Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2014, Article ID 175417, 15 pages
http://dx.doi.org/10.1155/2014/175417

*Research Article*

# An Improved Differential Evolution Method Based on the Dynamic Search Strategy to Solve Dynamic Economic Dispatch Problem with Valve-Point Effects

**Guangyu Chen and Xiaoqun Ding**

*College of Energy and Electrical Engineering, Hohai University, Nanjing 210098, China*

Correspondence should be addressed to Guangyu Chen; cgyhhu@163.com

Received 16 March 2014; Revised 4 July 2014; Accepted 13 July 2014; Published 3 August 2014

Academic Editor: Victor Kovtunenko



An improved differential evolution (DE) method based on the dynamic search strategy (IDEBDSS) is proposed to solve dynamic economic dispatch problem with valve-point effects in this paper. The proposed method combines the DE algorithm with the dynamic search strategy, which improves the performance of the algorithm. DE is the main optimizer in the method proposed. While chaotic sequences are applied to obtain the dynamic parameter settings in DE, dynamic search strategy which consists of two steps, global search strategy and local search strategy, is used to improve algorithm efficiency. To accelerate convergence, a new infeasible solution handing method is adopted in the local search strategy; meanwhile, an orthogonal crossover (OX) operator is added to the global search strategy to enhance the optimization search ability. Finally, the feasibility and effectiveness of the proposed methods are demonstrated by three test systems, and the simulation results reveal that the IDEBDSS method can obtain better solutions with higher efficiency than the standard DE and other methods reported in the recent literature.

## 1. Introduction

The dynamic economic dispatch (DED) is very important optimization problems in the power system operation, which is a complicated nonlinear dynamic constrained problem [1], and its purpose is to find the optimal combination of power outputs of all generating units to minimize the total fuel cost and meanwhile satisfy all equality and inequality constraints during all the dispatch periods. Generally, the generating unit fuel cost function is represented approximately as a nonlinear convex quadratic function. In this approximation, the value-point effects of generator are not considered, so that the inaccuracy of the dispatch results is inevitable. However, in reality, convex fuel cost function cannot be exhibited by the generating units due to steam valves in large steam turbines, and the generator exhibits the characteristics of nonsmooth and nonconvex mathematically. From the perspective of math, the DED problem with valve-point effects can be categorized as a dynamic nonlinear optimization problem with nonsmooth and nonconvex characteristics subjected to various equality and inequality constraints. Therefore, it is a challenge to find the optimal dispatch result in a reasonable computation time.

Over the past decades, many traditional mathematic methods have been developed to solve the DED problem, including linear programming (LP) [2], quadratic programming (QP) [3], Lagrangian relaxation (LR) [4], and dynamic programming (DP) [5]. However, when applying these traditional methods to DED problem with valve-point effects, the global optimal solution can hardly be achieved due to their drawbacks. For example, large errors would be generated during the process of linearizing the DED model when using LP to solve DED problem. For QP, the objective function needs to be transformed for the reason that the objective function must be continuous and differentiable, which would bring inaccuracy to the final solution. Though DP can solve the DED problem without imposing any restrictions, it suffers from the "curse of dimensionality," when applied in large scale power systems; it may not converge in a possible time.

Recently, many modern heuristics stochastic search algorithms such as genetic algorithm (GA) [6], evolutionary programming (EP) [7], tabu search (TS) [8], particle swam

optimization (PSO) [9, 10], differential evolution algorithm (DE) [11–21], biogeography-based optimization (BBO) [22], chaotic self-adaptive differential harmony search algorithm (CSADHS) [23], quadratically constrained quadratic program method (QCQP) [24], Krill herd algorithm (KHA) [25], and harmony search with new pitch (NPAHS) [26] have shown great potentials in solving the nonlinear ELD or DED problems. Although these heuristics stochastic optimization methods mentioned do not guarantee the obtaining of the globally optimal solution in finite computation time, a satisfactory result can always be acquired. Unfortunately, for DED problems with value-point effects, these methods seem to lack the ability to find the global optimal solution and often get local optima owing to their drawbacks.

GA suffers from the premature convergence, and the encoding and decoding strategies adopted by GA causes it to take a long computation time to achieve convergence. For EP, though it can obtain a reasonable solution occasionally, the slow convergence may lead to a long computation time in DED problem. TS algorithm can escape from local optima and fast convergence to the global optimum; however, it may not reach the global optimum solution in a reasonable computational time when the initial solution is far away from the region where the global optimal solution exists. For PSO and DE methods, the premature convergence may trap the algorithm into the local optimum and reduce their optimization ability when solving DED problems. Recently, some hybrid methods are proposed to solve DED problems with nonsmooth fuel cost functions, including PSO-SQP [27], hybrid BBO-DE [28], hybrid EP-SQP [29], and BCO-SQP [30]. These hybrid methods utilize the features of different algorithms to keep balance between global search capability and local search capability and obtain good actual effects. However, these hybrid methods still have drawbacks for solving the DED problem, such as choosing suitable parameter values and slowing the convergence because of the large amount of iteration procedures.

Differential evolution algorithm (DE), first proposed by Storn and Price, is one of the best global optimization methods [31]. Compared with other evolutionary algorithms, DE is a simple yet efficient optimizer with fewer parameters. In recent years, DE has been applied successfully to solve optimization problems in various fields due to its high efficiency, such as economic dispatch optimization problem [11], transient stability constrained optimal power flow problem [32], network reconfiguration problem of distribution systems problem [33], optimal reactive power dispatch problem [34], and electromagnetic inverse scattering problems [35]. With the popularization of DE, it has drawn more and more attention of the scholars all over the world [36–39]. However, DE still has its drawbacks that need to be improved, such as how to select suitable parameter values for DE, how to avoid premature convergence for DE, and how to enhance the global search capability in searching the global optimal solution rapidly and efficiently. Furthermore, it does not consider the constraints of the complicated optimization problem in the canonical version of DE. In view of the defects of the standard DE, some improved DE algorithms are proposed to solve DED or ELD problems. In [11], a combined version of the DE algorithm with the generator of chaos sequences and sequential quadratic programming (SQP) technique is proposed to optimize the performance of economic dispatch problems of which the DE with chaos sequences is the global optimizer, and the SQP is used to fine-tune the DE run in a sequential manner. Duvvuru and Swarup [12] proposed a novel hybrid algorithm that integrated interior point method (IPM) and differential evolution (DE) for solving economic load dispatch (ELD) problem with valve-point effect. Firstly, interior point method is used to solve the problem without valve-point loading, and the obtained solution is called $X$. Then, an initial population around $X$ was generated using a new strategy. Finally, the ELD problem with valve-point loading could be solved using DE. He et al. [13] combined the GA algorithm with the deferential evolution (DE) and sequential quadratic programming (SQP) technique to improve the performance of the algorithm, of which GA is the main optimizer, while the DE and SQP are used to adjust the solution of the GA running. In [18], a modified differential evolution approach (MDE) is proposed to solve the DED problem, and its most important contribution is to handle constraints effectively by devising feasibility-based selection comparison techniques and heuristic search rules. Although some improved DE algorithm has already been proposed to deal with the constraints of the complicated optimization problem in DED problem, it still needs further study on how to improve the overall implementation efficiency of combining DE algorithm and constraint handling.

Therefore, to overcome the defects of the mentioned above, an improved differential evolution method based on the dynamic search strategy is proposed to solve DED problem in this paper. The research of this paper mainly focuses on the following three issues. Firstly, it is difficult to select suitable parameter values in the traditional DE methods, so that a dynamic parameter control mechanism based on chaotic sequences is applied to determine the parameter settings adaptively by virtue of the randomness and regularity of the chaos mechanism. Secondly, it is very difficult to handle the constraints in DED. In this paper, multiobjective concepts without penalty function method are used to handle the complex constraints of DED problem, and DED problem is converted into a biobjective optimization problem. Meanwhile, Pareto dominance which is usually used in multiobjective optimization is adopted to compare the individuals in the population. In order to improve the efficiency of constraint handling during the process of evolution, an effective rough preadjustment method is proposed to handle constraint violation of the infeasible individuals. Notably, violation of equality constraint can be handled by allocating constraint violations to units according to their regulatory abilities, which is different from handling constraint violation by selecting generators randomly [26, 40]. By using the proposed method, regulation of each unit becomes more reasonable, and the solving efficiency is also improved. Thirdly, in order to improve algorithm performance in solving DED problem, a new dynamic searching strategy including global search strategy and local search strategy is proposed to solve DED problem in this paper, which is different from dynamic search strategy in [41].

The main improvement is reflected in the following three aspects. (1) DE/rand/1 strategy is replaced by DE/best/1 in local search strategy. The DE/best/1 strategy can make the population search toward the best feasible solution and guide the population into the feasible region quickly. (2) Constraint pretreatment (Section 4.5) is used to handle infeasible individuals of the population in local search strategy in order to reduce the number of infeasible solutions. (3) Quantization orthogonal crossover (QOX) operator is introduced into global search strategy. QOX operator can search the better solution with less computation time, so that it can enhance global search ability. At the same time, considering that infeasible solutions may be obtained by QOX operator. The information of infeasible solutions is utilized reasonably, which can provide important help to search the globally optimal solution, especially when the optimal solution is in the feasible region boundary. Finally, the proposed method is implemented to solve the DED problem by three test systems, and the feasibility and efficiency of the proposed algorithm are shown by simulation results. Compared with other optimization methods reported in the literature, the proposed IDEBDSS method can obtain better solutions in a shorter computation time along with higher effectiveness and robustness.

This paper is organized as follows. DED problem formulation is introduced in Section 2. Since DE is used as the search algorithm in this paper, it is briefly introduced in Section 3. In Section 4, IDEBDSS algorithm is presented for solving DED problem in detail. In Section 5, the effectiveness of the proposed algorithm is verified through a numerical example. Finally, some conclusions are given in Section 6.

## 2. Formulation of DED Problem

*2.1. Objective Function.* The DED problem is nonconvex and nondifferentiable considering valve-point effects [42]. The objective of the classic DED problem is to minimize the total fuel cost function associated with the $N$ generating units for $T$ intervals in the given dispatch horizon as follows [43]:

$$F = \min \sum_{t=1}^{T} \sum_{i=1}^{N} f_i\left(p_i^t\right), \tag{1}$$

where $F$ is the total fuel cost over the whole dispatch periods, $T$ is the number of intervals over the dispatch horizon, $N$ is the number of generating units, $p_i^t$ is the power output of the $i$th unit at the $t$th dispatch interval, and $f_i(p_i^t)$ is the fuel cost of $i$th unit at the output of $p_i^t$. Traditionally, the fuel cost function of each unit can be described as a quadratic function as shown in

$$f_i\left(p_i\right) = a_i + b_i p_i + c_i p_i^2, \tag{2}$$

where $a_i$, $b_i$, and $c_i$ are cost coefficients of the $i$th unit.

