Research Article

Hierarchical Clustering Cuckoo Search Optimization Implemented in Optimal Setting of Directional Overcurrent Relays

Lvbin Ma and Jiangtao Yu

1Zhejiang Huayun Information Technology Co Ltd, Hangzhou, Zhejiang 310012, China
2College of Mechanical and Electrical Engineering, China Jiliang University, No. 258 Xueyuan Street, Hangzhou, Zhejiang 310012, China
3Department of Electrical Engineering, Yeungnam University, 280 Daehak-Ro, Kyonsan 38541, Republic of Korea

Correspondence should be addressed to Jiangtao Yu; yujiangtao0221@163.com

Received 3 May 2022; Revised 10 July 2022; Accepted 15 July 2022; Published 30 August 2022

Copyright © 2022 Lvbin Ma and Jiangtao Yu. 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.

This study has proposed a hierarchical clustering cuckoo search optimization (HCSO) to solve the optimal setting problems of directional overcurrent relays (DOCRs) in the power system. Different from the randomly generated stepsize $\theta$ in traditional CSO, stepsize $\theta$ in HCSO is updated by a hierarchical clustering mechanism, which can balance the searching priority of exploration and exploitation for each solution. The superiority of HCSO is evaluated by solving the DOCRs coordination problem with the 3-bus, 8-bus, and 9-bus system, presented by linear programming (LP), nonlinear programming (NLP), and mixed integer nonlinear programming (MINLP) forms. Results show that under the same parameter setting conditions, the minimum value of the objective function by HCSO has reduced by 10.93%, 2.93%, and 0.46%, respectively, compared to that of CSO in the experimented cases 2, 3, and 4. It verifies that HCSO is more competitive than CSO in solving the DOCRs optimal setting problem irrespective of the DOCRs number and formulation type.

1. Introduction

Relays coordination is important in power system protection. The purpose of relays coordination is to cut the faulted parts as fast as possible to maintain services by the rest of the normal parts. Due to great progress in directional overcurrent relays (DOCRs) technology, they have been employed in the relays coordination field in the power system.

Modern optimization algorithms are used to solve the optimal setting problems of DOCRs. GA and three improved versions of GA were used in Refs. [1–3], while two modified PSO were used in Refs. [4, 5]. In Ref. [6], the DOCRs problem is regarded as NLP and MINLP formulation, and solved by a Seeker algorithm. In Ref. [7], IGSO is proposed to enhance GSO’s searching ability for solving DOCRs with NLP formulation. In Ref. [8], BBO-LP is proposed to cope with the optimal coordination of DOCRs with MINLP formulation. In Ref. [9], MEFO is proposed by modifying the expression of the force for each EMP, and it is proved to be effective in minimizing the operating time for the DOCRs problem. Recently, an adaptive coordination scheme of numerical DOCRs is proposed in Ref. [10] by utilizing a mathematical programming language based interior point optimization solver. In Ref. [11], optimal settings of DOCRs with different characteristic curves for AC microgrids are presented. Enhanced versions of GWO are proposed to solve the coordination of DOCRs in Refs. [12, 13]. WCA is recently applied to this field, and the results are promising [14, 15]. Most lately, the DOCRs problem is solved by the newly proposed algorithms such as ISOA and GSA-SQP [16, 17].
Cuckoo search optimization (CSO) is developed by imitating the obligate brood parasitic behavior of some cuckoo species together with the Lévy flights behavior of some birds and fruit flies [18]. Though CSO has only one parameter to be tuned, it can balance the exploration and exploitation very well to reach prominent results in continuous and discrete optimization problems. For example, in Ref. [19], CSO with fuzzy logic and Gauss–Cauchy is used for minimizing the localization errors of WSN. In Ref. [20], the traveling salesman problem and path planning for the autonomous trolley inside the warehouse is solved by a discrete variant of CSO. In Ref. [21], the combined heat and power economic dispatch problem is solved by an adaptive CSO. In Ref. [22], a multiobjective CSO is used to optimize the pathways for 75% renewable electricity mix by 2050 in Algeria. The implementations above have verified CSO’s performances in a variety of scientific and engineering fields.

However, some works are devoted to developing new parameter control methods to make CSO better suited to various problems. For example, in Ref. [23], Sekhar and Mohanty suggested that the probability and stepsize in CSO should better be adopted as a dynamic variable to improve the solution quality. In Ref. [24], Nguyen and Vo suggested to merge new solutions generated from both Lévy flights and replacement of a fraction of eggs together in order to evaluate and rank the solutions at only one time, which is quite different from traditional CSO. Recently in Ref. [25], a four-solution factor, a fitness ratio $FR_d$, and a predetermined threshold $\varepsilon_d$ were introduced to update the step size in CSO to extend the feasible searching zones. Actually, in order to make CSO more effective and efficient, there are two issues that should always be considered:

(1) How to construct a proper host nest structure for the cuckoos
(2) How to guide each cuckoo lay its eggs on more promising host nests

To answer the first question, a hierarchical clustering mechanism is introduced to construct the host nests in this article. At the beginning, each host nest is regarded as a separate hierarchy, then the hierarchies are merged successively into larger hierarchies according to the similarity of each hierarchy, and then the hierarchy for host nests (sub-swarms) is dynamically build. This dynamic procedure can effectively improve the host nest structure by sharing information among different hierarchies via merging the nearest two hierarchies together. For the second question, since the host nests lie in different hierarchy, they own different abilities, and we take the average objective function value as an example. For each hierarchy, if its average objective function value ($\psi$) is at an inferior state, then we speed up its global search to escape from its current area by guiding the cuckoos lay its eggs at farther nests; if its average objective function value ($\psi$) is at a superior state, then we perform local search to exploit better nests nearby by guiding the cuckoos lay its eggs in neighborhood. Based on the two answers, a hierarchical clustering cuckoo search optimization (HCSO) is proposed in this article. Specifically, the main contributions of this article are as follows:

(1) The objective function of DOCRs optimal setting problem is tested by three types as LP, NLP, and MINLP
(2) HCSO is proposed by replacing the randomly generated stepsize with a hierarchical clustering stepsize
(3) The sensitivity of stepsize in HCSO is analyzed on benchmark functions
(4) The superiority of HCSO is verified by solving the DOCRs optimal setting problem

Rest of this article is arranged as follows. In Section 2, the objective function and constrained functions of DOCRs optimal setting problem are illustrated. Related works on HCSO are analyzed in detail in Section 3, including a brief introduction of CSO, hierarchical clustering mechanism, hierarchical clustering stepsize $\theta$, new solutions generated by hierarchical clustering stepsize $\theta$, and alien eggs discovery action. The sensitivity analysis of stepsize $\theta$ by a series of constant values vs a hierarchical value is carried on benchmark functions in Section 4. Experimental cases on DOCRs optimal setting problem with 3-bus, 8-bus, and 9-bus systems by LP, NLP, and MINLP formulations are conducted in Section 5, and the results and comparisons are provided simultaneously. Discussions on the effects of egg abandon fraction $P_s$ and hierarchy number $|O|$ are presented in Section 6. Finally, conclusions are given in Section 7.

2. Problem Formulation

2.1. Objective Function. From a mathematical point of view, the DOCRs optimal setting problem is an engineering optimization problem, and the goal of the objective function (OF) is to minimize the summation of the working time of all the primary DOCRs, as expressed below:

$$\chi = \sum_{i=1}^{N} v_i Q_i,$$

where $\chi$ denotes the OF value, $N$ denotes the number of primary DOCRs, and $v_i$ denotes the relay coefficient and is set to be 1 for all the DOCRs throughout this article. $Q_i$ denotes the working time of relay $i$, which is calculated by

$$Q_i = TDS_i \times \frac{\phi}{(IF/IP)^\gamma - \eta} + \zeta,$$

$$IP_i = PS_i \times CTR_i,$$

where $\phi$, $\rho$, $\eta$, and $\zeta$ are coefficients with fixed value as 0.14, 0.02, 1.0, and 0, respectively, according to the IEC curves. IF denotes the fault current, IP denotes the pickup current, and CTR denotes the CT ratio.
2.2. Constrained Functions

2.2.1. DOCRs Characteristic Constraints. The DOCRs characteristic constraints are the physical limitations which are given as

\[ Q_{i}^{\min} \leq Q_{i} \leq Q_{i}^{\max}, \]
\[ TDS_{i}^{\min} \leq TDS_{i} \leq TDS_{i}^{\max}, \]
\[ PS_{i}^{\min} \leq PS_{i} \leq PS_{i}^{\max}, \]

where Eq. (4)–Eq. (6) are the minimum and maximum bounds for the abovementioned parameters in Eq. (1)–Eq. (3).

