Applied Intelligence (2022) 52:15325–15344
https://doi.org/10.1007/s10489-022-03179-y

Chaotic slime mould algorithm for economic load dispatch problems

Tribhuvan Singh[1]

Accepted: 24 December 2021 / Published online: 14 March 2022

Abstract

The economic load dispatch (ELD) problem strives to optimize the division of total power demand among the power generators under specified constraints. It is solved by scheduling the generating units of a power plant that meet the load demand with minimum generation cost while satisfying various equality and inequality constraints. Achieving global optimal points is considered difficult due to the involvement of a non-linear objective function and large search domain. The slime mould algorithm (SMA) was recently proposed to solve complex problems. Its convergence rate and capability of capturing optimal global solutions are pretty satisfactory. In this paper, a chaotic number-based slime mould algorithm (CSMA) is suggested for ELD problems the first time. Five test cases with different power demands have been considered to compare the performance of the proposed approach against SMA, salp swarm algorithm (SSA), moth flame optimizer (MFO), grey wolf optimizer (GWO), biogeography based optimizer (BBO), grasshopper optimization algorithm (GOA), multi-verse optimizer (MVO) on 6, 13, 15, 40, and 140 generators ELD problems. The experimental results show that the proposed algorithm reduces the total generation cost significantly. CSMA outperformed SMA in all test cases that justify the effectiveness of chaotic sequences used in the proposed work. Further, three statistical tests have been conducted to justify the competitiveness of the suggested approach.

Keywords Economic load dispatch · Slime mould algorithm · Chaotic maps · Optimization problems

1 Introduction

Economic load dispatch (ELD) is one of the prominent problems of the power systems domain [1]. It has attracted the attention of many researchers in the past few decades, and it is still a hot topic of research in the field of power systems. This problem aims to minimize the total generation cost in producing specific power demands while satisfying a set of constraints. A power plant can have multiple generating units (generators) with their respective cost coefficients and limits constraints. These generating units are treated as energy-producing resources in a power plant. The proper utilization of generating units is required to optimize fuel consumption and energy as fossil fuels used to generate electricity are limited in nature. The generating units can be appropriately utilized through an efficient optimization algorithm with the following characteristics.

- The algorithm must produce the best scheduling strategy that meets a certain load demand.
- The algorithm must satisfy all input constraints and work effectively on non-linear functions.
- The algorithm must be capable of handling problems of high dimensions.

Various optimization algorithms have already been suggested to deal with ELD problems while achieving the aforementioned conditions. Still, none of them can defeat all other algorithms in all benchmark datasets. Optimization algorithms for handling ELD problems are being developed in the hope of getting a globally optimal solution while avoiding premature convergence. These algorithms use certain mathematical equations to perform exploitation and exploration and follow the same steps in uni-modal and multi-modal problems, even though they may have different requirements. An algorithm that effectively deals with uni-modal problems may not be effective in multi-modal

✉ Tribhuvan Singh
tribhuvansingh@soa.ac.in

1 Department of Computer Science and Engineering, Siksha O Anusandhan (Deemed to be University), Bhubaneswar, Odisha, India

problems; however, the converse is not valid. An algorithm designed to solve multi-modal problems will work efficiently in uni-modal problems. As multi-modal problems may have more than one global or local optimal solution, a thorough investigation of search space is required to target all optimal solutions. In general, algorithms encounter difficulty in achieving the desired objective if the dimension of the problem increases. An algorithm that performs well in lower-dimensional optimization problems may not work efficiently in higher-dimensional optimization problems. Therefore, combining a diversity-producing mechanism with a fast converging algorithm works more appropriately in high-dimensional optimization problems.

ELD is a multi-modal high-dimensional constrained optimization problem with a large search domain and non-linear objective function. ELD aims to determine the best schedule of generating units to minimize the total generation cost while meeting certain load demand under specified constraints [2, 3] with each generating unit producing electricity (power) in its generation range. In the conventional formulation of ELD, some practical features such as ramp rate limits, prohibited operating zones, and valve point effect were not considered. However, in real power systems, these features are usually encountered. Therefore, neglecting these features may lead to inaccurate solutions of the ELD problems [4, 5].

The large search domain and high dimensionalities of ELD need an optimization algorithm with a powerful exploration method that helps in avoiding the problem of local entrapment during optimization. This is one of the NP-hard problems that require more computational time during optimization. These problems may be solved by giving more attention to implementing an efficient diversity-preserving approach. In solving such problems, nature-inspired algorithms have gained a wide range of acceptability due to their population-based search techniques [6–9]. However, some nature-inspired algorithms do not contain much novelty. Authors in [10] have argued that the concepts used in the cuckoo search are similar to that of $(\mu + \lambda)$ evolutionary strategies. Authors in [11] have presented sufficient evidence that grey wolf, bat, and firefly algorithms are not novel. These algorithms have reiterated the ideas introduced first for particle swarm optimization. Some other research articles have highlighted valuable insight into optimization algorithms in [12–14]. Authors in [15] proposed a new dendritic neuron model for solving complex problems.

The presence of constraints makes it difficult to solve ELD. These constraints are handled by using the respective penalty functions. Hence, an optimization algorithm should have the ability to handle these constraints by implementing a penalty function while solving ELD. In such a case, optimization algorithms need more intelligent exploitation and exploration methods. An exploration method targets the promising solutions throughout the search space, whereas an exploitation method tries to search around the neighborhood of the good solutions found so far. A powerful exploration method improves the global search ability, while an intelligent exploitation method improves the convergence rate of the optimization algorithm. These two methods need to be balanced during the program execution to avoid the problems of random search and local entrapment.

In ELD problems, the search domain for ith generator is given by the range $[P_i^{\min}, P_i^{\max}]$. Here, $P_i^{\min}$ and $P_i^{\max}$ represent the lower and upper limits, respectively for *ith* generator. The number of generators in a power plant is treated as the dimension of the problem. In most of the datasets available in the literature, the number of generators is 6, 13, 15, 20, 40, 80, 140, and 160. In general, optimization algorithms perform well in solving ELD problems with up to 20 generators. As the number of generators in a power plant increases, the optimization algorithms find it difficult to solve ELD problems efficiently. To handle hard problems of different domains, recently, some meta-heuristics have been suggested. These algorithms use some natural phenomena to achieve the desired goal. Faris et al. [16] proposed a monarch butterfly optimization (MBO) algorithm. Moth search algorithm (MSA) is proposed for global optimization in [17]. Yang et al. [18] presented a hunger games search (HGS) algorithm for solving complex problems. An efficient approach based on the Runge Kutta method (RUN) is proposed in [19] for optimization problems. Tu et al. [20] suggested colony predation algorithm (CPA) for complex problems. Heidari et al. [21] proposed Harris hawks optimization (HHO) algorithm for hard problems. Modified versions of the HHO algorithm are used to solve data clustering problems in [22, 23].

Recently, the slime mould algorithm (SMA) [24] is proposed for solving complex problems. This algorithm implements the natural phenomenon of slime behavior during the foraging to the food source. SMA, being a new algorithm, there is a lot of scope of its performance improvement and its applicability in various real-world problems of science and engineering. Authors of the base paper of SMA and other researchers have argued the effectiveness of this algorithm in solving complex problems. These points could be the motivation for using SMA to tackle the *ELD* Problems. The replacement of random numbers by chaotic sequences improves the performance of an optimization algorithm [25]. In this paper, a chaotic sequence guided slime mould algorithm (CSMA) is presented for solving ELD problems with prohibitive operating zones and ramp rate limits. In short, the originality and major contributions of this paper are summarized as follows.

- Integration of merits of chaotic sequences generated by the logistic chaotic map into a fast converging algorithm.
- Validation of efficacy of CSMA by five test cases with 6, 13, 15, 40, and 140 generators *ELD* problems.
- Comparison of the performance of CSMA with seven recent state-of-the-art algorithms SMA, SSA, MFO, GWO, BBO, GOA, and MVO based on the experimental values.
- Total generation cost for specified load demand is used as the main performance metric for comparing the performances of the algorithms.
- Validation for the effectiveness of CSMA using three statistical tests: Friedman test, Iman-Davenport test, and Holm test.

The rest of the paper is organized as follows. We present the literature review in Section 2. Section 3 describes the mathematical formulation of ELD and various constraints associated with it. Section 4 describes CSMA in detail. The description of the datasets and parameter setting is given in Section 5. The analysis of experimental results is given in Section 6. Section 7 highlights the conclusions and future research directions.

2 Literature review

Pothiya et al. [26] proposed an approach based on multiple tabu search (MTS) algorithms to solve the dynamic economic dispatch problem with generator constraints. They used experimental data to show the efficacy of MTS against genetic algorithms, simulated annealing, and particle swarm optimization (PSO). Lin et al. [27] suggested an improved tabu search (ITS) algorithm for ELD problems. They suggested the idea of a flexible memory system to avoid the problem of local entrapment. After the first proposal of tabu search for the ELD problems, other variants have been suggested to speed up the performance during optimization. PSO, on the other hand, has been applied with different variations to solve ELD [28]. Another well-known and powerful algorithm to solve ELD is based on differential evolution (DE) [29].

Jayabarathi et al. [30] used the crossover and mutation operators combined with a grey wolf optimizer to solve ELD. They claimed the efficacy of their proposal through experimental results. Pradhan et al. [31] integrated opposition-based learning into the basic GWO algorithm to improve the convergence rate and solution quality while solving ELD. Elsakaan et al. [32] suggested an enhanced moth-flame optimizer (MFO) to solve non-smooth economic dispatch problems. They combined the merits of levy flight with MFO to achieve the desired goal. Mandal et al. [33] incorporated mutation and crossover operators of differential evolution in krill herd algorithm (KHA) to solve ELD. They used experimental data to argue for the importance of integrated operators in KHA. Bulbul et al. [34] proposed an opposition-based krill herd algorithm for ELD. Coelho et al. [35] proposed an improved harmony search (IHS) algorithm based on exponential distribution for ELD. The involvement of exponential distribution with the harmony search algorithm yielded better performance for IHS. ELD has also been solved by a tournament-based harmony search (THS) algorithm [36]. THS replaced the random selection process in the memory consideration operator with the tournament selection process to achieve the desired objective while improving the convergence rate. Pothiya et al. [37] suggested an ant colony-based optimizer for ELD. They introduced the concept of the priority list, variable reduction, and zoom feature to improve the overall performance of the suggested approach. Elsayed et al. [38] presented an approach based on the social spider algorithm for solving the economic dispatch problem.

Recently, hybrid approaches have been widely used to solve various science and engineering optimization problems. These approaches leverage the merits of existing techniques to perform specific tasks efficiently. Various hybrid algorithms have been developed in the field of power systems. Bhattacharya et al. [39] proposed a hybrid algorithm by combining differential evolution with biogeography-based optimization (DE/BBO) algorithm to solve convex and nonconvex ELD problems considering transmission losses and constraints such as ramp rate limits, valve-point loading, and prohibited operating zones. In addition, various other hybrid algorithms have been proposed to solve a variety of ELD problems, and the efficacy of the approaches has been justified based on experimental values and comparative performance analysis [40–43].

