Egyptian Computer Science Journal Vol. 39 No. 2 May 2015 ISSN-1110-2586

A Memetic Backtracking Search Optimization Algorithm for Economic Dispatch Problem

Ahmed Fouad Ali

Computer Science Department, Faculty of Computers and Information
Suez Canal University, Ismailia, Egypt.

ahmed_fouad@ci.suez.edu.eg

Abstract

Economic load dispatch (ELD) is a non-linear optimization problem for determining the power shared among the generating units to satisfy the generation limit constraints of each unit and minimizing the cost of power production. In this paper, we present a new hybrid algorithm by combing the standard backtracking search optimization algorithm (BSA) and the random wake with direction exploitation algorithm (RWDE). The proposed algorithm is called a memetic backtracking search optimization algorithm (MBSOA). Invoking the RWDE algorithm as a local search algorithm in the standard BSA can accelerate the convergence and refine the best obtain solution at each iteration in the proposed algorithm. The general performance of the proposed MBSOA is tested on a six-generator test system for a total demand of 700MW and 800MW and compared against three natural inspired algorithms. The experimental results show that the proposed algorithm is a promising algorithm to solve economic load dispatch problem.

Keyboard: *Backtracking search optimization, Random walk algorithm, Economic dispatch problem, Optimization problems*.

1. Introduction

Meta-heuristics methods can be classified into two classes, population-based methods and point to point methods. In the population-based methods, the initial population is generated randomly and the replacement phase is started by selecting a new population from the previous population and some operators are applied in order to generate a new offspring. The overall process is iterated until stopping criteria satisfied. Some of the population based methods are called nature-inspired methods or swarm intelligence methods such as evolutionary algorithms (EAs) [17], genetic algorithm (GA)[17], deferential evolutionary (DE) [33], particle swarm optimization (PSO) [19], firefly algorithm (FA) [43], [44], bat algorithm (BA) [45], bee colony optimization (BCO) [36], wolf search [35], bacterial foraging [20], cat swarm [8], cuckoo search [44], fish swarm/school [24], ant colony optimization (ACO)[11], group search optimizer (GSO)[16], artificial immune system (AIS) [12]. The second meta-heuristics class is called point to point methods. In the point to point methods the neighborhood solutions are generated from the current single solution in iterative way until termination criteria satisfied. The most popular examples of point to point methods are tabu search (TS) [15], simulated annealing (SA)[21], iterated local search (ILS), guided local search (GLS) [37], [38], [39], variable neighborhood search (VNS), greedy randomized adaptive search procedure (GRASP) .

Evolutionary algorithms (EAs) are the most commonly used population-based meta-heuristics methods. They are flexible to solve global optimization problems because they have a good abilities to perform a global exploration and a local exploitation [18].

Backtracking search optimization algorithm (BSA) is a recent population-based evolutionary algorithm proposed by P. Civicioglu [9] for solving numerical optimization problems. BSA is different from other evolutionary algorithms because it has a single control parameter and it generates a trail population, which enables it to solve numerical optimization problems rapidly.

Although the efficiency of the BSA when applied to solve many of optimization problems and real life problems it likes most of the meta-heuristics methods that they suffer from the slow convergence and they wander around the optimal solution when high accuracy is needed.

In this paper, we propose a new hybrid algorithm by combining the backtracking search optimization algorithm and the random walk with direction exploitation algorithm (RWDE) in order to solve economic load dispatch problem (ELD). The proposed algorithm is called a memetic backtracking search optimization algorithm (MBSOA). The RWDE algorithm is used in the MBSOA as a local search algorithm in order to refine the best-obtained solution at each iteration in the proposed algorithm. The proposed MBSOA is tested on a six-generator test system for a total demand of 700MW and 800MW and compared against three natural inspired algorithms.

The rest of this paper is organized as follow. Some of the recent related works are presented in Section 2. The formulation of the economic load dispatch problem is highlighted in Section 3. In Section 4, we give an overview of the backtracking search optimization algorithm (BSA) and its main processes. The proposed algorithm is presented in Section 5. The numerical results are described in Section 6. The conclusion of the paper makes up Section 7.

2. Related work

The economic load dispatch problem (ELD) problem can be mathematically formulated as a continuous optimization problem. In the last decade, many researchers have applied their algorithms to solve the global optimization problems and its applications. These algorithms are natural inspired based algorithms such as genetic algorithm [34], bat algorithm [1], [2], [4], [22], [26], [32], particle swarm optimization [3], [7], [10], [23], [32], [46], firefly algorithm [5], [42], cuckoo search [40], [41], Bacterial Foraging Algorithm [14], [20].

Due to the efficiency of the natural-inspired algorithms, they have been applied to solve economic load dispatch problem (ED) problem such as many variant of genetic algorithm with good results to solve non-convex ELD problems [1], [25]. GA has main advantage over other algorithms that it can use a chromosome representation technique tailored to the specific problem. However, it suffers from the slow convergence and the execution time is very long.