2.2.2. DOCRs Coordination Constraint. When failure happens in the power system, the primary DOCR and the backup DOCR will detect it at the same time. This DOCRs coordination constraint ensures that the primary and backup DOCRs operate accordingly and prevent unnecessary tripping or uncoordinated tripping. To ensure proper coordination, the operating time for the backup DOCR must be more than the operating time of the primary DOCR by the predefined coordination time interval (CTI), which is defined as follows:

\[ Q^{bc} - Q^{pri} \geq CTI, \]

where \( Q^{bc} \) and \( Q^{pri} \) denote the operating time of the backup and primary DOCRs.

3. Hierarchical Clustering Cuckoo Search Optimization (HCSO)

3.1. Original Cuckoo Search Optimization (CSO). There are three basic rules in original CSO, which could be referred to Ref. [18]. On one side, original CSO has been proved to be super effective for lots of kinds of real-world optimization problem since it keeps excellent balance between global search and local search by switching the parameter \( p_{a} \in [0, 1] \). On the other side, original CSO is suffering difficulties of achieving high-quality solutions and fast convergence speeds due to the randomly generated value for stepsize \( \theta \) within \([0, 1]\), which cannot adapt to the environmental changes as the iteration goes on.

However, in this article, the cluster analysis is combined with original CSO to replace the fixed value of stepsize \( \theta \) instead of an adaptive hierarchical clustering stepsize \( \theta \); hence, the disadvantage of CSO mentioned ahead is solved.

3.2. Hierarchical Clustering Mechanism. Hierarchical clustering is a member of cluster family, and fundamental information has been provided in Ref. [26]. It needs to be noted that in this article, hierarchical clustering refers to the agglomerative clustering and not the divisive clustering.

Supposing that set \( H = \{ h_{1}; h_{2}; \ldots; h_{M} \} \) contains \( M \) solutions, where the \( m^{th} \) solution is described as \( h_{m} = \{ h_{m,1}, h_{m,2}, h_{m,3}, \ldots, h_{m,N} \} \) with \( N \) dimensions, the fundamental mechanism of hierarchical clustering can be as illustrated below:

B1: Every row in \( H \) is regarded as a separate hierarchy \( O \) with only one solution, which is \( O_{m} = \{ h_{m} \} \), where \( m = \{1, 2, \ldots, M\} \)

B2: Euclidean distance \( \text{dis}(h_{\alpha}, h_{\beta}) \) between any two solutions \( h_{\alpha} \) and \( h_{\beta} \) is calculated by

\[ \text{dis}(h_{\alpha}, h_{\beta}) = \sqrt{\sum_{i=1}^{N} (h_{\alpha,i} - h_{\beta,i})^2}, \]

where \( \alpha, \beta \in \{1, 2, \ldots, M\} \), \( N \) is the dimension number of \( h_{\alpha} \) and \( h_{\beta} \)

B3: The distance matrix \( \text{Dis} = \{ \text{dis}_{ij} \}_{O \times (O)} \) is built, where \( \text{dis}_{ij} = \text{dis}(h_{i}, h_{j}) \), where \(|O|\) is the present hierarchy number

B4: From the matrix Dis, choose the closest two hierarchies, suppose they are \( O_{\gamma} \) and \( O_{\delta} \), then combine them together as a new hierarchy \( O_{\gamma} \), which is \( O_{\gamma} = O_{\gamma} \cup C_{\delta} \), meanwhile, update \(|O| = |O| - 1\)

Here, note that the single linkage (SL) formula is adopted to measure the distance between the \( O_{\gamma} \) and \( O_{\delta} \) as follows:

\[ \text{dis}(O_{\gamma}, O_{\delta}) = \min_{h \in O_{\gamma}, k \in O_{\delta}} \text{dis}(h, k), \]

where \( \text{dis}(h, k) \) is calculated by equation (8)

B5: Go to B2 until the cutoff condition is satisfied

In addition, here, three points need to be clarified in particular. First, in B4, to measure the distance between \( O_{\gamma} \) and \( O_{\delta} \), there are three ways as complete linkage (CL), single linkage (SL), and average linkage (AL). In this article, we adopt SL as the way. But the other two ways as CL and AL are discussed and compared in Section 6.3 to comprehensively analyze the performances of hierarchical clustering mechanism.

Second, in B5, there are different ways for “cutoff condition”. In Ref. [27], the maximum number of hierarchies was set as the “cutoff condition.” In Ref. [28], the authors used the parameter subSize, which is a prefixed maximum subswarm size, to control the “cutoff condition.” In this article [27], the maximum number of hierarchies (denoted as \(|O|\)) is used to be the “cutoff condition,” and the effects of \(|O|\) is discussed in Section 6.2.
Third, the hierarchical clustering mechanism in this article is extended from Ref. [29]. In Ref. [29], the clustering task is fulfilled by a 4-step $K$-means algorithm, while in this article, it is by a 5-step agglomerative clustering method. Moreover, Ref. [29] focused on the economic load dispatch (ELD) problem with NLP formulations only, but this article extended its applications on various formulations as LP, NLP, and MINLP in the DOCRs coordination problem (see Section 5).

3.3. Hierarchical Clustering Stepsize. Based on Section 3.2, all solutions in $H$ are classified into different hierarchies. Because different solution $h_m$ owns different objective function (OF) value; hence, each hierarchy has a different average OF value as well. Thus, we can classify the hierarchies based on this value to find a targeted way to improve its searching ability. To describe the hierarchical clustering stepsize clearly, the following definitions are given first.

**Definition 1.** The hierarchy $\sigma$ is denoted as $O_\sigma$. For $O_\sigma$, its solution size is expressed as $|O_\sigma|$. Suppose solution $h_m$ is within hierarchy $O_\sigma$, then the OF value of $h_m$ is denoted as $f_{m,\sigma}$; hence, the average OF value of hierarchy $O_\sigma$ is

$$\psi_\sigma = \frac{\sum_{m=1}^{|O_\sigma|} f_{m,\sigma}}{|O_\sigma|}$$  

(10)

**Definition 2.** For the whole solution set $H$, suppose $f_m$ is the OF value of solution $h_m$ within $H$, then the average OF value of $H$ is

$$\psi = \frac{\sum_{m=1}^M f_m}{M}$$  

(11)

where $M$ is the solution size of $H$.

Next, for hierarchy $O_\sigma$, let us compare $\psi_\sigma$ with $\Psi$, and see what does the comparison mean to evaluate the state of $O_\sigma$. Assuming that the to-be-solved problem is a minimization type, then two different situations will appear as follows:

**Situation 1.** $\psi_\sigma \geq \Psi$, in this situation, hierarchy $O_\sigma$ is in a relatively poorer position because its average OF value ($\psi_\sigma$) is greater than that of the entire solution set ($\Psi$). Therefore, the searching range of hierarchy $O_\sigma$ should be increased appropriately to find more competitive solutions.

**Situation 2.** $\psi_\sigma < \Psi$, in this situation, hierarchy $O_\sigma$ is in a relatively better position. Hence, the searching scope of hierarchy $O_\sigma$ should be reduced to enhance the local search and to avoid missing the optimal solutions in neighborhood.

According to the analysis above, hierarchical clustering stepsize (denoted as $\theta_\sigma$) for hierarchy $O_\sigma$ is proposed, which is calculated as follows:

$$\theta_\sigma = \begin{cases} \left[ \frac{(U - (U - U) \times \Psi_\sigma)}{\Psi_\sigma} \right]^2, & \text{if } \psi_\sigma \geq \Psi \\ \left[ \frac{(L - (L - L) \times \Psi - \psi_\sigma)}{\Psi} \right]^2, & \text{if } \psi_\sigma < \Psi \end{cases}$$  

(12)

where $U$ and $U$ are the upper-limit and lower-limit for stepsize in Situation 1, which is chosen to be 1.0 and 0.5, respectively, in this article; $L$ and $L$ are the upper-limit and lower-limit for stepsize in Situation 2, which is chosen to be 0.5 and 0, respectively, in this article. Here, we should note that the chosen values for $U$, $U$, $L$, and $L$ are just examples to illustrate the hierarchical clustering mechanism, and these values are maintained in Sections 4 and 5. However, they may not be the best values for other problems, and different problems may prefer different choices.

In original CSO, stepsize $\theta$ is a fixed value within $[0, 1]$ [18]. Differently, in this article, hierarchical clustering stepsize $\theta_\sigma$ is introduced to modify its value according to different hierarchy conditions, which is expressed as follows:

**Condition 1.** If solution $h_m \in O_\sigma$ and $O_\sigma$ belongs to Situation 1, then stepsize $\theta_\sigma$ is updated from $U$ to $U$. Meanwhile, based on Equation (12), the larger the value of $\psi_\sigma$, the smaller the distance between $\theta_\sigma$ and $U$; then a bigger stepsize $\theta_\sigma$ for $h_m$ is made according to the current state of hierarchy $O_\sigma$. Therefore, the new solutions are enabled to explore wider searching ranges in a unique and targeted manner.

