Information Sciences 192 (2012) 213–227

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch

Yong Zhang [a,*], Dun-Wei Gong [a], Zhonghai Ding [b]

[a] School of Information and Electronic Engineering, China University of Mining and Technology, Xuzhou 221008, China
[b] Department of Mathematical Sciences, University of Nevada, Las Vegas, NV 89154-4020, USA

ARTICLE INFO

Article history:
Available online 6 June 2011

Keywords:
Environmental/economic dispatch
Multi-objective optimization
Particle swarm

ABSTRACT

In this paper, we propose a new bare-bones multi-objective particle swarm optimization algorithm to solve the environmental/economic dispatch problems. The algorithm has three distinctive features: a particle updating strategy which does not require tuning up control parameters; a mutation operator with action range varying over time to expand the search capability; and an approach based on particle diversity to update the global particle leaders. Several trials have been carried out on the IEEE 30-bus test system. By comparing with seven existing multi-objective optimization algorithms and three well-known multi-objective particle swarm optimization techniques, it is found that our algorithm is capable of generating excellent approximation of the true Pareto front and can be used to solve other types of multi-objective optimization problems.

1. Introduction

The primary objective of the classical economic dispatch is to operate the electric power systems by meeting the load demand at a minimum fuel cost regardless of emissions produced. However, with the increasing public awareness of the environmental pollution caused by the fossil fuel fired thermal power plants, the total fuel cost can no longer be considered alone. Limiting the emission of pollutants becomes another crucial objective. In this circumstance, as a short-term alternative to reduce the atmospheric emissions, the environmental/economic dispatch (EED) is becoming more and more desirable, because it results in not only greater economical benefits, but also less pollutant emissions [35].

To handle the EED problem with conflicting objectives, many techniques have been proposed, which can be mainly divided into three groups [1].

- The first group treats the emission as a constraint with a permissible limit [6,17,25]. However, it fails to provide any information regarding the tradeoff front.
- The second group treats the emission as a different objective in addition to the usual fuel cost objective. However, the EED problem is handled as a single-objective optimization problem by using the linear weighted sum method [8,25] and the price penalty factor [26], or by considering one objective at a time [1,8]. These techniques require multiple runs to obtain a set of non-dominated solutions, and cannot be used to find the Pareto-optimal solutions for those problems with a non-convex Pareto-optimal front.

* Corresponding author.
E-mail address: Yongzh401@126.com (Y. Zhang).

0020-0255/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.06.004

- The third group handles both the fuel cost and the emission simultaneously as competing objectives. Over the past decade, this approach has attracted many researchers' interests due to the new development of multi-objective evolutionary search techniques. Several multi-objective evolutionary algorithms such as the niched Pareto genetic algorithm (NPGA) [2], the nondominated sorting genetic algorithm (NSGA) [3], the strength Pareto evolutionary algorithm (SPEA) [4] and the NSGA-II [7,29] have been introduced to solve the EED problem. In addition, some other optimization approaches, such as the multi-objective mathematical programming method [43] and the biogeography-based optimization algorithm [44] have also been proposed for generating Pareto-optimal solutions of the EED problem.

The particle swarm optimization (PSO) is a heuristic search technique that is inspired by the behavior of bird flocks [18]. Due to its advantages such as simplicity, fast convergence and population-based feature, the PSO is a favorable technique to tackle the optimization problems of power systems [9,27,39,42]. In recent years, several PSO-based approaches such as the multi-objective PSO [1], the comprehensive learning particle swarm optimizer (MOCLPSO) [37], the fuzzified multi-objective particle swarm optimization (FMOPSO) [40], the multi-objective chaotic particle swarm optimization (MOCPSO) [10], the fuzzy clustering-based particle swarm optimization (FCPSO) [5], and the Pareto archive-based particle swarm optimization [45], have been proposed to solve the EED problem. However, all of them require users to tune control parameter such as inertia weight, acceleration coefficients and velocity clamping in order to obtain the desirable solutions. Moreover, empirical and theoretical studies have shown that the convergence behavior of PSO depends strongly on the values of these control parameters [11,38,46]. In other words, the settings of these control parameters depend on individual applications, and need to be adjusted for different problems.

In this paper, a new multi-objective optimization algorithm, called the bare-bones multi-objective particle swarm optimization (BB-MOPSO), is proposed for solving the EED problems. It is worth to point out that the concept of bare-bones particle swarm optimization (BBPSO) was used in some early literatures to deal with the single-objective problems [19,22,23]. In this paper, we extend the idea of the BBPSO to solve the multi-objective optimization problems. The BB-MOPSO has three distinctive features: a particle updating strategy which does not require tuning up control parameters; a mutation operator with action range varying over time to expand the search capability and avoid the premature convergence; and an approach based on particle diversity to update the global particle leaders. A technique dealing with constraints is introduced for fast adjusting the unfeasible solutions of the EED problems. Several existing techniques such as the external repository of elite particles, the crowding distance [14] and the fuzzy-based mechanism to extract the compromise solutions, are incorporated into the BB-MOPSO.

This paper is organized as follows. The mathematical model for the EED problem is given in Section 2. A brief review of the BBPSO is given in Section 3. In Section 4, we present the BB-MOPSO algorithm. Section 5 describes how to implement the BB-MOPSO algorithm. The performance analysis of the BB-MOPSO algorithm and the comparison with several existing optimization algorithms are given in Sections 6–8. The conclusions and remarks are given in Section 9.

2. Mathematical model for the EED problem

A classical EED problem is to minimize simultaneously two competing objective functions, fuel cost and emission, subject to several equality and inequality constraints. The mathematical model for the EED problem is described as follows.

2.1. Objective functions

Considering a power system with N generators: $G1, G2, \ldots, GN$. The total fuel cost $FC(\vec{P}_G)$ (dollars per hour) can be represented as

$$FC(\vec{P}_G) = \sum_{i=1}^{N} a_i + b_i P_{Gi} + c_i P_{Gi}^2, \tag{1}$$

where $\vec{P}_G = (P_{G1}, P_{G2}, \ldots, P_{GN})$; P_{Gi} is the power output of the ith generator; a_i, b_i and c_i are the cost coefficients of the ith generator.

The total emission $EM(\vec{P}_G)$ (tons per hour) of atmospheric pollutants such as sulfur oxides SO_X and nitrogen oxides NO_X caused by the fossil-fueled thermal generators can be represented as [2–5,10,40]

$$EM(\vec{P}_G) = \sum_{i=1}^{N} 10^{-2}(\alpha_i + \beta_i P_{Gi} + \gamma_i P_{Gi}^2) + \zeta_i \exp(\lambda_i P_{Gi}), \tag{2}$$

where α_i, β_i, γ_i, ζ_i and λ_i are the emission coefficients of the ith generator.

2.2. Constraints

In this paper, we consider one equality constraint on the power balance and several inequality constraints on generation capacity.

Generation capacity constraints: for stable operation, the power output of each generator is bounded by the lower and upper bounds as follows:

$$P_{Gi}^{\min} \leqslant P_{Gi} \leqslant P_{Gi}^{\max}, \quad i = 1, 2, \ldots, N. \tag{3}$$

Power balance constraint: the total power generation must cover the total demand P_D and the power loss P_L in transmission lines, namely,

$$\sum_{i=1}^{N} P_{Gi} = P_D + P_L. \tag{4}$$

Here a reduction method is applied to model transmission losses as a function of the generator outputs with Kron's loss coefficients introduced in [30]. The Kron's loss formula can be expressed as

$$P_L = \sum_{i=1}^{N}\sum_{j=1}^{N} P_{Gi}B_{ij}P_{Gj} + \sum_{i=1}^{N} B_{0i}P_{Gi} + B_{00}, \tag{5}$$

where B_{ij}, B_{0i} and B_{00} are the transmission network power loss coefficients.