However, in reality, when steam admission valve starts to open, a sharp increase in fuel loss would be added to the fuel cost curve due to the wire drawing effects, which is named as valve-point effects. In order to model the DED problem with the consideration of valve-point effects more accurately, the objective function of the problem is described as a superposition of quadratic and sinusoidal functions, and meanwhile a set of nonsmooth cost functions are imported into DED problem. The cost function with valve-point effects can be represented as follows [44]:

$$f_i\left(p_i\right) = a_i + b_i p_i + c_i p_i^2 + \left|e_i \times \sin\left(h_i \times \left(p_{i,\min} - p_i\right)\right)\right|, \tag{3}$$

where $e_i$ and $h_i$ are cost coefficients of generator $i$.

*2.2. Constraints*

*(1) Real Power Balance Constraint.* Consider

$$\sum_{i=1}^{N} p_i^t = p_D^t + p_L^t, \tag{4}$$

where $p_D^t$ is the total load demand at $t$ interval. $p_L^t$ is the transmission loss. System loss is a function of unit power production which can be calculated using the results of load flow problem [42] or Kron's loss formula known as $B$-matrix coefficients [45]. In this work $B$-matrix coefficients method is used to calculate system loss as follows:

$$p_L^t = \sum_{i=1}^{N} \sum_{j=1}^{N} p_i^t B_{i,j} p_j^t + \sum_{j=1}^{N} B_{0i} p_i^t + B_{00}, \tag{5}$$

where $B_{i,j}$ is the $i,j$th element of the loss coefficient square matrix, $B_{0i}$ is the $i$th element of the loss coefficient vector, and $B_{00}$ is the loss coefficient constant.

*(2) Power Operating Limits.* Consider

$$p_{i,\min} \le p_i^t \le p_{i,\max}, \quad i = 1, 2, \ldots, N, \ t = 1, 2, \ldots, T, \tag{6}$$

where $p_{i,\min}$ and $p_{i,\max}$ are the minimum and maximum outputs of $i$th generator, respectively.

*(3) Generating Unit Ramp Rate Limits.* The output power change rate of the thermal unit must be in an acceptable range to avoid undue stress on the boiler and combustion equipments [42]. The ramp rate limits of generation units are stated as follows:

$$\begin{aligned} &p_i^t - p_i^{t-1} \le \mathrm{UR}_i, \\ &p_i^{t-1} - p_i^t \le \mathrm{DR}_i, \\ &i = 1, 2, \ldots, N, \quad t = 1, 2, \ldots, T, \end{aligned} \tag{7}$$

where $\mathrm{UR}_i$ and $\mathrm{DR}_i$ are the up-ramp and down-ramp limits of the $i$th generator, respectively.

## 3. Overview of Differential Evolution Algorithm

Differential evolution algorithm (DE) is a simple and powerful population-based stochastic optimization algorithm [31]. During the evolution, DE implements mutation, crossover,

and selection operations to update the population. The initial population of DE is randomly generated within the decision space, which consists of NP (NP is population size) $n$-dimensional vector, namely, $\overrightarrow{X_{i,g}} = (X_{i,1,g}, X_{i,2,g}, \ldots, X_{i,n,g})$, $i = 1, 2, \ldots, \mathrm{NP}$, where $g$ denotes the current generation number. The key idea behind DE is to make use of mutation and crossover operations to yield a trial vector $\overrightarrow{u_{i,g}}$ for each target vector $\overrightarrow{x_{i,g}}$. Afterwards, a selection operation is executed between the trial vector $\overrightarrow{u_{i,g}}$ and the target vector $\overrightarrow{x_{i,g}}$ to get better generation individuals. The mutation, crossover, and selection operations of this DE algorithm are explained as follows.

*Mutation Operation.* A mutant individual $\overrightarrow{v_{i,g}} = (v_{i,1,g}, v_{i,2,g}, \ldots, v_{i,n,g})$ is generated by a mutation operator. The following two mutation strategies are frequently used in many literatures [46]; that is,

(1) DE/rand/1: $\overrightarrow{v_{i,g}} = \overrightarrow{x_{r_1,g}} + F \cdot (\overrightarrow{x_{r_2,g}} - \overrightarrow{x_{r_3,g}})$;

(2) DE/rand/1: $\overrightarrow{v_{i,g}} = \overrightarrow{x_{\mathrm{best},g}} + F \cdot (\overrightarrow{x_{r_2,g}} - \overrightarrow{x_{r_3,g}})$,

where $r_1$, $r_2$, and $r_3$ are generated randomly in the range of $[1, \mathrm{NP}]$ and satisfy $r_1 \neq r_2 \neq r_3$, $\overrightarrow{x_{\mathrm{best},g}}$ is the best individual in the population at generation $g$, and $F$ is a control parameter, often called as scaling factor. The control parameter $F \in [0, 1]$ is a real constant parameter supplied by users, which controls the amplification of the differential variation.

*Crossover Operation*. Crossover operation is applied to increase the diversity of the population. After the mutation operation, the trial vector $\overrightarrow{u_{i,g}}$ is generated by a binomial crossover operation on the target vector $\overrightarrow{x_{i,g}}$ and the mutant vector $\overrightarrow{v_{i,g}}$ using the following scheme:

$$u_{ij,g} = \begin{cases} v_{ij,g}, & \text{if } (\mathrm{Rand}(j) \leq \mathrm{CR}) \text{ or } j = j_{\mathrm{rand}}, \\ x_{ij,g}, & \text{otherwise}, \end{cases} \quad (8)$$
$$i = 1, 2, \ldots, \mathrm{NP}; \quad j = 1, 2, \ldots, D,$$

where $j_{\mathrm{rand}}$ is a randomly chosen integer from $[1, D]$, which ensures that $\overrightarrow{u_{i,g}}$ gets at least one element from $\overrightarrow{v_{i,g}}$. Otherwise, the population may not evolve for there is no new generated solution. Rand $(j)$ is the $j$th evaluation of a uniform random number generator between 0 and 1 and $\mathrm{CR} \in [0, 1]$ is a crossover control parameter called as crossover rate, which is the user-defined crossover constant that controls the recombination.

*Selection Operation*. Selection operation is implemented by comparing the target vector $\overrightarrow{x_{i,g}}$ against the trial vector $\overrightarrow{u_{i,g}}$. According to the value of fitness, the better one will be selected to participate in the next generation. The selection operation can be expressed as follows:

$$x_{i,g+1} = \begin{cases} u_{i,g}, & \text{if } f(u_{i,g}) < f(x_{i,g}), \\ x_{i,g}, & \text{else}, \end{cases} \quad (9)$$

where $f(x_{i,g})$ is the fitness value of the target vector $\overrightarrow{x_{i,g}}$ and $f(u_{i,g})$ is the fitness value of the trial vector $\overrightarrow{u_{i,g}}$.

## 4. Implementation of IDEBDSS Method for Dynamic Economic Dispatch with Value-Points Effects

*4.1. Chaotic Sequences for Adjusting the Parameter Value Settings of DE Adaptively.* The performance of DE is significantly influenced by the value settings of control parameters $F$ (scale factor) and CR (crossover rate). Proper value settings are important for the successful application of the DE algorithm. Recently, some successful applications of an evolutionary algorithm (EA) combined with chaotic sequences have been reported in optimization problems [11, 55]. Due to the randomness, ergodicity, and regularity of the chaos mechanism, chaotic sequences applied in an EA can increase the exploitation capability of the algorithm in the search space and enhance its convergence property. Therefore, in this paper, a dynamic parameter control mechanism based on chaotic sequences is applied for adjusting the parameter value settings of DE adaptively during the searching progress. The Logistic map used in this paper is described as follows:

$$\beta^{k+1} = u\beta^k\left(1 - \beta^k\right), \quad k = 1, 2, \ldots,$$
$$\beta \in (0, 1), \quad \beta \neq 0.25, 0.5, 0.75, \quad (10)$$

where $k$ is the iterative number, $\beta^k$ is a stochastic number between 0 and 1, and $u = 4$. From (10), it can be known that, during the iterations of the Logistic map, the value of $\beta^k$ will distribute between 0 and 1 when the initial $\beta^k \in (0, 1)$ and $\beta \neq 0.25, 0.5, 0.75$.

The value setting of parameter $F$ of DE is adjusted dynamically as follows:

$$F^0 \in (0, 1), \quad F^0 \notin \{0.25, 0.5, 0.75\},$$
$$F^{g+1} = 4F^g\left(1 - F^g\right), \quad g = 1, 2, \ldots, g_{\max}, \quad (11)$$

where $g_{\max}$ is the maximum iteration number.

Similarly, the parameter CR can be updated adaptively by

$$\mathrm{CR}^0 \in (0, 1), \quad \mathrm{CR}^0 \notin \{0.25, 0.5, 0.75\},$$
$$\mathrm{CR}^{g+1} = 4\mathrm{CR}^g\left(1 - \mathrm{CR}^g\right), \quad g = 1, 2, \ldots, g_{\max}. \quad (12)$$

*4.2. Orthogonal Crossover.* In a discrete single objective optimization problem, when there are $K$ factors with each factor having $Q$ levels, the search space consists of $Q^K$ combinations of levels. To find the best level for each factor, it is generally inevitable to do one experiment for every combination of factor levels. If $K$ and $Q$ are very big, it will take a long time to do all experiments. In this case, experimental design methods can be used for sampling a small number of well representative combinations for testing.

Orthogonal design is regarded as a very popular experimental design tool. In orthogonal design, a series of orthogonal arrays with different numbers of factors and different levels can be provided, and all columns in orthogonal array can be evaluated independently without considering the influence of one another (a number of such arrays can be found in http://www.york.ac.uk/depts/maths/tables/orthogonal.htm).

An orthogonal array for $N$ factors with $Q$ levels and $M$ combinations is often denoted by $L_M(Q^N)$. Quantization orthogonal crossover (QOX) proposed in [56] has been successfully applied into differential evolution [57] and genetic algorithm [56] to improve the exploration ability. In this paper, QOX is imported into the proposed algorithm to enhance the global search ability during evolution procedure.

