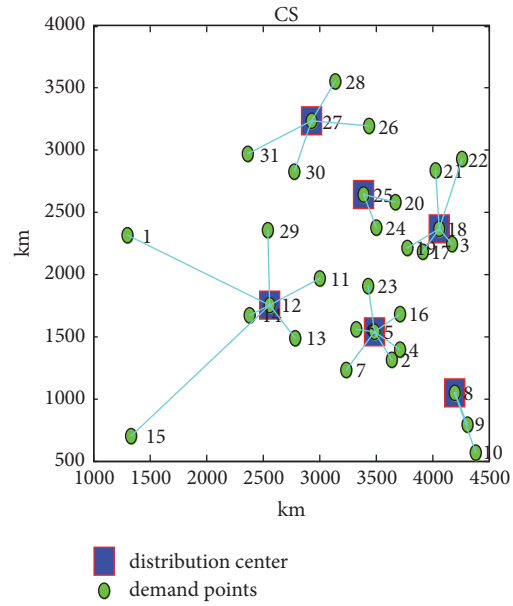


Figure 18: The selection scheme of CS.

CSPSO ranks three and it achieves the function value $F^*(\vec{x}) = 5.71E+05$ at $\vec{x}^* = (5, 21, 9, 17, 27, 12)$. In summary, the proposed CSDE algorithm performs best among other six algorithms. ICS and CSPSO are only inferior to the CSDE. BA achieves the worst function value. The end of Table 6 shows the results of the 5 trials of the calculation by the proposed CSDE. According to the calculation result, we can see that the maximum function value is 5.69E+05, the minimum function value is 5.65E+05, the average function value is 5.68E+05, and the standard deviations function value is 2.00E+01.

Figures 18–24 demonstrate the selection schemes of the proposed CSDE and other six algorithms according to the obtained optimal distribution centers in Table 6. The blue squares represent distribution centers, and the green circles represent demand points. Each distribution center is built to satisfy the corresponding customer requirements of demand points. We demonstrate the selection scheme of CS as an example. In Figure 18, the distribution center 8 is responsible for satisfying the quantity demand of demand points 9 and 10. The distribution center 25 is responsible for demand points 20 and 24. The distribution center 18 is responsible for demand points 3, 17, 19, 21, and 22. The distribution center 5 is responsible for demand points 2, 4, 6, 7, 16, and 23. The distribution center 12 is responsible for demand points 1, 11,

Table 6: Optimal results for minimization of the cost.

Methods	Optimal distribution center (j)						Cost
	j_1	j_2	j_3	j_4	j_5	j_6	
CS [20]	8	25	18	5	12	27	5.74E+05
PSO [37]	29	14	17	9	25	5	5.94E+05
BA [38]	19	9	25	20	8	4	6.08E+05
ICS [40]	14	24	27	5	9	18	5.69E+05
CSPSO [41]	5	21	9	17	27	12	5.71E+05
IA [42]	6	27	12	17	9	20	5.78E+05
CSDE	27	19	12	20	5	9	5.65E+05
CSDE	Cost (5 times)						
Min	5.65E+05						
Max	5.69E+05						
Mean	5.68E+05						
SD	2.00E+01						

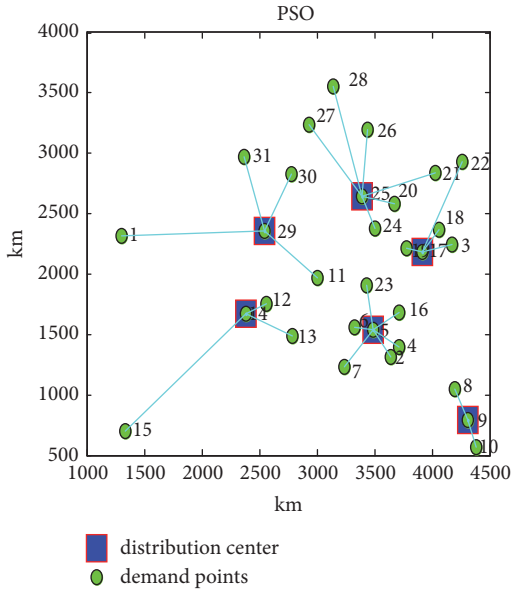


Figure 19: The selection scheme of PSO.