**Condition 2.** If solution $h_m \in O_\sigma$ and $O_\sigma$ belongs to Situation 2, then $\theta_\sigma$ is updated from $L$ to $L$ to narrow its searching area. Similar to Condition 1, the smaller the value of $\psi_\sigma$, the smaller the distance between $\theta_\sigma$ and $L$, then a smaller stepsize $\theta_\sigma$ for $h_m$ is made according to the current state of $O_\sigma$. Hence, the updated solutions can adaptively perform a more targeted neighboring search to achieve the global optimal solution.

3.4. New Solutions via Hierarchical Clustering Stepsize. In original CSO, there are two phases of generating new solutions including Lévy flights and the replacement of a fraction of eggs [18]. However, in HCSO, the generation of new solution $h^\text{new}_m$ via hierarchical clustering stepsize $\theta_\sigma$ based on Lévy flights is described as follows:

$$h^\text{new}_m = h_{m,\sigma} + \theta_\sigma \times \text{step} \times (h_{m,\sigma} - h_{\text{best}}) \times \text{Levy} (\beta),$$  

(13)

where $\text{step} \in [0, 1]$ is the standard step in original CSO, $\theta_\sigma$ is the stepsize by Equation (12), $h_{m,\sigma}$ is the $m^{th}$ solution belongs to hierarchy $\sigma$, $h_{\text{best}}$ is the best solution at present. It is noted that the definition and calculation process of Lévy ($\beta$) are all drawn from [24], and it is not repeated here because of the limited space.
Here, we should mention that in Equation (13), except for \( \theta_o \), the original step is also important for new solutions. If the step is too large, the new solutions may vary in a very large range that the optimal solution is either obtained fast or omitted; however, if it is too small, the new solutions may move around such a small area that it cannot contribute to the solution diversity. This is the reason why we combine \( \theta_o \) with the step.

3.5. Alien Eggs Discovery Action. In cuckoo’s behavior, there is a possibility that alien eggs will be discovered by the host bird, and this discovery action with probability \( p_a \) is expressed as

\[
h_m^{\text{dis}} = h_m^* + p_a \times \Delta h_m^{\text{dis}}, \tag{14}
\]

where the explanation of probability \( p_a \) and the calculation of increased value \( \Delta h_m^{\text{dis}} \) are as the same as in Ref. [24], so it is not repeated here.

3.6. Steps of Implementation. Based on the hierarchical clustering stepsize, a hierarchical clustering cuckoo search optimization (HCSO) is proposed. The steps and pseudo code of implementing HCSO on solving the optimal setting of DOCRs problem are demonstrated in the following. Here we should note that the penalty function, which is denoted as \( \Delta(s) \) in equation (15), is applied to handle the DOCRs coordination constraint [30]. Thus, the constrained objective function is transformed into an unconstrained one as follows:

\[
\chi = \sum_{i=1}^{N} Q_i^{\text{pri}} + \sum_{i=1}^{I} \Delta(s), \tag{15}
\]

where

\[
\Delta(s) = \begin{cases} 
0, & \text{if } (Q_{bc}^i - Q_{pr}^i) \geq \text{CTI} \\
\xi \times \text{CTI} - (Q_{bc}^i - Q_{pr}^i), & \text{otherwise}
\end{cases},
\tag{16}
\]

where \( \xi \) is the penalty factor and should be a relatively large value to achieve zero penalties in optimal solutions.

1. C1. Set values for solution size \( \alpha_{\text{sol}} \), maximum number of iterations \( \alpha_{\text{iter}} \), fraction of abandoned eggs \( p_a \), and number of hierarchies \( |O| \)
2. C2. Initial solution \( H \) is produced by a matrix of \([\alpha_{\text{sol}} \times \alpha_{\text{var}}]\), which includes \( \alpha_{\text{sol}} \) solutions, where each solution includes \( \alpha_{\text{var}} \) variables
3. C3. Repair each solution \( h_m \) by equations (5) and (6), where \( m \in \{1, 2, 3, \ldots, \alpha_{\text{sol}}\} \)
4. C4. Check if equation (7) is satisfied; if not, the violations is calculated by equation (16)
5. C5. OF value is calculated by equation (15)
6. C6. Set \( \text{iter} = 1 \)
7. C7. Choose \( h_{\text{test}} \) according to the OF value in C5
8. C8. Divide the whole \( H \) into \( |O| \) hierarchies according to B1–B5 in Section 3.2

(9) C9. Calculate \( \psi_o \) and \( \Psi \) by (10) and (11)
(10) C10. The hierarchical clustering stepsize \( \theta_o \) is obtained by (12)
(11) C11. New solutions \( h_{\text{new}} \) are updated by (13)
(12) C12. Repeat C3–C4 to check the constraint
(13) C13. Repeat C5 to obtain the new OF value
(14) C14. New solutions \( h_{\text{dis}} \) are updated by (14)
(15) C15. Repeat C3–C4 to check the constraint
(16) C16. Repeat C5 to obtain the new OF value
(17) C17. If \( \alpha_{\text{iter}} \) is reached, stop the loop; otherwise, set \( \text{iter} = \text{iter} + 1 \) and go to C6 to continue the loop

4. Benchmark Functions for Stepsize Analysis

To observe the sensitivity of stepsize \( \theta \) for HCSO, six benchmark functions as Sphere, Rosenbrock, Ackley, Rastrigin, Griewank, and Weierstrass with \( \text{dim} = 100 \) are tested in this section [31, 32]. The value of \( \theta \) is from \{0.01, 0.1, 0.25, 0.5, 0.75, 1.0\}, hierarchical clustering value]. All the benchmark functions are given the same conditions as \( \alpha_{\text{sol}} = 30, \alpha_{\text{iter}} = 5000, p_a = 0.1, \) and hierarchy number \( |O| = 10 \).

Figure 1 shows the convergence curves with the same randomly generated initial solution \( H \). In general, when \( \theta = 0.01 \) (blue color) or \( \theta = 1.0 \) (red color), all the convergence curves from Figures 1(a)–1(f) show the worst performance. This is because when \( \theta = 0.01 \), it is so small that the exploitation range is limited in a very small space; hence, it cannot perform a comprehensive search. On the contrary, when \( \theta = 1.0 \), it is so large that the convergent performance is affected adversely. However, when it comes to the hierarchical clustering value (in brown), the fastest convergent speed can be observed with the lowest \( y \)-axis value for all the six functions. It is not hard to understand that constant \( \theta \) cannot adjust to the environmental changes as the iteration carries on, but hierarchical clustering stepsize \( \theta \) can update its value according to the hierarchy changes for every iteration, thereby improving the searching balance between exploration and exploitation of each individual solution. So it explains why hierarchical clustering stepsize \( \theta \) is more competitive than fixed stepsize \( \theta \).

Let us further take Figure 1(e) as an example, which is the convergence characteristic of Griewank function to further analyse the influences of stepsize \( \theta \). From Figure 1(e), \( \theta = \{0.1, 0.25, \text{hierarchical clustering step}\} \) is the top-performer since they converged to the global optima as \( y \)-axis = 0 (circled by dotted lines). Moreover, \( \theta = \) hierarchical clustering step converges with the fastest speed, followed by \( \theta = 0.25 \) and 0.1. It is also observed that the second-rate-performer is achieved by \( \theta = 0.5 \) and 0.75, and they have reached \( y \)-axis = \( 10^{-10} \) and \( y \)-axis = \( 10^{-8} \), which are relatively good results but not as good as the top-performer. Finally, the third-rate-performer belongs to \( \theta = 0.01 \) and 1.0, which is the smallest \( \theta \) and the largest \( \theta \), respectively, and they show the worst results. The reason has been explained above, so it is not repeated here.
5. Optimal Setting of the DOCRs Problem

5.1. Case 1: 3-Bus System with LP Formulation. This 3-bus system is shown in Figure 2. All the related data are provided in Ref. [6]. This case is formulated as the LP problem, and the design variable is TDS.

Parameter conditions are set as $\alpha_{sol} = 20$, $\alpha_{iter} = 50$, $\alpha_{var} = 6$ (6 TDS), $p_a = 0.1$, and $|O| = 3$. To observe the convergence characteristics of CSO and HCSO visually, Figure 3 picks one randomly chosen convergence curve among 30 running times. It shows that even though CSO and HCSO achieve the same result in the end, CSO takes up around 26 iterations, while HCSO takes up only 19 iterations, which shows HCSO has advantages over CSO in terms of convergence efficiency. Figure 4 is the outlines of OF value, and it is surprising that all the independent runs achieve the same optimal result (OF = 1.7804 s), no matter by HCSO or by CSO. The reason is that LP formula with six variables in this case is a simple optimization problem, so it is quite easy to achieve its best result.

The optimal settings of TDS by CSO and HCSO are given in Table 1, along with the simplex method [33], LP using MATLAB [4], PSO [4], the Seeker algorithm (SA) [6], and OJaya [34]. It is obvious that both CSO and HCSO achieved more optimized value as 1.7804(s) than 1.9258 (s). Meanwhile, the standard deviation (Std) reached 0 by both HCSO and CSO, which means every time of its independent run reaches the optima, which can also be observed in Figure 4.