*4.3. Structure of Individuals.* For an individual $P$, which consists of $N$ generating units and $T$ intervals, the array of control variable vector can be described as

$$P = \begin{bmatrix} p_1^1 & p_2^1 & \cdots & P_N^1 \\ p_1^2 & p_2^2 & \cdots & P_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ p_1^T & p_2^T & \cdots & P_N^T \end{bmatrix}, \tag{13}$$

where $p_i^t$ is the output of the $i$th generating unit at the $t$th dispatch interval, satisfying the real power output limit constraints as is shown by (6).

*4.4. Initialization of the Population.* In this paper, the power outputs are initialized randomly in the feasible range, which satisfy the constraints given by (6). The initial power outputs can be generated randomly by

$$\begin{aligned} p_i^t &= p_{i,\min} + \left(p_{i,\max} - p_{i,\min}\right) \times \text{Rand}\,(0,1), \\ &\qquad i = 1,2,\ldots,N, \quad t = 1,2,\ldots,T, \end{aligned} \tag{14}$$

where Rand $(0,1)$ is a randomly generated number between 0 and 1, which obeys uniform distribution, and $p_{i,\max}$ and $p_{i,\min}$ are the maximum and minimum outputs of generating unit $i$, respectively.

*4.5. Constraint Handling Method.* The traditional constraint handling strategy generally applies a penalty function to convert a constrained problem into an unconstraint one. Although, this strategy is very convenient to handle the constraints for evolutionary algorithm, it is difficult to find suitable penalty factors. In order to overcome the drawbacks of penalty method, constrained optimization evolutionary algorithms (COEAs) based on multiobjective concepts have been gradually developed in recent years [58, 59]. In this paper, DED problem is converted into a biobjective optimization problem $F(x) = (f(x), G(x))$ by treating $G(x)$ as an additional objective ($f(x)$ is to minimize the original objective function for DED problem and $G(x)$ is the degree of all constraint violation for DED). As the original problem has been converted into a biobjective problem (i.e., multiobjective problem), Pareto dominance usually used in multiobjective optimization is adopted to compare the individuals in the population. Let $x_1$ and $x_2$ be two individuals in the population. If

(1) $f(x_1) \le f(x_2) \wedge G(x_1) \le G(x_2)$ and
(2) $f(x_1) < f(x_2) \vee G(x_1) < G(x_2)$,

then $x_1$ Pareto dominates $x_2$ (denoted as $x_1 \prec x_2$). $x_1$ and $x_2$ are considered nondominated with each other if they cannot Pareto dominate each other. If there is no other $x^*$ satisfying $x^* \prec x$ in the population, $x$ is called a nondominated individual in the population.

Above all, it is important to deal with $G(x)$. If the $G(x)$ of infeasible individuals during the evolution is not properly dealt with, it may produce more infeasible individuals in the population and thus the $G(x)$ values of these individuals tend to be large. Hence, more iteration is needed to handle these constraint violations in the evolution. Consequently, in every evolution in this paper, in order to increase the number of feasible individuals as well as cut down the constraint violation of infeasible individuals in the present population, a rough preadjustment is made on the constraint violation of the infeasible individuals firstly in the case that there are not too many infeasible individuals. Notably, this preadjustment can help to improve the efficiency at the same time. This paper comes up with a new constraint handing approach in which constraint violation of infeasible individual is allocated proportionally to each unit according to their regulatory abilities. The steps are as follows.

*Step 1.* Set current interval $t = 1$, where $t$ denotes the dispatch interval index.

*Step 2.* Calculate the feasible horizon of each unit at current interval by

$$\begin{aligned} P_{i,\min}^t &= \begin{cases} P_{i,\min}, & \text{if } t = 1, \\ \max\left(P_i^{t-1} - \text{DR}_i, P_{i,\min}^t\right), & \text{others}, \end{cases} \\ P_{i,\max}^t &= \begin{cases} P_{i,\max}, & \text{if } t = 1, \\ \min\left(P_i^{t-1} + \text{UR}_i, P_{i,\max}^t\right), & \text{others}, \end{cases} \\ &\qquad i = 1,2,\ldots,N, \quad t = 1,2,\ldots,T, \end{aligned} \tag{15}$$

where $P_{i,\min}^t$ and $P_{i,\min}^t$ are the up and down limits of the $i$th unit during the $t$th dispatch interval. Then, the outputs of each unit at current interval are adjusted to feasible horizon by

$$\begin{aligned} &P_{i,\min}^t \quad \text{if } P_i^t < P_{i,\min}^t, \\ &P_i^t \quad\;\; \text{if } P_{i,\min}^t < P_i^t < P_{i,\max}^t, \\ &P_{i,\max}^t \quad \text{if } P_i^t > P_{i,\max}^t, \\ &\qquad i = 1,2,\ldots,N, \quad t = 1,2,\ldots,T. \end{aligned} \tag{16}$$

*Step 3.* Calculate $\Delta P_{i-\min}^t$ and $\Delta P_{\max-i}^t$ by

$$\begin{aligned} \Delta P_{i-\min}^t &= P_i^t - P_{i,\min}^t \\ \Delta P_{\max-i}^t &= P_{i,\max}^t - P_i^t, \end{aligned} \qquad i = 1,2,\ldots,N, \quad t = 1,2,\ldots,T, \tag{17}$$

where $\Delta P_{i-\min}^t$ and $\Delta P_{\max-i}^t$, respectively, denote distance between $P_i^t$ and up and down limits of the $i$th unit at current interval and $N$ denotes the number of units.

*Step 4.* Calculate the amount of load balance constraint violation at the $t$th dispatch interval $P^t_{\text{voil}}$ according to (18). Set $l = 0$, where $l$ denotes the iteration number of the preadjustment operation. Consider

$$P^t_{\text{voil}} = \sum_{i=1}^{N} P^t_i - P^t_D - P^t_L. \tag{18}$$

*Step 5.* If $|P^t_{\text{voil}}| < P_{\text{Viol}}$, where $P_{\text{Viol}}$ is defined by user, then go to Step 10; otherwise, if $P^t_{\text{voil}} > 0$, then go to Step 6; if $P^t_{\text{voil}} < 0$, then go to Step 7.

*Step 6.* When $P^t_{\text{voil}} > 0$, the outputs of unit would decrease, and $\Delta p^t_{i-\min} = 0$ should not be taken into account at this time. Calculate $P^t_{i-\text{coef}-\min}$ value of each unit by (19), where $P^t_{i-\text{coef}-\min}$ denotes allocation coefficient of each unit at the $t$th dispatch interval. Then, calculate $\Delta P^t_{i-\text{app}}$ value of each unit by (20), where $\Delta P^t_{i-\text{app}}$ denotes allocation value of each unit at the $t$th dispatch interval. Afterwards, modify the power outputs of all units at the $t$th dispatch interval by (21) and go to Step 8. Consider

$$P^t_{i-\text{coef}-\min} = \frac{\Delta P^t_{i-\min}}{\left(\sum_{i=1}^{N} \Delta P^t_{i-\min}\right)}, \quad \left(\Delta P^t_{i-\min} \neq 0\right), \tag{19}$$

$$\Delta P^t_{i-\text{app}} = P^t_{i-\text{coef}-\min} \times P^t_{\text{voil}}, \tag{20}$$

$$P^t_i = P^t_i - \Delta P^t_{i-\text{app}}. \tag{21}$$

*Step 7.* When $P^t_{\text{voil}} < 0$, the outputs of unit would increase, and $\Delta P^t_{\max-i} = 0$ should not be taken into account at this time. Calculate $P^t_{i-\text{coef}-\max}$ value and $\Delta P^t_{i-\text{app}}$ value of each unit by (22) and (23), respectively. Afterwards, modify the power outputs of all units at the $t$th dispatch interval by (24), and then go to Step 8. Consider

$$P^t_{i-\text{coef}-\max} = \frac{\Delta P^t_{\max-i}}{\left(\sum_{i=1}^{N} \Delta P^t_{\max-i}\right)}, \quad \left(\Delta P^t_{\max-i} \neq 0\right), \tag{22}$$

$$\Delta P^t_{i-\text{app}} = P^t_{i-\text{coef}-\max} \times P^t_{\text{voil}}, \tag{23}$$

$$P^t_i = P^t_i + \Delta P^t_{i-\text{app}}. \tag{24}$$

*Step 8.* If the modified $P^t_i$ does not violate the power output limit constraints given by (6), then go to Step 9; otherwise, modify the value of $P^t_i$ by (16) and then go to Step 9.

*Step 9.* Calculate $P^t_{\text{voil}}$ by (18), and then set $l = l + 1$. If $l < l_{\max}$, where $l_{\max}$ is the maximum iteration number of the preadjustment operation, then go to Step 5; otherwise, go to Step 10.

*Step 10.* If $t$ is not the last interval, set $t = t + 1$, and then return to Step 2 or else go to Step 11.

*Step 11.* Sum the violation value $G_{\text{all}}(x)$ of this individual by

$$G_{\text{all}}(x) = \sum_{t=1}^{T} P^t_{\text{voil}}, \quad t = 1, \ldots, 24. \tag{25}$$

*Step 12.* The process is terminated.

*4.6. Dynamic Search Strategy Considering Constraints Violation.* By virtue of dynamic hybrid framework (DHF) [41], a dynamic search strategy considering constraint violation is proposed to solve DED optimization problem in this paper. QOX is introduced to enhance the capability of searching the optimal solution in global search strategy, while in local search strategy, constraint handling method (refer to Section 4.5) and the DE/best/1 mutation strategy are combined to impel the population to enter the feasible region promptly, thus accelerating convergence.

*4.6.1. Global Search Strategy.* During the evolution, the primary advantage of the global search strategy is that it can discover more promising regions. DE is the main searching algorithm in this paper. In order to further improve the global search capability of DE, a combination of DE and orthogonal crossover method [57] is applied to produce more trial vectors in the later stage of evolution. Here, only the best feasible solutions in population are selected to implement QOX operator during every evolution. Take the best individual of the set $M$, which is produced by QOX operator, as trial vectors to compare with target vector. If there are infeasible solutions in set $M$, some infeasible solutions should be utilized, and the best individual (i.e., the infeasible solution with the lowest degree of constraint violation, denoted as $M^{\text{best}}_{\text{infeasible}}$) from the nondominated individuals of set $M$ will be compared with other infeasible solutions in population. Figure 1 shows infeasible solutions obtained by QOX in the two-dimensional search space. From Figure 1, no matter if the mutant vector is feasible solution or infeasible solution (the triangle points represent the trial vectors obtained by QOX), infeasible solution may exist in trial vectors obtained by QOX. The procedure of the proposed method is shown in Algorithm 1.