2.3. The optimization problem

By integrating two objectives (1) and (2) with constraints (3) and (4), the EED problem can be described as follows: find the vector $\vec{P}_G^* = (P_{G1}^*, P_{G2}^*, \ldots, P_{GN}^*)$ such that

$$\vec{F}(\vec{P}_G^*) = \min \qquad (FC(\vec{P}_G), EM(\vec{P}_G)) \tag{6}$$

$$\text{subject to} \quad \begin{cases} \sum_{i=1}^{N} P_{Gi} = P_D + P_L, \\ P_{Gi}^{\min} \leqslant P_{Gi} \leqslant P_{Gi}^{\max}, \quad i = 1, 2, \ldots, N. \end{cases} \tag{7}$$

2.4. Optimization objective

For the EED problem with two conflicting objective functions, relationship between any two of its feasible solutions has one of two possibilities: one dominates the other or none dominates the other. In particular, for feasible solutions $\vec{P}_G^1$ and $\vec{P}_G^2$, $\vec{P}_G^1$ is said to dominate $\vec{P}_G^2$ (denoted by $\vec{F}(\vec{P}_G^1) \prec \vec{F}(\vec{P}_G^2)$), if one of the following two conditions is satisfied:

(a) $FC(\vec{P}_G^1) \leqslant FC(\vec{P}_G^2)$ and $EM(\vec{P}_G^1) < EM(\vec{P}_G^2)$;

(b) $FC(\vec{P}_G^1) < FC(\vec{P}_G^2)$ and $EM(\vec{P}_G^1) \leqslant EM(\vec{P}_G^2)$.

A feasible solution $\vec{P}_G$ is said to be non-dominated with respect to set Ω, if there does not exist $\vec{P}'_G \in \Omega$ such that $\vec{F}(\vec{P}'_G) \prec \vec{F}(\vec{P}_G)$. Furthermore, the feasible solutions that are non-dominated within the entire search space are called the Pareto optimal solutions, which constitute the Pareto optimal set. Objective function values of these Pareto optimal solutions constitute the Pareto front of the EED problem. Unless there is some preference information, the main goal of the EED problem is to find a Pareto-optimal set, instead of a single optimal solution. The detailed discussion of these basic concepts can be found in [12].

3. Particle swarm optimization

The PSO is inspired by the social behavior of some biological organisms, especially the group's ability of some animal species to locate a desirable position in the given area. It was proposed first by Kennedy and Eberhart in 1995 [18]. In the PSO, a swarm consists of a set of particles; and each particle represents a potential solution of an optimization problem. Considering the ith particle in the swarm, its position and velocity at iteration t are denoted by $\vec{X}_i(t) = (x_{i,1}(t), x_{i,1}(t), \ldots, x_{i,n}(t))$ and $\vec{V}_i(t) = (v_{i,1}(t), v_{i,1}(t), \ldots, v_{i,n}(t))$. Then, the new position and velocity of this particle at iteration $t + 1$ will be calculated by using the following equations:

$$\begin{cases} v_{i,j}(t+1) = w * v_{i,j}(t) + r_1 c_1 * (Pb_{i,j}(t) - x_{i,j}(t)) \\ \qquad\qquad\quad + r_2 c_2 * (Gb_j(t) - x_{i,j}(t)), \\ x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1). \end{cases} \tag{8}$$

where $\vec{P}b_i(t) = (Pb_{i,1}(t), Pb_{i,2}(t), \ldots, Pb_{i,n}(t))$, called the local leader or the personal best position, represents the best position found by the ith particle itself so far; $\vec{G}b(t) = (Gb_1(t), Gb_2(t), \ldots, Gb_n(t))$, called the global leader or the global best position, represents the global best position found by neighbors of this particle so far; acceleration coefficients c_1 and c_2 are nonnegative constants which control the influence of $\vec{G}b(t)$ and $\vec{P}b_i(t)$ on the search process; and w is the inertia weight to control particle's exploration in the search space [31]. Note that a large inertial weight is helpful for global search performance while a small inertia weight facilitates a local search. r_1 and r_2 are two random numbers within [0,1].

The bare-bones particle swarm optimization (BBPSO) proposed by Kennedy [19] is a simple version of PSO. The BBPSO algorithm does not use the particle velocity, but uses a Gaussian sampling based on $\vec{G}b(t)$ and $\vec{P}b_i(t)$, and (8) is replaced by

$$x_{i,j}(t+1) = N\left(\frac{Pb_{i,j}(t) + Gb_j(t)}{2}, \left|Pb_{i,j}(t) - Gb_j(t)\right|\right). \tag{9}$$

In Eq. (9), the position of each particle is randomly selected from the Gaussian distribution with the average of the personal best position and the global best position. Recently, Pan et al. [28] demonstrated that the BBPSO can be deduced from the PSO. Kennedy [19] proposed also an alternative version of the BBPSO, denoted by BBExp, where (9) is replaced by

$$x_{i,j}(t+1) = \begin{cases} N\left(\frac{Pb_{i,j}(t)+Gb_j(t)}{2}, \left|Pb_{i,j}(t) - Gb_j(t)\right|\right), & U(0,1) < 0.5, \\ Pb_{i,j}(t), & \text{otherwise}. \end{cases} \tag{10}$$

Since there is 50% chance that the jth dimension of the particle changes to the corresponding personal best position, the BBExp inclines to search for personal best positions, as pointed out in [23].

Unlike the PSO, the BBPSO is parameter-free and is suitable for those real application problems where the information on parameters such as inertia weights and acceleration coefficients of particles is lacking or hard to obtain. Some successful applications of the BBPSO can be found in [22,24]. However, there has been very little discussion in the existing literature to use the BBPSO to solve the multi-objective optimization problems, in particular, the multi-objective EED problem.

4. Description of the BB-MOPSO algorithm

This section describes the proposed BB-MOPSO algorithm. The motivation for this algorithm is to design a parameter-free multi-objective optimization technique, which not only has a good performance on tackling the multi-objective optimization problems, but also is easy to implement. We use the pseudo-code to describe the BB-MOPSO algorithm. In the pseudo-code, //* and *// are used to provide comments and distinguish them from the code.

In Step 1 of the BB-MOPSO algorithm, the swarm is initialized in the search space; the personal best position of each particle is set as the particle itself; and the archive Ar_0 stores all the non-dominated particles with respect to the swarm. After that, the same iteration steps are run circularly to find the optimal set of the optimization problem, until the maximum iteration number $T_{\max}$ is reached. Within the iteration, each particle updates its personal best position and its global best position by using the functions ***GET_PBEST*** and ***GET_GBEST***. Based on the two best positions obtained, the position of each particle gets then an update by using the function ***UP_PARTICLE***. The function ***MUTATE*** showed in Step 2.2 is activated next to improve the diversity of particles, and the function ***EVALUATE***is used to calculate the fitness value of each new particle. In Step 2.4, the archive $Ar(t)$ is updated and pruned to store the best non-dominated solutions found up to the current iteration time. The algorithm returns then to Step 2.1 with the new loop counter $t + 1$, and the search ends when loop counter t reaches $T_{\max}$. Note that, in this algorithm, whenever the position of a new particle goes beyond its lower or upper bound, the particle will take the value of its corresponding lower or upper bound. Fig. 1 shows the flowchart of the proposed algorithm.

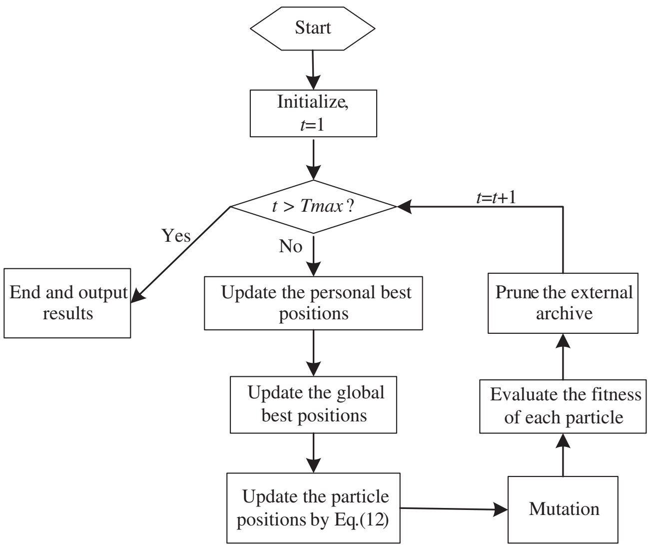

Fig. 1. Flowchart of the proposed algorithm.