The use of chaotic sequences in place of random numbers improves the convergence rate and quality of solutions during optimization [44]. A chaotic sequence-based differential evolution algorithm for solving complex problems is proposed in [45]. Yang et al. [46] proposed an adaptive chaotic spherical evolution algorithm for optimization. Xu et al. [47] implemented chaotic local search into grey wolf optimizer to avoid the problem of local entrapment during the search process. By utilizing the evidence argued by many researchers, chaotic sequences are used in various optimization algorithms to solve real-world global optimization problems in science and engineering. Adarsh et al. [48] introduced a chaotic map-based bat algorithm for ELD problems. They used chaotic sequences to enhance the performance of their suggested approach.

They used experimental data to claim the effectiveness of their proposal. Arul et al. [49] proposed a chaotic self-adaptive differential harmony search (CSADHS) algorithm to solve the dynamic economic dispatch problem. They replaced the pitch adjustment operator in the harmony search algorithm with a chaotic self-adaptive differential mutation operator to improve the searching ability with less computational cost. Lu et al. [50] proposed a chaotic map-based differential evolution for dynamic economic dispatch problems. Coelho et al. [51] integrated chaotic sequences and implicit filtering local search methods in PSO to solve ELD problems. A multi-population-based chaotic JAYA algorithm is proposed to solve ELD problems in [52]. In [52], random numbers are replaced by chaotic numbers to improve the convergence rate of the JAYA algorithm. In addition, the population is divided into sub-populations to enhance diversity during optimization.

Zhao et al. [53] proposed a cuckoo search algorithm-guided approach by introducing a self-adaptive step size and neighbor-study strategies to improve the global search ability while solving the ELD problems. Moreover, they proposed an improved lambda iteration strategy to create offspring. Mohammadi and Abdi [54] suggested a modified crow search algorithm guided approach for ELD problems. They proposed an approach to capture optimal global solutions by introducing an adaptive adjustment of the flight length. A harmony search-based method is proposed in [55] to solve ELD problems. In [55], the update process of the harmony search algorithm based on a greedy approach is replaced by another efficient method to enhance the global search capability of the algorithm during the search process. The effectiveness of the simplex search method is integrated into artificial algae algorithm to solve ELD problems in [56].

Prakash et al. [57] proposed a quasi-oppositional self-learning teacher-learner-based-optimization algorithm to solve ELD problems. Kaboli et al. [58] proposed an artificial cooperative search algorithm to solve ELD problems. This algorithm tries to balance exploitation and exploration during the optimization to avoid the problem of stagnation and random search. Trivedi et al. [59] proposed an interior search algorithm to solve ELD and combined economic emission dispatch (CEED) problems in microgrids. An Ameliorated GWO algorithm is presented in [60] to solve ELD by synergizing the exploration and exploitation mechanism. In addition, an opposition-based learning approach is used to target the global optimal solution in [60].

Srivastava et al. [61] proposed an aggrandized class topper optimization algorithm for solving ELD. A crow search algorithm guided approach for ELD is presented in [62]. Some other approaches for ELD are given in [63, 64]. A clustering-based cuckoo search approach for ELD problems is presented in [65]. Authors have shown the effectiveness of clustering in cuckoo search for solving ELD problems. A moth flame optimizer-guided approach for ELD problems is presented in [66]. Kamboj et al. [67] presented an approach based on a grey wolf optimizer for ELD. A biogeography-based optimizer is proposed for ELD problems in [68]. Salp swarm algorithm (SSA) [69] is developed to solve complex problems. The main motivation of SSA is the swarming behavior of salps when navigating and foraging in oceans. After the first version of SSA, it is being applied to solve various real-world problems. A grasshopper optimization algorithm (GOA) [70] is presented for solving hard problems. The main inspiration of GOA is the behavior of grasshopper swarms. Mirjalili et al. [71] proposed a multi-verse optimizer (MVO) for solving challenging real-world problems. The authors have justified the competitiveness of MVO using experimental results.

3 Mathematical formulation and constraints

In this section, we present the basic concepts and mathematical formulation of ELD and various constraints associated with it.

3.1 Total generation cost

The total generation cost with D generators is given by

$$\begin{aligned} \min F_t &= \sum_{i=1}^{D} F_i(P_i) \\ &= \sum_{i=1}^{D} \left(a_i P_i^2 + b_i P_i + c_i \right) + e_i \\ &\quad \times |sin(f_i \times (P_i^{\min} - P_i))| \end{aligned} \tag{1}$$

where F_t and F_i are the total generation cost and cost function, respectively of ith generator. P_i is the power generated by ith generator, and a_i, b_i, c_i, e_i, f_i are its cost coefficients.

3.2 Constraints

3.2.1 Power equality constraint

This constraint states that the power generated by all generating units should be equal to the sum of load demand and power loss. Mathematically,

$$\sum_{i=1}^{D} P_i = P_D + P_L \tag{2}$$

where P_D is the load demand in a power plant. P_L is the power loss that is computed using B-coefficients as follows:

$$P_L = \sum_{i=1}^{D}\sum_{j=1}^{D} P_i B_{ij} P_j + \sum_{i=1}^{D} B_{0i} P_i + B_{00} \tag{3}$$

where B_{ij}, B_0, B_{00} are the transmission loss coefficients.

3.2.2 Generation limits constraint

This constraint states that the ith generating unit can generate power between the lower and upper limits as follows:

$$P_i^{\min} \le P_i \le P_i^{\max} \tag{4}$$

where $P_i^{\min}$ and $P_i^{\max}$ are the lower and upper limits of ith generator, respectively.

3.3 Ramp rate constraints

The ramp rate constraint restricts the operating range of the physical lower and upper limits to the effective lower limit P_i^{min} and upper limit P_i^{max}, respectively. According to [72], the inequality constraints due to ramp rate limits for unit generation changes are given

1) as generation increases

$$P_i - P_i^0 \le UR_i \tag{5}$$

2) as generation decreases

$$P_i - P_i^0 \le DR_i \tag{6}$$

where P_i and P_i^0 are the current and previous output, respectively. UR_i and DR_i are the up ramp limit and down ramp limit, respectively, of the ith generator in MW/time-period.

3.4 Prohibited operating zone constraints

A power generator may have prohibited operating zones (POZs) due to the physical limitation of power plant components [73]. A generator with POZs has discontinuous input-output characteristics. Each generator with $(Z-1)$ POZs is characterized by Z disjoint operating sub-regions. The POZ constraints are given by [74]:

$$P_{iz}^{L} \le P_i \le P_{iz}^{U}, \quad z = 1, 2, \cdots, Z \tag{7}$$

Note that $P_{i1}^{L} = P_i^{\min}$, $P_{iZ}^{U} = P_i^{\max}$. Z is the number of prohibited zones for each generator. The cost function of generator with prohibited zones is given in Fig. 1.

Respective penalty functions handle all the constraints mentioned above during the program execution. The description of these penalty functions is given below.

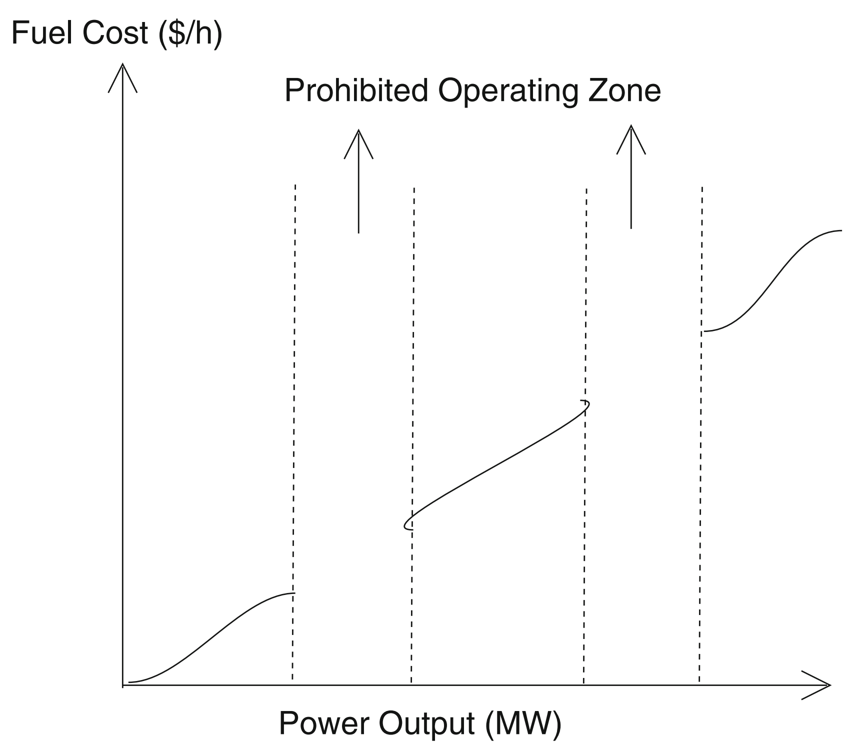

Fig. 1 Cost function with prohibited operating zones

3.4.1 Power balance penalty (PBP)

This penalty function is used to handle power equality constraint described in (2) and is defined as:

$$PBP = |P_D + P_L - \sum_{i=1}^{D} P_i| \tag{8}$$

In the ideal case, the value of PBP is zero.

3.4.2 Capacity limits penalty (CLP)

This penalty function is used to handle generation limits constraint described in (4). Its mathematical formulation is

$$\begin{aligned} CLP &= \sum_{i=1}^{D} |P_i - P_i^{\min}| - \left(P_i - P_i^{\min}\right) \\ &+ \sum_{i=1}^{D} |P_i^{\max} - P_i| - \left(P_i^{\max} - P_i\right) \end{aligned} \tag{9}$$

In the ideal case, the value of CLP is zero.

3.4.3 Ramp limits penalty (RLP)

This penalty function is used to handle ramp rate constraints given in (5) and (6). This penalty function is mathematically formulated as

$$\begin{aligned} RLP &= \sum_{i=1}^{D} |P_i - DR_i| - (P_i - DR_i) \\ &+ \sum_{i=1}^{D} |UR_i - P_i| - (UR_i - P_i) \end{aligned} \tag{10}$$

3.4.4 Prohibited operating zone penalty (POZP)

This penalty function is used to handle the POZ constraints given in (7). Its mathematical formulation is

$$POZP = \sum_{i=1}^{D} (P_{poz})^2 \tag{11}$$

where

$$P_{poz} = \left\{ \begin{matrix} \min(P_i - P_{iz}^L, P_{iz}^U - P_i), & P_{iz}^L \le P_i \le P_{iz}^U \\ 0, & \text{otherwise} \end{matrix} \right\} \tag{12}$$

where P_{iz}^L and P_{iz}^U are the lower bound and upper bound of the ith generator for the zth prohibited zone.

The penalty functions mentioned-above give either zero or non-zero values. The zero value of the penalty function indicates that the respective constraint is satisfied. In that case, the multiplication of penalty function value with any value of λ will be zero. Therefore, the total generation cost will remain the same. A non-zero value of the penalty function indicates that the respective constraint is not satisfied. The non-zero value can be treated as an error value. A solution that is unable to satisfy the constraint must be discarded. Since, in this work, we are dealing with a minimization problem. All optimization algorithms try to minimize the total generation cost of the generating units. Each penalty function value is multiplied by a constant value to magnify the error if it occurs. The resultant value is added to the total generation cost (fitness value). Hence, during the selection, the solutions that do not satisfy the constraints will not be selected and will not be able to create offspring.