*4.6.2. Local Search Strategy.* In the local search, both the addition of feasible individuals and the quick entry into the feasible region are necessary due to the small number of feasible individuals in the population. Firstly, the constraint handling method (refer to Section 4.5) is used for constraint pretreatment on the infeasible individual of the population, which increases the number of feasible individuals and reduces the constraint violation of those infeasible ones. Secondly, DE/best/1, one of the classic versions of DE, is considered in local search strategy, and it usually has fast convergence speed because the DE/best/1 strategy mainly relies on the best individual in the population. The procedure of the proposed method is described as follows.

*Step 1.* For infeasible individual, the constraint handling method is adopted to carry out constraint pretreatment.

*Step 2.* Each target vector (i.e., $\overrightarrow{x_{i,G}}$) in the population NP is used to produce a mutation vector (i.e., $\overrightarrow{v_{i,G}}$) by the DE/best/1 mutation operation.

**Parameters**:
$\overrightarrow{x_{i,G}}$: Target vector; $\overrightarrow{v_{i,G}}$: Mutant vector; $\overrightarrow{u_{i,G}}$: Trial vector; $x_i$: $i$th individual in population;
$P_{\text{infeasible}}$: The set of all the infeasible solution of current population;
***Step 1.*** Choose the smallest of $f(x)$ value forms the feasible solutions, denoted as $P_{\text{best}}$
***Step 2.*** For $i = 1, \ldots, \text{NP}$, do
*Step 2.1.* Generate a $\overrightarrow{v_{i,G}}$ by DE/rand/1
*Step 2.2.* If $(x_i = P_{\text{best}})$
*Step 2.2.1.* Mix $\overrightarrow{v_{i,G}}$ and $\overrightarrow{x_{i,G}}$ by making use of QOX based on $L_M(Q^k)$ to generate $M$ trial vectors. Calculation $f(x)$ values and $G(x)$ values of $M$ trial vectors
*Step 2.2.2.* Choose the one feasible solutions with the smallest of $f(x)$ value as $\overrightarrow{u_{i,G}}$ in $M$ trial vectors.
*Step 2.2.3.* If there are infeasible solutions in $M$ trial vectors, it will be added into set $T$
Else
*Step 2.2.4.* Mix $\overrightarrow{v_{i,G}}$ and $\overrightarrow{x_{i,G}}$ by (8) to generate $\overrightarrow{u_{i,G}}$.
End if
*Step 2.3.* Calculate the $f(x)$ value and the $G(x)$ value for the trial vector $\overrightarrow{u_{i,G}}$.
*Step 2.4.* If $\overrightarrow{u_{i,G}} \prec \overrightarrow{x_{i,G}}$, $\overrightarrow{u_{i,G}}$ will replace the $\overrightarrow{x_{i,G}}$, else no replacement occurs.
***Step 3.*** If $T$ are not empty set. Select the best infeasible individual (i.e., the infeasible solution with the lowest degree of constraint violation) of the non-dominated individuals in the set $T$, denoted as $M^{\text{best}}_{\text{infeasible}}$.
***Step 4.*** If $M^{\text{best}}_{\text{infeasible}}$ exists. Compared with pareto dominance between $M^{\text{best}}_{\text{infeasible}}$ and $P_{\text{infeasible}}$, $M^{\text{best}}_{\text{infeasible}}$ is used to replace a randomly selected pareto dominated individual in $P_{\text{infeasible}}$.

Algorithm 1: Procedure of global search strategy.

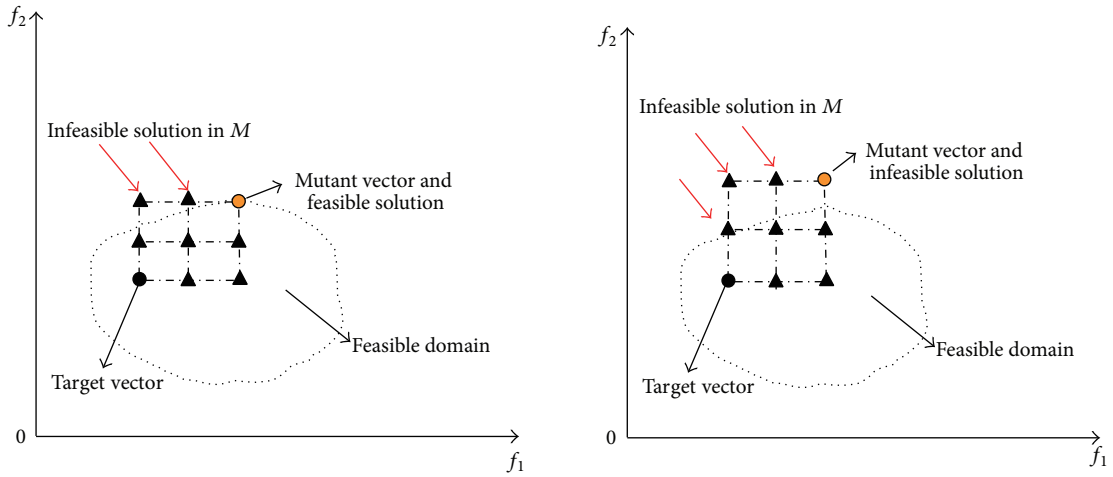


Figure 1: Infeasible solutions obtained by QOX in the two-dimensional search space.

*Step 3.* Crossover operation by (8) is used to produce a trial vector $\overrightarrow{u_{i,G}}$.

*Step 4.* $f(x)$ and $G(x)$ for the trial vector $\overrightarrow{u_{i,G}}$ are computed.

*Step 5.* If $\overrightarrow{u_{i,G}} \prec \overrightarrow{x_{i,G}}$, the $\overrightarrow{x_{i,G}}$ is replaced by the $\overrightarrow{u_{i,G}}$ or else no replacement occurs.

*4.7. Procedure of IDEBDSS for DED Problems.* To sum up, the flow chart of the proposed IDEBDSS can be illustrated as shown by Figure 2.

## 5. Case Study

*5.1. Description of the Test Systems.* In order to verify the feasibility and effectiveness of the proposed IDEBDSS algorithm for practical application that involves nonsmooth valve-point effects, three test systems are designed for solving DED problem.

*Test System 1.* This test system is a ten-unit system which considers nonsmooth valve-point effects but neglects the transmission losses.

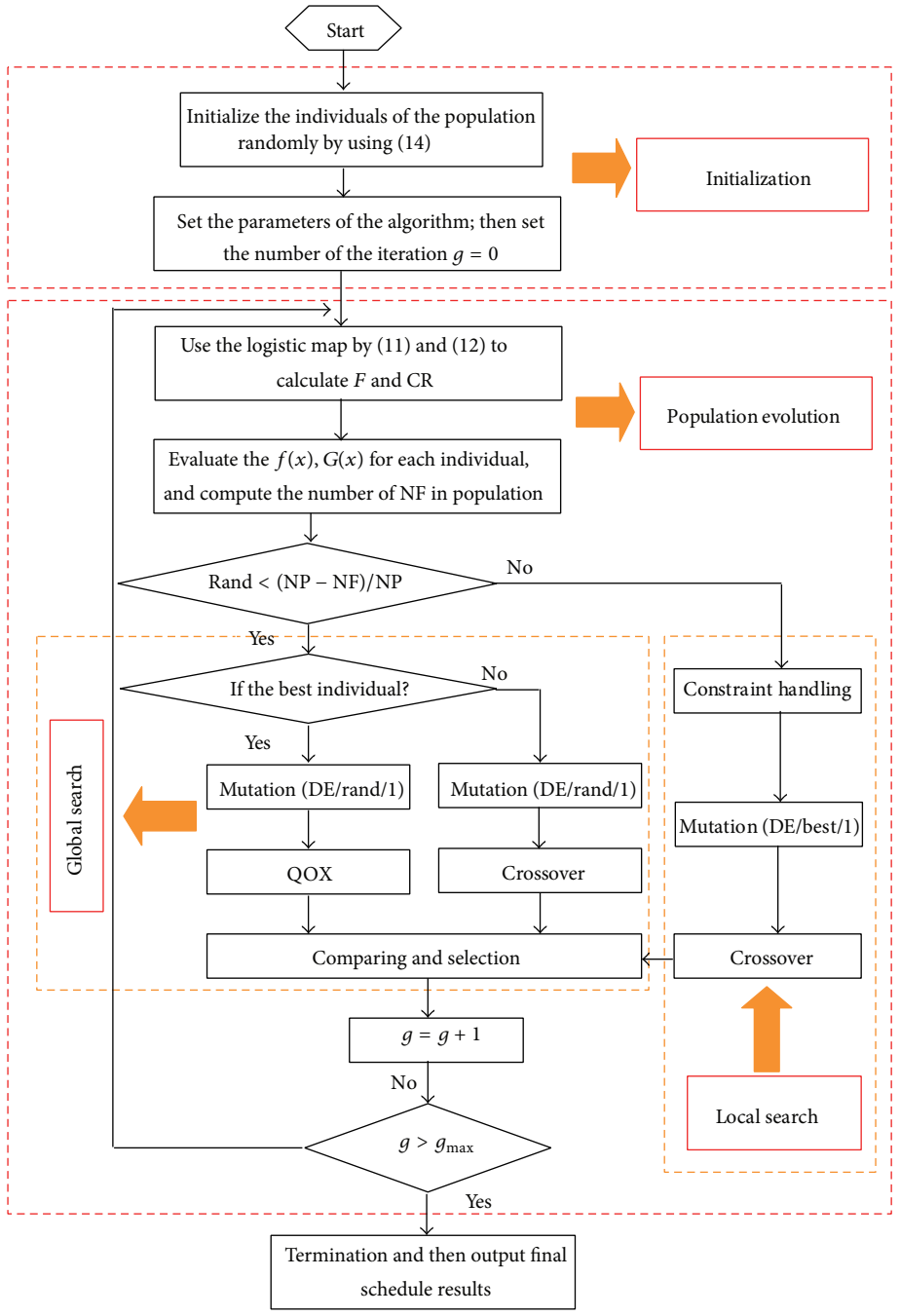


Figure 2: The flow chart of IDEBDSS.

*Test System 2.* The DED problem of the 10-unit system discussed in system 1 is solved with the consideration of transmission losses.

*Test System 3.* 30 units are included in this large-scale power system, which is tripled to test system 1.

For all the three test systems, the dispatch horizon $T$ is set as one day which is divided into 24 intervals. The technical data of the units, as well as the demand for the load, are taken from [52].