Table 2 shows the operating times of primary and backup relays, as well as the CTI. We can tell that the CTI constraints are satisfied in every P/B pair.

5.2. Case 2: 8-Bus System with NLP Formulation. This 8-bus system is shown in Figure 5. All the related data are provided in [35]. This case is formulated as the NLP problem, and the design variables are TDS and PS, which are continuous values in $[0, 1]$ and $[0.5, 2.0]$, respectively. The CTI is selected as 0.3.

In this case, $\alpha_{sol} = 50$, $\alpha_{iter} = 20000$, $\alpha_{var} = 28$ (14 TDS + 14 PS), $p_a = 0.1$, and $|O| = 10$. The optimal settings of TDS and

\begin{algorithm}
\begin{algorithmic}
\State Set initial values for $\alpha_{sol}$, $\alpha_{iter}$, $\alpha_{display}$, $p_a$ and $|O|$;
\State Generate the initial solution $H = [\alpha_{sol} \times \alpha_{var}]$;
\State Check constraints (I) and (II);
\State Evaluate the objective function by Equation (15);
\State Set $\text{iter} = 1$;
\While{$\text{iter} < \alpha_{iter}$}
\State $\text{distance} = \text{pdist}(H,'euclidean')$;
\State $\text{cluster} = \text{linkage}(\text{distance}, 'single')$;
\State $\text{clu\_index} = \text{cluster}(\text{cluster}, 'maxclust', |O|)$;
\State Calculate $\psi_\sigma$ by (10);
\State Calculate $\Psi$ by (11);
\For{$m = 1 \rightarrow \alpha_{sol}$}
\State $\sigma = \text{clu\_index}(m)$;
\State Update stepsize $\theta_\sigma$ by (12) according to $\psi_\sigma$ and $\Psi$;
\State Update $h_{m, \sigma} \rightarrow h_{m, \text{new}}$ by (13);
\State Check the constraints and calculate $f(h_{m, \text{new}})$;
\If{$f(h_{m, \text{new}})$ is better than $f(h_{m, \sigma})$}
\State $h_{m, \sigma} = h_{m, \text{new}}$
\State $f(h_{m, \sigma}) = f(h_{m, \text{new}})$
\Else
\State Keep $h_{m, \sigma}$;
\EndIf
\EndFor
\EndWhile
\If{$\text{rand}[0, 1] < p_a$}
\State Build new nests by (14);
\EndIf
\If{$f(h_{m, \text{dis}})$ is better than $f(h_m)$}
\State $h_m = h_{m, \text{dis}}$
\State $f(h_m) = f(h_{m, \text{dis}})$
\Else
\State Keep $h_m$;
\EndIf
\EndIf
\State $\text{iter} = \text{iter} + 1$;
\EndAlgorithm
\end{algorithm}

Algorithm 1: HCSO.
Figure 1: Continued.
PS are displayed in Table 3. It is observed that the OF value by HCSO is 6.2083(s), while it is 6.9703(s) by CSO. The corresponding CTI for each pair of P/B relays are shown in Table 4.

The convergence curves are shown in Figure 6, from which we can see, generally speaking, CSO and HCSO converge at similar trends. However, if we observe the zoomed-in view in Figure 6(a), it shows that after 50 iterations, HCSO begins to outperform CSO. Then from Figure 6(b), it shows that from 10000 iterations to the end, HCSO keeps surpassing CSO all the time. The amplitudes of 30 running times are shown in Figure 7, it can be clearly observed that the OF value fluctuates in quite a large range by CSO but not that large by HCSO; meanwhile, the general curve by HCSO is below the curve by CSO, which means less OF values for most of the runs. Above analysis illustrates that in aspects of convergence rate, solution quality, and robustness, HCSO behaves much better than CSO.
The optimal results are being compared with published data in Table 5, and we can notice that HCSO achieves the best \( \text{OF}_\text{min} \) as 6.2083(s). Apart from \( \text{OF}_\text{min} \), HCSO achieves the best value of \( \text{OF}_\text{max} \), \( \text{OF}_\text{ave} \), and Std as well, among the compared methods.

Here, we should mention that there are unfeasible solutions by DE [9], PSO [9], and EM [9] with a number of 9/30 (9 unfeasible runs among 30 runs in total), 25/30, and 16/30. Moreover, the HHO [36] cannot converge to the optimal solutions because the CTI constraints are not satisfied. In fact, this case has a big number of constraints with a big number of decision spaces; hence, it is hard to achieve optimal solutions under the condition of satisfying all the constraints. However, HCSO has completed this task very well.

### 5.3. Case 3: 8-Bus System with MINLP Formulation

The system structure of this case is same as that of Case 2, but the system data are different, which is given in Refs. [2, 6]. The parameters are set as \( \alpha_{\text{ref}} = 50 \), \( \alpha_{\text{iter}} = 20000 \), \( p_u = 0.1 \), \( \alpha_{\text{var}} = 28 \), and \( |O| = 10 \). Continuous variable TDS is within \([0.1, 1.1]\), and discrete PS is chosen from \([0.5, 0.6, 0.8, 1.0, 1.5, 2.0, 2.5]\). CTI is selected to be 0.3(s).

Seen from the convergence curves in Figure 8, within around 4500 iterations, CSO and HCSO perform equally overall. However, as the iteration goes on, HCSO overtakes CSO in convergence rate and \( \text{OF} \) value. The reason is that hierarchic clustering stepsize \( \theta \) is able to adapt its value according to the solution changes; hence, HCSO provides a more targeted neighborhood search, while the CSO with fixed stepsize \( \theta \) cannot. The distribution of \( \text{OF} \) value is shown in Figure 9, from which we can see that among 30 runs, HCSO shows stronger ability in approaching the minimum \( \text{OF} \) value and maintaining its results stable (with Std as small as 0.2518), while CSO suffers several times of premature problem with relatively bigger \( \text{OF} \) value (with Std as 0.6024).

Optimum settings of TDS and PS are presented in Table 6. It should be noted that this case is a highly constrained network with limited number of discrete PS values, so it cannot get feasible and optimal solutions easily. Results obtained by Seeker [6], Olaya [34], GA [2], GA-LP [2], BBO [8], BBO-LP [8], Jaya [36], and HHO [36] are displayed in Table 7. It is noticed that GA [2] and GA-LP [2] are not capable of achieving feasible solutions, which has also been mentioned in Ref. [8]. HHO [36] suffers the same problem of unfeasible solutions because some P/B pairs cannot satisfy the CTI constraint. Even though the Seeker [6] provides a better \( \text{OF} \) value (8.4270s) than HCSO (8.6563s), some of its CTI constraints are not strictly satisfied. For example, the CTI for P/B pairs as 11/12 and 14/9 by Seeker is 0.2982s and...
0.2980 s, which is a little bit less than the required 0.3 s [34]. However, HCSO are able to strictly satisfy the CTI constraint because all the P/B pairs yield to CTI ≥ 0.3 s, as shown in Table 8. Hence, HCSO is still the best performer in this case among all the compared methods.

### 5.4 Case 4: 9-Bus System with NLP Formulation

This case is with single-end fed and equal impedance for all the lines, and 3 φ fault at the midpoint of each line is considered. All the DOCRs have the same CTR as 500:1, and the CTI is selected to be 0.2 s. The TDS is considered continuous in [0.025, 1.2]; PS is also considered continuous, and its maximum value and minimum value are given in Ref. [38]. The P/B pairs, the fault current, and the maximum and minimum fault current are given in Ref. [3]. The algorithm parameters are set as \( \alpha_{\text{sol}} = 30, \ \alpha_{\text{iter}} = 10000, \ \rho_a = 0.1, \ \alpha_{\text{var}} = 48 \) (24 TDS + 24 PS), and \(|O| = 15\).

The convergence characteristic is shown in Figure 10, and we can observe that during the whole iteration period, HCSO is always converging faster than CSO, and obtaining lower OF value as well. This is because HCSO has stronger searching ability than CSO by adjusting its hierarchical clustering stepsize \( \theta \) according to the current location of each solution. Moreover, when we come to Figure 11, the superiorities of HCSO over CSO become more obvious.