Algorithm BB-MOPSO: Op = BB-MOPSO (Ns, Na, T_{max})
//∗ Ns: size of the swarm, Na : maximum capacity of the archive, T_{max}: maximum number of iterations∗//
1. $t = 0$, initialize the swarm S_0 //∗S_t and Ar_t: swarm and archive at iteration t , respectively∗//
   1.1 **FOR** $i = 1$ to Ns
      $INITIALIZE(\vec{X}_i(0))$ //∗Initialize the position of the ith particle∗//
      $\vec{P}b_i(0) \leftarrow \vec{X}_i(0)$ //∗ Initialize the personal best position of the ith particle∗//
   ENDFOR
   1.2 $F(S_0) \leftarrow EVALUATE(S_0)$ //∗ Evaluate the fitness of each particle in the swarm S_0∗
   1.3 $Ar_0 \leftarrow NON\_DOMINATED(S_0)$ //∗ Return the non-dominated solutions from the swarm S_0∗//
2. **WHILE** $t < T_{\max}$, **DO**
   2.1 **FOR** $i = 1$ to Ns //∗Update the position of each particle in S_t ∗//
      $\vec{P}b_i(t) \leftarrow GET\_PBEST()$ //∗ Return the personal best position of the ith particle∗//
      $\vec{G}b_i(t) \leftarrow GET\_GBEST()$ //∗ Return the global best position of the ith particle ∗//
      $\vec{X}_i(t+1) \leftarrow UP\_PARTICLE(\vec{G}b_i(t), \vec{P}b_i(t), \vec{X}_i(t))$ //∗ Update the ith particle using the improved BBPSO ∗//
   ENDFOR
   2.2 $S_{t+1} \leftarrow MUTATE(S_{t+1})$ //∗Implement the mutation operator on S_{t+1} ∗//
   2.3 $F(S_{t+1}) \leftarrow EVALUATE(S_{t+1})$ //∗ Return the fitness value of each new particle ∗//
   2.4 //∗ Update the archive, where $|Ar_{t+1}|$ isthe element number of Ar_{t+1}∗//

$$Ar_{t+1} \leftarrow NON\_DOMINATED(S_{t+1} \cup Ar_t)$$
$$\text{if}(|Ar_{t+1}| > Na), \quad \text{then} \quad PRUNE\_ARCHIVE(Ar_{t+1})$$

   2.5 $t \leftarrow t + 1$
   ENDWHILE
3. $Op \leftarrow Ar_t$ and stop the algorithm //∗Output the obtained Pareto optimal front ∗//

4.1. Initialization

In the initialization phase of the BB-MOPSO, the swarm with size Ns is randomly generated. Each particle in the swarm is assigned random values for each dimension from the respective domain. The initial value for the personal best position of each particle is set to be the particle itself, $\vec{P}b_i(0) = \vec{X}_i(0)$, where $\vec{X}_i(0)$ is the position of the ith particle in the swarm.

The BB-MOPSO maintains an archive for storing the non-dominated solutions found during the entire search process. In Step 1 of the BB-MOPSO, the archive is initialized to contain the non-dominated solutions from the swarm S_0. Function ***NON_DOMINATED*** returns the non-dominated solutions from the swarm S_0.

4.2. Update of the personal best positions

The personal best position (*Pbest*) is the best position achieved by the particle itself so far. If the current position of a particle is dominated by the position contained in its memory, then we keep the position in memory; otherwise, the current position of this particle replaces it. The update equation of *Pbest* is shown in Eq. (11). Function ***GET_PBEST*** returns the personal best position in Step 2.1 of the BB-MOPSO.

$$\vec{P}b_i(t+1) = \begin{cases} \vec{P}b_i(t), & \text{if } \vec{F}(\vec{P}b_i(t)) \prec \vec{F}(\vec{x}_i(t+1)), \\ \vec{x}_i(t+1), & \text{otherwise.} \end{cases} \quad (11)$$

4.3. Update of the global best positions

The global best position (*Gbest*) is the best solution obtained from neighbors of the particle so far. When solving the single-objective optimization problems, it is completely determined once a neighborhood topology is established. However, in the case of the multi-objective optimization problems, the conflicting nature of multiple objectives makes it difficult to choose a single optimal solution. To resolve this problem, the BB-MOPSO maintains an external archive with its maximum capacity to store the non-dominated solutions found (see Section 4.1). The *Gbest* of each particle is selected from the archive based on the diversity of non-dominated solutions. The concept of crowding distance is introduced in [14] to develop the multi-objective genetic algorithms. In the BB-MOPSO algorithm, we use the crowding distance to estimate the diversity of non-dominated solutions. For the example of two objective functions, Fig. 2 shows the calculation method for the crowding distances of solutions. In Fig. 2, solid dots represent solutions in the archive. The crowding distance of the ith solution is the average side length of the cuboid (a dashed rectangle in Fig. 2). Note that all of the boundary solutions with respect to each dimension of the objective space (solutions with smallest and largest objective function values) are assigned with an infinite distance value.

By using the method discussed above, the density (the crowding distance) of each solution in the archive is obtained at each iteration. Then the binary tournament with these crowding distances is utilized to obtain the *Gbest* for each particle from the archive. The higher crowding distance indicates a better chance to be selected as the *Gbest*. At Step 2.1 of the BB-MOPSO algorithm, Function ***GET_GBEST*** returns the updated *Gbest*s.

4.4. Update of the particle's positions

To deal with the multi-objective optimization problems, a variation of the BBExp, called the BBVar, is proposed here to update a particle's position:

$$x_{i,j}(t+1) = \begin{cases} N\left(\frac{r_3 Pb_{i,j}(t)+(1-r_3)Gb_{i,j}(t)}{2}, \left|Pb_{i,j}(t)-Gb_{i,j}(t)\right|\right), & U(0,1)<0.5, \\ Gb_{i,j}(t), & \text{otherwise}, \end{cases} \tag{12}$$

where r_3 is a random number within [0,1]. By comparing (12) with (10), the replacement of $Pb_{i,j}(t)$ with $Gb_{i,j}(t)$ will not diminish the particles' diversity, but provide a good scheme as fast as the crossover operator of evolution algorithms, since the archive contains the best non-dominated set found so far. Furthermore, when $U(0,1) < 0.5$, the stochastic weighted average of the *Pbest* and *Gbest*, actually widen the search domain of particles.

4.5. Mutation operator

The PSO is known to have a fast convergence speed. However, such convergence speed may be harmful in the case of multi-objective optimization, since a PSO-based algorithm may converge to a false Pareto front [13]. This has motivated the introduction of mutation operators in the literatures.

In the BB-MOPSO algorithm, we use Function ***MUTATE*** where mutation parameter α is set to control the decreasing speed of the mutation probability. Fig. 3 displays the mutation probability. At the beginning of the algorithm, all particles in the swarm S_t are affected by the mutation operator with the full range of decision variables (see Function ***MUTATE***). When the number of iterations increases, the effect of the mutation operator decreases, which is reflected by the fact that the mutation probability decreases and the mutation range shrinks. Function ***MUTATE*** is activated and returns to the swarm at Step 2.2 of the BB-MOPSO algorithm.

Function MUTATE: Op = *MUTATE* (S_t, *bound*, *n*)
//∗*bound*: matrix with regard to the lower or the upper boundaries of the decision variable,
n: dimension of the decision variable ∗//
  FOR *i*=1 to *Ns*   //∗ *Ns*: size of the swarm∗//
    IF $e^{(-\alpha*t/T_{max})} > r_4$ //∗ r_4: a random number within [0, 1]; α: mutation parameter ∗//
      dim = *rand*(1,*n*)   //∗ Pick a random integer between 1 to *n*∗//
        $range = [upper\_bound(dim) - low\_bound(dim)] * e^{(-\alpha*t/T_{max})}$
        $x_{i,dim} = x_{i,dim} + N(0,1) * range$ // $N(0,1)$ is a Gaussian distribution with 0 and 1∗//
      ENDIF
    ENDFOR
$Op \leftarrow S_t$ //Return the swarm after mutation ∗//

4.6. Pruning the external archive

In order to report a good optimal set at the end of the BB-MOPSO algorithm and to provide particles with the good global leaders, it is important to retain the non-dominated solutions found during the entire search process. Like most of the existing multi-objective evolutionary algorithms, the external archive with maximal capacity is adopted to retain those non-dominated solutions. When the archive has reached its maximal capacity, an approach based upon the crowding distance [20,32] is adopted to reduce the archive size without damaging its distribution characteristics. In particular, at each iteration, all of the non-dominated solutions from both the current swarm and the archive are stored first in the archive by Function ***NON_DOMINATED***. If the archive has reached its maximal capacity *Na*, then the most sparsely spread *Na* solutions, i.e., *Na* solutions with the largest crowding distance values, are retained in the archive.