PSO with different variants have been applied for solving non-convex ELD problems [6], [13], [30], [31]. The main advantage of the PSO is it easy to perform and it has a few adjustable parameters. In addition, PSO is very efficient in exploring the search space (diversification). The main drawbacks of PSO are the slow convergence of it at refined search stage and its weak local search ability (intensifications).

Bat algorithm (BA) and its variants have also been used to solve the ELD problem [27], [29], [32]. BA has a big advantage over other algorithms that it has a number of tunable parameters giving a greater control over the optimization process. BA is a promising algorithm when applied for solving lower dimensional optimization problem but it becomes ineffective for high dimensional problems because of fast initial convergence [28].

Although the efficiency of these algorithms when they applied to solve ELD, they suffer from the slow convergence. In this paper, we propose a new algorithm to overcome the slow convergence of the meta-heuristics algorithms by invoking the RWDE algorithm as a local search algorithm in the proposed algorithm in order to accelerate the convergence and refine the best-obtained solution at each iteration.

3. Economic Dispatch Load Problem

The main objective of economic load dispatch problem (ELD) problem is to find the optimal combination of power generation in such a way that the total production cost of the system is minimized. The basic concepts of the ELD problem are highlighted in the following subsections.

3.1 Problem Objective Function

The cost function of a generating unit can be represented as a quadratic function with a sine component. The sine component denotes the effect of steam valve operation. The quadratic refers to a convex objective whereas the valve-point effect makes the problem non-convex. The convex and non-convex cost function of a generator can be expressed as follow.

$$F_i(P_i) = a_i P_i^2 + b_i P_i + c_i \tag{1}$$

$$F_i(P_i) = a_i P_i^2 + b_i P_i + c_i + \left|d_i \ \sin\left[e_i * \left(P_i^{min} - P_i\right)\right]\right| \tag{2}$$

Where P_i is the active power output, $F_i(P_i)$ is the generation cost, P_i^{min} is the minimum output limit of the ith generator and the a_i, b_i, c_i, d_i, e_i are the cost coefficients of the generator. The fuel cost of all generators can be defined by the following equation

$$Min\ F_i(P_i) = \sum_{i=1}^{Ng} a_i\, P_i^2 + b_i P_i + c_i + \left|d_i \sin\left[e_i * \left(P_i^{min} - P_i\right)\right]\right| \tag{3}$$

Where N_g is the number of generating units.

3.2 Problem Constraints

Power balance constraint: the total power generated should cover the power demand and the active power losses

$$\sum_{i=1}^{Ng} P_i = P_D + P_{loss} \tag{4}$$

Where P_D is the total demand load and P_{loss} is the total transmission losses computed using the quadratic approximation as follow.

$$P_{loss} = \sum_{i=1}^{Ng}\sum_{j=1}^{Ng} P_i\, B_{ij} P_j \qquad (5)$$

Where B_{ij} is the loss coefficient matrix.

Power generation limits. The real output power of each generator should be within a lower and an upper limit as follow.

$$P_i^{min} \le \; P_i \le P_i^{max} \qquad (6)$$

Where $P_i^{min}$and P_i^{max} are the lower and the upper limit of the ith generators.

3.3 Penalty Function

The penalty function technique is used to transform the constrained optimization problems to unconstrained optimization problem by penalizing the constraints and forming a new objective function as follow.

$$F(x) = \begin{cases} F(x), & If\ x\ \in feasible\ region \\ F(x) + penalty, & If\ x\ \notin\ feasible\ region \end{cases} \qquad (7)$$

Where

$$penalty(x) = \begin{cases} 0, & If \text{ no constraint is violated} \\ 1, & If \text{ otherwise} \end{cases} \qquad (8)$$

In this paper, the constraint of the ELD problem can be handled as follow.

$$\begin{gathered} \min(Q(P_i)) = F_c(P_i) + pen * G[h_k(P_i)] \\ Subject\ to\ :\ g_j(P_i) \le 0, j = 1, \dots, J \end{gathered} \qquad (9)$$

Where pen is the penalty factor and $G[h_k(P_i)]$ is the penalty function, which is calculated as follow.

$$G[h_k(P_i)] = \varepsilon^2 + \omega^2 \qquad (10)$$

Where ε and ω are equality and inequality constraint violation penalties, respectively and can be calculated as follow.

$$\varepsilon = \left| P_D + P_{loss} - \sum_{j=1}^{Ng} P_i \right| \qquad (11)$$

and

$$\omega = \begin{cases} |P_i^{min} - P_i| & P_i^{min} > P_i \\ 0 & P_i^{min} < P_i < P_i^{max} \\ |P_i - P_i^{max}| & P_i^{max} < P_i \end{cases} \qquad (12)$$

4. An Overview of the Backtracking Search Optimization Algorithm (BSA)

In this section, we give an overview of the BSA and its main processes. There are five main processes in BSA such as initialization, selection- I, mutation, crossover, and selection-II [9]. In the following subsections, we highlight these processes as follow.

4.1. Initializing Population

The initial population *P* in the BSA generated randomly and consists of N individuals and *D* variables. The initial population can be represented as follow.

$$P_{ij} \sim U(low_j, up_j) \tag{13}$$

For $i = 1, 2, 3, \dots, N$ and $j = 1, 2, 3, \dots, D$, where *N*, *D* are the population size and the problem dimension, respectively, *U* is the uniform distribution.