### Table 2: Operating times and CTI for case 1.

| P/R | B/R | CSO T_pr | CSO T_bc | CSO CTI | HCSO T_pr | HCSO T_bc | HCSO CTI |
|-----|-----|----------|----------|---------|------------|-----------|---------|
| 1   | 5   | 0.3641   | 0.8873   | 0.5232  | 0.3641     | 0.8873    | 0.5232  |
| 2   | 4   | 0.2094   | 0.8465   | 0.6371  | 0.2094     | 0.8465    | 0.6371  |
| 3   | 1   | 0.3216   | 0.9633   | 0.6417  | 0.3216     | 0.9633    | 0.6417  |
| 4   | 6   | 0.3390   | 0.8202   | 0.4812  | 0.3390     | 0.8202    | 0.4812  |
| 5   | 3   | 0.2319   | 1.0661   | 0.8342  | 0.2319     | 1.0661    | 0.8342  |
| 6   | 2   | 0.3144   | 0.7842   | 0.4698  | 0.3144     | 0.7842    | 0.4698  |

### Table 3: TDS and PS values for case 2.

| No. | TDS | PS  | No. | TDS | PS  | No. | TDS | PS  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 1   | 0.0803 | 2.0000 | 9   | 0.0500 | 1.8371 | 17  | 0.2600 | 1.8287 |
| 2   | 0.4400 | 0.5000 | 10  | 0.1892 | 0.5000 | 18  | 0.0804 | 2.0000 |
| 3   | 0.2843 | 0.7848 | 11  | 0.1716 | 1.9427 | 19  | 0.1716 | 1.9998 |
| 4   | 0.0799 | 1.9745 | 12  | 0.3947 | 0.7105 | 20  | 0.2216 | 1.9816 |
| 5   | 0.0510 | 1.2369 | 13  | 0.0804 | 2.0000 | 21  | 0.2600 | 1.8287 |
| 6   | 0.1718 | 1.7901 | 14  | 0.2600 | 1.8287 | 22  | 0.0500 | 1.8371 |
| 7   | 0.2832 | 1.5797 | 15  | 0.1716 | 1.9998 | 23  | 0.0502 | 1.6993 |
| 8   | 0.2515 | 0.7613 | 16  | 0.1716 | 1.9854 | 24  | 0.0502 | 1.6993 |
| 9   | 0.2717 | 1.3863 | 17  | 0.2600 | 1.8287 | 25  | 0.0502 | 1.6993 |
| 10  | 0.0500 | 1.8371 | 18  | 0.2600 | 1.8287 | 26  | 0.0502 | 1.6993 |
| 11  | 0.0502 | 1.6993 | 19  | 0.2600 | 1.8287 | 27  | 0.0502 | 1.6993 |

### Figure 5: IEEE 8-bus DOCRs coordination problem model.
because HCSO maintains its OF value within a much smaller area with respect to $y$-axis than CSO among 30 times of independent runs.

Optimum settings of TDS and PS are presented in Table 9. The comparisons with published data are presented in Table 10. It is noticed that the best result of OF_min is obtained by HCSO as 4.8000 s. In addition to OF_min, HCSO has also achieved the best results of OF_max and Std, and the Std reaches as less as 0.0180, which is a very obvious advantage over the other compared methods. Table 11 shows the operating time and CTI, and we can see that no selectivity constraint is violated by both CSO and HCSO.

### Table 4: Operating times and CTI for case 2.

| P/R | B/R | CSO | HCSO |
|-----|-----|-----|------|
|     |     | T_pr | T_bc | CTI | T_pr | T_bc | CTI |
| 1   | 6   | 0.3222 | 0.6468 | 0.3246 | 0.2898 | 0.5898 | 0.3000 |
| 2   | 1   | 0.7797 | 1.0823 | 0.3026 | 0.6644 | 0.9650 | 0.3007 |
| 2   | 7   | 0.7797 | 1.0807 | 0.3010 | 0.6644 | 0.9654 | 0.3011 |
| 3   | 2   | 0.5877 | 0.8967 | 0.3090 | 0.5316 | 0.8331 | 0.3014 |
| 4   | 3   | 0.3569 | 0.6734 | 0.3165 | 0.3454 | 0.6454 | 0.3000 |
| 5   | 4   | 0.2340 | 0.5347 | 0.3007 | 0.2193 | 0.5196 | 0.3003 |
| 6   | 5   | 0.4791 | 1.1586 | 0.6796 | 0.4871 | 0.9286 | 0.4415 |
| 6   | 14  | 0.4791 | 1.0808 | 0.6017 | 0.4871 | 0.9689 | 0.4818 |
| 7   | 5   | 0.6829 | 1.1586 | 0.4757 | 0.6254 | 0.9286 | 0.3032 |
| 7   | 13  | 0.6829 | 1.0815 | 0.3986 | 0.6254 | 0.9676 | 0.3422 |
| 8   | 7   | 0.5154 | 1.0807 | 0.5652 | 0.4204 | 0.9654 | 0.5451 |
| 8   | 9   | 0.5154 | 1.1039 | 0.5884 | 0.4204 | 0.8883 | 0.4679 |
| 9   | 10  | 0.2186 | 0.5227 | 0.3041 | 0.2090 | 0.5093 | 0.3003 |
| 9   | 11  | 0.4345 | 0.7380 | 0.3035 | 0.3501 | 0.6501 | 0.3000 |
| 11  | 12  | 0.5861 | 0.8892 | 0.3031 | 0.5145 | 0.8169 | 0.3024 |
| 12  | 13  | 0.7733 | 1.0815 | 0.3083 | 0.6667 | 0.9676 | 0.3009 |
| 12  | 14  | 0.7733 | 1.0808 | 0.3075 | 0.6667 | 0.9689 | 0.3022 |
| 13  | 8   | 0.3375 | 0.6580 | 0.3205 | 0.3021 | 0.6029 | 0.3007 |
| 14  | 1   | 0.6623 | 1.0823 | 0.4200 | 0.5824 | 0.9650 | 0.3826 |
| 14  | 9   | 0.6623 | 1.1039 | 0.4415 | 0.5824 | 0.8883 | 0.3058 |

### Figure 6: Comparison of convergence curves for case 2.

### Figure 7: Comparison of independent runs for case 2.

#### 6. Further Discussions

6.1. Effects of Egg Abandon Fraction $p_a$. Here, to observe the effects of fraction $p_a$ comprehensively, a series of $p_a$ from 0.1 to 1.0 in increments of 0.1 is experimented by 30 running times on the 9-bus system (Case 4). The other parameters as $\alpha_{tol}$, $\alpha_{iter}$, and $|O|$ are held to be 30, 300, and 10 during the whole process.

Observed from Figure 12, when $p_a = 0.9$ and 1.0, their performances are apparently different from the other $p_a$ because their $y$-axis range varies as large as $[6, 70]$ and $[30, 110]$, respectively, while $y$-axis for the other $p_a$ varies
Table 5: Comparisons of different methods for case 2.

| Algorithm | OF_max    | OF_ave    | OF_min    | Std      | Feasible |
|-----------|-----------|-----------|-----------|----------|----------|
| EFO [9]   | 10.8670   | 9.8870    | 7.6110    | 0.7670   | ○        |
| MEFO [9]  | 10.7300   | 8.6570    | 6.3490    | 1.1480   | ○        |
| DE [9]    | —         | —         | 6.6540    | —        | □/○      |
| PSO [9]   | —         | —         | 10.4210   | —        | □/○      |
| BH [9]    | 16.0910   | 12.8170   | 11.4010   | 0.8440   | ○        |
| EM [9]    | —         | —         | 15.9130   | —        | □/○      |
| BBO [9]   | 10.3040   | 8.6060    | 7.5470    | 0.5840   | ○        |
| HS [9]    | 13.3010   | 12.5250   | 11.7600   | 0.3970   | ○        |
| EM [36]   | —         | —         | 8.1996    | 3.0687   | ○        |
| HH [36]   | —         | —         | 3.7108    | 3.3709   | ×         |
| Jaya [36] | 8.0177    | 6.9731    | 6.9703    | 0.3375   | ○        |
| HCSO      | **6.8101**| **6.5929**| **6.2083**| **0.1521**| ○        |

Bold denotes the best result in that column. — means there is no data provided in the cited article. × denotes that the solution of “OF_min” in that row is unfeasible. □ denotes that the solution of “OF_min” in that row is feasible. □/○ denotes that there are both un-feasible and feasible solutions among independent runs.

Figure 8: Comparison of convergence curves for case 3.

Figure 9: Comparison of independent runs for case 3.
within a much smaller range as [5.5, 8.0] overall, which can be observed clearly from the zoomed-in view. This shows that when \( p_a < 0.9 \), its role in the final result is very limited. The authors believe it is because in original CSO, \( p_a \) plays an important role in the exploitation stage. However, in HCSO, the exploitation stage is performed by gradually adjusting stepsize \( \theta \) through the hierarchical clustering mechanism, so \( p_a \) is no longer as important as in CSO.