5. Implementation of the BB-MOPSO

In this section, the proposed BB-MOPSO algorithm is applied to the standard IEEE 30-bus six-generator test system [4,16]. This power system is connected through 41 transmission lines, and the total system demand amounts to 2.834 P.U. The system parameters including fuel cost and emission coefficients are listed in Table 1. The *B*-coefficients [25,39] are shown.

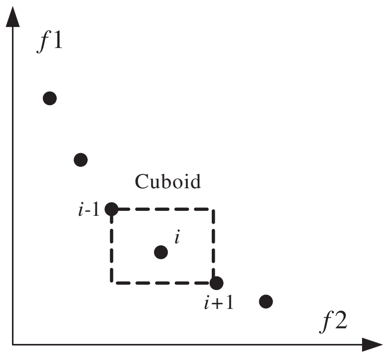

Fig. 2. Calculation of crowding distance.

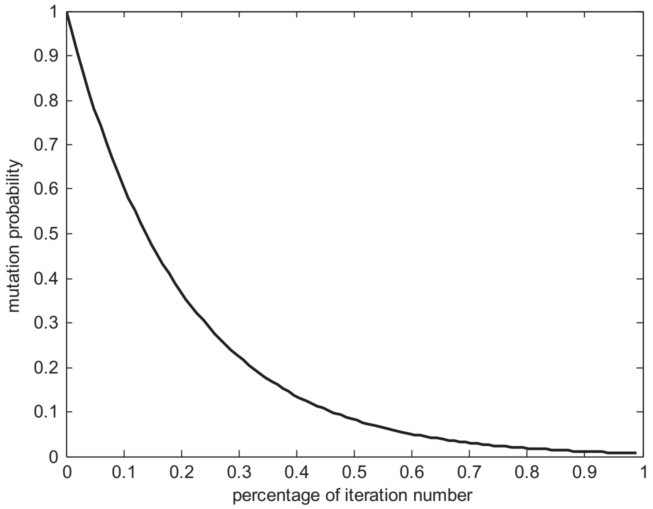

Fig. 3. Behavior of our mutation operator.

$$B = \begin{bmatrix} 0.1382 & -0.0299 & 0.0044 & -0.0022 & -0.0010 & -0.0008 \\ -0.0299 & 0.0487 & -0.0025 & 0.0004 & 0.0016 & 0.0041 \\ 0.0044 & -0.0025 & 0.0182 & -0.0070 & -0.0066 & -0.0066 \\ -0.0022 & 0.0004 & -0.0070 & 0.0137 & 0.0050 & 0.0033 \\ -0.0010 & 0.0016 & -0.0066 & 0.0050 & 0.0109 & 0.0005 \\ -0.0008 & 0.0041 & -0.0066 & 0.0033 & 0.0005 & 0.0244 \end{bmatrix},$$
$$B_0 = [-0.01070.0060 - 0.00170.00090.00020.0030],$$
$$B_{00} = 9.8573E - 4.$$

5.1. Encoding of particles

When the proposed algorithm is applied to the EED problem, the encoding of decision variables must be introduced first. In this paper, the power output of each generator is taken as the encoded element, and multiple elements comprise a particle representing a candidate solution of the EED problem. Thus, each particle consists of a 6-bit real coded string for optimizing the IEEE 30-bus six-generator test system, i.e.

$$\vec{P}_{Gi} = (P_{Gi,1}, P_{Gi,2}, \ldots, P_{Gi,6}), \quad i = 1, 2, \ldots, N_s,$$

where N_s is the swarm size, and $P_{Gi,j}$ is the generation power output of the ith generator.

Table 1
Generator cost and emission coefficients.

		G_1	G_2	G_3	G_4	G_5	G_6
Cost	a	10	10	20	10	20	10
	b	200	150	180	100	180	150
	c	100	120	40	60	40	100
Emission	α	4.091	2.543	4.258	5.326	4.258	6.131
	β	−5.554	−6.047	−5.094	−3.550	−5.094	−5.555
	γ	6.490	5.638	4.586	3.380	4.586	5.151
	ζ	2.0E−4	5.0E−4	1.0E−6	2.0E−3	1.0E−6	1.0E−5
	λ	2.857	3.333	8.000	2.000	8.000	6.667

5.2. Handling of the constraints

Since the BB-MOPSO algorithm is developed essentially for optimization problems without constraints, a constraint handling scheme needs to be introduced to deal with the constrained EED problem. A straightforward constraint checking approach, called the rejecting strategy, was proposed in [10,39,40] to deal with the constraints of the EED problem. However, our experiences indicate that this approach is time-consuming to produce the Pareto optimal solutions satisfying the equality constraints. In this paper, a technique that is suitable for equality constraints is proposed to deal with the power balance constraint.

Function ***HANDLE_CONS*** is developed to handle the constraints of the EED problem. For a particle $\vec{X}_i$, Function ***HANDLE_CONS*** checks first the feasibility of $\vec{X}_i$ by calculating the difference value *Dif* between $P_D + P_L$ and the sum of elements of $\vec{X}_i$. If the absolute value of *Dif* is greater than a small value ε (we set $\varepsilon = 10^{-10}$ in this paper), then the elements of X_i are adjusted in turn from its *k*th dimension until $\vec{X}_i$ satisfies this constraints, where *k* is a random integer within $[1, n]$. Step 3.2 of Function ***HANDLE_CONS*** truncates the position of $\vec{X}_i$ to satisfy the generation capacity constraints. By using Function ***HANDLE_CONS***, any unfeasible solution produced by the BB-MOPSO can be converted into a feasible one. This ensures the feasibility of the non-dominated solutions obtained.

Function *HANDLE CONSTRINTS*: $Op = HANDLE\_CONS(X_i, P_G^{\min}, P_G^{\max})$

//∗ X_i: a particle, $P_G^{\min}$and$P_G^{\max}$: matrixes with regard to the lower or the upper limits of generation capacity∗//

1. //∗Calculate and return the difference value between P_D+P_L and the element sum of X_i∗//

   $$Dif = P_L + P_D - sum(X_i)$$

2. $k = rand(1,n)$ //∗Pick a random integer between 1 to n∗//
3. WHILE ($|Dif| > \varepsilon$) // Check the feasibility of X_i∗//
   - 3.1 $x_{i,k} \leftarrow x_{i,k} + Dif$
   - 3.2 //∗ Make the position of X_i satisfy the generation capacity constraints ∗//
     - IF $x_{i,k} < P_G^{\min}$ THEN $x_{i,k} \leftarrow P_G^{\min}$
     - IF $x_{i,k} > P_G^{\max}$ THEN $x_{i,k} \leftarrow P_G^{\max}$
   - 3.3 $Dif = P_L + P_D - sum(X_i)$ // Calculate and return the difference value again ∗//
   - 3.4 $k = \mathrm{mod}(k, n) + 1$ //∗ Go to next dimension of variable, modulo n ∗//

   ENDWHILE
4. $Op \leftarrow X_i$ //Return the particle position after modified ∗//

5.3. Compromise solution

Due to the imprecision of judgments by the decision makers in real applications, a fuzzy membership function [33] is adopted to simulate the decision maker's preference and to identify the compromise solution from the Pareto optimal set obtained. In order to compare with the results in [2–5], the compromise solution is also identified at the end of our algorithm. Consider a non-dominated solution $\vec{X}_k$ in the archive*Ar*. The satisfactory degree of $\vec{X}_k$ for the *i*th objective function F_i is expressed by a membership function

$$\mu_i^k = \begin{cases} 1, & F_i(\vec{X}_k) \leqslant F_i^{\min}, \\ \frac{F_i^{\max} - F_i(\vec{X}_k)}{F_i^{\max} - F_i^{\min}}, & F_i^{\min} < F_i\left(\vec{X}_k\right) < F_i^{\max}, \\ 0, & F_i(\vec{X}_k) \geqslant F_i^{\max}, \end{cases} \tag{13}$$

where $F_i^{\max}$ and $F_i^{\min}$ are the maximum and minimum of the *i*th objective function F_i. Then, the normalized membership function μ^k of $\vec{X}_k$ is calculated by

$$\mu^k = \frac{\sum_{i=1}^{M} \mu_i^k}{\sum_{k=1}^{|Ar|} \sum_{i=1}^{M} \mu_i^k}, \tag{14}$$

where M is the objective number of the optimized problem, $|Ar|$ is the element number of the archive. The compromise solution is the one having the maximum of μ^k in the archive Ar.