4.2. BSA's Selection -I

BSA has two selection operators, the first selection is called selection-I, which is used to determine the historical population (P^{old}) in order to calculate the search direction. The initial historical population is generated randomly as follow.

$$P_{ij}^{old} \sim U(low_j, up_j) \tag{14}$$

The P^{old} is redefined at the beginning of each iteration through the following rule

$$If\ a < b\ then\ P^{old} \coloneqq P | a, b \sim U(0,1), \tag{15}$$

Where := is the update operation, a, b are random numbers. After the historical population is determined, the order of its individuals is changed as follow.

$$P^{old} \coloneqq permuting(P^{old}) \tag{16}$$

The permuting function is a random shuffling function.

4.3. BSA's Mutation

BSA generates the initial trail population *Mutant* by applying the mutation operator as follow.

$$Mutant = P + F \cdot (P^{old} - P) \tag{17}$$

Where *F* controls the amplitude of the search direction matrix *(oldP - P).*

4.4 BSA's Crossover

The final form of the trial population *T* is generated by using BSA's crossover operator. There are two steps in BSA's crossover process. The first step generates a binary integer-valued matrix (*map*), where *map* size is $N \times D$, which indicates the individuals of the trail population *T*. The initial value of the binary integer matrix $map_{n,m}$ is set to 1, where $n \in \{1,2,3, \dots, N\}$ and $m \in \{1,2,3, \dots, D\}$. The trail population *T* is updated as follow.

$$T_{n,m} \coloneqq P_{n,m} \tag{18}$$

The main steps of the BSA's crossover are presented in Algorithm 1 as follow.

Algorithm 1. BSA's Crossover.

```
1: **For** $i = 1; i \le N; i++$
2:     **For** $j = 1; j <= D; j++$
3:     Set $map_{ij} = 1$
4:     **End**
5: **End**
6: Generate random numbers $a, b \in [0,1]$
7: **If** $(a < b)$ **Then**
8:     **For** $i = 1; i \le N; i++$
9:         Set $map_{i,u(1:\lceil mixrate\,.rnd\,.D \rceil)} = 0$
10:    **End**
11: **Else**
12:    **For** $i = 1; i \le N; i++$
13:        Set $map_{i,rand(D)} = 0$
14:    **End**
15: **EndIf**
16:Set $T = mutant$
17:**For** $i = 1; i \le N; i++$
18:    **For** $j = 1; j <= D; j++$
19:        **If** $(map_{ij} = 1$ **Then**
20:            Set $T_{ij} = P_{ij}$
21 :       **EndIf**
22 :   **End**
23: **End**
```

In the crossover process, the mix rate parameter (mixrate) controls the number of variables in the individuals, which will mutate in a trail population T by using $\lceil mixrate\,.rnd\,.D \rceil$.

4.5 Boundary Control Mechanism of BSA

The obtained individuals from BSA's crossover may overflow the allowed search space limit; these individuals are regenerated using Algorithm 2.

Algorithm 2. Boundary control mechanism of BSA.

```
1: **For** $i = 1; i \le N; i++$
2:     **For** $j = 1; j <= D; j++$
3:         **If** $(T_{ij} < low_j)$ $or$ $(T_{ij} > up_j$ **Then**
4:             Set $T_{ij} = rnd\,.(up_j - low_j) + low_j$
5:         **End**
6:     **End**
**7: End**
```

4.6. BSA's Selection-II

The second selection operator in the BSA is a greedy selection, which is called selection-II. The individual of the trail population T are replaced with the individuals in the population P, when their fitness values are better than the fitness values of the individuals of the population P. The overall best individual with the best fitness value is selected to be the global best solution P^{best}.

4.7. Backtracking Search Optimization Algorithm (BSA)

Backtracking search optimization algorithm (BSA) is a population-based algorithm designed to obtain a global minimum of functions. The main steps of BSA are reported in Algorithm 3 and the flowchart in Figure 1.

Algorithm 3. BSA algorithm.

```
1: Set the initial values of population size $N$, $mat\_itr$, $mix\_rate$.
2: For $i = 1; i \le N; i++$
3:     For $j = 1; j <= D; j++$
4:     Set $P_{ij} = rnd.(up_j - low_j) + low_j$
5:     Set $P_{ij}^{old} = rnd\,.\left(up_j - low_j\right) + low_j$
6:     End
7:Evaluate each solution in the population $P$ by calculating its fitness function $f(P)$
8: End
9: Set $t = 0$
10: Repeat
11:      Generate random numbers $a, b$, where $a, b \in [0,1]$
12:      If $(a < b\ P|a, b \sim U(0,1))$ Then
13:           Set $P^{old} := P$
14:      End
15:      $P^{old} := permuting(P^{old})$ant
16:     Set $P^{mutant} = P + 3 \cdot rndn \cdot (P^{old} - P)$
17:     Apply the crossover operator on the solutions in $P^{mutant}$ as shown in Algorithm 1
18:     If individuals beyond the search space Then
19:        Regenerate the individuals as shown in Algorithm 2
20:    End
21:    Evaluate each solution in the trail population $T$ , $f(T)$
22:    For $i = 1; i \le N; i++$
23:       If $f(T_i) < f(P_i)$ Then
24:           Set $f(P_i) = f(T_i)$
25:           Set $P_i = T_i$
26:       End
27:   End
28:   Set $P^{best} = \min f(P)$
29:   If $P^{best} < global\_min$ Then
30:     $global\_min = P^{best}$
31:   End
32    Set $t = t + 1$
33: Until $t <$ max_$itr$
34: Produce the best obtained solution
```