*5.2. Parameter Settings for Simulation.* The proposed IDEBDSS method is coded with C++ programming language and executed in P-IV 2.2 GHz personal computer to solve DED problems mentioned above. To verify modification work effectiveness, the simulation is executed for 40 times from different initial populations (every population consists of 70 solutions), and the best dispatch result among these 40 independent simulations is selected as the final optimization solution. Meanwhile, in order to verify effectiveness of the proposed algorithm for solving DED problems, both

Table 1: Parameter settings for IDEBDSS and DE.

| Method | $F$ | CR | $G_{\max}$ | NP | $L_M(Q^N)$ | $l_{\max}$ | $P_{\text{Viol}}$ |
|---|---|---|---|---|---|---|---|
| DE | 0.25 | 0.45 | 600 | 70 | — | — | — |
| IDEBDSS | — | — | 600 | 70 | $L_9(3^4)$ | 3 | 30 |

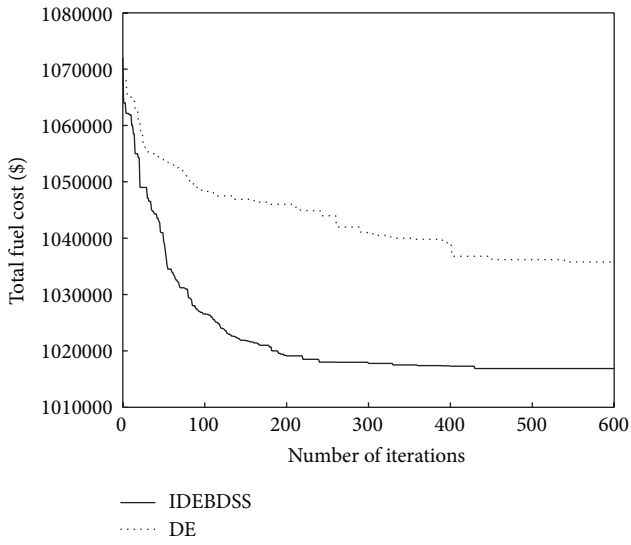


Figure 3: Convergence process of the best solution obtained by IDEBDSS and DE methods for test system 1.

IDEBDSS and standard DE are implemented to solve the same problems. The parameters of these two algorithms are listed in Table 1.

### 5.3. Computation Results and Comparison

*5.3.1. Test System 1.* According to the dispatch results and the convergence performance of the proposed IDEBDSS method and other methods, a comparison study is implemented to solve the DED problem of this test system. Table 2 gives the comparison of the best total fuel cost, average total fuel cost, average CPU execution time, and the number of independent simulations to obtain the dispatch results by IDEBDSS method and other methods. Convergence process of the best total fuel cost obtained by IDEBDSS method compared and standard DE method are shown in Figure 3. The details of the best dispatch result obtained by the proposed IDEBDSS method are provided in Table 3, which also demonstrates whether the constraints of the problem are satisfied or not.

In order to have a relatively fair comparison of the computation effort, the CPU times obtained from different methods are converted into a common base by the CPU chip frequency [45]. Notably, the total CPU time of the different algorithms is estimated based on the CPU speed of 2.2 GHz as they are all executed using a Pentium IV-2.2 GHz CPU personal computer. The equalized CPU time of an algorithm can be computed by (26). By means of the equalized transformation in computation time, it would be more meaningful to compare computational effort of different methods. The results are shown in column 6 of Table 2. Consider

$$\text{equalized CPU tims (s)} = C_{\text{eq}} \times \text{actual CPU time (s)},$$
$$C_{\text{eq}} = \frac{\text{given CPU Speed (GHz)}}{2.2\,\text{GHz}}. \tag{26}$$

From the dispatch results in Table 2, it is clear that the proposed IDEBDSS method can provide the best dispatch result compared with other methods. The total fuel cost of the best dispatch result for test system 1 in all dispatch periods achieves 1016873$, which is the best dispatch result of all. At the same time, in terms of the average results, the proposed IDEBDSS method also achieves the best result among all the algorithms. Besides, it is noted that the dispatch results of the proposed IDEBDSS are very close to those of the EPSO. However, the average result of IDEBDSS is only 0.0395% better than that of EBSO, but it is even better than the best result of EBSO. It also indicates that the proposed IDEBDSS method is very competitive compared with EBSO.

From the average CPU time in Table 2, it can be noticed that the proposed method needs less computational effort and they are significantly faster in speed than other methods except for EBSO. Although the proposed IDEBDSS method takes slightly longer CPU time than EBSO, the number of iterations of IDEBDSS is less than EBSO. To be specific, the number of iterations in EBSO method is 700 [54], while that in IDEBDSS is only 600. What is more, the results of the IDEBDSS method are better than those of the EBSO method. Therefore, it can be concluded that the proposed IDEBDSS method will take less iteration number to get better results than the EBSO method.

In order to better display the robustness of the proposed IDEBDSS method, the results of the 40 independent trials are provided in Figures 5 and 6. From Figure 5, it can be seen that the total fuel cost of the best dispatch result obtained by the proposed IDEBDSS method from each independent trial fluctuates in a relatively small range. At the same time, Figure 6 shows that the distribution of the results falls into the same numerical interval that the 40 independent trials produce and that the results of the 40 independent trials approximately follow a normal distribution. From Figures 5 and 6, all these total fuel costs distribute between the minimum and the maximum values without obvious bias, which approximately follow a normal distribution. Hence, the proposed method has a good robustness for solving the DED problem.

In addition, in order to verify the improvement in convergence property of the proposed method relative to the standard DE for solving DED problem, the variation of the best total fuel cost during the evolutionary process is examined. From Figure 3, it is clear that the best total fuel cost of the proposed IDEBDSS method declines sharply at the beginning of iteration compared with that of standard DE, but slowly at later stages. Meanwhile, it can be seen that the proposed IDEBDSS method can get better solutions much more quickly than standard DE method. Therefore, the proposed method has better convergence property and

Table 2: Comparison of results with different methods for test system 1.

| Method | Total generation cost ($) | | Average CPU time (min) | | | Different trial |
|---|---|---|---|---|---|---|
| | Minimum | Average | Actual | CPU | Equalized | |
| DE | 1035479 | 1038680 | 0.21 | 2.2 GHz | 0.21 | 40 |
| SQP [29] | 1051163 | — | 1.19 | 850 MHz | 0.45 | 20 |
| EP [29] | 1048638 | — | 42.49 | 850 MHz | 16.41 | 20 |
| BCO-SQP [30] | 1032200 | — | 2.68 | 3.0 GHz | 3.65 | — |
| EP-SQP [47] | 1031746 | 1035748 | 20.51 | 850 MHz | 7.92 | 20 |
| MHEP-SQP [47] | 1028924 | 1031179 | 21.23 | — | — | 30 |
| MDE [18] | 1031612 | 1033630 | 5.30 | 2.0 GHz | 4.82 | 30 |
| HDE [36] | 1031077 | — | — | 2.4 GHz | — | — |
| DGPSO [48] | 1028835 | 1033630 | 15.39 | 750 GHz | 5.25 | 30 |
| CE [45] | 1022701 | 1024024 | 0.5237 | 1.5 GHz | 0.36 | 30 |
| ECE [45] | 1022271 | 1023334 | 0.5271 | 1.5 GHz | 0.36 | 30 |
| AIS [49] | 1021980 | 1023156 | 19.01 | 3.2 GHz | 27.65 | 30 |
| AHDE [40] | 1020082 | 1022474 | 1.10 | 2.4 GHz | 1.20 | 10 |
| HHS [50] | 1019091 | — | 12.23 | 2.0 GHz | 11.11 | 25 |
| ICPSO [51] | 1019072 | 1020027 | 0.467 | 1.8 GHz | 0.38 | 30 |
| CSAPSO [52] | 1018767 | 1019874 | 0.467 | 1.8 GHz | 0.38 | 40 |
| EAPSO [53] | 1018510 | 1018701 | 0.5 | 3.0 GHz | 0.68 | 40 |
| CSADHS [23] | 1018681 | 1018718 | 2.72 | 1.6 GHz | 1.98 | 100 |
| EBSO [54] | 1017147 | 1017526 | 0.205 | 1.8 GHz | 0.17 | 40 |
| IDEBDSS | 1016873 | 1017124 | 0.33 | 2.2 GHz | 0.33 | 40 |

Table 3: Best solutions obtained by the proposed IDEBDSS method for test system 1 (MW).