### Table 6: TDS and PS values for case 3.

| No. | TDS  | PS   | TDS  | PS   | No. | TDS  | PS   |
|-----|------|------|------|------|-----|------|------|
| 1   | 0.1000 | 2.5000 | 0.1000 | 2.5000 | 9   | 0.1606 | 2.5000 |
| 2   | 0.3082 | 2.0000 | 0.2998 | 2.0000 | 10  | 0.1862 | 2.5000 |
| 3   | 0.2991 | 1.5000 | 0.2296 | 2.5000 | 11  | 0.1944 | 2.5000 |
| 4   | 0.1650 | 2.5000 | 0.1642 | 2.5000 | 12  | 0.2739 | 2.5000 |
| 5   | 0.1047 | 2.5000 | 0.1037 | 2.5000 | 13  | 0.1000 | 2.5000 |
| 6   | 0.1734 | 2.5000 | 0.2277 | 1.5000 | 14  | 0.3356 | 1.5000 |
| 7   | 0.3422 | 1.5000 | 0.3355 | 1.5000 | 5.9077 | 8.6563 |
| 8   | 0.1700 | 2.5000 | 0.1701 | 2.5000 | Time | 21.6406 | 37.1221 |

### Table 7: Comparison of different methods for case 3.

| Algorithm | OF_max | OF_ave | OF_min | Std | Feasible |
|-----------|--------|--------|--------|-----|----------|
| PSO [5]   | —      | —      | 20.7975 | —   | ○        |
| MPSO [5]  | —      | —      | 17.3300 | —   | ○        |
| Seeker [6] | —      | —      | 8.4270  | —   | ×        |
| DJaya [34] | —      | —      | 9.9661  | 3.1145 | ○        |
| OJaya [34] | —      | —      | 9.8520  | 1.7749 | ○        |
| GA [2]    | —      | —      | 11.0010 | —   | ×        |
| GA-LP [2] | —      | —      | 10.9499 | —   | ×        |
| BBO [8]   | —      | —      | 10.5495 | —   | ○        |
| BBO-LP [8]| —      | —      | 8.7556  | —   | ○        |
| Jaya [36] | —      | —      | 10.2325 | 3.4972 | ○        |
| HHO [36]  | —      | —      | 7.28    | 2.9254 | ×        |
| CSO       | 10.6252 | 9.4496 | 8.9177  | 0.6042 | ○        |
| HCSO      | 9.5841  | 8.9777 | 8.6563  | 0.2519 | ○        |

Bold denotes the best result in that column. — means that there is no data provided in the cited article. × denotes that the solution of "OF_min" in that row is un-feasible. ○ denotes that the solution of "OF_min" in that row is feasible.

### Table 8: Operating times and CTI for case 3.

| PR | BR | T_pr | T_bc | CTI | T_pr | T_bc | CTI |
|----|----|------|------|-----|------|------|-----|
| 1  | 6  | 0.4087 | 0.7087 | 0.3000 | 0.4088 | 0.7104 | 0.3016 |
| 2  | 1  | 0.8371 | 1.3742 | 0.3000 | 0.8144 | 1.3744 | 0.3002 |
| 3  | 7  | 0.8371 | 1.1371 | 0.3000 | 0.8144 | 1.1145 | 0.3002 |
| 4  | 2  | 0.7558 | 1.0558 | 0.3000 | 0.7195 | 1.0272 | 0.3002 |
| 5  | 3  | 0.6157 | 0.9157 | 0.3000 | 0.6127 | 0.9158 | 0.3002 |
| 6  | 4  | 0.5213 | 0.8213 | 0.3000 | 0.5163 | 0.8173 | 0.3002 |
| 7  | 5  | 0.5110 | 1.0540 | 0.3000 | 0.5430 | 1.0439 | 0.3002 |
| 8  | 6  | 0.5110 | 1.1198 | 0.3000 | 0.5471 | 1.1013 | 0.5541 |
| 9  | 14 | 0.7540 | 1.0540 | 0.3000 | 0.7391 | 1.0439 | 0.3047 |
| 10 | 13 | 0.7540 | 1.3994 | 0.3000 | 0.7391 | 1.3995 | 0.6604 |
| 11 | 7  | 0.5015 | 1.1371 | 0.3000 | 0.5018 | 1.1145 | 0.6128 |
| 12 | 9  | 0.5015 | 1.0406 | 0.3000 | 0.5391 | 1.0566 | 0.4548 |
| 13 | 10 | 0.6045 | 0.9045 | 0.3000 | 0.5557 | 0.8558 | 0.3001 |
| 14 | 11 | 0.6851 | 0.9851 | 0.3000 | 0.6481 | 0.9486 | 0.3005 |
| 15 | 12 | 0.7337 | 1.0337 | 0.3000 | 0.7066 | 1.0101 | 0.3036 |
| 16 | 13 | 0.8198 | 1.3994 | 0.3000 | 0.8011 | 1.3995 | 0.5984 |
| 17 | 14 | 0.8198 | 1.1198 | 0.3000 | 0.8011 | 1.1013 | 0.3002 |
| 18 | 8  | 0.4288 | 0.7288 | 0.3000 | 0.4288 | 0.7292 | 0.3004 |
| 19 | 1  | 0.7406 | 1.3742 | 0.3000 | 0.6563 | 1.3744 | 0.7181 |
| 20 | 9  | 0.7406 | 1.0406 | 0.3000 | 0.6563 | 0.9566 | 0.3003 |
Figure 10: Comparison of convergence curves for case 4.

Figure 11: Comparison of independent runs for case 4.

Table 9: TDS and PS values for case 4.

| No. | CSO TDS | PS | HCSo TDS | PS | No. | CSO TDS | PS | HCSo TDS | PS |
|-----|---------|----|----------|----|-----|---------|----|----------|----|
| 1   | 0.0666  | 0.9968 | 0.0568   | 1.3849 | 14  | 0.0597  | 1.0880 | 0.0514   | 1.4265 |
| 2   | 0.0410  | 0.7940 | 0.0479   | 0.6292 | 15  | 0.0542  | 1.3006 | 0.0535   | 1.3258 |
| 3   | 0.0416  | 1.3382 | 0.0397   | 1.4283 | 16  | 0.0643  | 0.8174 | 0.0524   | 1.2179 |
| 4   | 0.0452  | 1.1022 | 0.0493   | 0.9583 | 17  | 0.0773  | 1.1033 | 0.0770   | 1.1040 |
| 5   | 0.0482  | 0.6774 | 0.0500   | 0.6369 | 18  | 0.0373  | 1.2591 | 0.0251   | 1.9011 |
| 6   | 0.0539  | 1.3766 | 0.0753   | 0.6708 | 19  | 0.0769  | 1.1032 | 0.0762   | 1.1057 |
| 7   | 0.0541  | 1.3636 | 0.0736   | 0.7111 | 20  | 0.0368  | 1.4731 | 0.0473   | 1.0281 |
| 8   | 0.0391  | 0.9217 | 0.0480   | 0.6822 | 21  | 0.0785  | 1.1078 | 0.0764   | 1.1256 |
| 9   | 0.0414  | 1.2545 | 0.0412   | 1.2579 | 22  | 0.0371  | 1.2830 | 0.0346   | 1.3742 |
| 10  | 0.0517  | 0.9511 | 0.0518   | 0.9488 | 23  | 0.0689  | 1.5001 | 0.0710   | 1.3983 |
| 11  | 0.0427  | 0.7512 | 0.0530   | 0.5288 | 24  | 0.0279  | 1.2875 | 0.0268   | 1.3146 |
| 12  | 0.0665  | 0.5809 | 0.0531   | 0.9065 | 25  | 4.8224  | 4.8000 | 0.0268   | 1.3146 |
| 13  | 0.0735  | 0.6396 | 0.0516   | 1.2520 | 26  | 15.7702 | 19.6230 |
### Table 10: Comparison of different methods for case 4.

| Algorithm     | OF_max | OF_ave | OF_min | Std  | Feasible |
|---------------|--------|--------|--------|------|----------|
| GA [3]        | —      | —      | 32.6058| —    | ○        |
| NLP [3]       | —      | —      | 19.4041| —    | ×        |
| GA-NLP [3]    | —      | —      | 6.1786 | —    | ○        |
| GA [39]       | 16.8083| 15.6592| 14.5426| 0.4110| ○        |
| PSO [39]      | 31.0790| 20.6491| 13.7150| 3.6823| ○        |
| DE [39]       | 9.4859 | 8.7162 | 8.6822 | 0.1233| ○        |
| HS [39]       | 10.1411| 9.6180 | 9.2339 | 0.2152| ○        |
| SOA [39]      | 19.9820| 16.5710| 14.2338| 1.2133| ○        |
| Jaya [34]     | —      | —      | 7.1378 | 2.83  | ○        |
| DJaya [34]    | —      | —      | 6.8319 | 2.28  | ○        |
| OJaya [34]    | —      | —      | 6.3713 | 1.45  | ○        |
| BBO [8]       | 56.9419| 43.6807| 28.8348| 5.25  | ○        |
| BBO-LP [8]    | 4.9126 | **4.8188**| **4.8000**| 0.02 | ○        |
| EFO [9]       | 7.5250 | 6.5810 | 6.0500 | 0.39  | ○        |
| MEFO [9]      | 12.1010| 6.0880 | 5.2250 | 1.22  | ○        |
| CSO           | 5.0774 | 4.8992 | 4.8224 | 0.0618| ○        |
| HCSO          | **4.8998**| 4.8384| **4.8000**| 0.0180| ○        |

Bold means the best result in that column. — denotes that there are no data provided in the cited article. × denotes that the solution of “OF_min” in that row is un-feasible. ˆ denotes that the solution of “OF_min” in that row is feasible.