5.4. Settings of the proposed approach

To demonstrate the effectiveness of the BB-MOPSO algorithm, two different cases have been considered.

- **Case 1:** In order to compare our algorithm to some of the existing algorithms, we consider the IEEE 30-bus six-generator test system without loss. In this case, the problem constraints are the generation capacity constraint and the power balance constraints without transmission loss.
- **Case 2:** In order to validate the constraint handling strategy and evaluate the performance of our algorithm, transmission losses are considered in this case.

In all of the simulation runs, the swarm size and the capacity of the archive are fixed at 50 particles. The number of fitness function evaluations is taken as the termination criteria of the proposed approach, which is restricted to 10,000 and 20,000 for Case 1 and Case 2, respectively. The mutation parameter α is set to be 8 for compromising the exploitation and exploration capabilities of our algorithm.

6. Performance for extreme points and compromise solutions

In this section, we present an experiment that is designed to evaluate the BB-MOPSO algorithm's performance by looking for the extreme points on the Pareto front of the EED problem and the compromise solutions of Case 1. Initially, fuel cost and emission objectives are optimized individually to obtain two extreme points of the problem. The existing results obtained by optimizing fuel cost and emission individually [4] are provided in Table 2. In order to compare the compromise solutions, the average satisfactory degree (ASD) of decision-maker with compromise solution is calculated based on the maximum and minimum of each objective function shown in Table 2. For a compromise solution $\vec{X}_{com}$, its ASD value is given by

$$\bar{\mu}_{com} = \frac{1}{M} \sum_{i=1}^{M} \mu_i^{com}. \tag{15}$$

Applying the BB-MOPSO algorithm to Case 1, the approximation to the true Pareto front is displayed in Fig. 4, which indicates clearly that the solutions found are well-distributed and covered almost the entire Pareto front of the problem. Tables 3 and 4 compare our solutions for the fuel cost and the pollutants emission with those results reported the literatures, which were obtained by using LP [16], MOSST [15], NSGA [3], NPGA [2], SPEA [4], NSGA-II [29] and FCPSO [5]. Table 5 compares our result for the compromise solution with those results reported the literatures, which were obtained by using NSGA [3], NPGA [2], SPEA [4], and FCPSO [5]. Where the bold values in the Tables 2–5 are the best results obtained for each case.

As shown in Tables 3 and 4, it is quite evident that the BB-MOPSO algorithm performs better than the LP and MOSST algorithms for the problem under discussion, since a reduction of more than 5 dollars per hour is observed with less level of pollutant emission. By comparing with the four evolutionary algorithms and the FCMOPSO, both the minimums of fuel cost and pollutant emission are improved slightly by using BB-MOPSO algorithm. Moreover, the BB-MOPSO algorithm requires only 10,000 fitness function evaluations to obtain a Pareto front including these best solutions in all trials, while the SPEA, NSGA and NPGA take about 100,000 fitness function evaluations, and the FCMOPSO takes also about 20,000 fitness function evaluations. Thus, running time of the BB-MOPSO algorithm to generate a good Pareto front is much less than those existing algorithms. In addition, the BB-MOPSO algorithm attains the best compromise solutions among five algorithms except LP and MOSST, as shown in Table 5. The above discussions indicate that the BB-MOPSO algorithm is more efficient than those existing algorithms.

7. Comparison of the multi-objective performance

In this section, we evaluate the multi-objective performance of the BB-MOPSO algorithm by considering Case 2. The algorithm was set to conduct 30 runs to collect the statistical results for all simulations. The best results obtained with respect to each adopted performance metric are shown in boldface in Tables 6–11.

7.1. Selected algorithms and parameter settings

Three well-known MOPSO algorithms are selected for the performance comparison, which are the multi-objective particle swarm optimization algorithm (CMOPSO) [13], the multi-objective particle swarm with the sigma method (SMOPSO) [21], and the time variant multi-objective particle swarm optimization (TV-MOPSO) [36]. Table 6 shows the parameter

Table 2
Best solutions for cost and emission optimized individually for Case 1.

	1	2	3	4	5	6	Fuel cost	Emission
Best cost	0.1095	0.2997	0.5245	1.0160	0.5247	0.3596	600.112	0.22214
Best emission	0.4058	0.4592	0.5380	0.3830	0.537	0.5101	638.260	0.194203

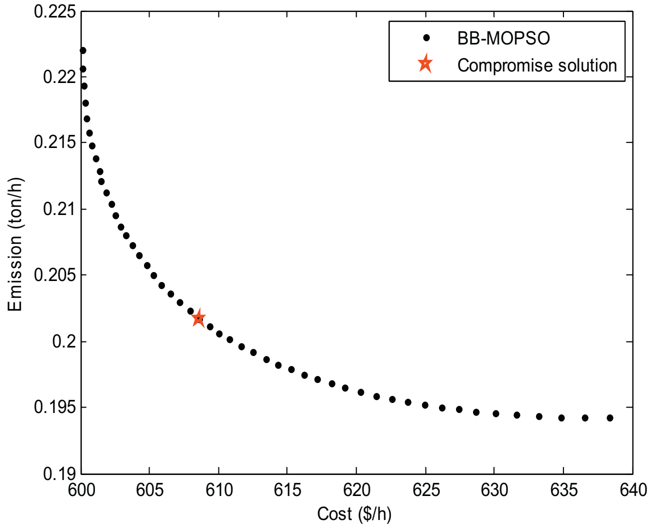

Fig. 4. Pareto front obtained by the BB-MOPSO for Case 1.

Table 3
Best solutions for cost with eight algorithms on optimizing Case 1.

	P_{G1}	P_{G2}	P_{G3}	P_{G4}	P_{G5}	P_{G6}	Fuel cost	Emission
BB-MOPSO	**0.109**	**0.3005**	**0.5234**	**1.017**	**0.5238**	**0.3603**	**600.112**	**0.22220**
LP	0.1500	0.3000	0.5500	1.0500	0.4600	0.3500	606.314	0.22330
MOSST	0.1125	0.3020	0.3020	1.0208	0.5311	0.3625	605.889	0.22220
NSGA	0.1567	0.2870	0.4671	1.0467	0.5037	0.3729	600.572	0.22282
NPGA	0.1080	0.3284	0.5386	1.0067	0.4949	0.3574	600.259	0.22116
SPEA	0.1062	0.2897	0.5289	1.0025	0.5402	0.3664	600.150	0.22151
NSGA-II	0.1059	0.3177	0.5216	1.0146	0.5159	0.3583	600.155	0.22188
FCPSO	0.1070	0.2897	0.525	1.015	0.5300	0.3673	600.132	0.22226

Table 4
Best solutions for emission with eight algorithms on optimizing Case 1.

	P_{G1}	P_{G2}	P_{G3}	P_{G4}	P_{G5}	P_{G6}	Fuel cost	Emission
BB-MOPSO	0.4071	0.4591	0.5374	0.3838	0.5369	0.5098	0.194203	638.262
LP	0.4000	0.4500	0.5500	0.4000	0.5500	0.5000	0.194227	639.600
MOSST	**0.4095**	**0.4626**	**0.5426**	**0.3884**	**0.5427**	**0.5152**	**0.194182**	**644.112**
NSGA	0.4394	0.4511	0.5105	0.3871	0.5553	0.4905	0.194356	639.209
NPGA	0.4002	0.4474	0.5166	0.3688	0.5751	0.5259	0.194327	639.180
SPEA	0.4116	0.4532	0.5329	0.3832	0.5383	0.5148	0.194210	638.507
NSGA-II	0.4074	0.4577	0.5389	0.3837	0.5352	0.5110	0.194204	638.249
FCPSO	0.4097	0.4550	0.5363	0.3842	0.5348	0.5140	0.194207	638.358

Table 5
Best solutions for compromise solution with five algorithms on optimizing Case 1.