| Hour | Unit 1 | Unit 2 | Unit 3 | Unit 4 | Unit 5 | Unit 6 | Unit 7 | Unit 8 | Unit 9 | Unit 10 | Total load |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 150.0000 | 135.0000 | 194.0932 | 60.0000 | 122.8666 | 122.4498 | 129.5904 | 47.0000 | 20.0000 | 55 | 1036 |
| 2 | 150.0000 | 135.0000 | 268.1132 | 60.0000 | 122.8666 | 122.4498 | 129.5704 | 47.0000 | 20.0000 | 55 | 1110 |
| 3 | 226.6250 | 215.0000 | 309.3354 | 60.0000 | 73.0000 | 122.4492 | 129.5904 | 47.0000 | 20.0000 | 55 | 1258 |
| 4 | 303.2484 | 295.0000 | 300.7051 | 60.0000 | 73.0000 | 122.4561 | 129.5904 | 47.0000 | 20.0000 | 55 | 1406 |
| 5 | 303.2485 | 310.0043 | 309.8324 | 60.0000 | 122.8674 | 122.4570 | 129.5904 | 47.0000 | 20.0000 | 55 | 1480 |
| 6 | 379.8747 | 390.1823 | 301.0241 | 60.0000 | 122.8663 | 122.4622 | 129.5904 | 47.0000 | 20.0000 | 55 | 1628 |
| 7 | 379.8729 | 396.8001 | 318.5557 | 60.0000 | 172.7351 | 122.4458 | 129.5904 | 47.0000 | 20.0000 | 55 | 1702 |
| 8 | 379.8726 | 396.7994 | 297.1018 | 105.5865 | 222.5996 | 122.4497 | 129.5904 | 47.0000 | 20.0000 | 55 | 1776 |
| 9 | 456.4968 | 396.7994 | 299.2045 | 144.8595 | 222.5997 | 122.4497 | 129.5904 | 77.0000 | 20.0000 | 55 | 1924 |
| 10 | 456.4968 | 396.7994 | 330.5748 | 185.6150 | 222.6186 | 160.0000 | 129.5904 | 85.3050 | 50.0000 | 55 | 2072 |
| 11 | 456.4968 | 396.7994 | 324.5355 | 231.7083 | 222.5997 | 160.0000 | 129.5904 | 117.2128 | 52.0571 | 55 | 2146 |
| 12 | 456.4968 | 460.0000 | 325.5698 | 240.7433 | 222.5997 | 160.0000 | 129.5904 | 120.0000 | 50.0000 | 55 | 2220 |
| 13 | 456.4968 | 396.7994 | 325.1248 | 186.3889 | 222.5997 | 160.0000 | 129.5904 | 120.0000 | 20.0000 | 55 | 2072 |
| 14 | 456.4968 | 396.7994 | 290.5637 | 140.5003 | 222.5996 | 122.4498 | 129.5904 | 90.0000 | 20.0000 | 55 | 1924 |
| 15 | 379.8726 | 396.7994 | 303.9199 | 110.3227 | 172.7331 | 122.4498 | 129.5904 | 85.3121 | 20.0000 | 55 | 1776 |
| 16 | 303.2484 | 396.7994 | 287.5221 | 61.2113 | 122.8665 | 122.4498 | 129.5904 | 55.3121 | 20.0000 | 55 | 1554 |
| 17 | 226.6243 | 396.8011 | 299.6654 | 60.0000 | 122.8664 | 122.4524 | 129.5904 | 47.0000 | 20.0000 | 55 | 1480 |
| 18 | 303.2484 | 396.7996 | 321.1787 | 60.0000 | 172.7333 | 122.4496 | 129.5904 | 47.0000 | 20.0000 | 55 | 1628 |
| 19 | 379.8726 | 396.7994 | 297.2775 | 105.4103 | 222.5997 | 122.4501 | 129.5904 | 47.0000 | 20.0000 | 55 | 1776 |
| 20 | 456.4968 | 460.0000 | 340.0000 | 121.3131 | 222.5997 | 160.0000 | 129.5904 | 77.0000 | 50.0000 | 55 | 2072 |
| 21 | 456.4968 | 395.5979 | 317.1186 | 120.5966 | 222.5997 | 160.0000 | 129.5904 | 47.0000 | 20.0000 | 55 | 1924 |
| 22 | 379.8726 | 309.2646 | 273.5396 | 68.7188 | 222.5642 | 122.4498 | 129.5904 | 47.0000 | 20.0000 | 55 | 1628 |
| 23 | 302.9124 | 229.5981 | 193.3172 | 60.0000 | 172.1321 | 122.4498 | 129.5904 | 47.0000 | 20.0000 | 55 | 1332 |
| 24 | 226.6242 | 222.2665 | 178.2025 | 60.0000 | 122.8666 | 122.4498 | 129.5904 | 47.0000 | 20.0000 | 55 | 1184 |

Table 4: Comparison of results with different methods for test system 2.

| Method | Total generation cost ($) | | Average CPU time (min) | | | Different trial |
|---|---|---|---|---|---|---|
| | Minimum | Average | Actual | CPU | Equalized | |
| DE | 1053278 | 1058118 | 0.23 | 2.2 GHz | 0.23 | 40 |
| EP [47] | 1054685 | 1057323 | 47.23 | 850 MHz | 18.25 | 20 |
| EP-SQP [47] | 1052668 | 1053771 | 27.53 | 850 MHz | 10.64 | 20 |
| MHEP-SQP [47] | 1050054 | 1052394 | 24.33 | — | — | 30 |
| AIS [49] | 1045715 | 1047050 | 23.22 | 3.2 GHz | 33.77 | 30 |
| CSAPSO [52] | 1038251 | 1039543 | — | 1.8 GHz | — | 40 |
| EBSO [54] | 1038915 | 1039188 | 0.22 | 1.8 GHz | 0.27 | 40 |
| EAPSO [53] | 1037898 | 1038109 | 2.3 | 3 GHz | 3.13 | 40 |
| CSADHS [23] | 1035199 | 1035259 | 2.8 | 1.6 GHz | 2.03 | 100 |
| IDEBDSS | 1035061 | 1035104 | 0.35 | 2.2 GHz | 0.35 | 40 |

Table 5: Best solutions obtained by the proposed IDEBDSS method for test system 2 (MW).

| Hour | Unit 1 | Unit 2 | Unit 3 | Unit 4 | Unit 5 | Unit 6 | Unit 7 | Unit 8 | Unit 9 | Unit 10 | PLoss | Total load |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 150.0000 | 135.0000 | 206.2132 | 60.0000 | 122.8666 | 122.4495 | 129.5901 | 47.0000 | 20.0000 | 55 | 12.1194 | 1036 |
| 2 | 150.0000 | 135.0000 | 283.4721 | 60.0000 | 122.8621 | 122.3805 | 129.5934 | 47.0000 | 20.0000 | 55 | 15.3081 | 1110 |
| 3 | 226.6255 | 140.5172 | 303.5180 | 60.0023 | 172.7118 | 122.4501 | 129.5903 | 47.0000 | 20.0000 | 55 | 19.4150 | 1258 |
| 4 | 303.2481 | 217.1263 | 298.3105 | 60.0000 | 172.7201 | 122.5314 | 129.5903 | 47.0000 | 20.0000 | 55 | 19.5267 | 1406 |
| 5 | 378.1103 | 221.4563 | 298.1562 | 60.0000 | 172.7364 | 122.4495 | 129.5902 | 47.0000 | 20.0000 | 55 | 24.4989 | 1480 |
| 6 | 379.2190 | 395.1559 | 292.5124 | 60.7754 | 122.1191 | 122.0033 | 129.5931 | 77.0000 | 20.0000 | 55 | 25.3782 | 1628 |
| 7 | 379.3315 | 308.1123 | 301.6784 | 110.2891 | 222.6871 | 122.3588 | 129.5937 | 85.0021 | 20.0000 | 55 | 32.0530 | 1702 |
| 8 | 456.4955 | 307.2651 | 301.6512 | 160.4467 | 172.8113 | 122.4713 | 129.5913 | 85.1197 | 20.0000 | 55 | 34.8521 | 1776 |
| 9 | 456.4961 | 381.5338 | 299.2182 | 191.7144 | 222.0122 | 122.1125 | 129.5991 | 85.7833 | 20.0000 | 55 | 39.4696 | 1924 |
| 10 | 456.5337 | 396.1244 | 325.5543 | 242.3067 | 222.5977 | 159.9998 | 129.5902 | 115.1221 | 20.0038 | 55 | 50.8327 | 2072 |
| 11 | 456.5051 | 396.8771 | 340.0000 | 291.8865 | 226.9819 | 160.0000 | 129.8112 | 120.0000 | 20.0073 | 55 | 51.0691 | 2146 |
| 12 | 456.4981 | 460.0000 | 325.9193 | 293.2254 | 222.7892 | 160.0000 | 129.1218 | 120.0000 | 50.0673 | 55 | 52.6211 | 2220 |
| 13 | 456.4972 | 396.7975 | 305.1244 | 293.0451 | 222.5877 | 122.4543 | 129.5967 | 120.0000 | 20.0008 | 55 | 49.1037 | 2072 |
| 14 | 456.8864 | 315.9902 | 310.5337 | 240.9335 | 222.6017 | 122.8966 | 129.5998 | 90.0000 | 20.0000 | 55 | 40.4419 | 1924 |
| 15 | 379.9812 | 307.6411 | 296.8875 | 191.5335 | 222.7889 | 122.4366 | 129.5832 | 85.3165 | 20.0000 | 55 | 35.1685 | 1776 |
| 16 | 303.5614 | 233.1066 | 283.7054 | 177.6954 | 172.0887 | 122.9389 | 129.5899 | 85.3044 | 20.0000 | 55 | 28.9907 | 1554 |
| 17 | 226.6345 | 309.7312 | 305.2144 | 126.8105 | 122.4622 | 122.7871 | 129.5899 | 85.3137 | 20.0000 | 55 | 23.5435 | 1480 |
| 18 | 303.2955 | 309.5308 | 293.0189 | 165.3517 | 172.8655 | 122.0773 | 129.5944 | 85.3002 | 20.0000 | 55 | 28.0343 | 1628 |
| 19 | 379.8966 | 309.7066 | 305.8871 | 180.8354 | 222.9094 | 122.9508 | 129.5997 | 85.3392 | 20.0000 | 55 | 36.1248 | 1776 |
| 20 | 456.5441 | 389.3176 | 335.1308 | 230.8341 | 222.7566 | 160.0000 | 129.5993 | 115.7032 | 20.0000 | 55 | 42.8857 | 2072 |
| 21 | 456.4844 | 389.4533 | 305.3274 | 180.8443 | 222.6377 | 122.4544 | 129.5873 | 85.3088 | 20.0000 | 55 | 43.0976 | 1924 |
| 22 | 379.8764 | 310.8876 | 283.1745 | 130.8401 | 172.7224 | 122.8655 | 129.5897 | 55.3114 | 20.0000 | 55 | 32.2676 | 1628 |
| 23 | 303.2518 | 231.6654 | 205.1876 | 118.9013 | 122.8588 | 122.4473 | 129.5905 | 47.0000 | 20.0000 | 55 | 23.9027 | 1332 |
| 24 | 226.6188 | 222.2679 | 184.6386 | 120.0015 | 73.0000 | 122.4577 | 129.5844 | 47.0001 | 20.0000 | 55 | 16.5690 | 1184 |

can get the optimal solution much more quickly compared with standard DE. Moreover, the following conclusions about the modification work can be drawn from Figure 3: (1) in order to improve the parameter settings of DE, the dynamic parameter control mechanism based on chaotic sequences is adopted in the proposed IDEBDSS method to improve the convergence property of DE, without consuming extra computation time; (2) by using the local search strategy, the population can approach the feasible region more quickly in the early stage of evolution, which can accelerate convergence; (3) by using the global search strategy, search capability of the globally optimal solution can be enhanced. Although the implementation of QOX operator may need a little extra computation time, it can improve the search ability for globally optimal solution when QOX is embedded into the proposed IDEBDSS method. Thus, the IDEBDSS method can obtain the globally optimal solution with larger probability in the later stage of evolution.