### Table 11: Operating times and CTI for case 4.

| P/R | B/R | T_pr | T_bc | CTI | T_pr | T_bc | CTI |
|-----|-----|------|------|-----|------|------|-----|
| 1   | 15  | 0.2002| 0.6434| 0.4433 | 0.2000| 0.6577| 0.4577|
| 1   | 17  | 0.2002| 0.6296| 0.4295 | 0.2000| 0.6271| 0.4271|
| 2   | 4   | 0.2001| 0.4915| 0.2914 | 0.2000| 0.4393| 0.2393|
| 3   | 1   | 0.2002| 0.4595| 0.2594 | 0.2000| 0.5840| 0.3840|
| 4   | 6   | 0.2001| 0.6494| 0.4493 | 0.2000| 0.4015| 0.2015|
| 5   | 3   | 0.2001| 0.5587| 0.3587 | 0.2000| 0.6094| 0.4094|
| 6   | 8   | 0.2000| 0.6286| 0.4285 | 0.2000| 0.4536| 0.2536|
| 6   | 23  | 0.2000| 0.8211| 0.6211 | 0.2000| 0.7547| 0.5547|
| 7   | 5   | 0.2000| 0.4513| 0.2513 | 0.2000| 0.4321| 0.2321|
| 7   | 23  | 0.2000| 0.8211| 0.6211 | 0.2000| 0.7547| 0.5547|
| 8   | 10  | 0.2001| 0.4170| 0.2169 | 0.2000| 0.4162| 0.2162|
| 9   | 7   | 0.2001| 0.6421| 0.4419 | 0.2000| 0.4109| 0.2109|
| 10  | 12  | 0.2001| 0.4620| 0.2619 | 0.2000| 0.6697| 0.4697|
| 11  | 9   | 0.2002| 0.5651| 0.3650 | 0.2000| 0.5667| 0.3667|
| 12  | 14  | 0.2003| 0.5425| 0.3422 | 0.2000| 0.7252| 0.5252|
| 12  | 21  | 0.2003| 0.6425| 0.4422 | 0.2000| 0.6369| 0.4369|
| 13  | 11  | 0.2055| 0.5363| 0.3308 | 0.2000| 0.4063| 0.2063|
| 13  | 21  | 0.2055| 0.6425| 0.4370 | 0.2000| 0.6369| 0.4369|
| 14  | 16  | 0.2009| 0.4817| 0.2808 | 0.2000| 0.6917| 0.4917|
| 14  | 19  | 0.2009| 0.6433| 0.4424 | 0.2000| 0.6399| 0.4399|
| 15  | 13  | 0.2002| 0.4343| 0.2341 | 0.2000| 0.7187| 0.5187|
| 15  | 19  | 0.2002| 0.6433| 0.4431 | 0.2000| 0.6399| 0.4399|
| 16  | 2   | 0.2003| 0.5732| 0.3729 | 0.2000| 0.4549| 0.2549|
| 16  | 17  | 0.2003| 0.6296| 0.4293 | 0.2000| 0.6271| 0.4271|
| 17  | —   | —     | —     | —     | —     | —     | —   |
| 18  | 2   | 0.2006| 0.5732| 0.3726 | 0.2000| 0.4549| 0.2549|
| 18  | 15  | 0.2006| 0.6434| 0.4428 | 0.2000| 0.6577| 0.4577|
| 19  | —   | —     | —     | —     | —     | —     | —   |
| 20  | 13  | 0.2003| 0.4343| 0.2340 | 0.2000| 0.7187| 0.5187|
| 20  | 16  | 0.2003| 0.4817| 0.2814 | 0.2000| 0.6917| 0.4917|
| 21  | —   | —     | —     | —     | —     | —     | —   |
| 22  | 11  | 0.2027| 0.5363| 0.3335 | 0.2000| 0.4063| 0.2063|
| 22  | 14  | 0.2027| 0.5425| 0.3398 | 0.2000| 0.7252| 0.5252|
| 23  | —   | —     | —     | —     | —     | —     | —   |
| 24  | 5   | 0.2032| 0.4513| 0.2481 | 0.2000| 0.4321| 0.2320|
| 24  | 8   | 0.2032| 0.6286| 0.4254 | 0.2000| 0.4536| 0.2536|

— denotes no relay considered in that row.
Figure 12: The effects of changing $p_a$ for case 4.

Figure 13: The effects of changing hierarchy number $|O|$ for case 4.

Figure 14: Two random simulation results for comparison of SL, CL, and AL on case 4.
6.2. Effects of Hierarchical Cluster Number |Ω|. Let us still take case 4 as an example, where |Ω| is set within [3, 30] in step 3, while $\alpha_{sol} = 30$, $\alpha_{iter} = 300$, and $p_e = 0.1$ are kept the same by different |Ω|. For each |Ω|, we record its average OF value (OF_ave), as well as the average CPU time, to observe the effects of different |Ω|

In Figure 13, let us observe the situation of two extreme values as 3 and 30 (in red color). When |Ω| = 3, y-axis shows the worst value because it is as large as 7.24 s; when |Ω| = 30, y-axis shows the second to last. This performance proves that |Ω| value cannot be too small or too large, but the problem is on how to decide the most suitable value. The answer is, there is no “perfect” value for |Ω|, and it depends on the characteristics of the problem to be solved. According to Figure 13, we choose |Ω| = 15 in case 4, which is an integer within [1, $\alpha_{sol}$].

However, for CPU time (in blue color), the situation changes since the y-axis keeps almost unchanged within [0.935, 0.94] regardless of |Ω| value. It is because the time complexity of the hierarchical clustering mechanism is $O(\alpha_{sol}^3)$, and it has nothing to do with |Ω|.

6.3. Different Hierarchy Distance Measurement. As mentioned in Section 3.2, there are three ways for measuring the distance between any two hierarchies $O_1$ and $O_2$ as single linkage (SL), complete linkage (CL), and average linkage (AL). In SL, we merge in each step $O_1$ and $O_2$, whose two closest solutions have the smallest distance, as shown in Ref. (9). In CL, we merge in each step $O_1$ and $O_2$ with the smallest maximum pairwise distance, as shown in Ref. (17). In AL, the distance between $O_1$ and $O_2$ is defined as the average distance between all pairs of solutions, as shown in Ref. (18).

$$\text{dis}(O_1, O_2) = \max_{h \in O_1, k \in O_2} \text{dis}(h, k). \quad (17)$$

$$\text{dis}(O_1, O_2) = \frac{1}{|O_1||O_2|} \sum_{h \in O_1, k \in O_2} \text{dis}(h, k), \quad (18)$$



where $\text{dis}(h, k)$ is calculated by equation (8).

Here, we still take case 4 as an example, where $\alpha_{sol} = 30$, $\alpha_{iter} = 300$, $p_e = 0.1$, and |Ω| = 15 are kept the same by SL, CL, and AL. For each measurement, the OF_ave is recorded by 30 independent runs. We can observe from the two randomly generated simulation results in Figures 14(a) and 14(b) that from a statistical point of view, there is no significant difference between (a) and (b). For each figure, no matter we choose SL, CL, or AL, it has few impacts on the statistical results.

7. Conclusion

In this article, a hierarchical clustering mechanism is cooperated with CSO to solve the optimal setting problem of DOCRs. To observe the sensitivity of HCSO, the parameters as hierarchical clustering stepsize $\theta$, eggs abandon fraction $p_e$, and hierarchy number |Ω| are analyzed comprehensively on benchmark functions. The conclusion is that in the DOCRs optimal setting cases, HCSO is more successful than CSO in reducing the objective function value and in maintaining its robustness. Taking case 2 as an example, compared with CSO, the minimum objective function value reduced by 10.93% and the Std value reduced by 54.93% in HCSO. In future works, more improved CSO will be studied and compared with the proposed HCSO in aspects of adaptive change of parameters, convergence rate, and solution stability. Furthermore, larger test systems of DOCRs’ optimal setting problems as 15-bus, 30-bus, and 42-bus systems are supposed to be experimented in the following research to deeply expand HCSO’s applications in this field.

Data Availability

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

Conflicts of Interest

The authors declare that they have no competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

References

[1] F. Razavi, H. A. Abyaneh, M. Al-Dabbagh, R. Mohammadi, and H. Torkaman, “A new comprehensive genetic algorithm method for optimal overcurrent relays coordination,” Electric Power Systems Research, vol. 78, no. 4, pp. 713–720, Apr. 2008.

[2] A. S. Noghabi, J. Sadeh, and H. R. Mashhadi, “Considering different network topologies in optimal overcurrent relay coordination using a hybrid GA,” IEEE Transactions on Power Delivery, vol. 24, no. 4, pp. 1857–1863, 2009.