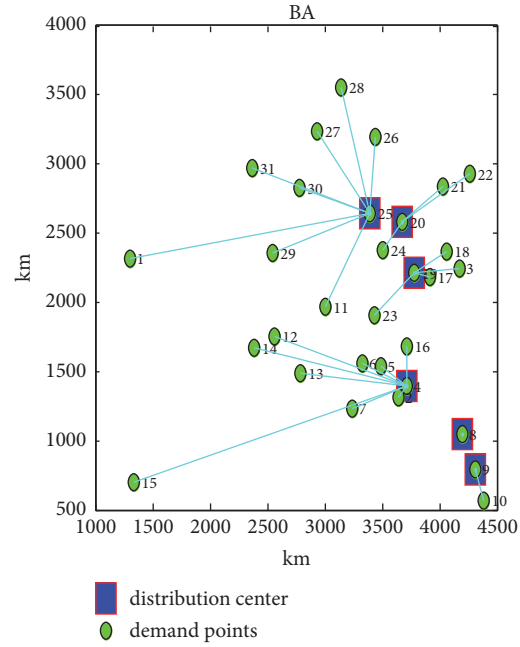


Figure 20: The selection scheme of BA.

13, 14, 15, and 29. The distribution center 27 is responsible for demand points 26, 28, 30, and 31. By analogy, the selection scheme of PSO, BA, ICS, CSPSO, IA, and CSDE can be clearly understood from Figures 19–24, respectively.

Figure 25 presents the convergence curves of the proposed CSDE and other six algorithms for the best result. The values of Y-axis in Figure 25 are obtained optimal values according to the model of the logistics distribution center location problem mentioned before. The values of X-axis are the iteration numbers. As it can be seen from Figure 25, CSDE has the fastest convergence speed in comparison with other six algorithms and it shows the best performance. The convergence speed of ICS and CSPSO is only inferior to CSDE. ICS ranks two and CSPSO ranks three. BA has the slowest convergence speed.

6. Discussion and Conclusion

In this paper, we have formulated an effective hybrid algorithm called CSDE to solve logistics distribution center location problem. From the formulation of CSDE to its implementation and comparison, it can be clearly seen that CSDE is a promising algorithm. The proposed CSDE has been compared with other metaheuristic algorithms including CS [20], DE [23], PSO [37], BA [38], HS [39], ICS [40], CSPSO [41], and IA [42]. The comparison results show that the proposed CSDE algorithm is more superior to other algorithms for most of optimized problems. The main reason is that CSDE takes the search advantage of DE and combined it to CS. From the framework of CSDE, we can see that the

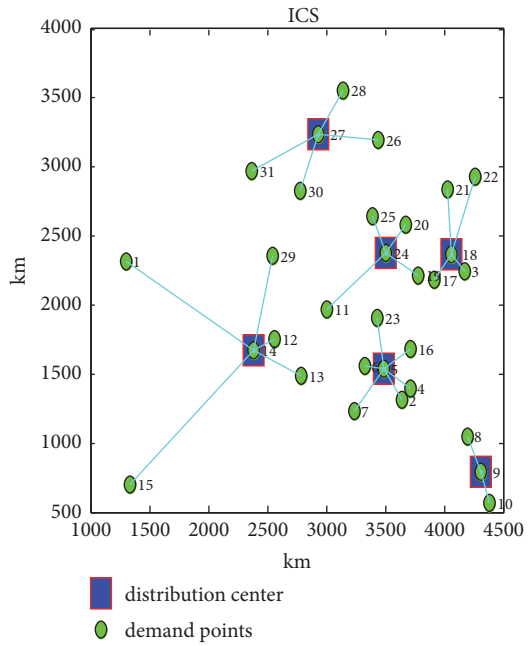


Figure 21: The selection scheme of ICS.

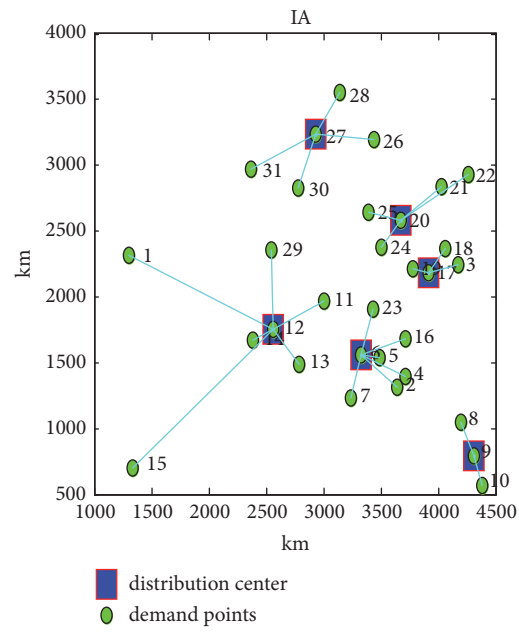


Figure 23: The selection scheme of IA.