The values presented in the Table 1 are chosen in such a way that a solution gets discarded if it does not satisfy the constraint. For example, if we consider test case 1, the values are $\lambda_1 = 1000$, $\lambda_2 = 1000$, $\lambda_3 = 100000$, $\lambda_4 = 10000$. These values are multiplied by the power balance penalty, capacity limits penalty, ramp limits penalty, and prohibited operating zone penalty, respectively. The resultant value is then added to the total generation cost. In such a case, the possibility of a solution getting discarded is very high if it does not satisfy the constraints. Test cases 2 and 4 are not given the data to calculate ramp limits and prohibited operating zone penalties. Therefore, there is a dash (-) corresponding to these values in Table 1. The total penalty (TP) is computed by:

$$TP = \lambda_1 \times PBP + \lambda_2 \times CLP + \lambda_3 \times RLP + \lambda_4 \times POZP \tag{13}$$

where λ_1, λ_2, λ_3, and λ_4 are constants given in Table 1.

The fitness function (1) with the above set of constraints (2), (4), (5), (6), (7) is considered as in (14) using the penalty function method.

$$\begin{aligned} \min F_t &= \sum_{i=1}^{D} F_i(P_i) \\ &= \sum_{i=1}^{D} (a_i P_i^2 + b_i P_i + c_i) + e_i \\ &\times |\sin(f_i \times (P_i^{\min} - P_i))| + TP \end{aligned} \tag{14}$$

Table 1 Value of λ in different test cases

Test Cases	λ_1	λ_2	λ_3	λ_4
1	10^3	10^3	10^5	10^5
2	10^5	10^3	–	–
3	10^3	10^3	10^5	10^5
4	10^5	10^3	–	–
5	10^7	10^5	10^7	10^5

We will use the above concepts to describe our algorithm in the next section.

4 Proposed algorithm

This section describes the proposed algorithm (CSMA) in detail. The optimization algorithms start their execution with the randomly initialized candidate solutions in the specified boundary as the initial search direction is not known in complex problems. Like other optimization algorithms, the proposed method starts with the randomly initialized population. Mostly, in the random initialization of solutions, the boundaries of the search domain and a minimum of one random value are used. The boundaries of the search domain depend on the problem taken under consideration. In this study, the lower and upper limits of generating units in the power plants decide the boundaries of the search domain. Each randomly initialized solution is a vector of size $1 \times D$. Here, D is treated as the dimension of the problem taken under consideration. In this work, D is the number of generating units in the power plant. The solutions for each of the D generators are randomly initialized by:

$$P_{ji} = P_i^{\min} + r \times \left(P_i^{\max} - P_i^{\min}\right) \tag{15}$$

where i varies from 1 to D and j varies from 1 to N (N is the population size). $P_i^{\min}$ and $P_i^{\max}$ are the lower and upper limits, respectively, of the ith generator. P_{ji} is the power generated by the ith generator at jth individual in the population, and r is a random value between 0 to 1. After random initialization, each solution needs to be evaluated using (14). Here, each solution can be treated as a slime mould. Power demand is passed in the program code during the optimization process. The algorithms are supposed to generate supplied power demand with the least generation cost. Meanwhile, it is also desirable to have minimum power loss and power balance penalty. In this study, all algorithms motivate costly generating units to produce minimum power to minimize the total generation cost.

Based on the total generation cost, the most appropriate scheduling of generating units can be identified for the first iteration, and relevant values can be saved for future use. Afterward, each solution needs to be updated to identify other possible generating units scheduling. The solution updating method depends on the algorithm taken into consideration. Different algorithms have different solution updating methods. These methods intentionally try to implement exploration and exploitation of solutions to achieve the desired goal. The mathematical expression for updating the position of slime mould is given in (16).

$$X = \left\{ \begin{array}{ll} (P_i^{\min} + \alpha.(P_i^{\max} - P_i^{\min}), & r < z \\ X_b(t) + V_b.(W.X_A(t) - X_B(t)), & r < p \\ V_c.X(t), & r \geq p \end{array} \right\} \tag{16}$$

where $V_b \in [-a, a]$ and V_c decreases linearly from 1 to 0. X_b and t represent the individual location with the highest odour concentration currently found, and current iteration, respectively. X_A and X_B are two randomly selected individuals from slime mould. W and X represent the weight and position, respectively of a slime mould. In this study, parameter z is set to 0.03. Here, it should be noted that each variable in (16) is in the form of a vector of size $1 \times D$.

p is defined according to (17)

$$p = \tanh |F(i) - BF| \tag{17}$$

where $i \in 1, 2, \cdots, N$, $F(i)$ and BF are the fitness of i-th individual and the best fitness, respectively found so far. The value of a is calculated using (18):

$$a = \operatorname{arctanh}\left(-(\frac{t}{T}) + 1 \right) \tag{18}$$

where T represents the maximum number of iterations considered in this study. For the best $N/2$ solutions, W is calculated using (19).

$$W = 1 + r.log\left(\frac{BF - F(i)}{BF - WF} + 1\right) \tag{19}$$

For the remaining $N/2$ solutions, W is calculated using (20).

$$W = 1 - r.log\left(\frac{BF - F(i)}{BF - WF} + 1\right) \tag{20}$$

where BF and WF represent the best and the worst fitness values, respectively found in the current iteration.

In (16), $\alpha \in (0, 1)$ is a chaotic sequence of size $1 \times D$ created by the logistic chaotic map. This map is mathematically formulated as follows [75]:

$$x_{t+1} = cx_t(1 - x_t) \tag{21}$$

here, $c = 4$, and $x_t = 0.75$.

The proposed method for solving ELD problems is given in Algorithm 1.

Algorithm 1 CSMA.

Input: Objective function (14), $N = 25$, initial population given in (15), independent runs $R = 30$, maximum iterations $T = 600$, cost coefficient, B-coefficient, lower and upper limits of power generators, and the power demand.
Output: Optimal scheduling of generators, total power generation, power loss, power balance penalty, total generation cost.
$m = 1$.
while $m \leq R$ **do**
  Initialize each solution according to (15).
  $t = 1$.
  while $t \leq T$ **do**
    Evaluate each solution of the population using (14).
    Sort the fitness values in ascending order.
    Use (19) and (20) to calculate W.
    for each slime mould **do**
      Update V_b, V_c, p.
      Update the position of slime mould using (16).
    end for
    $t = t + 1$.
  end while
  Save optimal scheduling of generators, total power generated by the generators, power loss, power balance penalty, and total generation cost.
  $m = m + 1$.
end while
Save the final outputs.
return outputs.

4.1 Analysis of computational complexity

The proposed algorithm mainly consists of random initialization, fitness evaluation, sorting, and population update. The computational complexity of random initialization and fitness evaluation is $O(N \times D)$ and $O(N)$, respectively. The computational complexity of sorting and population update is $O(\text{N log N})$ and $O(N \times D)$, respectively. In this study, the maximum iterations and number of independent runs are T and R, respectively. Therefore, the total computational complexity of the proposed algorithm is $O(R \times (N \times D + T \times N(1 + \text{log N} + D)))$.

In this study, the computational complexity of algorithms depends on N, T, R, D. These parameters are the same for all algorithms considered. In this work, the flow of the program execution is the same for all the algorithms, which are (1) the random initialization of population, (2) fitness evaluation, (3) creation of new solutions, (4) selection of the top solutions. Therefore, the computational complexity of the remaining algorithms will be similar to that of the proposed approach.

5 Datasets and parameter setting

We used five test cases with 6, 13, 15, 40, and 140 generators to compare the performance of the algorithms. Power demands and references of these test cases are given in Table 2. The cost coefficients, minimum and maximum power generation capacity of generating units, B-coefficients, and other relevant information of used datasets can be found in detail in respective references.

5.1 Experimental setup

We compared the performance of the proposed approach against seven algorithms: slime mould algorithm (SMA) [24], salp swarm algorithm (SSA) [69], moth flame optimizer (MFO) [66], grey wolf optimizer (GWO) [67], biogeography-based optimizer (BBO) [68], grasshopper optimization algorithm (GOA) [70], and multi-verse optimizer (MVO) [71]. The individual parameters of these algorithms are set according to the respective paper. However, three parameters that are common for all algorithms are set as follow:

- Population size *(N)* = 25.
- Maximum iterations *(T)* = 600.
- Independent runs *(R)* = 30.

For a fair comparison, all algorithms are implemented in MATLAB R2017a on a machine with 8GB of RAM and a Core-i5 processor. All algorithms are executed over 30 independent runs. We compute a solution that meets the load demand with minimum generation cost in each run. In other words, we compute a solution with a minimum balance penalty. If two solutions have the same balance penalty, we save the solution with minimum generation cost. At the same time, we keep the total generation cost for the minimum balance penalty concerning iteration. Out of 30 independent runs, we stored the best solution for the respective algorithms and test cases.

Table 2 Description of datasets used in this study

Sr. No.	Test Case	# generators	Power Demand (MW)	Reference
1	Test case 1	6	1263	[72]
2	Test case 2	13	1800	[76]
3	Test case 3	15	2630	[77]
4	Test case 4	40	10500	[76]
5	Test case 5	140	49342	[78]

Table 3 Comparison of the experimental results for test case 1

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P1	452.1021	429.0046	454.133	449.3141	469.165	453.7971	452.0043	496.0728
P2	172.096	173.0249	199.9928	179.4009	169.1438	175.3897	175.4473	177.9637
P3	251.8451	261.2887	257.3872	252.4426	264.303	246.2134	259.7947	248.1452
P4	128.9404	132.2047	128.738	150	128.0659	140.4549	135.1351	148.6548
P5	164.1094	173.4473	163.5624	150.2007	157.1685	167.6643	157.803	139.7994
P6	106.7149	106.8766	71.9534	93.7543	87.9716	91.9848	95.3758	64.6089
TPG	1275.8079	1275.8468	1275.7668	1275.1126	1275.8178	1275.5042	1275.5602	1275.245
PL	12.8079	12.8469	12.7668	12.1126	12.7637	12.5039	12.5602	12.25
PBP	0	1.00E-04	0	2.27E-13	0.0541	0.0003	2.27E-13	0.0052
TGC	15451.89584	15453.11477	15455.49006	15448.51678	15505.49444	15448.13172	**15446.49854**	15479.2

Bold entries represent the best (optimized) cost to meet specified power demands

6 Analysis of experimental results

This section presents the analysis of experimental results. We performed the analysis in two ways based on (1) the quantitative values of the power balance penalty, power loss, total generation cost, and (2) the statistical test. We conducted various experiments to identify the best possible method to solve ELD problems. Tables 3, 4, 5, 6, 7 and 8 represent the comparison of experimental values of algorithms for test cases 1-5 ELD problems on load demands of 1263MW, 1800MW, 2630MW, 10500MW, and 49342MW, respectively. These tables represent the optimal power output of each generating unit, total power output, power loss, power balance penalty, and total generation cost. The minimum values of total generation cost, power loss, and power balance penalty are desirable.