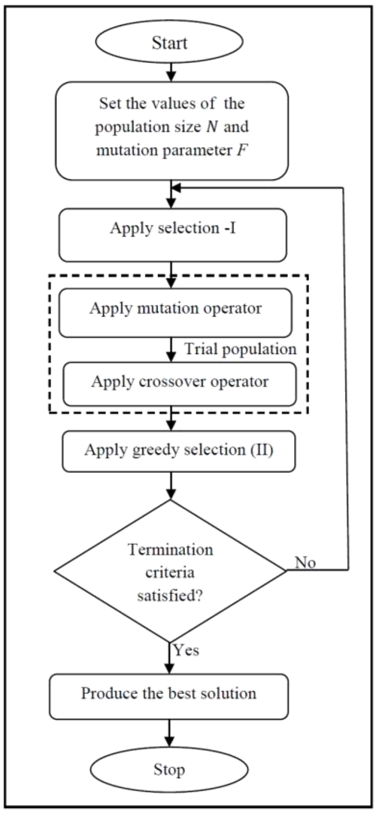
Start
Set the values of the population size N and mutation parameter F
Apply selection -I
Apply mutation operator
Trial population
Apply crossover operator
Apply greedy selection (II)
Termination criteria satisfied?
No
Yes
Produce the best solution
Stop

Fig 1. General structure of BSA.

The main steps of BSA are summarized as follow.

Step 1: The algorithm starts by setting its initial values of the main parameters in the algorithm (line 1)

Step 2: The initial population P and the initial historical population P^{old} are generated randomly as shown in Equations 13, 14, respectively and each individual (solution) is evaluated by calculating its fitness function. (lines 2-8)

Step 3: The iteration counter t is initialized. (line 9)

Step 4: The following process are repeated until termination criteria satisfied

Step 4.1: Two random numbers a, b are generated randomly and if a is less than b, the historical population P^{old} is redefined as shown in Equation 15 and the order of the individuals in the historical population is changed randomly as shown in Equation 15. (lines 11-15)

Step 4.2: The mutation operator is applied in order to form the initial trail population as shown in Equation 17. (line 16)

Step 4.3: The crossover operator is applied on the mutant trail population as shown in Algorithm 1. (line 17)

Step 4.4: The boundary control mechanism of BSA is applied on the individuals, which are overflow the allowed search space limit. (lines 18-20)

Step 4.5: Each solution in the trail population is evaluated by calculating its fitness function and the greedy selection in applied (selection-II operator). (lines 22-27)
Step 4.6: The best optimal solution P^{best} is selected and if its fitness value is less than the global minimum value, it assigned to the global minimum at each iteration. (lines 28-31)
Step 4.7: The iteration counter is increased and the termination criteria is satisfied. (lines 32-33)

Step 5: The optimal solution is produced. (line 34)

5. The Proposed Memetic Backtracking Search Optimization Algorithm (MBSOA)

Before presenting the proposed algorithm, we highlight a random walk with direction exploitation (RWDE) algorithm as a local search algorithm as follow.

5.1 Random Walk with Direction Exploitation

The proposed MBSOA algorithm uses a Random Walk with Direction Exploitation (RWDE) as a local search algorithm as shown in Algorithm 4 in order to accelerates the convergence of the proposed algorithm and refine the best obtained solution at each iteration in the standard algorithm.

Algorithm 4. Random Walk with direction exploitation

```
1: Set the initial values of a scalar step length Δ, $t_{max}$
2: Set $t = 0$
3: Start with an initial solution $x^0$
4: Evaluation the initial solution $f(x^0)$
5: repeat
6:     Set $t = t + 1$
7:     Generate a unit length random vector $u^{(t)}$
8:     Set $x^{(t+1)} = x^{(t)} + \Delta u^{(t)}$
9:     Evaluate the new solution $f(x^{(t+1)})$
10:    if $f(x^{(t+1)}) < f(x^{(t)})$ then
11:       Set $f(x^{(t+1)}) = f(x^{(t+1)})$
12:    else
13:       Set $f(x^{(t+1)}) = f(x^{(t)})$
14:       Set Δ=Δ/2
15:    end if
16: Until ($t \leq t_{max}$)
17: Produce the best solution
```

The main steps of the RWDE algorithm in Algorithm 4 are summarized as follows.