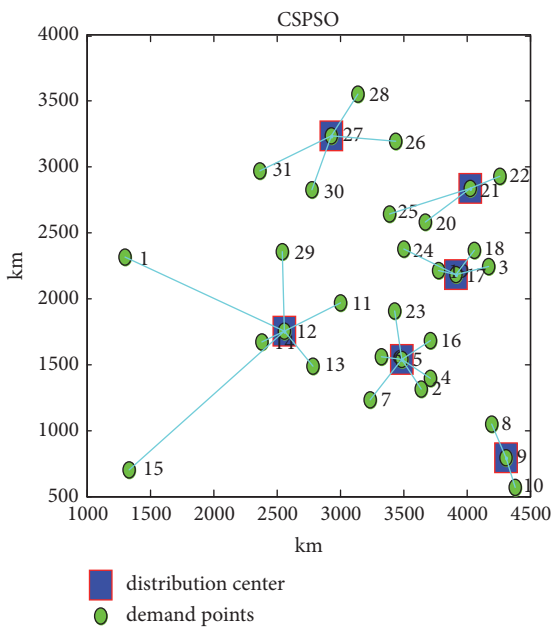


Figure 22: The selection scheme of CSPSO.

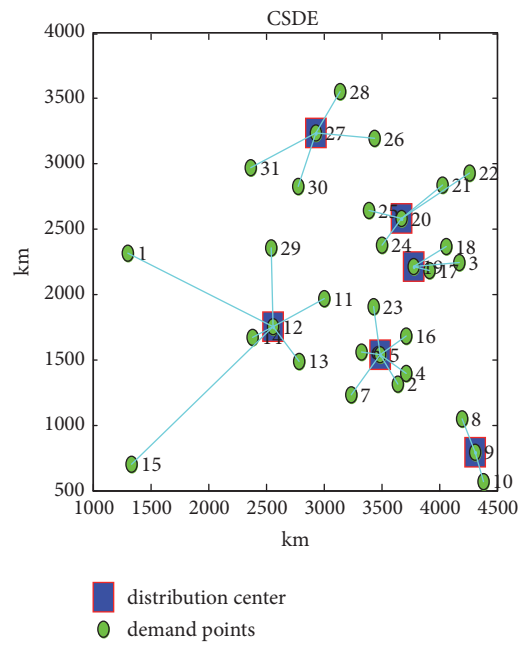


Figure 24: The selection scheme of CSDE.

individuals of the population in CSDE evolve according to two different mechanisms and then share their information with each other. Besides, it can be also considered as a kind of coevolutionary. With the constantly interactive iterations of the CS and DE, the CSDE is well able to find the improved solutions with a high speed.

In addition, it should be noticed that the mutation operation of DE and the value of scaling factor are all different and adjusted dynamically with the increase of the iterations. The effectiveness of these improvements is verified by a series of benchmark functions and the logistics distribution center location problem.

The comparison results on 10 benchmark functions and the logistics distribution center location problem have shown that the proposed CSDE is competitive and efficient when compared with the other considering metaheuristic algorithms. However, the present proposed CSDE is only suitable for single objective optimization problem. Application of the proposed algorithm to multiobjective optimization problem and large-scale optimization problem is the future work.

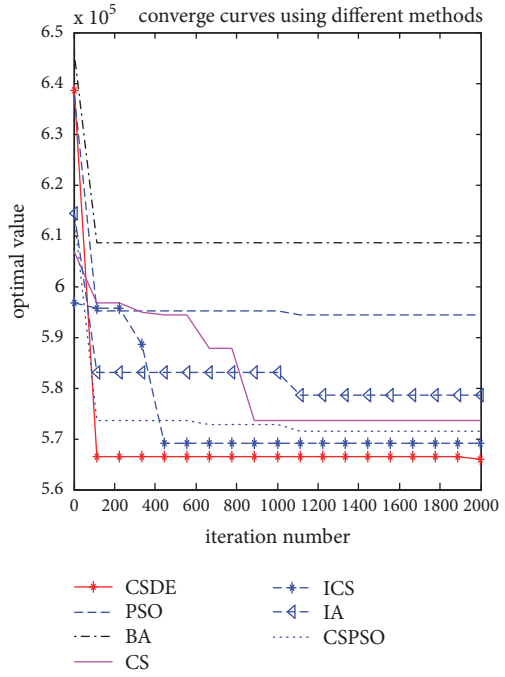


FIGURE 25: Convergence curve of seven algorithms for the best result.