Analysis of Tables 3–8 prove the effectiveness of the proposed approach in solving ELD problems. Table 4 shows that SSA crosses the boundary of the search domain in most of the cases, which is not desirable. In such cases, algorithms find difficulty in identifying the optimal power output of generating units. To avoid such problems, modular clamping could be more effective instead of boundary clamping. From Tables 5–8, it is easy to conclude that MFO crosses the search domain for some generating units. The reason could be the updating method adopted in MFO that

Table 4 Comparison of the experimental results for test case 2

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P1	456.1793	635.4627	679.9999	537.601	356.5023	530.5871	430.8568	472.721
P2	359.9911	0	0.0001	0	170.6964	222.6766	279.3771	127.6703
P3	58.1085	132.0909	360	225.5374	157.6633	142.9125	225.7135	106.2248
P4	166.6687	142.1908	60	180	82.7489	124.5816	91.2189	177.3811
P5	60.354	177.3557	60	144.0892	140.9382	162.5963	108.9245	75.8999
P6	170.912	177.7333	60	105.0287	169.5707	126.8706	156.7571	171.3589
P7	60.4646	60	60	102.2321	115.0123	68.1776	78.4613	61.4193
P8	168.3948	60.1755	60	60	162.6712	66.7819	109.8634	180
P9	60.0187	112.0158	60.0001	108.9372	153.7838	73.1143	60	135.203
P10	40.1758	41.5267	40	40	42.0602	54.6918	71.9579	110.4757
P11	40	88.989	119.9999	115.0877	66.4048	56.6178	76.8698	41.3109
P12	55.1596	79.1544	120	98.5675	108.3154	78.831	55	84.3163
P13	103.5729	93.3052	120	82.9192	73.5981	91.6423	55	56.0184
TPG	1800	1800	1800	1800	1799.966	1800.081	1800	1800
PL	0	0	0	0	0	0	0	0
PBP	0	0	0	2.27E-13	0.0344	0.0814	0.0003	0.0004
TGC	18701.48614	18932.83	18751.11	**18631.65**	22542.18	27127.16	18716.94	19488.14

Bold entries represent the best (optimized) cost to meet specified power demands

Table 5 Comparison of the experimental results for test case 3

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P1	442.3627	411.9042	453.3456	414.669	449.2663	417.1372	441.9616	395.9525
P2	362.1448	364.9595	370.8865	347.7588	373.4517	361.1331	361.4166	270.7895
P3	129.9984	129.9996	128.8434	130	122.9923	126.9073	127.9938	129.0111
P4	129.9994	127.8854	127.6552	130	123.8456	119.7826	127.9885	123.3212
P5	150.1493	154.2636	152.2417	150	155.5002	161.1515	158.0847	153.985
P6	459.9591	455.8144	458.0856	459.1144	458.5167	415.3898	460	459.0193
P7	429.9094	404.2705	400.735	397.9086	404.5571	413.3875	429.6221	416.6748
P8	60.4653	60.2894	105.7626	60	133.8719	127.0909	153.6797	137.7381
P9	161.4039	157.3844	110.4698	162	117.0195	116.3765	25	143.191
P10	155.5549	158.7252	111.7072	135.1363	140.4845	129.1154	81.2059	158.1521
P11	20.7804	70.5312	62.8821	70.8672	32.6746	59.5413	80	78.3053
P12	79.9997	74.6833	51.8546	79.7251	22.0828	72.9551	80	65.6436
P13	25.6061	29.1402	57.1767	45.7809	44.6857	73.21	46.5231	75.2504
P14	42.5303	17.3164	54.9603	50.0125	51.0037	32.7375	44.5052	40.5552
P15	15.2454	47.5947	15.8565	29.8765	37.9998	38.2458	42.0039	21.0709
TPG	2666.1091	2664.762	2662.463	2662.849	2667.952	2664.162	2659.985	2668.66
PL	36.0888	34.762	32.4628	32.8493	37.8874	34.1967	29.9864	38.6566
PBP	0.0203	0	0	0	0.065	0.0352	0.0013	0.0034
TCG	32911.81303	32929.19	32934.52	32963.97	33123.18	33091.4	**32873.84**	33144.73

Bold entries represent the best (optimized) cost to meet specified power demands

motivates the solutions to cross the boundary range of the search domain.

Table 9 represents the ranking of each algorithm for different test cases and average ranking. This ranking has been calculated based on the total generation cost. An algorithm with the least generation cost got rank 1 (best). This table shows that CSMA outperforms SMA in all test cases considered in this study. The reason behind the competitive performance of CSMA could be the inclusion of chaotic sequences generated by the logistic chaotic map. For test cases 4 and 5, CSMA got the first rank. These test cases correspond to 40 and 140 generators ELD problems. In this study, the number of generators is treated as the dimension of the optimization problem. Therefore, it can be concluded that the proposed algorithm will perform well in higher dimensional optimization problems. The average ranking of MVO is 7, which is the worst among all approaches considered. In this work, GOA got the second position.

In this study, each algorithm is executed 30 times. Each time, the maximum number of iterations is taken as 600. The best value of the total generation cost in each independent run is saved. Therefore, the best 30 values of total generation cost are obtained at the end of the program execution. These values are used in box plots. Figures 4b, 5 and 6b represent the box plots of algorithms for different test cases. A careful observation of these plots justifies the effectiveness of CSMA. On the other hand, an observation of total generation cost concerning iterations is also a method of comparing the performances of the algorithms. To do so, convergence curves of algorithms for all test cases have been plotted. Figures 2a, 3 and 4a show the convergence curves. In minimization problems, a high rate of decrease in objective values is desirable. The suggested algorithm has shown similar behavior during the program execution (Figs. 5 and 6).

To validate the competitiveness of the proposed algorithm statistically, three statistical tests (Friedman test [79], Iman-Davenport test [80], Holm test [81]) have been performed at a 5% significant level. Friedman test is one of the nonparametric tests mathematically formulated as follows.

$$FT = \frac{12}{nA(A+1)} \sum_{j=1}^{A} R_j^2 - 3n(A+1) \tag{22}$$

Here, FT represents the Friedman test statistical value. n and A are the number of test cases and the number of algorithms, respectively. In this work, $n = 5$, and $A = 8$. R_j represents the sum of ranks for the j-th algorithm. The calculated Friedman test value is now compared with the table of Chi-square statistics by considering degree of freedom = 7(number of algorithms - 1) at a significance level $\alpha = 0.05$. The p-value for a given Friedman test value can be calculated from https://www.socscistatistics.com/pvalues/chidistribution.aspx.

Table 6 Comparison of the experimental results for test case 4

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P1	56.6088	111.7044	105.3791	48.1435	53.4627	79.2026	97.0355	81.8181
P2	110.27	62.0484	106.2879	36	62.2056	97.3916	106.2604	114
P3	60.0012	70.3685	101.4398	117.5921	106.3848	82.0864	114.796	112.4868
P4	188.7847	170.9622	189.3892	93.3786	104.5871	113.3507	148.9523	150.2621
P5	92.6865	80.1288	59.3508	47.437	75.4158	65.7503	97	56.4089
P6	68.534	136.5748	84.3609	123.7405	122.6863	105.0801	135.3812	139.1904
P7	274.6126	110.022	150.1128	300	178.6965	213.7192	272.769	177.0855
P8	191.2145	299.9953	294.2138	300	269.2612	267.9599	299.4974	298.2995
P9	280.5952	292.3559	290.7083	166.0045	287.871	259.0893	299.9718	268.4233
P10	206.0184	298.3512	292.1319	130	156.2688	242.1177	255.6323	235.2451
P11	294.9264	317.5832	186.932	263.7924	229.8713	311.0256	370.7238	371.8589
P12	302.018	105.6665	170.1992	304.4923	168.6575	308.731	205.1913	324.8202
P13	148.9083	463.8908	131.1266	489.7033	425.5147	416.3951	474.0741	313.5656
P14	461.5775	496.4607	499.1159	432.9267	497.3051	404.9229	252.2187	395.2939
P15	471.8658	354.9193	327.6567	500	424.8819	392.0007	371.0152	497.5048
P16	499.6436	494.7099	281.3945	500	467.5326	426.574	285.0095	464.6349
P17	499.855	499.1105	499.416	466.3732	479.6155	441.1167	439.5455	447.0674
P18	499.9579	263.3216	335.296	500	440.9322	469.0858	428.5404	499.0463
P19	510.941	545.2287	549.5749	256.1748	514.9695	476.6316	496.5081	247.1646
P20	548.8863	549.9944	541.4355	460.7275	525.1793	499.8877	470.4407	324.3734
P21	513.5603	540.1189	548.8805	511.0406	546.9783	502.3242	549.7075	501.9908
P22	549.9838	532.7856	530.9896	549.9962	525.4683	491.1793	548.1523	544.7195
P23	549.741	524.4338	546.1555	550	525.9502	438.4428	518.0482	419.9503
P24	254.0001	532.6146	533.1362	545.3031	514.0755	514.751	398.5565	518.1818
P25	549.988	460.1178	456.2145	332.2258	531.2003	439.6494	549.9979	550
P26	548.2001	512.6982	542.6873	497.5983	505.8669	524.8136	372.0683	411.9595
P27	13.0883	32.1278	72.2192	10	46.9861	65.0718	51.1433	125.9788
P28	10.0006	119.0028	19.197	72.345	22.9725	92.0527	56.8809	148.8649
P29	10.9041	40.204	66.7804	39.6085	11.8075	52.876	58.1429	48.3594
P30	94.5227	53.1188	96.1897	97	78.279	79.7771	96.9761	69.4158
P31	188.4519	189.7552	150.9867	190	155.9839	161.7881	189.9999	175.699
P32	189.5851	139.992	173.3299	162.2198	91.4303	143.4472	181.6041	93.5086
P33	61.9416	189.6109	186.1879	190	184.9948	150.9609	77.9802	137.0916
P34	178.7561	91.6728	180.0767	170.7627	125.0216	138.5553	129.6379	199.6099
P35	199.9947	171.2064	199.6233	200	140.0354	155.4158	196.7152	200
P36	190.3114	92.3108	137.5377	200	176.2554	171.1119	189.7362	150.6107
P37	32.9694	25.2744	101.7723	25	55.6671	59.9581	103.3023	75.1156
P38	25.0633	97.4334	107.748	45.4136	37.2699	85.2821	105.7938	47.496
P39	27.582	109.8424	109.3305	25	98.1883	77.4087	25.0001	66.6809
P40	543.4498	322.2824	545.4353	550	534.2801	483.0384	479.9932	496.2229
TPG	10500	10500.0001	10500	10500	10500.01	10500.02	10500	10500.01
PL	0	0	0	0	0	0	0	0
PBP	0	1.00E-04	0	0	0.0108	0.0233	0	0.0057
TCG	**129612.3173**	137380.41	131430.9	134038.8	131228.4	138667.4	133676.8	150002.6