	P_{G1}	P_{G2}	P_{G3}	P_{G4}	P_{G5}	P_{G6}	Fuel cost	Emission	ASD
BB-MOPSO	**0.2595**	**0.3698**	**0.5351**	**0.6919**	**0.5500**	**0.4277**	**609.747**	**0.20083**	**0.7555**
NSGA	0.2571	0.3774	0.5381	0.6872	0.5404	0.4337	610.067	0.20060	0.7551
NPGA	0.2696	0.3673	0.5594	0.6496	0.5396	0.4486	612.127	0.19941	0.7491
SPEA	0.2785	0.3764	0.5300	0.6931	0.5406	0.4153	610.254	0.20055	0.7527
FCPSO	0.3193	0.3934	0.5359	0.5921	0.5457	0.447	619.998	0.19715	0.7267

Table 6
Parameter configurations for selected algorithms.

	Population size	Archive size	No. of fitness evaluations	Other parameters
MOPSO	50	50	20,000	Thirty divisions for adaptive grid; mutation probability = 0.5; w=0.4
SMOPSO				Turbulence factor R_T is a random value in [0,1]; probability of adding the turbulence factor is 0.05; w = 0.4
TV-MOPSO				$w = (0.7 - 0.4) * (tmax - t)/tmax + 0.4$; $c_1 = (0.5 - 2.5) * t/tmax + 2.5$; $c_2 = (2.5 - 0.5) * t/tmax + 0.5$; mutation parameter $b = 5$

Table 7
Statistical results of the metric SP for Case 2.

	BB-MOPSO	SMOPSO	CMOPSO	TV-MOPSO
Best	**0.0032**	0.0047	0.0105	0.0095
Worst	**0.0092**	0.0237	0.023	0.0319
Median	**0.0052**	0.0076	0.0132	0.0164
Average	**0.0057**	0.0093	0.0143	0.0179
Std.	**0.0012**	0.0045	0.0032	0.0065

Table 8
Statistical results of the normalized distance metric for Case 2.

	BB-MOPSO	SMOPSO	CMOPSO	TV-MOPSO
Best	0.8955	0.904	0.9138	**0.9656**
Worst	**0.8800**	0.8219	0.8548	0.7986
Median	**0.887**	0.8571	0.8799	0.8448
Average	**0.8881**	0.8619	0.8801	0.8653
Std.	**0.0051**	0.0262	0.0207	0.0571

Table 9
Best solutions out of 30 trials for cost with four algorithms on optimizing Case 2.

	1	2	3	4	5	6	Fuel cost	Emission	Loss
SMOPSO	0.1225	0.2899	0.5741	0.9932	0.5255	0.3547	606.0038	0.220522	0.02583
CMOPSO	0.1198	0.2928	0.5778	0.99	0.527	0.3524	606.0062	0.220414	0.02576
TV-MOPSO	0.1011	0.2883	0.5852	0.9832	0.5271	0.3749	606.1114	0.220503	0.02587
BB-MOPSO	**0.1229**	**0.288**	**0.5792**	**0.9875**	**0.5255**	**0.3564**	**605.9817**	**0.220190**	**0.02562**

Table 10
Best solutions out of 30 trials for emission with four algorithms on optimizing Case 2.

	1	2	3	4	5	6	Fuel cost	Emission	Loss
SMOPSO	0.4078	0.4824	0.5388	0.3977	0.5335	0.5093	646.0817	0.194216	0.03549
CMOPSO	0.4097	0.4648	0.5523	0.394	0.5361	0.5123	645.7762	0.194186	0.03515
TV-MOPSO	0.4188	0.4582	0.553	0.3803	0.5345	0.5251	647.665	0.194203	0.03583
BB-MOPSO	**0.4103**	**0.4661**	**0.5432**	**0.3883**	**0.5447**	**0.5168**	**646.4847**	**0.194179**	**0.03537**

Table 11
Statistical results of the metric SC for Case 2.

	BB-MOPSO	SMOPSO	CMOPSO	TV-MOPSO
SC (BB-MOPSO,∗)	**–**	0.4125	0.3148	0.6629
SC (SMOPSO,∗)	**0.0471**	–	0.2037	0.4802
SC (CMOPSO,∗)	**0.0628**	0.2243	–	0.5867
SC (TV-MOPSO,∗)	**0**	0.1255	0.0926	–

configurations for these three algorithms according to [13]. To deal with the constraints, the strategy proposed in Section 5.2 is incorporated into the three MOPSO algorithms.

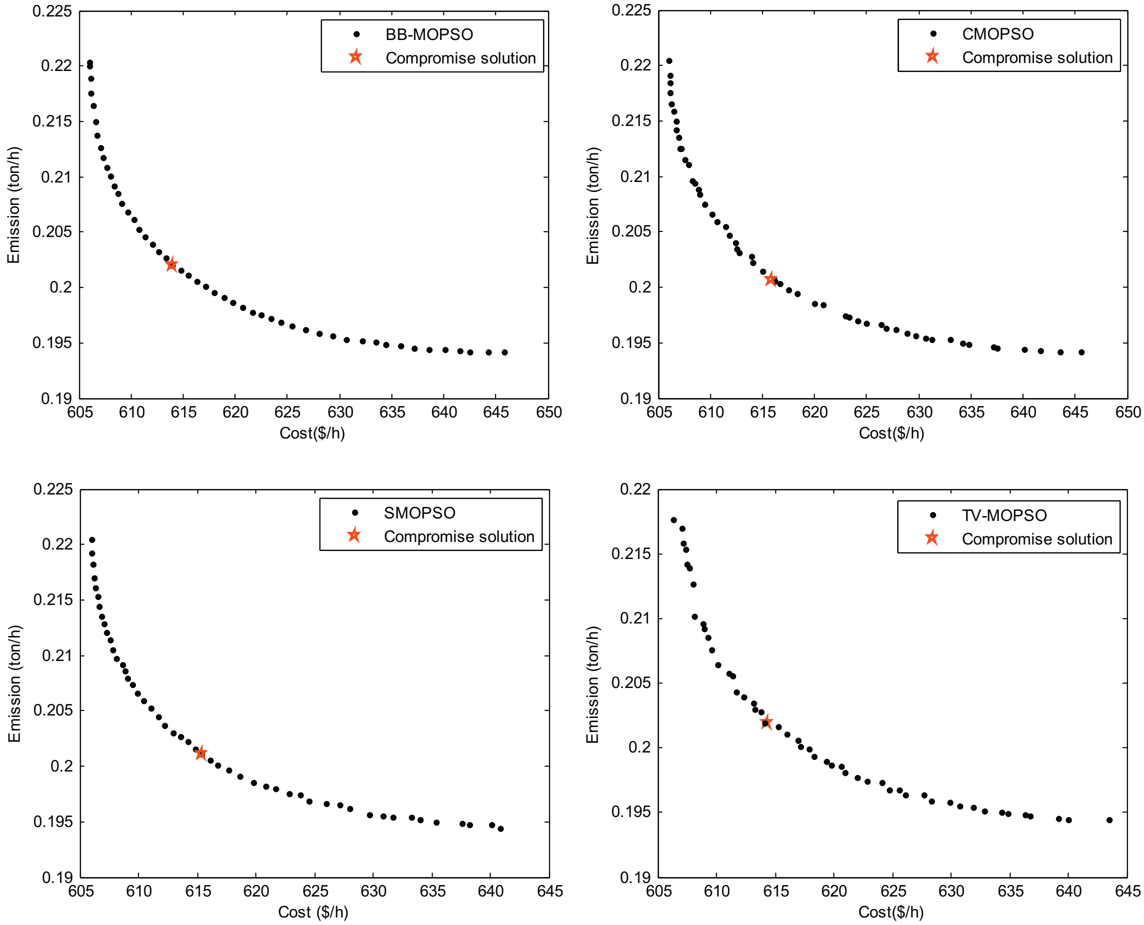

Fig. 5. Pareto fronts and compromise solutions produced by the four algorithms for Case 2.

7.2. Comparison results and analysis

Unlike single-objective optimization problems, the following criteria are considered in order to evaluate the solution quality for multi-objective optimization problems [41]:

- It is desirable to have a good distribution of the solutions found (uniform in most cases).
- The extent of the obtained Pareto optimal solutions should be maximized, which means, for each objective function, a wide range of values should be covered by the Pareto optimal solutions.
- The distance of the obtained Pareto optimal front to the true Pareto-optimal front should be minimized.