Data Availability

All data included in this study are available upon request by contact with the corresponding author.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The work is supported by the Natural Science Foundation of Hubei Province, China (no. 2015cfb586), and the National Natural Science Foundation of China (no. 51567008).

References

[1] R. Gevaers, E. Voorde, and T. Vanelslander, "Characteristics of Innovations in Last-Mile Logistics—Using Best Practices, Case Studies And Making The Link with Green And Sustainable Logistic," in *Proceedings of the European Transport Conference, Freight and Logistics Track*, pp. 1–21, 2009.

[2] I. Bongartz, P. H. Calamai, and A. R. Conn, "A Projection Method for l_p Norm Location-Allocation Problems," *Mathematical Programming*, vol. 66, no. 1–3, pp. 283–312, 1994.

[3] C. Rao, M. Goh, Y. Zhao, and J. Zheng, "Location Selection of City Logistics Centers under Sustainability," *Transportation Research Part D: Transport and Environment*, vol. 36, pp. 29–44, 2015.

[4] M. G. Resende and R. F. Werneck, "A Hybrid Multistart Heuristic for The Uncapacitated Facility Location Problem," *European Journal of Operational Research*, vol. 174, no. 1, pp. 54–68, 2006.

[5] A. Weber and C. J. Friedrich, *Alfred Weber's Theory of The Location of Industries*, vol. 23, University of Chicago Press, Illinois, Ill, USA, 1929.

[6] S. L. Hakimi, "Optimum Locations of Switching Centers And The Absolute Centers And Medians of A Graph," *Operations Research*, vol. 12, no. 3, pp. 450–459, 1964.

[7] C. H. Aikens, "Facility Location Models for Distribution Planning," *European Journal of Operational Research*, vol. 22, no. 3, pp. 263–279, 1985.

[8] A. Klose and A. Drexl, "Facility Location Models for Distribution System Design," *European Journal of Operational Research*, vol. 162, no. 1, pp. 4–29, 2005.

[9] H. Sun, Z. Gao, and J. Wu, "A Bi-Level Programming Model And Solution Algorithm for The Location of Logistics Distribution Centers," *Applied Mathematical Modelling*, vol. 32, no. 4, pp. 610–616, 2008.

[10] S.-Y. Chou, Y.-H. Chang, and C.-Y. Shen, "A Fuzzy Simple Additive Weighting System under Group Decision-Making for Facility Location Selection with Objective/Subjective Attributes," *European Journal of Operational Research*, vol. 189, no. 1, pp. 132–145, 2008.

[11] C. Kahraman, D. Ruan, and I. Doğan, "Fuzzy Group Decision-Making for Facility Location Selection," *Information Sciences*, vol. 157, no. 1–4, pp. 135–153, 2003.

[12] J. Heizer and B. Render, *Principles of Operations Management*, Prentice Hall, New Jersey, USA, 2004.

[13] R.-L. Tang, Z. Wu, and Y.-J. Fang, "Adaptive Multi-Context Cooperatively Coevolving Particle Swarm Optimization for Large-Scale Problems," *Soft Computing*, vol. 21, no. 16, pp. 4735–4754, 2017.

[14] C. H. Peng, L. Xu, X. Gong, H. J. Sun, and L. Pan, "Molecular Evolution Based Dynamic Reconfiguration of Distribution Networks with Dgs considering Three-Phase Balance And Switching Times," *IEEE Transactions on Industrial Informatics*, pp. 1-1, 2018.

[15] S. Li, "A 1.488 approximation algorithm for the uncapacitated facility location problem," *Information and Computation*, vol. 222, pp. 45–58, 2013.

[16] S. Elhedhli and R. Merrick, "Green supply chain network design to reduce carbon emissions," *Transportation Research Part D: Transport and Environment*, vol. 17, no. 5, pp. 370–379, 2012.

[17] A. Rahmani and S. A. MirHassani, "A hybrid firefly-genetic algorithm for the capacitated facility location problem," *Information Sciences*, vol. 283, pp. 70–78, 2014.

[18] S. C. Ho, "An Iterated Tabu Search Heuristic for The Single Source Capacitated Facility Location Problem," *Applied Soft Computing*, vol. 27, pp. 169–178, 2015.

