International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:3, No:4, 2009

Application of soft computing methods for Economic Dispatch in Power Systems

Jagabondhu Hazra, *Member, IEEE,* and Avinash Sinha, *Member, IEEE,*

Abstract—Economic dispatch problem is an optimization problem where objective function is highly non linear, non-convex, non-differentiable and may have multiple local minima. Therefore, classical optimization methods may not converge or get trapped to any local minima. This paper presents a comparative study of four different evolutionary algorithms i.e. genetic algorithm, bacteria foraging optimization, ant colony optimization and particle swarm optimization for solving the economic dispatch problem. All the methods are tested on IEEE 30 bus test system. Simulation results are presented to show the comparative performance of these methods.

Keywords—Ant colony optimization, bacteria foraging optimization, economic dispatch, evolutionary algorithm, genetic algorithm, particle swarm optimization.

I. Introduction

Economic Dispatch (ED) in Power System deals with the determination of optimum generation schedule of available generators so that total cost of generation is minimized within the system constraints [1]. Several classical optimization techniques such as lambda iteration method, gradient method, Newton's method, linear programming, Interior point method and dynamic programming have been used to solve the basic economic dispatch problem [2]. Lambda iteration method has the difficulty of adjusting lambda for complex cost functions. Gradient methods suffer from the problem of convergence in the presence of inequality constraints. Newton's method is very much sensitive to the selection of initial conditions. Linear programming approach provides optimal results in less computational time but results are not accurate due to linearization of the problem. Interior point method is faster than linear programming but it may provide infeasible solution if the step size is not chosen properly. Dynamic programming suffers from curse of dimensionality.

Most of the classical optimization techniques need derivative information of the objective function to determine the search direction. But actual fuel cost functions are non-linear, non-convex and non-differentiable because of ramp rate limits, prohibited operating zones, valve point effects and multi-fuel options [3]. Recently some heuristic techniques such as genetic algorithm [4], genetic algorithm combined with simulated annealing [5], evolutionary programming [6], improved tabu search [7], ant swarm optimization [8] and particle swarm optimization [9]-[12] have been used to solve the complex non-linear optimization problem.

Jagabondhu Hazra is with the Department of Power and Energy systems, SUPELEC, France, e-mail: jagabondhu.hazra@supelec.fr
Avinash Sinha is with the Department of Electrical Engineering, IIT Kharagpur, India, e-mail: aksinha@ee.iitkgp.ernet.in

In this paper ED problem has been solved using four different evolutionary algorithms i.e. genetic algorithm (GA), bacteria foraging optimization (BFO), ant colony optimization (ACO) and particle swarm optimization (PSO). Performance of each algorithm for solving the ED problem has been investigated and simulation results are presented in terms of accuracy, reliability and execution time.

The paper is organized as follows. Section II formulates the Economic Dispatch problem, Section III presents the brief overview of different evolutionary algorithms, Section IV describes the constrain handling method, Section V presents the simulation results, and Section VI concludes.

II. Problem Formulation

The objective of solving the ED problem is to minimize the total generation cost of a power system while satisfying various equality and inequality constraints.

Typically generator cost functions are approximately modeled by quadratic functions. But, actual fuel cost function of any large turbo generator may be much more complicated due to valve point loading effect [4]. The Economic Dispatch problem considering the valve point loading effect can be expressed as:

$$\begin{aligned} Minimize \quad F &= \sum_{i=1}^{NG} F_{gi} \\ &= \sum_{i=1}^{NG} (p_i + q_i P_{gi} + r_i P_{gi}^2) \\ &+ |e_i \times \sin(f_i \times (P_{gi} - P_{mini}))| \end{aligned} \quad (1)$$

where,

F	total generation cost;
F_{gi}	generation cost of generator i;
NG	number of generators;
P_{gi}	generation of generator i;
P_{mini}	minimum generation of generator i;
p_i, q_i, r_i	cost coefficients of generator i ;
e_i, f_i	coefficients of generator i reflecting valve point loading effect.

subjected to the following constraints:

$$P_{gi} - P_{di} - P_{loss} = 0 \quad (2)$$

$$Q_{gi} - Q_{di} - Q_{loss} = 0 \quad (3)$$

$$P_{mini} \leq P_{gi} \leq P_{maxi} \quad (4)$$

$$Q_{mini} \leq Q_{gi} \leq Q_{maxi} \quad (5)$$

where,

P_{gi}, Q_{gi}	real and reactive power generation at bus i;
P_{di}, Q_{di}	real and reactive power demand at bus i;
P_{loss}, Q_{loss}	total real and reactive power loss;
P_{mini}, P_{maxi}	minimum and maximum active power generation limits of generator i;
Q_{mini}, Q_{maxi}	minimum and maximum reactive power generation limits of generator i.