Finally, the last column in Table 3 provides the sum of power generations for all units. From the analysis of the dispatch results, it can be seen that the dispatch results obtained by the proposed IDEBDSS method satisfy all kinds

Table 6: Comparison of results with different methods for test system 3.

| Method | Total generation cost ($) | | Average CPU time (min) | | | Different trial |
|---|---|---|---|---|---|---|
| | Minimum | Average | Actual | CPU | Equalized | |
| DE | 3162709 | 3171416 | 0.56 | 2.2 GHz | 0.56 | 40 |
| EP [47] | 3164531 | 3200171 | 177.39 | 850 MHz | 68.54 | 20 |
| EP-SQP [47] | 3159204 | 3169093 | 27.53 | 850 MHz | 10.64 | 20 |
| MHEP-SQP [47] | 3151445 | 3157738 | 24.33 | — | — | 30 |
| CSAPSO [52] | 3066907 | 3075023 | 1.02 | 1.8 GHz | 0.84 | 40 |
| DGPSO [48] | 3148992 | 3154438 | 73.01 | 750 GHz | 24.88 | 30 |
| CE [45] | 3086109 | 3088869 | 2.0740 | 1.5 GHz | 1.41 | 30 |
| ECE [45] | 3084649 | 3087847 | 2.1375 | 1.5 GHz | 1.46 | 30 |
| ICPSO [51] | 3064497 | 3071588 | 1.03 | 1.8 GHz | 0.84 | 30 |
| HHS [50] | 3057313 | — | 27.65 | 2 GHz | 25.13 | 25 |
| EAPSO [53] | 3054961 | 3055252 | — | 3 GHz | — | 40 |
| CSADHS [23] | 3054709 | 3055070 | 7.37 | 1.6 GHz | 5.36 | 100 |
| EBSO [54] | 3054001 | 3054697 | 0.95 | 1.8 GHz | 0.78 | 40 |
| IDEBDSS | 3049736 | 3050492 | 0.80 | 2.2 GHz | 0.80 | 40 |

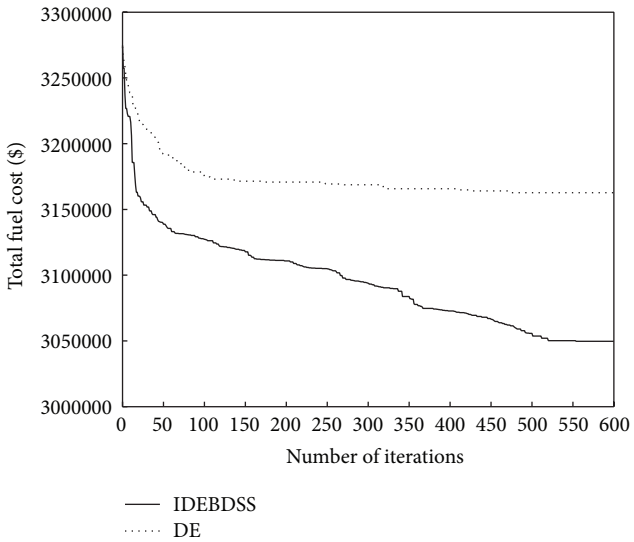


Figure 4: Convergence process of the best solution obtained by IDEBDSS and DE methods for test system 3.

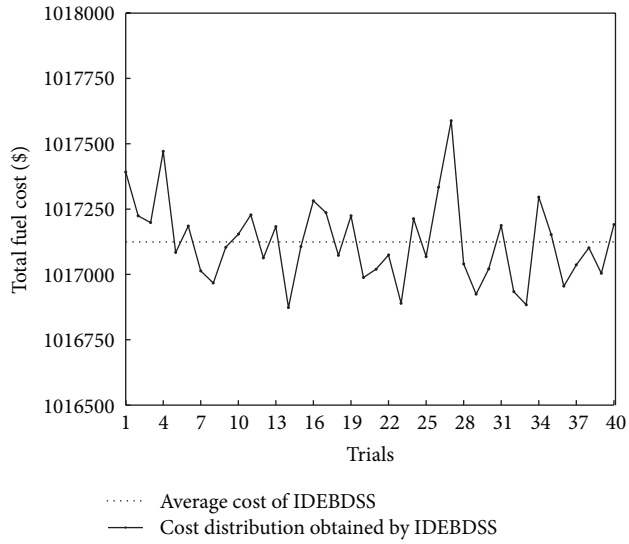


Figure 5: Cost distribution obtained by IDEBDSS for test system 1.

of complicated constraints of DED problem, thus reducing the total fuel cost effectively.

*5.3.2. Test System 2.* Test system 2 consists of ten generating units with the consideration of transmission losses. The $B$-coefficient values are taken from [47]. The total generation cost obtained by the proposed algorithm for 24 h is 1035061$ and the corresponding generation schedule is shown in Table 5. As can be seen from Table 4, the best and the average total fuel costs of the proposed IDEBDSS method for test system 2 are higher than test system 1 due to considering of the transmission losses. Actually, the power balance constraints given by (4) without transmission losses are easiy to implement, and CPU execution time can also be shortened, which leads to the computation time difference for the two test systems. At the same time, the comparison results of the best total cost, as well as the average total cost and the CPU execution time, between the proposed algorithm and other methods are shown in Table 4, from which it can be deduced that the proposed IDEBDSS method provides the lowest total fuel cost among all the above mentioned methods.

Besides, it can also be found that the dispatch results of the proposed IDEBDSS method are very close to those of the CSADHS method. The average total fuel cost of the IDEBDSS method is only 0.0150% better than that of the CSADHS method, but the average result of the IDEBDSS method is even better than the best result of the CSADHS method.

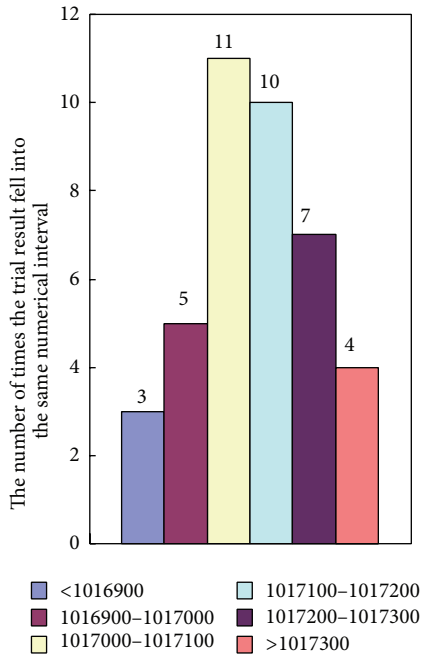


Figure 6: Distribution of the results of the 40 independent trials for test system 1.

Meanwhile, the IDEBDSS method spends no more than one-fifth of average CPU time of the CSADHS method, which is the shortest except for the EBSO method for calculation time. For the case with similar dispatch results in test system 1, the dispatch results of IDEBDSS are much better than those of EPSO in test system 2, which confirms the good performance of IDEBDSS in test system with transmission losses.

*5.3.3. Test System 3.* Compared with test system 1, test system 3 is a 30-unit large-scale power system, which is more complex in the nonconvex, nonlinear, and nonsmooth characteristics of the solution space. In order to prove the efficiency of the proposed algorithm for solving DED problems of large-scale system, the proposed IDEBDSS method and standard DE method are employed to optimize this test system, and the corresponding convergence property comparisons are illustrated in Figure 4. As can be seen, the proposed IDEBDSS method can get better results continuously in the later stage of evolution, indicating that the proposed IDEBDSS method can avoid being trapped into locally optimal area effectively. That is to say, IDEBDSS has better convergence property than standard DE. Table 6 shows the comparison results of the best total fuel cost, average total fuel cost, and average CPU execution time between the proposed IDEBDSS method and other mentioned methods. It can be found easily that IDEBDSS can get the best results both in minimum and average cost relative to other mentioned algorithms. Meanwhile, the results of IDEBDSS are very close to those of EBSO, and the solution of the former is just 0.1377% higher than that of the later. Although the improvement is relatively limited, it is more than three times higher than that observed in test system 1 (0.0395%). From viewpoint of the execution time, IDEBDSS consumes the shortest time among all methods but EBSO and the time difference between IDEBDSS and EBSO can dwindle to near insignificance with the scale-up of system from system 1 to 3.

In addition, for this test system, the average CPU time of IDEBDSS is 0.8 min, which is only about twice as that of test system 1. This demonstrates that the proposed method is efficient and superior for large-scale application.

## 6. Conclusions

In this paper, an improved differential evolution method based on the dynamic search strategy is presented to solve DED problem with valve-point effects. A new constraint pretreatment method is proposed to effectively handle complicated constraints of the DED problem, and an effective dynamic search strategy is adopted to improve convergence and global search capability during the evolution. Meanwhile, a dynamic parameter control mechanism based on Logistic map chaotic sequences is embedded into the proposed method to adjust the parameter values of DE adaptively. Finally, three different test systems are solved by the proposed method, and the dispatch results are compared directly with those of other methods in the recent literature. The result indicates that the proposed method can obtain not only the minimum total fuel cost but also a shorter CPU computation time compared with other methods. No matter in small scale or large scale, the simulation results confirm the superiority of the proposed method in both the solution quality and the computation efficiency. Therefore, the proposed IDEBDSS method can be a new and effective approach for solving DED problem.

## Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

## Acknowledgment

The authors would like to thank the anonymous reviewers for the constructive and helpful comments.

## References

[1] X. Xia and A. M. Elaiw, "Optimal dynamic economic dispatch of generation: a review," *Electric Power Systems Research*, vol. 80, no. 8, pp. 975–986, 2010.

[2] R. A. Jabr, A. Coonick, and B. Cory, "A homogeneous linear programming algorithm for the security constrained economic dispatch problem," *IEEE Transactions on Power Systems*, vol. 15, no. 3, pp. 930–936, 2000.

[3] G. P. Granelli and M. Montagna, "Security-constrained economic dispatch using dual quadratic programming," *Electric Power Systems Research*, vol. 56, no. 1, pp. 71–80, 2000.

[4] A. Keib, H. Ma, and J. Hart, "Environmentally constrained economic dispatch using the Lagrangian relaxation method," *IEEE Transactions on Power Systems*, vol. 9, no. 4, pp. 1723–1729, 1994.

[5] D. L. Travers and R. John Kaye, "Dynamic dispatch by constructive dynamic programming," *IEEE Transactions on Power Systems*, vol. 13, no. 1, pp. 72–78, 1998.

[6] C. Chiang, "Improved genetic algorithm for power economic dispatch of units with valve-point effects and multiple fuels," *IEEE Transactions on Power Systems*, vol. 20, no. 4, pp. 1690–1699, 2005.

[7] N. Sinha, R. Chakrabarti, and P. K. Chattopadhyay, "Evolutionary programming techniques for economic load dispatch," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 1, pp. 83–94, 2003.

[8] W. M. Lin, F. S. Cheng, and M. T. Tsay, "An improved tabu search for economic dispatch with multiple minima," *IEEE Transactions on Power Systems*, vol. 17, no. 1, pp. 108–112, 2002.