Bold entries represent the best (optimized) cost to meet specified power demands

Table 7 Comparison of the experimental results for test case 5

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P1	77.4728	73.3944	118.1117	119	76.4016	86.8417	71.9203	81.0578
P2	150.0797	121.0671	151.9062	121.704	148.0027	136.5155	124.5906	145.1823
P3	159.9411	189.9252	174.9891	151.8194	181.0803	162.7652	127.6347	154.953
P4	184.2693	155.1537	186.651	178.9847	178.6762	162.202	188.0234	164.2585
P5	173.7841	90.5935	152.1581	90	161.1127	152.6753	177.8161	107.2449
P6	188.2131	114.5141	189.4516	190	124.411	136.1504	93.6604	188.3866
P7	489.469	289.1811	375.6716	477.8952	393.7455	314.5827	414.9862	378.7311
P8	286.5923	394.2651	418.1627	490	467.4548	404.7332	290.2061	332.984
P9	405.6354	456.6575	425.0562	496	473.343	406.2614	471.485	419.446
P10	415.9994	260.1061	493.5377	284.6985	454.1539	354.5122	261.7837	319.6923
P11	391.5847	471.5842	483.3189	460.1181	494.8724	490.6619	486.318	327.2114
P12	399.1098	430.6027	361.7197	261.4077	279.6925	433.922	457.4423	407.6214
P13	496.1587	365.0601	377.2994	489.9187	442.4291	494.3088	472.9528	462.2952
P14	491.1538	404.0223	420.7256	260	445.8683	401.1594	505.9853	443.2314
P15	366.4118	453.1569	298.9294	427.4724	454.5995	408.661	435.3757	457.2514
P16	387.1772	347.3368	339.4679	260	343.8313	359.429	345.5399	289.1565
P17	496.0091	359.2237	376.5307	442.3007	451.0703	452.8235	506	262.804
P18	427.0241	345.9771	500.128	506	482.6971	398.6512	448.7479	415.8435
P19	260	504.5314	309.3918	505	458.0311	497.691	433.2677	366.4869
P20	477.8297	504.8196	473.8439	287.8062	341.4342	436.38	428.3365	418.1566
P21	447.1571	449.2819	486.7028	505	480.4599	374.4708	465.3488	429.4026
P22	485.306	448.0837	296.6246	366.0196	487.2223	395.6001	435.9382	436.8846
P23	490.0389	444.6373	490.4175	487.5186	370.9565	409.9914	350.1997	496.5669
P24	428.8585	468.4944	498.8792	395.1974	406.5087	404.3874	443.5176	481.1926
P25	536.9538	280.2722	524.3467	537	488.2437	298.286	344.1031	524.7636
P26	280.0138	514.9762	288.8348	537	424.0139	430.1844	382.3786	402.3752
P27	345.6383	528.0222	540.5842	495.5133	441.843	531.8759	407.0899	311.3758
P28	532.5583	543.2599	451.9365	280	367.6025	479.6886	516.2797	515.2719
P29	386.3472	499.0761	479.0983	501	450.3671	437.0765	453.086	408.4647
P30	443.8706	358.687	458.593	359.9835	377.7194	423.0413	482.4497	490.5552
P31	505.9825	398.0884	321.7369	435.8686	349.2668	483.8573	336.5195	473.4224
P32	454.1763	446.2488	389.5783	393.663	442.0896	347.628	492.0128	491.3192
P33	416.206	505.9855	461.8384	311.6391	434.6745	400.9106	347.4653	462.3993
P34	357.4958	338.0513	479.3187	506	307.2888	390.2708	467.7798	449.8601
P35	425.8886	348.9051	345.7191	260	344.0444	428.7789	293.1836	484.4252
P36	494.4212	499.997	326.8902	461.7627	475.0341	445.0952	449.7267	441.0361
P37	217.4733	183.6307	222.2423	241	235.1797	165.4058	203.4266	122.6452
P38	133.6161	126.7621	232.1038	241	215.6307	190.2664	126.8608	218.4799
P39	773.7813	595.402	765.8387	428.1906	721.2376	642.2526	744.1953	613.0105
P40	751.8299	657.437	731.7187	769	747.0165	703.2876	658.2712	582.4881
P41	7.7334	7.5313	18.2237	3	18.9109	4.9498	15.6144	11.1216
P42	7.6204	17.4129	16.7409	28	7.0689	14.0123	3.0251	15.7651
P43	161.14	219.0449	243.0378	249.9768	160.7254	184.5803	214.448	242.2967
P44	207.8936	165.4296	249.0376	250	228.982	218.2821	197.2887	245.3545
P45	242.8466	174.1658	249.5649	250	204.1362	217.5414	236.0512	220.6237
P46	202.0357	177.3213	175.199	236.5814	219.7275	218.1154	231.481	206.6866
P47	204.7325	243.142	200.2765	160.1511	225.4587	224.857	240.1183	178.1489
P48	242.5101	222.3759	160.0237	231.7339	214.7964	211.7637	187.2127	161.9942
P49	240.2459	174.0533	165.568	160	232.6451	185.7102	199.001	177.0373

Table 7 (continued)

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P50	244.4358	184.2352	183.7855	197.7211	227.1553	200.2953	183.3715	205.8226
P51	323.1738	475.9821	242.9835	165	402.9733	321.5263	346.896	287.3464
P52	504	447.8945	502.4677	504	344.8789	398.1737	182.3455	428.9521
P53	494.6062	492.4716	300.1375	468.4351	431.3104	445.3387	229.4769	492.855
P54	381.1127	298.9918	499.2836	165	315.0965	383.3798	305.1919	369.2838
P55	463.4207	219.4824	454.4044	468.7232	424.8132	413.1093	462.8156	465.6795
P56	466.7117	485.2498	509.4481	312.7879	401.1815	436.2861	277.0089	499.4454
P57	216.0202	103.3902	317.4957	103	129.6951	274.5811	238.0314	256.3737
P58	499.2885	556.5314	257.7325	617	603.8345	398.1626	479.6964	256.895
P59	165.9998	292.2943	199.5912	294.9718	294.1739	185.8127	172.6762	279.8837
P60	298.6933	454.9104	292.3085	471	299.2256	385.2088	313.6063	172.438
P61	302.3841	481.2713	329.1479	379.2972	199.3959	335.2788	427.2534	299.3742
P62	142.3553	283.9263	162.6819	290.6553	153.5669	246.6347	243.2729	284.4478
P63	507.5928	502.4973	278.0768	329.7275	267.6598	433.7826	284.3413	205.5805
P64	272.6857	500.353	273.0249	511	455.5026	351.6755	164.6184	465.5135
P65	372.437	474.2131	336.094	490	362.3292	277.3936	367.3103	348.23
P66	413.5545	348.3392	389.3089	196	217.686	478.7321	336.4191	394.7427
P67	377.1664	370.8394	416.6342	490	346.5332	406.1275	485.6841	489.3291
P68	236.2541	488.4581	301.9846	196	403.8326	442.5585	459.3381	347.3527
P69	330.2952	159.5684	381.4455	408.4331	343.553	268.3647	200.386	303.7244
P70	344.3013	289.455	301.0155	364.1906	279.32	282.3124	302.0286	403.7981
P71	167.6646	267.084	223.0095	454.9889	288.6672	280.318	246.4253	211.3846
P72	405.1083	454.7411	294.8444	429.7587	438.5352	339.892	453.4153	277.3599

Iman-Davenport test This test is derived from Friedman test [80] and mathematically expressed as follows.

$$IDT = \frac{(n-1) \times FT}{n \times (A-1) - FT} \quad (23)$$

where *IDT* is the Iman-Davenport test statistical value. The null hypothesis got rejected as the statistical value is greater than the critical value. The p-value for a given *IDT* can be evaluated from https://www.socscistatistics.com/pvalues/fdistribution.aspx.

Holm test This is one of the post hoc tests and mathematically expressed as follows.

$$HT = (R_i - R_j)/\sqrt{\frac{A \times (A+1)}{6n}} \quad (24)$$

Where *HT* is Holm's test statistical value. Here, R_j is the average rank of the proposed algorithm, whereas R_i represents the average rank of the algorithm that is taken under consideration from the remaining algorithms. The p-value for a given statistical value of the Holm test can be calculated from https://www.socscistatistics.com/pvalues/normaldistribution.aspx. These tests are conducted on the total generation cost values that meet the specific load demand. In these tests, the rejection of the NULL hypothesis indicates a significant difference in the performance of considered algorithms. However, the non-rejected NULL hypothesis indicates that the algorithms perform comparable, i.e., statistically, there is no difference in the performance of the algorithms.

Table 10 shows the experimental values of Friedman and Iman-Davenport tests. The rejection of the null hypothesis for both cases indicates a significant difference among the performances of the algorithms considered in this work. A post hoc test (Holm test) is performed to check whether there is a significant difference between the proposed algorithm and the rest of the approaches. The best performing approach (CSMA) is considered a control algorithm to calculate the statistical values and p-values. Table 11 shows the values of the Holm test. The rejection of the NULL hypothesis for MVO, GWO, BBO indicates that CSMA performs statistically better than these algorithms. The NULL hypothesis for SMA, SSA, MFO, and GOA is not rejected which indicates that CSMA does not perform statistically better than these algorithms. However, the effectiveness of CSMA against these algorithms can be easily seen from Table 9.

Based on the experimental results and various comparative performance analyses, CSMA exhibits the best performance compared to other algorithms. The competitiveness

Table 8 Comparison of the experimental results for test case 5 cont...