[19] T. Thongdee and R. Pitakaso, "Differential Evolution Algorithms Solving A Multi-Objective, Source And Stage Location-Allocation Problem," *Industrial Engineering & Management Systems*, vol. 14, no. 1, pp. 11–21, 2015.

[20] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing*, pp. 210–214, 2009.

[21] S. Burnwal and S. Deb, "Scheduling Optimization of Flexible Manufacturing System Using Cuckoo Search-Based Approach," *The International Journal of Advanced Manufacturing Technology*, vol. 64, no. 5-8, pp. 951–959, 2013.

[22] A. H. Gandomt, X. S. Yang, and A. H. Alavi, "Cuckoo Search Algorithm: A Metaheuristic Approach to Solve Structural Optimization Problem," *Engineering with Computers*, vol. 29, no. 1, pp. 17–35, 2013.

[23] R. Storn and K. Price, *Differential Evolution-A Simple And Efficient Adaptive Scheme for Global Optimization over Continuous Spaces*, ICSI, Berkeley, Calif, USA, 1995.

[24] A. Jha, K. Somani, M. K. Tiwari, F. T. S. Chan, and K. J. Fernandes, "Minimizing transportation cost of a joint inventory location model using modified adaptive differential evolution algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 60, no. 1–4, pp. 329–341, 2012.

[25] L. Wang, H. Qu, T. Chen, and F.-P. Yan, "An Effective Hybrid Self-Adapting Differential Evolution Algorithm for The Joint Replenishment And Location-Inventory Problem in A Three-Level Supply Chain," *The Scientific World Journal*, vol. 2013, pp. 1–11, 2013.

[26] W. Long, X. Liang, Y. Huang, and Y. Chen, "An Effective Hybrid Cuckoo Search Algorithm for Constrained Global Optimization," *Neural Computing and Applications*, vol. 25, no. 3-4, pp. 911–926, 2014.

[27] J. Huang, X. Li, and L. Gao, "A New Hybrid Algorithm for Unconstrained Optimisation Problems," *International Journal of Computer Applications in Technology*, vol. 46, no. 3, pp. 187–194, 2013.

[28] Z.-F. Hao, G.-H. Guo, and H. Huang, "A Particle Swarm Optimization Algorithm with Differential Evolution," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 1031–1035, IEEE, Hong Kong, August 2007.

[29] Y.-X. Su and R. Chi, "Multi-Objective Particle Swarm-Differential Evolution Algorithm," *Neural Computing and Applications*, vol. 28, no. 2, pp. 407–418, 2017.

[30] F. Shi, H. Wang, L. Yu, and F. Hu, *30 Cases Analysis of Intelligent Algorithm Based on Matlab*, Beihang University Press, Beijing, Chinese, 2011.

[31] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[32] W.-J. Zhang and X.-F. Xie, "DEPSO: Hybrid Particle Swarm with Differential Evolution Operator," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 4, pp. 3816–3821, IEEE, Washington, DC, USA, October 2003.

[33] L. H. Wu, Y. N. Wang, and Z. L. Chen, "Modified Differential Evolution Algorithm for Mixed-Integer Non-Linear Programming Problems," *Journal of Chinese Computer Systems*, vol. 28, no. 4, pp. 666–669, 2007.

[34] X. Yan, J. Yu, and F. Qian, "Kinetic Parameter Estimation of Oxidation in Supercritical Water Based on Modified Differential Evolution," *Journal of East China University of Science and Technology (Natural Science Edition)*, vol. 32, no. 1, pp. 94–97, 2006.

[35] L. Wang, F. Zou, X. Hei et al., "A Hybridization of Teaching–Learning-Based Optimization And Differential Evolution for Chaotic Time Series Prediction," *Neural Computing and Applications*, vol. 25, no. 6, pp. 1407–1422, 2014.

[36] P. N. Suganthan, N. Hansen, J. J. Liang et al., "Problem Definitions And Evaluation Criteria for The CEC 2005 Special Session on Real Parameter Optimization," Technical Report, Nanyang Technological University 2005005, Nanyang Technological University, Singapore, 2005.

[37] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.