Step 1: The algorithm starts by setting the initial values of the iteration counter t, maximum iteration t_{max} and a scalar length Δ (lines 1-2)
Step 2: The initial solution x^0 is given to the algorithm in order to start the search from this solution (line 3)
Step 3: The initial solution is evaluated by calculating its fitness function $f(x^0)$ (line 4)
Step 4: The following process are repeated until termination criteria satisfied
Step 4.1: The iteration counter is increased (line 6)
Step 4.2: A unit length random vector $u^{(t)}$ is generated (line 7)

Step 4.3: The solution x is updated and evaluated by calculating its fitness function $f(x^{(t+1)})$ (lines 8-9)

Step 4.4: If the fitness function value of the new solution is better than the fitness function value of the old solution, then the new solution is accepted and the operation is repeated until the current iteration reaches to the maximum number of iteration t_{max}, otherwise the new solution is rejected and the step length is reduced to Δ=Δ/2 (lines 10-16) .

Step 5: The best solution is produced (line 17).

5.2 The Proposed MBSOA

In the proposed algorithm, we try to balance between a global exploration and a deep exploitation by combining the standard backtracking algorithm with its ability to perform the exploration process and the random walk with direction exploitation with its capability to perform deep exploitation. The proposed algorithm is called a memetic backtracking search optimization algorithm (MBSOA). The main steps of the proposed MBSOA are presented in Algorithm 5 as follow.

Algorithm 5. MBSOA algorithm.

```
1: Set the initial values of population size $N$, $mat\_itr$, $mix\_rate$.
2: For $i = 1; i \le N; i++$
3:     For $j = 1; j <= D; j++$
4:     Set $P_{ij} = rnd.(up_j - low_j) + low_j$
5:     Set $P_{ij}^{old} = rnd\,.\left(up_j - low_j\right) + low_j$
6:     End
7:Evaluate each solution in the population $P$ by calculating its fitness function $f(P)$
8: End
9: Set $t = 0$
10: Repeat
11:     Generate random numbers $a, b$, where $a, b \in [0,1]$
12:      If $(a < b\ P | a, b \sim U(0,1))$ Then
13:         Set $P^{old} = P$
14:      End
15:      $P^{old} = permuting(P^{old})$ant
16:      Set $P^{mutant} = P + 3 \cdot rndn \cdot (P^{old} - P)$
17:      Apply the crossover operator on the solutions in $P^{mutant}$ as shown in Algorithm 1
18:      If individuals beyond the search space Then
19:         Regenerate the individuals as shown in Algorithm 2
20:      End
21:     Evaluate each solution in the trail population $T$ , $f(T)$
22:     For $i = 1; i \le N; i++$
23:        If $f(T_i) < f(P_i)$ Then
24:           Set $f(P_i) = f(T_i)$
25:           Set $P_i = T_i$
26:        End
27:     End
28:     Set $P^{best} = \min f(P)$
29:     If $P^{best} < global\_min$ Then
30:      $global\_min = P^{best}$
31:     End
32: Apply a local search algorithm as shown in Algorithm 4 on the best solution $P^{best}$ to obtain a new solution
33: Set $t = t + 1$
34: Until $t < \max\_itr$
35: Produce the best obtained solution
```

The main modification in the proposed MBSOA is invoking the RWDE algorithm as a local search algorithm in the standard BSA in order to refine the best optimal solution at each iteration and accelerate the convergence of the proposed algorithm. The proposed algorithm is terminated after reaching to a maximum number of iterations which is to 500 as reported in Table 1, increasing this number will increase the number of evaluations without improving the fitness function value.

6. Numerical Experiments

The general performance of the proposed MBSOA is tested on a convex load dispatch problem and compared against three nature-inspired algorithms. MBOSA was programmed in MATLAB. The parameter setting and the performance analysis of the proposed algorithm are presented as follow.

6.1. Parameter Setting

Before discussing the results, we summarize the parameter setting of MBSOA and their values as shown in Table 1.

Table 1. Parameter setting.

Parameters	Definitions	Values
N	population size	40
mat_itr	Maximum iteration number	500
mix_rate	Mix rate	1

6.2. Six Generator Test System with System Losses

The proposed MBSOA is tested on a six-generator test system. In order to simplify the problem, the values of parameters *d*, *e* in Equation 2 have been set to zero. The proposed algorithm is carried out for a total demand of 700MW and 800 MW. The generator active power limits and the fuel cost coefficients are given in Tables 2, 3, respectively.

Table 2. Generator active power limits

Generator	1	2	3	4	5	6
$P_{min}(MW)$	10	10	35	35	130	125
$P_{max}(MW)$	125	150	225	210	325	315

Table 3. Fuel cost coefficients

No.	a	b	c
1	0.15240	38.53973	756.79886
2	0.10587	46.15916	451.32513
3	0.02803	40.39655	1049.9977
4	0.03546	38.30553	1243.5311
5	0.02111	36.32782	1658.5596
6	0.01799	38.27041	1356.6592

6.3. The General Performance of the Proposed MBSOA with Economic Dispatch Problem.

The general performance of the proposed algorithm with economic dispatch problem are shown in Figure 2 by plotting the number iterations versus the cost ($/h) for total system demand 700 MW, 800 MW, respectively.