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P73	342.9759	294.6996	360.3213	541	535.7507	404.9785	528.1878	464.4063
P74	216.3166	532.334	478.4643	536	461.9887	350.7128	520.4232	387.0529
P75	203.8221	432.0613	353.7201	429.3069	524.8794	494.7253	176.6602	378.9769
P76	375.512	489.8397	492.6326	417.5693	343.4826	414.074	492.9229	354.3834
P77	327.0396	539.9354	227.4967	175	309.3388	213.4494	492.1318	537.8278
P78	443.6271	457.9752	353.8265	540.2377	551.7233	517.8524	459.6522	451.0421
P79	523.1989	402.4966	344.6953	522.2994	443.518	414.3765	445.0007	336.0092
P80	530.5542	185.3222	455.8446	279.9324	252.1218	438.7996	479.6594	378.2642
P81	499.7106	538.5554	334.0833	363.3908	530.567	394.5317	351.6283	472.2309
P82	56	104.6918	111.4338	74.4671	108.2281	121.9051	79.6545	70.6115
P83	216.7871	200.1433	173.6775	245	134.3887	186.0384	196.5856	242.8647
P84	178.3296	183.5177	179.143	189.9843	125.8314	174.8195	121.8445	164.7492
P85	143.1569	129.825	171.3209	115	220.0275	198.6934	154.3966	230.6023
P86	251.4759	243.1909	274.4427	303.5084	295.2932	273.9958	246.112	302.6432
P87	235.5074	254.9104	267.3516	215.2356	271.279	265.403	281.0073	229.8411
P88	176.5254	255.6926	251.3265	175	281.3763	310.1579	342.1578	340.3538
P89	338.2206	207.591	323.0475	203.6252	179.4915	322.9022	253.5462	315.5067
P90	230.402	247.2577	340.4937	323.2467	226.3251	247.7574	337.4549	215.3024
P91	178.5985	183.4469	322.5957	175	217.9268	240.0415	242.7773	225.3382
P92	575.2898	557.7463	573.9161	542.6587	555.101	562.3627	539.9199	569.599
P93	522.8033	521.9393	539.0381	518.3468	524.2908	525.9612	531.7254	535.2415
P94	815.9595	795.0052	817.3529	834.5254	824.9991	824.9902	825.6621	828.6131
P95	816.6081	834.8441	835.8697	795	806.2576	813.6664	835.53	825.6886
P96	590.6866	615.4733	675.1559	578	637.5226	618.2086	664.1675	580.8262
P97	719.9417	675.0375	701.2872	713.49	658.6637	675.3694	678.1231	701.4669
P98	620.6947	717.8999	663.6228	718	690.7655	666.1125	714.5943	634.8006
P99	694.4985	719.8944	703.4621	619.4024	679.8593	673.8758	718.438	632.9806
P100	957.2418	886.7449	929.9415	964	907.5994	922.4305	888.1742	893.7299
P101	954.2716	944.3924	955.6371	958	888.2694	891.4551	956.137	956.9913
P102	867.3491	943.2733	922.2077	859.9809	883.6153	924.497	945.2469	912.5357
P103	926.1141	915.3124	888.2142	864.7096	922.476	867.2955	846.1395	918.4437
P104	909.8812	898.4979	928.855	885.3699	909.8899	891.6031	927.5019	923.8367
P105	867.2403	859.8458	866.03	868.1344	851.7639	862.732	856.9918	854.4255
P106	865.1416	867.9925	833.7625	871.9649	868.8118	843.1751	873.6723	870.0345
P107	842.5247	844.2482	844.2763	837.1458	866.8342	863.5153	861.9037	872.1373
P108	871.2472	866.5193	875.6149	874.301	853.2344	861.051	876.8637	824.5837
P109	844.482	819.5897	801.1017	865.2799	840.901	839.7705	868.7925	805.4522
P110	799.8528	795.1354	858.6655	798.9687	818.4574	855.7987	862.8324	822.9444
P111	814.6186	848.4687	851.2424	861.9141	866.6489	869.7675	872.5109	861.8811
P112	159.359	121.4196	193.5488	94.1588	114.3979	150.582	128.7571	143.3379
P113	200.612	120.2216	201.9706	184.2298	139.8471	140.8712	143.5149	172.6618
P114	111.0953	95.0511	102.5404	203	125.7351	155.6551	190.3734	190.1394
P115	274.3602	350.7418	371.9708	316.956	363.6156	321.3182	376.7	365.6662
P116	287.1789	262.9812	270.7993	373.8839	276.3602	269.0315	357.2407	325.2179
P117	373.6938	255.5677	295.0682	378.9996	334.2428	307.7109	244	268.8996
P118	167.5294	188.1084	156.8319	95	148.1264	152.479	188.3918	167.3204
P119	152.7825	188.4822	179.6595	95.343	171.1407	161.8016	110.1991	103.7996
P120	116.0195	136.2572	188.9641	128.7835	174.4098	161.348	194	186.1987

Table 8 (continued)

Power	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
P121	302.4387	218.9639	198.9762	307.7353	316.5888	254.311	320.4385	302.3654
P122	4.2479	2.0641	9.2651	17.7702	6.1652	10.0783	15.2946	10.5674
P123	53.2732	35.9212	57.8264	59	48.4608	40.1275	37.2135	24.0524
P124	46.8711	16.789	50.6162	15	24.1628	42.4624	59.407	81.075
P125	48.3572	19.8603	13.6667	9.8639	14.0325	42.5497	41.4248	10.6652
P126	30.7797	21.011	29.1365	37	19.7208	30.2284	37	15.7831
P127	17.4196	25.8205	25.4816	10.5133	12.2315	28.3249	12.814	19.3954
P128	146.7311	124.4671	359.2557	112	234.8401	280.2164	276.9853	370.0679
P129	19.0518	5.3993	18.1868	16.8877	18.2136	7.8563	19.9864	15.7294
P130	20.5873	35.4891	23.245	29.3654	9.0641	15.3241	37.9884	36.4209
P131	13.3685	5.1227	13.3286	5	11.5232	9.0943	5.9504	5.3417
P132	60.484	67.1777	61.0187	50	79.2296	80.3598	74.9524	73.0282
P133	8.9211	5.0597	9.4342	5.2402	5.6159	5.6442	9.8519	7.4783
P134	44.6998	46.5964	68.7571	74	51.3174	66.504	72.8344	55.6672
P135	72.7236	50.0491	44.507	68.6575	65.1058	64.0164	61.4361	69.5997
P136	101.7478	41.4813	79.0642	41	45.584	85.7768	50.0428	95.5527
P137	38.2674	45.56	23.1646	17	38.1779	26.3798	50.9282	50.5777
P138	8.1225	8.4947	15.8652	19	13.652	12.7599	8.5672	17.8943
P139	13.3619	8.9059	15.4327	7	9.1468	12.3706	15.2215	10.5739
P140	26.1587	28.463	38.6144	26	39.4878	29.6289	26	39.8281
TPG	49342	49342	49342	49342	49342.37	49341.7037	49342	49341.9
PL	0	0	0	0	0	0	0	0
PBP	0.0001	0.0023	7.28E-12	7.28E-12	0.3734	0.2963	2.18E-11	0.1001
TCG	**1899719**	1945860	1915139	2308272	5649260	5211518.396	1907431	3953667

Bold entries represent the best (optimized) cost to meet specified power demands

and effectiveness of CSMA are due to its convergence rate and population diversity. CSMA can capture promising solutions from the entire search domain via its exploration and exploitation operators. The influential exploitation and exploration methods of an optimization algorithm avoid the problem of random search and local entrapment. In any optimization algorithm, the most important thing is the mechanism to update the position vectors in the search domain. Suppose the solutions of the population can visit a variety of locations from different sections of the search domain in a uniform fashion. In that case, the probability of getting the optimal global solution is very high.

In the proposed work, only a few parameters are needed to be adjusted. This could be one of the advantages of the suggested approach. In general, the performance of optimization algorithms degrades with an increase in dimensionalities. However, the experimental results indicate the superiority of the proposed algorithm in 40 and 140 generators ELD problems. This confirms that the suggested algorithm performs well in higher dimensional optimization problems. This could be another advantage of the proposed

Table 9 Ranking and average ranking of considered approaches based on total generation cost in different test cases

	Test Cases	CSMA	SMA	SSA	MFO	GWO	BBO	GOA	MVO
	Test case 1	4	5	6	3	8	2	1	7
	Test case 2	2	5	4	1	7	8	3	6
Ranking	Test case 3	2	3	4	5	7	6	1	8
	Test case 4	1	6	3	5	2	7	4	8
	Test case 5	1	4	3	5	8	7	2	6
Average ranking		2 (best)	4.6	4	3.8	6.4	6	2.2	7 (worst)

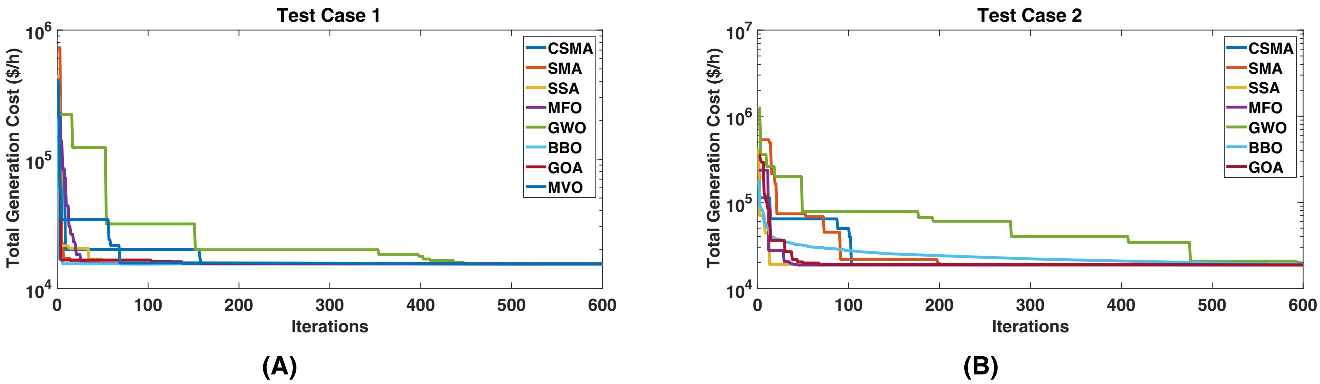

Fig. 2 Variation in total generation cost with respect to iterations for (**a**): test case 1, (**b**): test case 2

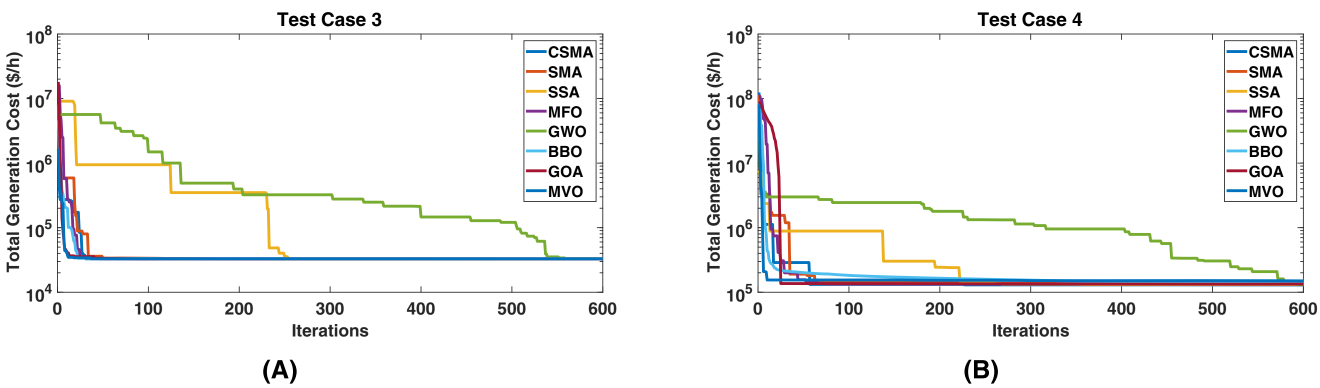

Fig. 3 Variation in total generation cost with respect to iterations for (**a**): test case 3, (**b**): test case 4