III. Evolutionary Algorithms

This section gives brief description of the evolutionary algorithms used in this paper as follows:

A. Genetic Algorithm (GA)

In last few decades, GA has been treated as benchmark for various optimization problems. GA consists of four steps i.e. representation, initialization, selection and reproduction with crossover and mutation. Depending on the type of representation genetic algorithms can be broadly classified into two groups (i)Binary coded genetic algorithm (BCGA) and (ii)Real coded genetic algorithm (RCGA).

1) Binary Coded Genetic Algorithm (BCGA): In BCGA, all the real world problems (phenotype space) are encoded to binary representation (genotype). Each element of a solution vector is represented by a binary string $z_i[z_i \in [x,y]] = [a_i^1, a_i^2, \ldots, a_i^L] \in [0,1]^L$. The length of the binary string (chromosome) L depend on how much accuracy is required.

For BCGA, there are several selection methods to form the parent pool such as proportionate reproduction, tournament selection, rank selection, genitor selection, etc. In this paper, binary tournament selection scheme is used because of its less time complexity [13].

Crossover is a method for sharing information between chromosomes. It combines the features of parent chromosomes to form offspring. Commonly used crossover techniques are one point crossover, two point crossover, n point crossover, uniform crossover, heuristic crossover, etc. In this paper uniform crossover is used to avoid positional bias [14].

Crossover operation makes a big jump to an area somewhere in between the two parent (explorative) whereas mutation creates random small diversions near the parent (exploitative). Typically mutation probability is very low (of the order of .01).

2) Real Coded Genetic Algorithm (RCGA): BCGA has difficulties of binary representation when dealing with continuous search space with large dimensions and improved numerical precision. RCGA does not have such difficulties. [15]-[16]. In RCGA genes are represented directly as real numbers and a chromosome is a vector of floating point numbers. As RCGA deals directly with real numbers, there is no need of phenotype to genotype conversion and vice versa. Any decision vector x is represented as follows: $\overline{x}$=$[x_1, x_2, \ldots, x_n]$, where n is the number of parameters to be optimized.

There are different types of crossover techniques for RCGA such as flat, simple, arithmetic, blend, linear, discrete, logical FCB (connectives Based Crossover) etc. Among different crossover techniques blend (with $\alpha = .5$), linear and logical FCB crossover techniques outperform the others [15]. In this paper blend crossover (with $\alpha = .5$) has been used to maintain a balance between exploitation and exploration.

There are different types of mutation techniques for RCGA such as nonuniform, real number creep, continuous modal mutation, discrete modal mutation. Among them non-uniform crossover is very effective for RCGA [15]. Therefore, In this study non-uniform crossover is chosen for RCGA.

B. Particle Swarm Optimization (PSO)

In 1995, Kennedy and Eberhart first introduced the PSO method [17] motivated by social behavior of organisms such as fish schooling and bird flocking. PSO is a population based search technique. Each individual potential solution in PSO is called particle. Each particle in a swarm fly around in a multidimensional search space based on its own experience and experience of neighboring particles.

Let, define the search space S in n-dimension and the swarm consists of N particles. Let, at instant t, particle i has its position defined by $X_t^i = \{x_1^i, x_2^i, \ldots, x_n^i\}$ and velocity defined by $V_t^i = \{v_1^i, v_2^i, \ldots, v_n^i\}$ in variable space S. Velocity and position of each particle in the next generation (time step) can be calculated as:

$$V_{t+1}^i = w \times V_t^i + c_1 \times rand() \times (P_t^i - X_t^i) + c_2 \times Rand() \times (P_t^g - X_t^i) \quad (6)$$

$$X_{t+1}^i = X_t^i + V_{t+1}^i \qquad \forall i = 1, \ldots, N \quad (7)$$

where

N	number of particles in the swarm;
w	inertia weight;
c_1, c_2	acceleration constant;
$rand()$, $Rand()$	uniform random value in the range [0,1];
P_t^g	global best at generation t;
P_t^i	best position that particle i could find so far.

Performance of PSO depends on selection of inertia weight (w), maximum velocity v_{max} and acceleration constants (c_1, c_2). The effect of these parameters is illustrated as follows:

Inertia weight(w): Suitable selection of weight factor w helps in quick convergence. A large weight factor facilitates global exploration (i.e. searching of new area) while small weight factor facilitate local exploration. Therefore, it is wiser to choose large weight factor for initial iterations and gradually smaller weight factor for successive iterations. In standard PSO linearly decreasing inertia weight w is set as 0.9 at beginning and 0.4 at the end [18].