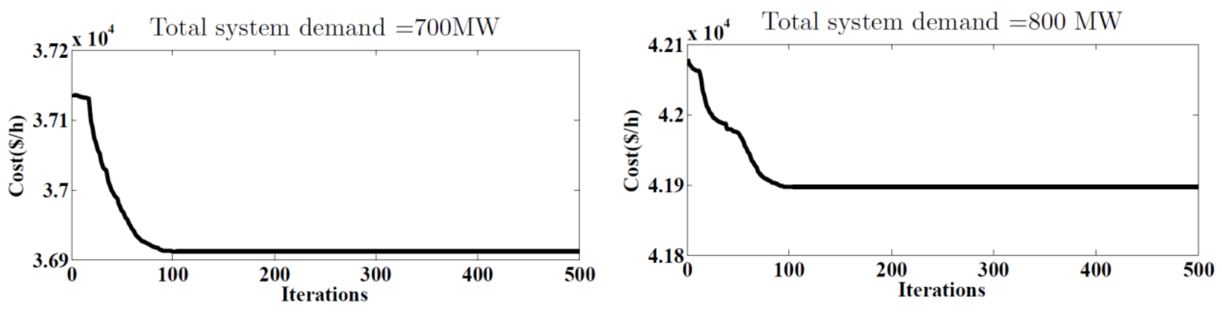

Fig. 2. The general performance of MBSOA with economic dispatch problem.

Figure 2 shows that the cost values are rapidly decrease with a few numbers of iterations. We can conclude from Figure 2 that the proposed algorithm is a promising algorithm and can obtain the desired power with the minimum cost.

6.4. MBSOA and Other Nature-Inspired Algorithms

In this subsection, we highlight the three compared nature- inspired algorithm as follow.

GA. Genetic algorithm is a population-based meta-heuristics algorithm. It has been developed by J. Holland [17] to understand the adaptive processes of natural systems. GA usually applies a crossover operator by mating the parents (individuals) and a mutation operator that randomly modifies the individual contents to promote diversity to generate a new offspring. GA uses a probabilistic selection that is originally the proposed proportional selection. The replacement (survival selection) is generational, that is, the parents are replaced systematically by the offspring.

PSO. Particle swarm optimization (PSO) is a population based meta-heuristics algorithm that inspired from the behavior (information exchange) of the birds in a swarm and it was proposed by Kennedy and Eberhart in 1995 [19]. PSO uses two techniques to find global optima, global best solution "gbest" and local best solution "lbest". The first technique is "gbest"` technique, in this technique, all particles share information with each other and move to global best position. However, this technique has drawback, because it is easy to trap in local optima. The second technique is "lbest" technique, a specific number of particles are neighbors to one particle, but this technique also has drawback, which is the slow of convergence.

BA. Bat algorithm (BA) is a population based meta-heuristics algorithm developed by Xin-She Yang [45]. BA is based on the echolocation of microbats, which use a type of sonar (echolocation) to detect prey and avoid obstacles in the dark. The main advantage of the BA that it can provide a fast convergence at a very initial stage by switching from exploration to exploitation, however, switching from exploration to exploitation quickly may lead to stagnation after some initial stage.

6.4.1. Comparison between GA, PSO, BA, and MBSOA.

The performance of the proposed MBSOA is tested on a six-generator test system and compared against three natural- inspired algorithms at total demand of 700 MW and 800 MW as shown in Tables 4, 5 respectively.

For all the experiments, all algorithms have applied the same termination criteria, which are the error tolerance was set to 0.01 MW or the maximum number of iterations was set to 500.

Table 4. Results for GA, PSO, BA and MBSOA at total system demand = 700 MW

Algorithm		$P1(MW)$	$P2(MW)$	$P3(MW)$	$P4(MW)$	$P5(MW)$	$P6(MW)$	Loss(MW)	Cost($/h)
	Best	26.7997	15.8931	107.307	123.932	228.3426	217.160	19.44	36924.15
GA	Avg	45.5736	48.619	105.805	106.478	211.4508	200.676	18.61	37505.72
	Std	19.770	28.673	43.328	36.2062	45.62	45.043	1.325	382.88
	Best	28.3023	9.9998	118.952	118.6706	230.7563	212.737	19.431	36911.54
PSO	Avg	28.3979	10.0233	119.0863	118.5947	230.588	212.723	19.4262	36911.75
	Std	0.858	0.139	0.83555	0.6229	1.1889	0.494	0.02786	1.4869
	Best	28.0739	10.0569	119.985	117.7729	231.1333	212.391	19.4238	36911.79
BA	Avg	28.3941	10.2677	119.159	119.0363	230.2951	212.244	19.4092	36912.54
	Std	0.6928	0.2676	2.2262	1.7091	2.9539	3.8	0.05999	1.0006
	Best	28.1483	10.0389	119.724	118.052	231.0219	212.419	19.4313	36911.28
MBSOA	Avg	28.3637	10.0213	119.0861	118.593	231.1245	212.713	19.4239	36911.81
	Std	0.1492	0.0124	0.4124	0.38249	0.08145	0.2079	0.0145	0.9154