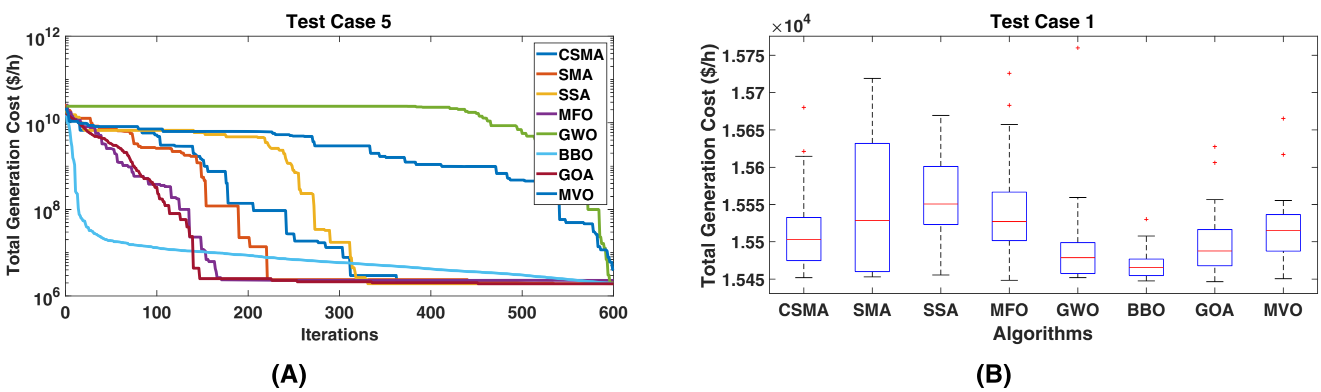

Fig. 4 (**a**): Variation in total generation cost with respect to iterations for test case 5, (**b**): Box plot for test case 1

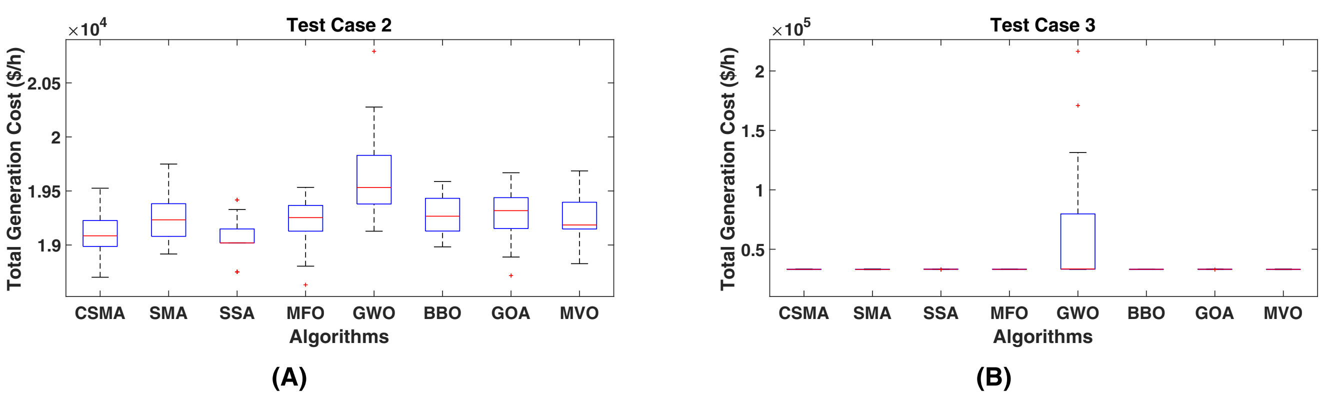

Fig. 5 Box plot for (**a**): test case 2 (**b**): test case 3

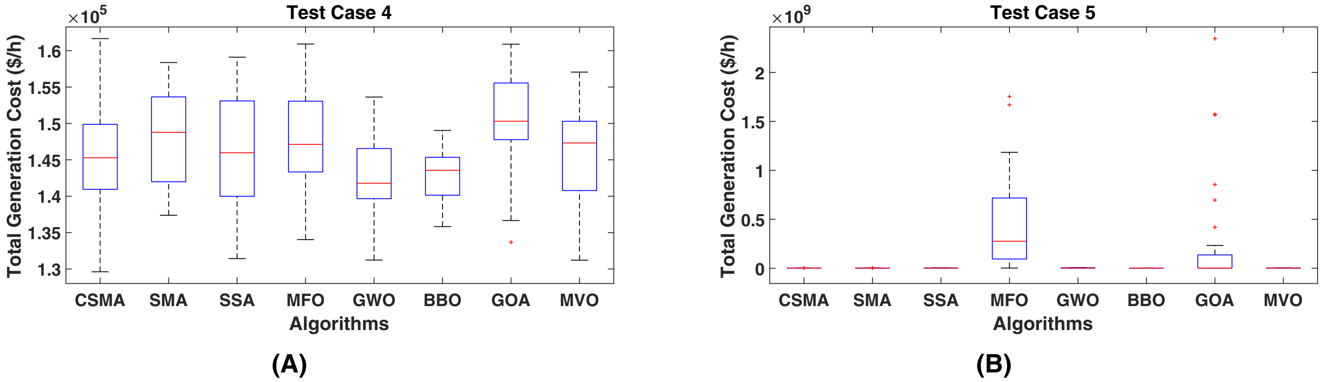

Fig. 6 Box plot for (**a**): test case 4 (**b**): test case 5

algorithm. The performance of the existing approaches decreases in higher dimensional optimization problems. Although the computational complexity of the considered algorithms is comparable, the suggested approach takes more execution time in some test cases. This could be the disadvantage of the proposed method. The computation for chaotic sequences for logistic map needs some execution time, which affects the overall computation time of the proposed algorithm. This might be the reason why the suggested algorithm takes more execution time in some test cases.

Although the suggested method outperformed other algorithms considered in this study in solving ELD problems, similar behavior in other complex applications is not guaranteed. To solve some special class of problems, algorithms needed to be adjusted according to the mathematical formulation of the objective function, set of constraints, dimensionalities, and search domain of the optimization problem. The proposed algorithm is modified to handle an ELD problem, a continuous optimization problem. Therefore, this algorithm might not be suitable for solving discrete optimization problems. This could be the limitation of the suggested approach. Some additional adjustment and parameter tuning needed to be done to solve discrete optimization problems.

7 Conclusions and future research directions

In this paper, the merits of chaotic sequences generated by a logistic chaotic map are integrated into a fast converging algorithm to solve economic load dispatch problems. The performance of the proposed approach is compared against seven recent state-of-the-art algorithms using five test cases. Based on the experimental values, it is easy to conclude that CSMA performs better than SMA in all test cases. The integration of chaotic sequences could be the main reason for the effectiveness of CSMA during the optimization process. We further validated the efficacy of the suggested approach by conducting three statistical tests (Friedman test, Iman-Davenport test, Holm test). Again, the proposed method has shown its robustness in solving ELD problems.

In the future, we would like to solve other real-world optimization problems such as electricity load forecasting, optimal controller placement problem in a software-defined network, and feature selection using CSMA. A multiobjective version of CSMA can be suggested to handle multiple conflicting objectives simultaneously.

Table 10 Friedman test (FT) and Iman-Davenport test (IDT) values based on total generation cost

Test	Values	p-Values	Null Hypothesis Rejected?
FT	20.3333	0.00489	Yes
IDT	5.5454	0.00044	Yes

Appendix: Abbreviations

ELD	Economic load dispatch
SMA	Slime mould algorithm
CSMA	Chaotic slime mould algorithm
GWO	Grey wolf optimizer
BBO	Biogeography-based optimization
SSA	Salp swarm algorithm

Table 11 Holm test (HT) values and p-values of different methods (CSMA is the control algorithm)

i	Algorithms	Values	p-Values	α/i	Null Hypothesis Rejected?
7	MVO	3.227681	0.000624	0.007142	Yes
6	GWO	2.840358	0.002254	0.008333	Yes
5	BBO	2.582144	0.00491	0.01	Yes
4	SMA	1.678393	0.046644	0.0125	No
3	SSA	1.291072	0.098352	0.016667	No
2	MFO	1.161965	0.122638	0.025	No
1	GOA	0.129107	0.448639	0.05	No

GOA	Grasshopper optimization algorithm
MFO	Moth-flame optimization
MVO	Multi-verse optimizer
TPG	Total power generation
P_L	Power loss
TGC	Total generation cost
PBP	Power balance penalty
CLP	Capacity limits penalty
RRLP	Ramp rate limits penalty
POZP	Prohibited operating zones penalty
N	Population size
D	Number of generating units
T	Maximum iterations
G.No.	Generating unit number
R	Independent runs
F_t	Total generation cost
P_i	Power generated by ith generating unit
$P_i^{\min}$	Minimum power generated by ith generating unit
$P_i^{\max}$	Maximum power generated by ith generating unit
P_D	Total power demand
$F_i(P_i)$	Fuel cost function of ith generator
a_i, b_i, c_i	Fuel cost coefficients of ith generator

Acknowledgements Funding information is not applicable/no funding was received.

Declarations

Conflict of Interests There is no conflict of interest.

References

1. Das D, Bhattacharya A, Ray RN (2020) Dragonfly algorithm for solving probabilistic economic load dispatch problems. Neural Comput and Applic 32(8):3029–3045
2. Chen G, Ren J, Feng EN (2016) Distributed finite-time economic dispatch of a network of energy resources. IEEE Transactions on Smart Grid 8(2):822–832
3. Sharma B, Prakash R, Tiwari S, Mishra KK (2017) A variant of environmental adaptation method with real parameter encoding and its application in economic load dispatch problem. Appl Intell 47(2):409–429
4. Elsayed WT, Hegazy YG, Bendary FM, El-Bages MS (2016) A review on accuracy issues related to solving the non-convex economic dispatch problem. Electr Power Syst Res 141:325–332
5. Chen G, Ding X (2015) Optimal economic dispatch with valve loading effect using self-adaptive firefly algorithm. Appl Intell 42(2):276–288
6. Singh T, Mishra KK et al (2019) Multiobjective environmental adaptation method for solving environmental/economic dispatch problem. Evol Intel 12(2):305–319
7. Singh T (2020) A novel data clustering approach based on whale optimization algorithm. Expert Syst, e12657
8. Singh T, Mishra KK (2020) Ranvijay A variant of eam to uncover community structure in complex networks. International Journal of Bio-Inspired Computation 16(2):102–110
9. Singh T, Saxena N, Khurana M, Singh D, Abdalla M, Alshazly H (2021) Data clustering using moth-flame optimization algorithm. Sensors 21(12):4086
10. Villalón CC, Stützle T, Dorigo M (2021) Cuckoo search≡ $(\mu + \lambda)$–evolution strategy
11. Villalón CLC, Stützle T, Dorigo M (2020) Grey wolf, firefly and bat algorithms: Three widespread algorithms that do not contain any novelty. In: International conference on swarm intelligence, pp 121–133. Springer
12. Sörensen K (2015) Metaheuristics—the metaphor exposed. Int Trans Oper Res 22(1):3–18
13. García-Martínez C, Gutiérrez PD, Molina D, Lozano M, Herrera F (2017) Since cec 2005 competition on real-parameter optimisation: a decade of research, progress and comparative analysis's weakness. Soft Comput 21(19):5573–5583
14. Dorigo M (2016) Swarm intelligence: a few things you need to know if you want to publish in this journal
15. Gao S, Zhou M, Wang Y, Cheng J, Yachi H, Wang J (2018) Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction. IEEE Transactions on Neural Networks and Learning Systems 30(2):601–614
16. Faris H, Aljarah I, Mirjalili S (2018) Improved monarch butterfly optimization for unconstrained global search and neural network training. Appl Intell 48(2):445–464
17. Wang G-G (2018) Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Memetic Computing 10(2):151–164
18. Yang Y, Chen H, Heidari AA, Gandomi AH (2021) Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. Expert Syst Appl 177:114864
19. Ahmadianfar I, Heidari AA, Gandomi AH, Chu X, Chen H (2021) Run beyond the metaphor: An efficient optimization algorithm based on runge kutta method. Expert Syst Appl 181: 115079
20. Tu J, Chen H, Wang M, Gandomi AH (2021) The colony predation algorithm. Journal of Bionic Engineering 18(3):674–710
21. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Futur Gener Comput Syst 97:849–872
22. Singh T (2020) A chaotic sequence-guided Harris Hawks optimizer for data clustering. Neural Comput and Applic 32:17789–17803
23. Singh T, Panda SS, Mohanty SR, Dwibedy A (2021) Opposition learning based harris hawks optimizer for data clustering. J Ambient Intell Humaniz Comput, 1–16
24. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: a new method for stochastic optimization. Future Generation Computer Systems
25. Singh T, Saxena N (2021) Chaotic sequence and opposition learning guided approach for data clustering. Pattern Anal Applic, 1–15
26. Pothiya S, Ngamroo I, Kongprawechnon W (2008) Application of multiple tabu search algorithm to solve dynamic economic dispatch considering generator constraints. Energy Convers Manag 49(4):506–516
27. Lin W-M, Cheng F-S, Tsay M-T (2002) An improved tabu search for economic dispatch with multiple minima. IEEE Transactions on Power Systems 17(1):108–112
28. Kamboj VK, Bhadoria A, Bath SK (2017) Solution of non-convex economic load dispatch problem for small-scale power systems using ant lion optimizer. Neural Comput and Applic 28(8):2181–2192
29. Neto JXV, Reynoso-Meza G, Ruppel TH, Mariani VC, dos Santos Coelho L (2017) Solving non-smooth economic dispatch by a new combination of continuous grasp algorithm and differential