Maximum velocity (v_{max}): With no restriction on the maximum velocity of the particles, velocity may become infinitely large. If v_{max} is very low particle may not explore sufficiently and if v_{max} is very high it may oscillate about optimal solution. Therefore, velocity clamping effect has been introduced

to avoid the phenomenon of "swarm explosion"[19]. In general maximum velocity is set as 10-20% of dynamic range of each variable. Velocity can be controlled within a band as:

$$v_{max} = v_{ini} - \frac{v_{ini} - v_{fin}}{iter_{max}} \times iter \tag{8}$$

where, v_{ini} is initial velocity, v_{fan} is final velocity, $iter$ is iteration number and $iter_{max}$ is number of maximum iterations.

Acceleration constants (c_1, c_2): Acceleration constant c_1 called cognitive parameter pulls each particle towards local best position whereas constant c_2 called social parameter pulls the particle towards global best position. Usually the values of c_1 and c_2 are chosen between 0 to 4.

C. Bacteria Foraging Optimization (BFO)

BFO method was invented by Kevin M. Passino [20] motivated by the natural selection which tends to eliminates the animals with poor foraging strategies and favor those having successful foraging strategies. The foraging strategy is governed by four processes namely Chemotaxis, Swarming, Reproduction, and Elimination & Dispersal.

1) Chemotaxis: Chemotaxis process is the characteristics of movement of bacteria in search of food and consists of two processes namely Swimming and Tumbling. A bacterium is said to be swimming if it moves in a predefined direction, and tumbling if it starts moving in an altogether different direction. Let, j be the index of chemotactic step, k be the reproduction step and l be the elimination dispersal event. Let, $\theta_i(j,k,l)$ is the position of i^{th} bacteria at j^{th} chemotactic step, k^{th} reproduction step and l_{th} elimination dispersal event. The position of the bacteria in the next chemotactic step after a tumble is given by:

$$\theta_i(j+1,k,l) = \theta_i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \tag{9}$$

where
$C(i)$ denotes step size;
$\Delta(i)$ random vector;
$\Delta^T(i)$ transpose of vector $\Delta(i)$.

If the health of the bacteria improves after the tumble, the bacteria will continue to swim to the same direction for the specified steps or until the health degrades.

2) Swarming: Bacteria exhibits swarm behavior i.e. healthy bacteria try to attract other bacterium so that together they reach the desired location (solution point) more rapidly. The effect of Swarming is to make the bacteria congregate into groups and move as concentric patterns with high bacterial density. Mathematically swarming behavior can be modeled as [20]:

$$\begin{aligned} J_{cc}(\theta, P(j,k,l)) &= \sum_{i=1}^{S} J_{cc}^{i}(\theta, \theta_i(j,k,l)) \\ &= \sum_{i=1}^{S}[-d_{attract}exp(-w_{attract})\sum_{m=1}^{p}(\theta^m - \theta_i^m)^2] \\ &+ \sum_{i=1}^{S}[-h_{repelent}exp(-w_{repelent})\sum_{m=1}^{p}(\theta^m - \theta_i^m)^2] \end{aligned} \tag{10}$$

where
J_{cc} the relative distances of each bacterium from the fittest bacterium
S number of bacteria
p number of parameters to be optimized
θ^m position of the fittest bacteria
$d_{attract}, w_{attract}, h_{repelent}, w_{repelent}$ different parameters

3) Reproduction: In this step, population members who have had sufficient nutrients will reproduce and the least healthy bacteria will die. The healthier population replaces unhealthy bacteria which gets eliminated owing to their poorer foraging abilities. This makes the population of bacteria constant in the evolution process.

4) Elimination and Dispersal: In the evolution process a sudden unforeseen event may drastically alter the evolution and may cause the elimination and/or dispersion to a new environment. Elimination and dispersal helps in reducing the behavior of stagnation i.e. being trapped in a premature solution point or local optima.

D. Ant Colony Optimization (ACO)

ACO was invented by Marco Dorigo and colleages [21]-[22] inspired by the foraging behavior of ant colony. While moving, each ant lays certain amount of pheromone on the path. Ants use the pheromone trails to communicate information among the individuals and based on that each ant decides the shortest path to follow.

ACO technique has been successfully used for difficult discrete combinatorial optimization problems such as Traveling Salesman Problem [23], Sequential Ordering [24], Routing in communication problems [25], etc. In recent years a large number of literature has been published [26]-[28] where ACO algorithm has been successfully used for continuous optimization problems.

In this paper, ACO algorithm proposed in [27] has been implemented for the solution of ED problem. The algorithm consists of four stages: Solution construction, Pheromone update, Local Search and Pheromone Re-initialization.