Table 5. Results for GA, PSO, BA and MBSOA at total system demand = 800 MW

Algorithm		$P1(MW)$	$P2(MW)$	$P3(MW)$	$P4(MW)$	$P5(MW)$	$P6(MW)$	Loss(MW)	Cost($/h)
	Best	39.63015	13.23341	170.317	155.1286	232.4949	213.4204	24.2359	41976.08
GA	Avg	55.35765	54.95395	130.4268	1342949	230.3903	218.5409	23.9787	42614.68
	Std	25.9155	30.1187	45.3717	39.9879	49.7911	45.6905	1.3105	436.61
	Best	32.59937	14.48227	141.5412	136.0392	257.6555	242.9997	25.3299	41895.98
PSO	Avg	32.5959	14.51256	141.4859	135.9388	257.6442	243.1419	25.3322	41896.02
	Std	0.19817	0.2575	0.31681	0.66662	0.33471	0.86126	0.020216	0.23259
	Best	32.46774	14.34427	141.9097	135.7294	257.7276	243.1421	25.3359	41895.88
BA	Avg	32.58662	14.49149	141.7122	136.2057	257.3597	242.9548	25.3232	41896.17
	Std	0.38275	0.49502	0.97076	0.88628	1.2144	1.3829	0.037035	0.25826
	Best	32.4968	14.34426	141.9089	135.7589	257.7274	243.1419	25.3358	41895.85
MBSOA	Avg	32.5899	14.4831	141.5440	135.8413	257.6588	243.0034	25.3396	41895.94
	Std	0.09245	0.09894	0.2145	0.05468	0.9458	0.9487	0.0781	0.09948

The results in Tables 4, 5 show that the proposed algorithms is a promising algorithm and can obtain the optimal or near optimal results of the economic dispatch problem better than the other compared algorithms.

7. Conclusions

In this paper, a new hybrid backtracking search optimization algorithm (BSOA) and a random walk with direction exploitation algorithm (RWDE) has been proposed in order to solve economic dispatch problem. The proposed algorithm is called a memetic backtracking search optimization algorithm (MBSOA). MBSOA was tested on a six-generator test system for a total demand of 700MW and 800MW and compared against three algorithms. The experimental results show that the proposed algorithm was a promising algorithm and more successful than all of the comparison algorithms.

References

[1]. M.A. Abido, A niched Pareto genetic algorithm for multi-objective environmental/economic dispatch. *Int. J.* Electr. Power Energy Sys, 25, 97–105, 2003.

[2]. A.F.Ali, Accelerated bat algorithm for solving integer programming problems, Egyptian computer science journal, vol 39, Jan, 2015.

[3]. A.F.Ali, A new hybrid particle swarm optimization with variable neighborhood search for solving unconstrained global optimization problems, the Fifth International Conference on Innov, Bio-Inspired Comput. and Appl. IBICA 2014.

[4]. N. Amjady, H. Nasiri-Rad, Solution of non-convex and non-smooth economic dispatch by a new adaptive real coded genetic algorithm. Expert Syst. App, 37, 5239–5245, 2010.

[5]. T. Apostolopoulos and A. Vlachos, Application of the Firfley Algorithm for Solving the Economic Emissions Load Dispatch Problem, International Journal of Combinatorics, Volume 2011, 2011.

[6]. J. Cai, X. Ma, L. Li, P. Haipeng, Chaotic particle swarm optimization for economic dispatch considering the generator constraints. Energy Convers. Manag, *48*, 645–653, 2007.

[7]. W.D. Chang, PID control for Chaotic synchronization using particle swarm optimization. Chaos, Solutions and Fractals, 39(2), 910-917, 2009.

[8]. S.A. Chu, P.W. Tsai, and J.S. Pan, Cat swarm optimization, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 4099 LNAI:854-858, 2006.

[9]. P. Civicioglu, Backtracking search optimization algorithm for numerical optimization problems / Applied Mathematics and Computation 219, 8121–8144, 2013.

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

[11]. M. Dorigo, Optimization learning and natural algorithms, Ph.D. Thesis, Politecnico di Milano, Italy, 1992.

[12]. J.D. Farmer, N.H. Packard, A.S. Perelson, The immune system, adaptation, and machine learning, Physica D2, 187-204, 1986.

[13]. Z. L. Gaing, Particle swarm optimization to solving the economic dispatch considering the generator constraints. Power Syst. IEEE Trans, *18*, 1187–1195, 2003.

[14]. V. Gazi, K.M. Passino, Stability analysis of social foraging swarms, IEEE Transactions on Systems, Man, and Cybernetics – Part B 34 (1), 539–557, 2004.

[15]. F. Glover, M. Laguna, Tabu search. Kluwer, Boston, 1997.

[16]. S. He, Q.H. Wu and J.R. Saunders. Group search optimizer - an optimization algorithm inspired by animal searching behavior. IEEE Transaction on Evolutionary Computation 13(5), 973-990, 2009.

[17]. J.H. Holland, Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor, MI, 1975.

[18]. D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 214 108–132, 2009.

[19]. J. Kennedy, RC. Eberhart, "Particle Swarm Optimization", Proceedings of the IEEE International Conference on Neural Networks, Vol 4, pp 1942-1948, 1995.