In this section, the results of different algorithms are compared in terms of the above three criteria.

To evaluate the distribution of solutions throughout the Pareto optimal set found, *the spacing metric (SP)* [34] is adopted. A value of zero for *SP* metric indicates that all members of the obtained Pareto front currently available are equidistantly spaced. Table 7 shows the comparison results among the four algorithms in terms of the *SP* metric. It can be seen that

Table 12
Statistical results of the three particular solutions for Case 1.

	Fuel cost	Emission	ASD values for compromise solution
Best	600.1117	0.194204	0.7552
Worst	600.1263	0.194220	0.7535
Median	600.1176	0.194207	0.7544
Average	600.1182	0.194208	0.7544
Std.	0.0043	5.12×10^{6}	0.0006

Table 13
Statistical results of the three particular solutions for Case 2.

	Fuel cost	Emission	ASD values for compromise solution
Best	605.9815	0.194179	0.7485
Worst	606.0065	0.194182	0.7470
Median	605.9942	0.194180	0.7481
Average	605.9944	0.194180	0.7479
Std.	0.0069	9.0×10^{-7}	0.0005

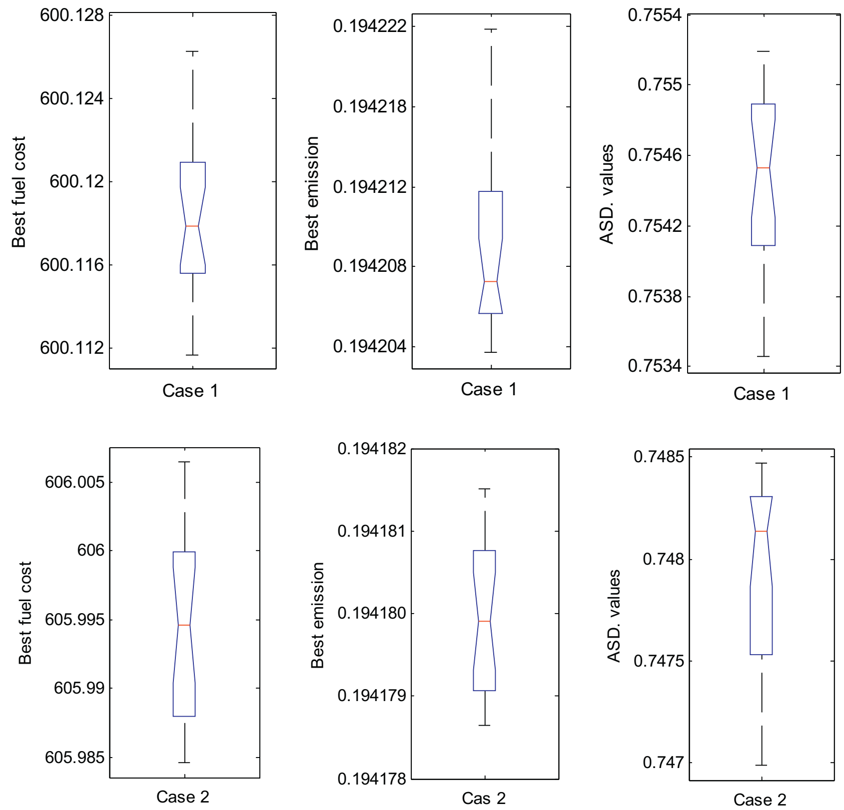

Fig. 6. Results obtained by the BB-MOPSO in 30 trials for both cases.

the BB-MOPSO algorithm has not only the best average performance in terms of the *SP*, but also the smallest standard deviation. In order to demonstrate further the distribution of solutions on the obtained Pareto front, Fig. 5 displays the graphical results produced by the BB-MOPSO algorithm and other three algorithms. It is evident from Fig. 5 that the distribution of CMOPSO and TV-MOPSO are not as good as that of BB-MOPSO and SMOPSO.

A performance measure of the extent of the Pareto optimal solutions was introduced in [41] and used in [4,5]. This measure estimates the range which the Pareto optimal fronts spread out. In other words, it measures the normalized distance of the two extreme solutions, i.e., the best cost solution and the best emission solution, on the Pareto optimal front. The statistical results of this measure are given in Table 8. These results indicate that the average performance of the BB-MOPSO algorithm is the best in terms of the extent of the Pareto optimal solutions, even though the TV-MOPSO finds the largest normalized distance value. The BB-MOPSO has also shown the best robustness in terms of this measure as reflected by its small standard deviation. In addition, Tables 9 and 10 show the smallest values out of 30 trials for the two extreme points obtained by four algorithms, respectively. It can be seen that the BB-MOPSO generates the best values in terms of both extreme points. It confirms further that the BB-MOPSO algorithm has a better performance than these of three existing algorithms..

In order to evaluate the closeness of the obtained Pareto front to the true Pareto front which is unknown in advance, the *two-set coverage* (*SC*) [41] is adopted. For given algorithms **A1** and **A2**, the value *SC* (**A1**,**A2**) = 1 represents that all solutions of **A2** are dominated by or equal to some solutions of **A1** (**A1** is said to cover **A2**), and indicates that the Pareto optimal front obtained by **A1** is closer to the true Pareto optimal front than that obtained by **A2**. Table 11 shows the comparison results among the four algorithms in terms of *SC*. It can be seen from this Table that at the worst case, near 7% solutions obtained by the BB-MOPSO algorithm are dominated by those of CMOPSO. However, the solutions obtained by the BB-MOPSO dominate by more than 30% the solutions obtained by SMOPSO, CMOPSO and TV-MOPSO. Thus, the BB-MOPSO is better than the three algorithms in terms of the convergence performance.

Therefore, it can be concluded from the above analysis that the BB-MOPSO algorithm has a better performance for Case 2 in terms of three aspects, convergence, extent and distribution of solutions, which can be attributed to collective efforts of the new techniques proposed in Section 4. On one hand, the update method of *Gbest* based on the crowding distances is able to assign promising elements, which locate at places with sparse solutions, as the Gbests of particles, while the BBVar method proposed is able to make particles exploit neighborhoods of those promising elements further. Hence their collective efforts guarantee the extent and distribution of solutions obtained by our algorithm. On the other hand, due to highly explorative behavior of the mutation operator proposed, the swarm is able to escape the local Pareto front at the first phase of our algorithm. This guarantees that our algorithm has a better convergence performance.

8. Robustness analysis

In order to check the robustness of the BB-MOPSO algorithm for solving the multi-objective optimization problems, 30 trials are performed to observe changes in solutions of three objectives: the minimal fuel cost, the minimal emission and the compromise solution. Note that, the differences between ASD values are used to describe changes in the compromise solutions, even though it is possible that different compromise solutions have the same ASD value.

For Case 1 and Case 2, Tables 12 and 13 show the statistical results with respect to solutions of the three objectives. Fig. 6 displays the distribution outline of the objective solutions obtained by 30 trials. For each Case, it is clear from Fig. 6 that the solution of each trial always remains close to the best solution (which can be seen from the extent of these plots) for the three objectives involved. As indicated by Tables 12 and 13, the standard deviations of solutions in terms of the three objectives are 0.0043, 5.12×10^{-6}, and 0.0006, respectively, for Case 1; the standard deviations of solutions in terms of the three objectives are 0.0069, 9.0×10^{-7}, and 0.0005, respectively, for Case 2. This indicates that the BB-MOPSO algorithm has the strong robustness to initial swarms for handling the IEEE 30-bus six-generator test system.

9. Conclusion

In this paper, a new MOPSO algorithm, called the BB-MOPSO algorithm, is proposed and applied successfully to solve the multi-objective EED problem with constraints. The algorithm extends the idea of bare-bones particle swarm optimization to the multi-objective optimization problems. It does not require fine tuning on control parameters such as inertia weight and acceleration coefficients in order to achieve a good performance of the algorithm. Moreover, the mutation operator and the constraint handling strategy proposed in this paper, together with several established techniques such as the external repository of elite particles, the crowding distance and the fuzzy membership function, have made the BB-MOPSO algorithm more effective in dealing with multi-objective optimization problems.