evolution. International Journal of Electrical Power & Energy Systems 84:13–24
30. Jayabarathi T, Raghunathan T, Adarsh BR, Suganthan PonnuthuraiNagaratnam (2016) Economic dispatch using hybrid grey wolf optimizer. Energy 111:630–641
31. Pradhan M, Roy PK, Pal T (2017) Oppositional based grey wolf optimization algorithm for economic dispatch problem of power system. Ain Shams Engineering Journal
32. Elsakaan AA, El-Sehiemy RA, Kaddah SS, Elsaid MI (2018) An enhanced moth-flame optimizer for solving non-smooth economic dispatch problems with emissions. Energy 157:1063–1078
33. Mandal B, Roy PK, Mandal S (2014) Economic load dispatch using krill herd algorithm. International Journal of Electrical Power & Energy Systems 57:1–10
34. Sk MdAliBulbul, Pradhan M, Roy PK, Pal T (2018) Opposition-based krill herd algorithm applied to economic load dispatch problem. Ain Shams Engineering Journal 9(3):423–440
35. dos Santos Coelho L, Mariani VC (2009) An improved harmony search algorithm for power economic load dispatch. Energy Convers Manag 50(10):2522–2526
36. Al-Betar MA, Awadallah MA, Khader AT, Bolaji AL (2016) Tournament-based harmony search algorithm for non-convex economic load dispatch problem. Appl Soft Comput 47:449–459
37. Pothiya S, Ngamroo I, Kongprawechnon W (2010) Ant colony optimisation for economic dispatch problem with non-smooth cost functions. International Journal of Electrical Power & Energy Systems 32(5):478–487
38. Elsayed WT, Hegazy YG, Bendary FM, El-Bages MS (2016) Modified social spider algorithm for solving the economic dispatch problem. Engineering Science and Technology, an International Journal 19(4):1672–1681
39. Bhattacharya A, Chattopadhyay PK (2010) Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch. IEEE Transactions on Power Systems 25(4):1955–1964
40. Ravikumar Pandi V, Panigrahi BK (2011) Dynamic economic load dispatch using hybrid swarm intelligence based harmony search algorithm. Expert Syst Appl 38(7):8509–8514
41. Niknam T (2010) A new fuzzy adaptive hybrid particle swarm optimization algorithm for non-linear, non-smooth and non-convex economic dispatch problem. Appl Energy 87(1):327–339
42. Alsumait JS, Sykulski JK, Al-Othman AK (2010) A hybrid ga–ps–sqp method to solve power system valve-point economic dispatch problems. Appl Energy 87(5):1773–1781
43. Kumar R, Sharma D, Sadu A (2011) A hybrid multi-agent based particle swarm optimization algorithm for economic power dispatch. International Journal of Electrical Power & Energy Systems 33(1):115–123
44. Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell 48(10):3462–3481
45. Gao S, Yu Y, Wang Y, Wang J, Cheng J, Zhou M (2019) Chaotic local search-based differential evolution algorithms for optimization. IEEE Transactions on Systems, Man and Cybernetics: Systems
46. Yang L, Gao S, Yang H, Cai Z, Lei Z, Todo Y (2021) Adaptive chaotic spherical evolution algorithm. Memetic Computing 13(3):383–411
47. Xu Z, Yang H, Li J, Zhang X, Lu B, Gao S (2021) Comparative study on single and multiple chaotic maps incorporated grey wolf optimization algorithms. IEEE Access
48. Adarsh BR, Raghunathan T, Jayabarathi T, Yang Xin-She (2016) Economic dispatch using chaotic bat algorithm. Energy 96:666–675
49. Arul R, Ravi G, Velusami S (2013) Chaotic self-adaptive differential harmony search algorithm based dynamic economic dispatch. International Journal of Electrical Power & Energy Systems 50:85–96
50. Lu Y, Zhou J, Qin H, Wang Y, Zhang Y (2011) Chaotic differential evolution methods for dynamic economic dispatch with valve-point effects. Eng Appl Artif Intel 24(2):378–387
51. dos Santos Coelho L, Mariani VC (2009) A novel chaotic particle swarm optimization approach using hénon map and implicit filtering local search for economic load dispatch. Chaos, Solitons & Fractals 39(2):510–518
52. Yu J, Kim C-H, Wadood A, Khurshiad T, Rhee S-B (2018) A novel multi-population based chaotic jaya algorithm with application in solving economic load dispatch problems. Energies 11(8):1946
53. Zhao J, Liu S, Zhou M, Guo X, Qi L (2018) Modified cuckoo search algorithm to solve economic power dispatch optimization problems. IEEE/CAA Journal of Automatica Sinica 5(4):794–806
54. Mohammadi F, Abdi H (2018) A modified crow search algorithm (mcsa) for solving economic load dispatch problem. Appl Soft Comput 71:51–65
55. Al-Betar MA, Awadallah MA, Khader AT, Bolaji AL, Almomani A (2018) Economic load dispatch problems with valve-point loading using natural updated harmony search. Neural Comput and Applic 29(10):767–781
56. Kumar M, Dhillon JS (2018) Hybrid artificial algae algorithm for economic load dispatch. Appl Soft Comput 71:89–109
57. Prakash T, Singh VP, Singh SP, Mohanty SR (2018) Economic load dispatch problem: quasi-oppositional self-learning tlbo algorithm. Energy Systems 9(2):415–438
58. Hr Aghay Kaboli S, Alqallaf AK (2019) Solving non-convex economic load dispatch problem via artificial cooperative search algorithm. Expert Syst Appl 128:14–27
59. Trivedi IN, Jangir P, Bhoye M, Jangir N (2018) An economic load dispatch and multiple environmental dispatch problem solution with microgrids using interior search algorithm. Neural Comput Applic 30(7):2173–2189
60. Singh D, Dhillon JS (2019) Ameliorated grey wolf optimization for economic load dispatch problem. Energy 169:398–419
61. Srivastava A, Das DK (2020) A new aggrandized class topper optimization algorithm to solve economic load dispatch problem in a power system. IEEE Transactions on Cybernetics
62. Spea SR (2020) Solving practical economic load dispatch problem using crow search algorithm. International Journal of Electrical and Computer Engineering 10(4):3431
63. Sheta A, Faris H, Braik M, Mirjalili S (2020) Nature-inspired metaheuristics search algorithms for solving the economic load dispatch problem of power system: a comparison study. In: Applied nature-inspired computing: algorithms and case studies, pp 199–230. Springer
64. X Chang YXu, Sun H, Khan I (2021) A distributed robust optimization approach for the economic dispatch of flexible resources. International Journal of Electrical Power & Energy Systems 124:106360
65. Yu J, Kim C-H, Rhee S-B (2020) Clustering cuckoo search optimization for economic load dispatch problem. Neural Comput Applic 32:16951–16969
66. Sulaiman MH, Mustaffa Z, Rashid MIM, Daniyal H (2018) Economic dispatch solution using moth-flame optimization algorithm. In: MATEC web of conferences, vol 214, pp 03007. EDP Sciences
67. Kamboj VK, Bath SK, Dhillon JS (2016) Solution of non-convex economic load dispatch problem using grey wolf optimizer. Neural Comput and Applic 27(5):1301–1316

68. Bhattacharya A, Chattopadhyay PK (2010) Solving complex economic load dispatch problems using biogeography-based optimization. Expert Syst Appl 37(5):3605–3615
69. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
70. Saremi S, Mirjalili S, Lewis A (2017) Grasshopper optimisation algorithm: theory and application. Adv Eng Softw 105:30–47
71. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Applic 27(2):495–513
72. Gaing Z-L (2003) Particle swarm optimization to solving the economic dispatch considering the generator constraints. IEEE Transactions on Power Systems 18(3):1187–1195
73. Lee FN, Breipohl AM (1993) Reserve constrained economic dispatch with prohibited operating zones. IEEE Transactions on Power Systems 8(1):246–254
74. Kılıç U (2015) Backtracking search algorithm-based optimal power flow with valve point effect and prohibited zones. Electr Eng 97(2):101–110
75. Ott E (2002) Chaos in dynamical systems. Cambridge University Press
76. Sinha N, Chakrabarti R, Chattopadhyay PK (2003) Evolutionary programming techniques for economic load dispatch. IEEE Transactions on Evolutionary Computation 7(1):83–94
77. dos Santos Coelho L, Lee C-S (2008) Solving economic load dispatch problems in power systems using chaotic and gaussian particle swarm optimization approaches. International Journal of Electrical Power & Energy Systems 30(5):297–307
78. Park J-B, Jeong Y-W, Shin J-R, Lee KY (2009) An improved particle swarm optimization for nonconvex economic dispatch problems. IEEE Transactions on Power Systems 25(1):156–166
79. Sheskin DJ (2003) Handbook of parametric and nonparametric statistical procedures. Chapman and Hall/CRC
80. Inman RL, Davenpot JM (1980) Approximations of the critical region of the friedman statistic. Communications in Statistics, Theory and Methods A 9:571–595
81. Holm S (1979) A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics, 65–70

Tribhuvan Singh did his M.Tech. and Ph.D. from NIT Durgapur and MNNIT Allahabad, respectively. His area of interest includes machine learning, nature-inspired algorithms, multiobjective optimization, and data clustering. Currently, he is working as an Assistant Professor in the Department of Computer Science and Engineering, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar, Odisha-751030, India.