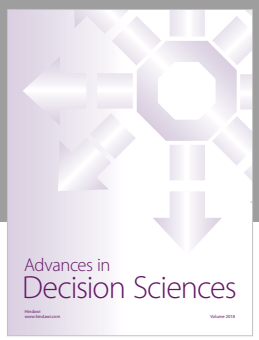
[38] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," in *Nature Inspired Cooperative Strategies for Optimization, Studies in Computational Intelligence*, Studies in Computational Intelligence, pp. 65–74, Springer, Berlin, Germany, 2010.

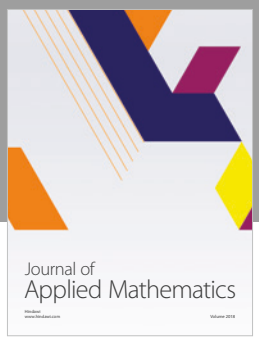
[39] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A New Heuristic Optimization Algorithm: Harmony Search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

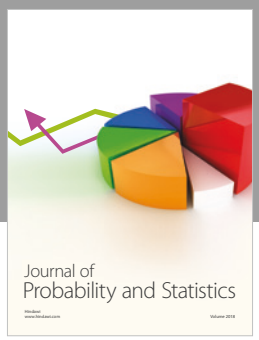
[40] E. Valian, S. Tavakoli, S. Mohanna, and A. Haghi, "Improved Cuckoo Search for Reliability Optimization Problems," *Computers & Industrial Engineering*, vol. 64, no. 1, pp. 459–468, 2013.

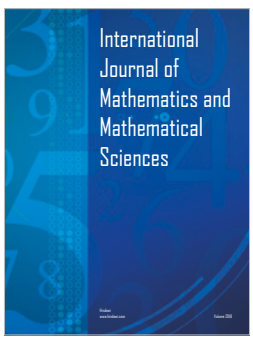
[41] R. Chi, Y. Su, D. Zhang, X. Chi, and H. Zhang, "A Hybridization of Cuckoo Search And Particle Swarm Optimization for Solving Optimization Problems," *Neural Computing and Applications*, pp. 1–18, 2017.

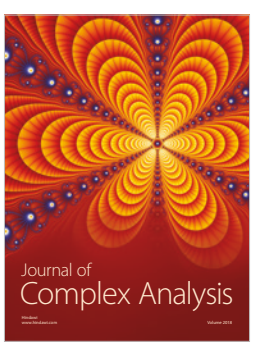
[42] K. M. Woldemariam and G. G. Yen, "Vaccine-Enhanced Artificial Immune System for Multimodal Function Optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, no. 1, pp. 218–228, 2010.

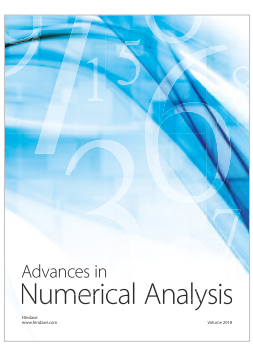


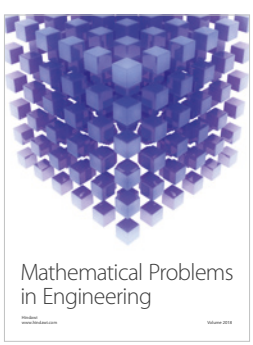


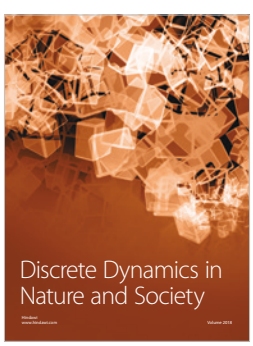


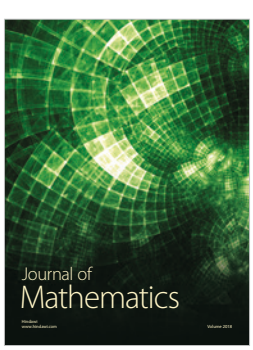


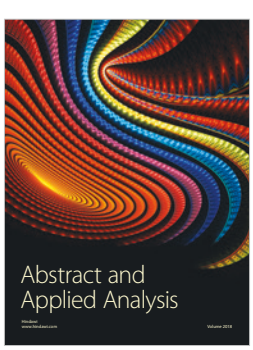


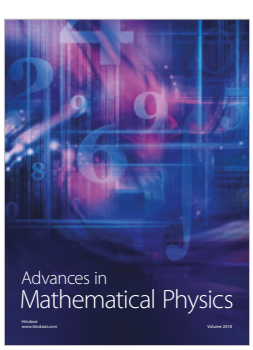












Hindawi

Submit your manuscripts at
www.hindawi.com