[3] P. P. Bedekar and S. R. Bhide, “Optimum coordination of directional overcurrent relays using the hybrid GA-NLP approach,” IEEE Transactions on Power Delivery, vol. 26, no. 1, pp. 109–119, Jan. 2011.

[4] M. M. Mansour, S. F. Mekhamer, and N. E. S. El-Kharbawe, “A modified particle swarm optimizer for the coordination of directional overcurrent relays,” IEEE Transactions on Power Delivery, vol. 22, no. 3, pp. 1400–1410, 2007.

[5] H. H. Zeineldin, E. F. El-Saady, and M. M. A. Salama, “Optimal coordination of overcurrent relays using modified particle swarm optimization,” Electric Power Systems Research, vol. 76, no. 11, pp. 988–995, Jul. 2006.

[6] T. Amraee, “Coordination of directional overcurrent relays using seeker algorithm,” IEEE Transactions on Power Delivery, vol. 27, no. 3, pp. 1415–1422, 2012.

[7] M. Alipour, S. Teimournazadeh, and H. Seyedi, “Improved group search optimization algorithm for coordination of directional overcurrent relays,” Swarm and Evolutionary Computation, vol. 23, pp. 40–49, 2015.

[8] F. A. Albasri, A. R. Alroomi, and J. H. Talaq, “Optimal coordination of directional overcurrent relays using
biogeography-based optimization algorithms,” IEEE Trans-
actions on Power Delivery, vol. 30, no. 4, pp. 1810–1820, 2015.
[9] H. R. E. H. Bouchekara, M. Zellagui, and M. A. Abido,
“Optimal coordination of directional overcurrent relays using a
modified electromagnetic field optimization algorithm,”
Applied Soft Computing, vol. 54, pp. 267–283, May. 2017.
[10] M. N. Alam, “Adaptive protection coordination scheme using
numerical directional overcurrent relays,” IEEE Transactions on
Industrial Informatics, vol. 15, no. 1, pp. 64–73, 2019.
[11] M. N. Alam, “Overcurrent protection of AC microgrids using
mixed characteristic curves of relays,” Computers & Electrical
Engineering, vol. 74, pp. 74–88, Mar. 2019.
[12] A. Korashy, S. Kamel, L. Nasrat, and F. Jurado, “Developed
multi-objective grey wolf optimizer with fuzzy logic decision-
making tool for direction overcurrent relays coordination,”
Soft Computing, vol. 24, no. 17, pp. 13305–13317, 2020.
[13] S. Kamel, A. Korashy, A. R. Yousef, and F. Jurado, “Develop-
ment and application of an efficient optimizer for optimal
coordination of directional overcurrent relays,” Neural
Computing & Applications, vol. 32, no. 12, pp. 8561–8583,
2020.
[14] A. Korashy, S. Kamel, E. H. Houssein, F. Jurado, and
F. A. Hashim, “Development and application of evaporation
rate water cycle algorithm for optimal coordination of di-
rectional overcurrent relays,” Expert Systems with Applica-
tions, vol. 185, Article ID 115538, 2021.
[15] A. A. El-Fergany and H. M. Hasien, “Water cycle algorithm
for optimal overcurrent relays coordination in electric power
systems,” Soft Computing, vol. 23, no. 23, pp. 12761–12778,
2019.
[16] M. Abdelhamid, E. H. Houssein, M. A. Mahdy, A. Selim, and
S. Kamel, “An improved seagull optimization algorithm for
optimal coordination of distance and directional overcurrent
relays,” Expert Systems with Applications, vol. 200, Article ID
116931, 2022.
[17] A. Assoua, R. Benabid, and A. A. Ladjici, “Optimal coor-
dination of directional overcurrent relays with non-standard
multi-characteristics for power systems transient instability
protection,” Electr. Eng., 2022.
[18] X. S. Yang and S. Deb, “Cuckoo search via Levy flights,” in
Proceedings of World congress on Nature and Biologically
Inspiried Computing, pp. 210–214, NaBIC, India, 2009.
[19] X. Ou, M. Wu, Y. Pu, B. Tu, G. Zhang, and Z. Xu, “Cuckoo
search algorithm with fuzzy logic and Gauss-Cauchy for
minimizing localization error of WSN,” Applied Soft Com-
puling, vol. 125, 2022.
[20] M. Reda, A. Onsy, M. A. Elhosseini, A. Y. Haikal, and
M. Badawy, “A discrete variant of cuckoo search algorithm to
solve the Travelling Salesman Problem and path planning
for autonomous trolley inside warehouse,” Knowledge-Based
Systems, vol. 252, 2022.
[21] Q. Yang, P. Liu, J. Zhang, and N. Dong, “Combined heat and
power economic dispatch using an adaptive cuckoo search
with differential evolution mutation,” Applied Energy,
vol. 307, 2022.
[22] S. Makhlofi, S. Khennas, S. Bouchaib, and A. H. Arab,
“Multi-objective cuckoo search algorithm for optimized
pathways for 75% renewable electricity mix by 2050 in
Algeria,” Renewable Energy, vol. 185, pp. 1410–1424, 2022.
[23] P. Sekhar and S. Mohanty, “An enhanced cuckoo search al-
gorithm based contingency constrained economic load dis-
patch for security enhancement,” International Journal of
Electrical Power & Energy Systems, vol. 75, pp. 303–310, 2016.
[24] T. T. Nguyen and D. N. Vo, “The application of one rank
cuckoo search algorithm for solving economic load dispatch
problems,” Applied Soft Computing, vol. 37, pp. 763–773,
2015.
[25] T. T. Nguyen, T. T. Nguyen, and D. N. Vo, “An effective
cuckoo search algorithm for large-scale combined heat and
power economic dispatch problem,” Neural Computing &
Applications, vol. 30, no. 11, pp. 3545–3564, 2018.
[26] S. Das, A. Abraham, and A. Konar, “Automatic clustering
using an improved differential evolution algorithm,” IEEE
Transactions on Systems, Man, and Cybernetics - Part A:
Systems and Humans, vol. 38, no. 1, pp. 218–237, 2008.
[27] Q. Liu, S. Du, B. J. Van Wyk, and Y. Sun, “Niching particle
swarm optimization based on Euclidean distance and hier-
archical clustering for multimodal optimization,” Nonlinear
Dynamics, vol. 99, no. 3, pp. 2459–2477, 2020.
[28] C. Li and S. Yang, “A general framework of multipopulation
methods with clustering in undetectable dynamic environ-
ments,” IEEE Transactions on Evolutionary Computation,
vol. 16, no. 4, pp. 556–577, 2012.
[29] J. Yu, C. H. Kim, and S. B. Rhee, “Clustering cuckoo search
optimization for economic load dispatch problem,” Neural
Computing & Applications, vol. 32, no. 22, pp. 16951–16969,
2020.
[30] J. Moirangthem, K. Kr, S. S. Dashi, and R. Ramaswami,
“Adaptive differential evolution algorithm for solving non-
linear coordination problem of directional overcurrent re-
lays,” IET Generation, Transmission & Distribution, vol. 7,
no. 4, pp. 329–336, Apr. 2013.
[31] S. Mirjalili, “Moth-flame optimization algorithm: a novel
nature-inspired heuristic paradigm,” Knowledge-Based Sys-
tems, vol. 89, pp. 228–249, 2015.
[32] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar,
“Comprehensive learning particle swarm optimizer for global
optimization of multimodal functions,” IEEE Transactions on
Evolutionary Computation, vol. 10, no. 3, pp. 281–295, 2006.
[33] A. J. Urdaneta, R. Nadira, and L. Perez Jimenez, “Optimal
coordination of directional overcurrent relays in inter-
connected power systems,” IEEE Transactions on Power De-
livery, vol. 3, no. 3, pp. 903–911, Jul. 1988.
[34] J. Yu, C. H. Kim, and S. B. Rhee, “Oppositional Jaya algorithm
with distance-adaptive coefficient in solving directional over
current relays coordination problem,” IEEE Access, vol. 7,
p. 150729–150742, 2019.
[35] M. Ezzeddine, R. Kaczmarek, and M. U. Iftikhar, “Coordi-
nation of directional overcurrent relays using a novel method
to select their settings,” IET Generation, Transmission &
Distribution, vol. 5, no. 7, pp. 743–750, Jul. 2011.
[36] J. Yu, C. Kim, and S. Rhee, “The Comparison of Lately
Proposed Harris Hawks Optimization and Jaya Optimization
in Solving Directional Overcurrent Relays Coordination
Problem,” Complexity, p. 2020, 2020.
[37] R. Correa, G. Cardoso Jr, O. C. d. Araújo, and L. Mariotto,
“Online coordination of directional overcurrent relays using
binary integer programming,” Electric Power Systems Re-
search, vol. 127, pp. 118–125, 2015.