[20]. D.H. Kim, A. Abraham, J.H. Cho, A hybrid genetic algorithm and bacterial foraging approach for global optimization, Information Sciences 177, 3918–3937, 2007.

[21]. S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science, 220(4598):671-680, 1983.

[22]. G. Komarasamy and A. Wahi, An optimized K-means clustering technique using bat algorithm, European J. Scientific Research, Vol. 84, No. 2, pp. 263-273, 2012.

[23]. M.D. Lei, A Pareto archive particle swarm optimization for multi-objective job shop scheduling. Computers & Industrial Engineering, 54(4), 960971, 2008.

[24]. X.L. Li, Z.J. Shao, and J.X. Qian, Optimizing method based on autonomous animates Fish-swarm algorithm, Xitong Gongcheng Lilun yu Shijian/System Engineering Theory and Practice, 22(11):32, 2002.

[25]. J.H. Lin, C.W. Chou, C.H. Yang, H.L. Tsai, A chaotic Levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems, J.Computer and Information Technology, Vol. 2, No. 2, pp. 5663, 2012.

[26]. R.Y.M. Nakamura, L.A.M. Pereira, K.A. Costa, D. Rodrigues, J.P. Papa, X.S. Yang, BBA: A binary bat algorithm for feature selection, in: 25th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), 22-25, IEEE Publication, pp. 291-297, 2012.

[27]. T. Niknam, R. Azizipanah-Abarghooee, M. Zare, B. Bahmani-Firouzi, Reserve constrained dynamic environmental/economic dispatch: A new multi-objective self-adaptive learning bat algorithm. Syst. J. IEEE, *7*, 763–776, 2012.

[28]. M.K. Passino, Biomimicry of bacterial foraging for distributed optimization and control, Control Systems, IEEE 22(3),52-67, 2002.

[29]. B. Ramesh, V. Chandra Jagan Mohan, V.C. Veera Reddy, Application of bat algorithm for combined economic load and emission dispatch. Int. J. Electr. Eng. Telecommun, 2, 1–9, 2013.

[30]. A.I. Selvakumar, K. A. Thanushkodi, A new particle swarm optimization solution to non-convex economic dispatch problems. Power Syst. IEEE Trans, *22*, 42–51, 2007.

[31]. A.I. Selvakumar, K. Thanushkodi, Anti-predatory particle swarm optimization: Solution to non-convex economic dispatch problems. Electr. Power Syst. Res. *78*, 2–10, 2008.

[32]. A. Sidi, A. Economic dispatch problem using bat algorithm. Leonardo J. Sci. *24*, 75–84, 2014.

[33]. R.M. Storn, K.V. Price, Differential evolution a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11, 341-359, 1997.

[34]. P. Subbaraj, R. Rengaraj, S. Salivahanan, Enhancement of self-adaptive real-coded genetic algorithm using Taguchi method for economic dispatch problem. Appl. Soft Comput. 11, 83–92, 2011.

[35]. R. Tang, S. Fong, X.S. Yang, and S. Deb, Wolf search algorithm with ephemeral memory, In Digital Information Management (ICDIM), 2012 Seventh International Conference on Digital Information Management, pages 165-172, 2012.

[36]. D. Teodorovic and M. DellOrco, Bee colony optimization a cooperative learning approach to complex transportation problems, In Advanced OR and AI Methods in Transportation: Proceedings of 16th MiniEURO Conference and 10th Meeting of EWGT (13-16 September 2005).Poznan: Publishing House of the Polish Operational and System Research, pages 51-60, 2005.

[37]. C. Voudouris, Guided Local Search for Combinatorial Optimization Problems, Ph.D Thesis, University of Essex, 1997.

[38]. C. Voudouris, Guided local search: an illustrative example in function optimization, BT Technology Journal 16 46-50, 1998.

[39]. C. Voudouris, E. Tsang, Guided local search, European Journal of Operational Research 113, 469-499, 1999.

[40]. X.S. Yang and S. Deb. Cuckoo search via levy ights. In Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, pages 210-214. IEEE, 2009.

[41]. Yang XS, Deb S, Engineering Optimisation by Cuckoo Search, Int. J. Mathematical Modeling and Numerical Optimization. Vol. 1, No. 4, pp 330–343, 2010.

[42]. X. S. Yang, Swarm-based meta-heuristic algorithms and no-free-lunch theorems, in: Theory and New Applications of Swarm Intelligence (Eds. R. Parpinelli and H. S. Lopes), Intech Open Science, pp. 1-16, 2012.

[43]. X.S. Yang, Firefly algorithm, stochastic test functions and design optimization, International Journal of Bio-Inspired Computation, 2(2):78-84, 2010.

[44]. X.S. Yang and S. Deb, Cuckoo search via levy flights, In Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, pages 210-214. IEEE, 2009.

[45]. X.S. Yang, A new meta-heuristic bat-inspired algorithm, Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), pages 65-74, 2010.

[46]. K. Zielinski, P. Weitkemper, R. Laur, Optimization of power allocation for interference cancellation with particle swarm optimization. IEEE Transactions on Evolutionary Computation, 13(1), 128150, 2009.