[9] Z. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *IEEE Transactions on Power Systems*, vol. 18, no. 3, pp. 1187–1195, 2003.

[10] V. Hosseinnezhad and E. Babaei, "Economic load dispatch using $\theta$-PSO," *International Journal of Electrical Power and Energy Systems*, vol. 49, no. 1, pp. 160–169, 2013.

[11] L. dos Santos Coelho and V. C. Mariani, "Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect," *IEEE Transactions on Power Systems*, vol. 21, no. 2, pp. 989–996, 2006.

[12] N. Duvvuru and K. S. Swarup, "A hybrid interior point assisted differential evolution algorithm for economic dispatch," *IEEE Transactions on Power Systems*, vol. 26, no. 2, pp. 541–549, 2011.

[13] D. He, F. Wang, and Z. Z. Mao, "A hybrid genetic algorithm approach based on differential evolution for economic dispatch with valve-point effect," *International Journal of Electrical Power and Energy Systems*, vol. 30, no. 1, pp. 31–38, 2008.

[14] N. Amjady and H. Sharifzadeh, "Solution of non-convex Economic Dispatch problem considering valve loading effect by a new Modified Differential Evolution algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 32, no. 8, pp. 893–903, 2010.

[15] C. H. Peng, H. J. Sun, J. F. Guo, and G. Liu, "Dynamic economic dispatch for wind-thermal power system using a novel bi-population chaotic differential evolution algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 42, no. 1, pp. 119–126, 2012.

[16] L. D. S. Coelho, B. T. Santos, and V. C. Mariani, "Differential evolution based on truncated Levy-type flights and population diversity measure to solve economic load dispatch problems," *International Journal of Electrical Power and Energy Systems*, vol. 57, pp. 178–188, 2014.

[17] A. S. Reddy and K. Vaisakh, "Shuffled differential evolution for economic dispatch with valve point loading effects," *International Journal of Electrical Power and Energy Systems*, vol. 46, no. 1, pp. 342–352, 2013.

[18] X. H. Yuan, L. Wang, Y. B. Yuan, Y. Zhang, B. Cao, and B. Yang, "A modified differential evolution approach for dynamic economic dispatch with valve-point effects," *Energy Conversion and Management*, vol. 49, no. 12, pp. 3447–3453, 2008.

[19] D. He, G. Dong, F.L. Wang, and Z. Mao, "Optimization of dynamic economic dispatch with valve-point effect using chaotic sequence based differential evolution algorithms," *Energy Conversion and Management*, vol. 52, no. 2, pp. 1026–1032, 2011.

[20] R. Balamurugan and S. Subramanian, "Hybrid integer coded differential evolution-dynamic programming approach for economic load dispatch with multiple fuel options," *Energy Conversion and Management*, vol. 49, no. 4, pp. 608–614, 2008.

[21] L. D. S. Coelho and V. C. Mariani, "Improved differential evolution algorithms for handling economic dispatch optimization with generator constraints," *Energy Conversion and Management*, vol. 48, no. 5, pp. 1631–1639, 2007.

[22] A. Bhattacharya and P. K. Chattopadhyay, "Biogeography-based optimization for different economic load dispatch problems," *IEEE Transactions on Power Systems*, vol. 25, no. 2, pp. 1064–1077, 2010.

[23] R. Arul, G. Ravi, and S. Velusami, "Chaotic self-adaptive differential harmony search algorithm based dynamic economic dispatch," *International Journal of Electrical Power and Energy Systems*, vol. 50, no. 1, pp. 85–96, 2013.

[24] H. W. Zhong, Q. Xia, Y. Wang, and C. Q. Kang, "Dynamic economic dispatch considering transmission losses using quadratically constrained quadratic program method," *IEEE Transactions on Power Systems*, vol. 28, no. 3, pp. 2232–2241, 2013.

[25] M. Barun, K. R. Provas, and M. Sanjoy, "Economic load dispatch using krill herd algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 57, pp. 1–10, 2014.

[26] Q. Niu, H. Y. Zhang, K. Li, and G. W. Irwin, "An efficient harmony search with new pitch adjustment for dynamic economic dispatch," *Energy*, vol. 65, no. 1, pp. 25–43, 2014.

[27] T. A. A. Victoire and A. E. Jeyakumar, "Hybrid PSO-SQP for economic dispatch with valve-point effect," *Electric Power Systems Research*, vol. 71, no. 1, pp. 51–59, 2004.

[28] A. Bhattacharya and P. K. Chattopadhyay, "Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch," *IEEE Transactions on Power Systems*, vol. 25, no. 4, pp. 1955–1964, 2010.

[29] P. Attaviriyanupap, H. Kita, E. Tanaka, and J. Hasegawa, "A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function," *IEEE Transactions on Power Systems*, vol. 17, no. 2, pp. 411–416, 2002.

[30] M. Basu, "Hybridization of bee colony optimization and sequential quadratic programming for dynamic economic dispatch," *International Journal of Electrical Power and Energy Systems*, vol. 44, no. 1, pp. 591–596, 2013.

[31] R. Storn and K. Price, "Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[32] H. R. Cai, C. Y. Chung, and K. P. Wong, "Application of differential evolution algorithm for transient stability constrained optimal power flow," *IEEE Transactions on Power Systems*, vol. 23, no. 2, pp. 719–728, 2008.

[33] J. Chiou, C. Chang, and C. Su, "Variable scaling hybrid differential evolution for solving network reconfiguration of distribution systems," *IEEE Transactions on Power Systems*, vol. 20, no. 2, pp. 668–674, 2005.

[34] M. Varadarajan and K. S. Swarup, "Differential evolutionary algorithm for optimal reactive power dispatch," *IEEE Transactions on Power Systems*, vol. 30, no. 8, pp. 435–441, 2008.

[35] A. Y. Qing, "Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 1, pp. 116–125, 2006.

[36] X. Yuan, L. Wang, Y. Zhang, and Y. Yuan, "A hybrid differential evolution method for dynamic economic dispatch with valve-point effects," *Expert Systems with Applications*, vol. 36, no. 2, pp. 4042–4048, 2009.

[37] L. L. Jiang, D. L. Maskell, and J. C. Patra, "Parameter estimation of solar cells and modules using an improved adaptive differential evolution algorithm," *Applied Energy*, vol. 112, pp. 185–193, 2013.

[38] M. Locatelli, M. Maischberger, and F. Schoen, "Differential evolution methods based on local searches," *Applied Soft Computing*, vol. 43, pp. 169–180, 2014.

[39] T. R. Chelliah, R. Thangaraj, and S. Allamsetty, "Coordination of directional overcurrent relays using opposition based chaotic differential evolution algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 55, pp. 341–350, 2014.

[40] Y. Lu, J. Zhou, H. Qin, Y. Li, and Y. Zhang, "An adaptive hybrid differential evolution algorithm for dynamic economic dispatch with valve-point effects," *Expert Systems with Applications*, vol. 37, no. 7, pp. 4842–4849, 2010.

[41] Y. Wang and Z. Cai, "A dynamic hybrid framework for constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 42, no. 1, pp. 203–217, 2012.

[42] T. A. A. Victoire and A. E. Jeyakumar, "Reserve constrained dynamic dispatch of units with valve-point effects," *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1273–1282, 2005.

[43] G. P. Granelli, P. Marannino, M. Montagna, and A. Silvestri, "Fast and efficient gradient projection algorithm for dynamic generation dispatching," *IEE Proceedings C: Generation Transmission and Distribution*, vol. 136, no. 5, pp. 295–302, 1989.

[44] D. C. Walters and G. B. Sheble, "Genetic algorithm solution of economic dispatch with value point loading," *IEEE Transactions on Power Systems*, vol. 8, no. 3, pp. 1325–1332, 1993.

[45] A. I. Selvakumar, "Enhanced cross-entropy method for dynamic economic dispatch with valve-point effects," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 3, pp. 783–790, 2011.

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

[47] T. A. A. Victoire and A. E. Jeyakumar, "A modified hybrid EP-SQP approach for dynamic dispatch with valve-point effect," *International Journal of Electrical Power and Energy Systems*, vol. 27, no. 8, pp. 594–601, 2005.

[48] T. A. A. Victoire and A. E. Jeyakumar, "Deterministically guided PSO for dynamic dispatch considering valve-point effect," *Electric Power Systems Research*, vol. 73, no. 3, pp. 313–322, 2005.

[49] S. Hemamalini and S. P. Simon, "Dynamic economic dispatch using artificial immune system for units with valve-point effect," *International Journal of Electrical Power and Energy Systems*, vol. 33, no. 4, pp. 868–874, 2011.

[50] V. Ravikumar Pandi and B. K. Panigrahi, "Dynamic economic load dispatch using hybrid swarm intelligence based harmony search algorithm," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8509–8514, 2011.

[51] Y. Wang, J. Zhou, H. Qin, and Y. Lu, "Improved chaotic particle swarm optimization algorithm for dynamic economic dispatch problem with valve-point effects," *Energy Conversion and Management*, vol. 51, no. 12, pp. 2893–2900, 2010.

[52] Y. Wang, J. Zhou, Y. Lu, and H. Qin, "Chaotic self-adaptive particle swarm optimization algorithm for dynamic economic dispatch problem with valve-point effects," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14231–14237, 2011.

[53] T. Niknam and F. Golestaneh, "Enhanced adaptive particle swarm optimisation algorithm for dynamic economic dispatch of units considering valve-point effects and ramp rates," *IET Generation, Transmission & Distribution*, vol. 6, no. 5, pp. 424–435, 2012.

[54] T. Niknam and F. Golestaneh, "Enhanced bee swarm optimization algorithm for dynamic economic dispatch," *IEEE Systems Journal*, vol. 7, no. 4, pp. 754–762, 2013.

[55] R. Caponetto, L. Fortuna, S. Fazzino, and M. G. Xibilia, "Chaotic sequences to improve the performance of evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 3, pp. 289–304, 2003.

[56] Y. Leung and Y. Wang, "An orthogonal genetic algorithm with quantization for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 41–53, 2001.

[57] Y. Wang, Z. Cai, and Q. Zhang, "Enhancing the search ability of differential evolution through orthogonal crossover," *Information Sciences*, vol. 185, pp. 153–177, 2012.

[58] C. M. Fonseca and P. J. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms part I: a unified formulation," *IEEE Transactions on Systems, Man, and Cybernetics A: Systems and Humans*, vol. 28, no. 1, pp. 26–37, 1998.

[59] Z. Cai and Y. Wang, "A multiobjective optimization-based evolutionary algorithm for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 658–675, 2006.