By comparing with those existing results obtained by seven established algorithms and three well-known MOPSO techniques, the solutions obtained by the BB-MOPSO algorithm exhibit certain superior characteristics. It seems that the BB-MOPSO algorithm is a viable alternative for solving the EED problem. We plan to apply the BB-MOPSO algorithm to some other multi-objective power system optimization problems and to explore its potentials.

Acknowledgment

This research is partially supported by the National Natural Science Foundation of China (No. 61005089) and the Specialized Research Fund for the Doctoral Program of Higher Education (No. 20100095120016).

References

[1] M.A. Abido, Multi-objective particle swarm optimization for environmental/economic dispatch problem, Electric Power Systems Research 79 (7) (2009) 1105–1113.
[2] M.A. Abido, A niched Pareto genetic algorithm for multiobjective environmental/economic dispatch, Electrical Power and Energy Systems 25 (2) (2003) 97–99.
[3] M.A. Abido, A novel multiobjective evolutionary algorithm for environmental/ economic power dispatch, Electric Power Systems Research 65 (2003) 71–91.
[4] M.A. Abido, Multiobjective evolutionary algorithms for electric power dispatch problem, IEEE Transactions on Evolutionary Computation 10 (3) (2006) 315–329.
[5] S. Agrawal, B.K. Panigrahi, M.K. Tiwari, Multiobjective particle swarm algorithm with fuzzy clustering for electrical power dispatch, IEEE Transactions on Evolutionary Computation 12 (5) (2008) 529–541.
[6] S.F. Brodesky, R.W. Hahn, Assessing the influence of power pools on emission constrained economic dispatch, IEEE Transactions on Power Systems PWRS 1 (1) (1986) 57–62.
[7] M. Basu, Dynamic economic emission dispatch using nondominated sorting genetic algorithm – II, Electrical Power and Energy Systems 30 (2) (2008) 140–149.
[8] C. Chang, K. Wong, B. Fan, Security-constrained multiobjective generation dispatch using bicriterion global optimization, IEE Proceedings – Generation, Transmission and Distribution 142 (4) (1995) 406–414.

[9] K.T. Chaturvedi, M. Pandit, L. Srivastava, Self-organizing hierarchical particle swarm optimization for nonconvex economic dispatch, IEEE Transactions on Power Systems 23 (3) (2008) 1079–1087.

[10] J. Cai, X. Ma, Q. Li, L. Li, H. Peng, A multi-objective chaotic particle swarm optimization for environmental/economic dispatch, Energy Conversion and Management 50 (5) (2009) 1318–1325.

[11] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 58–72.

[12] C.A. Coello Coello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary Algorithms for Solving Multi-objective Problems, Kluwer, 2002.

[13] C.A. Coello Coello, G.T. Pulido, M.S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 256–279.

[14] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[15] D.B. Das, C. Patvardhan, New multi-objective stochastic search technique for economic load dispatch, IEE Proceedings – Generation, Transmission and Distribution 145 (6) (1998) 747–752.

[16] A. Farag, S. Al-Baiyat, T.C. Cheng, Economic load dispatch multiobjective optimization procedures using linear programming techniques, IEEE Transactions on Power Systems 10 (2) (1995) 731–738.

[17] G.P. Granelli, M. Montagna, G.L. Pasini, P. Marannino, Emission constrained dynamic dispatch, Electric Power Systems Research 24 (1992) 56–64.

[18] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, 1995, pp. 1942–1948.

[19] J. Kennedy, Bare bones particle swarms, in: Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003, pp. 80–87.

[20] X.D. Li, A non-dominated sorting particle swarm optimizer for multi-objective optimization, in: Genetic and Evolutionary Computation Conference (GECCO), 2003, pp. 37–48.

[21] S. Mostaghim, J. Teich, Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO), in: Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003, pp. 26–33.

[22] M.G.H. Omran, A. Engelbrecht, A. Salman, Barebones particle swarm for integer programming problems, in: Proceedings of the 2007 IEEE Swarm Intelligence Symposium, 2007, pp. 170–175.

[23] M.G.H. Omran, A. Engelbrecht, A. Salman, Bare bones differential evolution, European Journal of Operational Research 196 (1) (2009) 128–139.

[24] M. Omran, S. Al-Sharhan, Barebones particle swarm methods for unsupervised image classification, in: Proceedings of IEEE Congress on Evolutionary Computation, 2007, pp. 3247–3252.

[25] R.E. Perez-Guerrero, J.R. Cedefio-Maldonado, Differential evolution based economic environmental power dispatch, in: Proceedings of the 37th Annual North American Power Symposium,2005, pp. 191–197.

[26] C. Palanichamy, N.S. Babu, Analytical solution for combined economic and emissions dispatch, Electric Power Systems Research 78 (7) (2008) 1129–1139.

[27] Y. Wang, B. Li, T. Weise, J.Y. Wang, et al, Self-adaptive learning based particle swarm optimization, Information Sciences 181 (2011) 4515–4538.

[28] F. Pan, X. Hu, R. Eberhart, Y. Chen, An analysis of bare bones particle swarm, in: Proceeding of the 2008 IEEE Swarm Intelligence Symposium, 2008, pp. 21–23.

[29] T.F. Robert, A.H. King, C.S. Harry, Rughooputh, K. Deb, Evolutionary multi-objective environmental/economic dispatch: stochastic versus deterministic approaches, KanGAL Rep. 2004019, 2004, pp. 1–15.

[30] H. Saadat, Power System Analysis, The McGraw-Hill Companies, 1999.

[31] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of the IEEE Congress on Evolutionary Computation, 1998, pp. 303–308.

[32] M.R. Sierra, C.A. Coello Coello, Improving PSO-based multiobjective optimization using crowding, mutation and ε-dominance, Proceeding of the Evolutionary Multi-Criterion Optimization (2005) 505–519.

[33] M. Sakawa, H. Yano, T. Yumine, An interactive fuzzy satisfying method for multiobjective linear programming problems and its application, IEEE Transactions on Systems, Man and Cybernetics SMC 17 (4) (1987) 654–661.

[34] J.R. Schott, Fault tolerant design using single and multicriteria genetic algorithm optimization, M.S. Thesis, Dept. Aeronautics and Astronautics, Massachusetts Inst. Technol., Cambridge, MA, 1995.

[35] J.H. Talaq, Ferial, M.E. EI-Hawary, A summary of environmental/economic dispatch algorithms, IEEE Transactions on Power Systems 9 (3) (1994) 1508–1516.

[36] P.K. Tripathi, S. Bandyopadhyay, S.K. Pal, Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients, Information Sciences 177 (22) (2007) 5033–5049.

[37] T.A.A. Victoire, P.N. Suganthan, Improved MOCLPSO algorithm for environmental/economic dispatch, Proceeding of IEEE Congress on Evolutionary Computation (2007) 3072–3076.

[38] F. Van den Bergh, A. Engelbrecht, A study of particle swarm optimization particle trajectories, Information Sciences 176 (8) (2006) 937–971.

[39] L.F. Wang, C. Singh, Balancing risk and cost in fuzzy economic dispatch including wind power penetration based on particle swarm optimization, Electric Power Systems Research 78 (8) (2008) 1361–1368.

[40] L.F. Wang, C. Singh, Environmental/economic power dispatch using a fuzzified multi-objective particle swarm optimization algorithm, Electric Power Systems Research 77 (12) (2007) 1654–1664.

[41] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evolutionary Computation 8 (2) (2000) 173–195.

[42] M. Amita, P. Vishnu, R. Saroj, Economic dispatch using particle swarm optimization: a review, Renewable and Sustainable Energy Reviews 13 (8) (2009) 2134–2141.

[43] V. Vahidinasab, S. Jadid, Joint economic and emission dispatch in energy markets: a multiobjective mathematical programming, Energy 35 (3) (2010) 1497–1504.

[44] A. Bhattacharya, P.K. Chattopadhyay, Application of biogeography-based optimization for solving multi-objective economic emission load dispatch problems, Electric Power Components and Systems 38 (3) (2010) 340–365.

[45] Y.M. Chen, W.S. Wang, Environmentally constrained economic dispatch using Pareto archive particle swarm optimization, International Journal of Systems Science 41 (5) (2010) 593–605.

[46] P. Chakraborty, S. Das, G.G. Roy, A. Abraham, On convergence of the multi-objective particle swarm optimizers, Information Sciences 181 (2011) 1411–1425.