1) Solution Construction: In this method, initial position of each ant i.e. initial solution vectors are generated randomly in the feasible search region. In each iteration, artificial ant construct the solution by generating a random number for each variable using the normal distribution $N(\mu_i, \sigma_i^2)$. Mean (μ_i) and standard deviation (σ_i^2) for each variable i changes with iteration number based on the experience of the colony.

This is synonymous to pheromone update in basic ant colony optimization.

Let $x = [x_1, \ldots, x_n]$ is a solution constructed by any ant using normal distribution $N(\mu_i, \sigma_i^2)$, $i \in \{1, \ldots, n\}$ associated with each variable x_i. Here n is the number of parameters to be optimized. After construction of each solution, upper and lower bound for each parameter is checked and the solution is modified if it goes out of search space:

$$x_i = \begin{cases} a_i & x_i < a_i \\ x_i & a_i \leq x_i \leq b_i \\ b_i & x_i > b_i \end{cases} \quad (11)$$

where, b_i and a_i are upper and lower bound of variable x_i respectively.

2) Pheromone update: Pheromone update procedure composed of pheromone evaporation and pheromone intensification. Pheromone initialization is done as follows:

$$\begin{aligned} \mu_i(0) &= a_i + rand(i)(b_i - a_i) \\ \sigma_i(0) &= (b_i - a_i)/2 \end{aligned} \quad (12)$$

After construction of solutions, pheromone evaporation phase is performed as follows:

$$\begin{aligned} \mu_i(t) &= (1 - \rho_1)\mu_i(t-1) \\ \sigma_i(t) &= (1 - \rho_1)\sigma_i(t-1) \end{aligned} \quad (13)$$

where, t is iteration number and $\rho_1 \in [0, 1]$ is the evaporation parameter, a uniform random number between 0 and 1.

The aim of pheromone intensification is to increase the pheromone value associated with promising solutions. This is done as follows:

$$\begin{aligned} \mu_i(t) &= \mu_i(t) + \rho_2 x^{gb} \\ \sigma_i(t) &= \sigma_i(t) + \rho_2 |x^{gb} - \mu_i(t-1)| \end{aligned} \quad (14)$$

where, $\rho_2 \in [0, 1]$ is the intensification parameter, a uniform random number between 0 and 1 and x^{gb} is the global best solution found in last (t-1) iteration.

3) Local Search: Local search in the vicinity of current solution may improve the solution constructed [29]. Local search is usually made before updating the pheromone. Local search technique proposed in [27] has been used in this paper.

IV. Constraints Handling

ED problem is associated with equality and inequality constraints. In this paper a dynamic penalty function method is used for constraint handling. A penalty proportional to the extent of constraint violation is added to the objective function value if any violation occurs. Penalty methods use a mathematical function that will increase the objective function value for any given constrain violation. The penalty function used in this paper is given by:

$$F(x) = f(x) + h_x H(x) \quad (15)$$

where,

$f(x)$	is the objective function of the constrained problem;
h_x	is the penalty value which is a function of iteration number;
$H(x)$	is the penalty factor and is given by:

$$H(x) = \sum_{i=1}^{m} (max[0, g_i(x)])^2 + \sum_{j=1}^{n} (h_j(x))^2 \quad (16)$$

where,

$g_i(x) < 0$	are the m inequality constraints;
$h_j(x) = 0$	are the n equality constraints.

V. Simulation Results and Discussions

A. Test system details

To verify the effectiveness of the evolutionary optimization techniques, simulations were carried out on IEEE 30 bus test system [30]. For the test system, minimum and maximum generation limits and transmission line capacities were taken from [31]. Generator cost coefficients used in the simulations are given in the Appendix A.

B. Parameter selection

Evolutionary computation techniques are sensitive to proper selection of the control parameters. In this paper parameters were selected based on results of many experiments available in the literature. Parameters selected for the simulations are as follows:

GA parameters [10]: Mutation probability, P_m=0.01; α=0.5 and Bit length=16.

ACO parameters: $cf_{th} = 0.5$.

PSO parameters [33]: Cognitive parameter (C_1)=1.5; social parameter (C_2)=2; maximum weight (w_{max})=0.8; minimum weight(w_{min})=0.5; initial velocity (V_{max})=0.25; and final velocity (V_{min})=0.02.

BFO parameters [20], [34]: Swimming length N_s= 4; number of iteration in a chemotactic loop(N_c)=100; no of reproduction (N_{re})=4; no of elimination and dispersal events (N_{ed})=2; probability of elimination and dispersal (P_{ed})=0.25 and step size $C(i)$=0.1 $\forall i = 1, \ldots, S$.

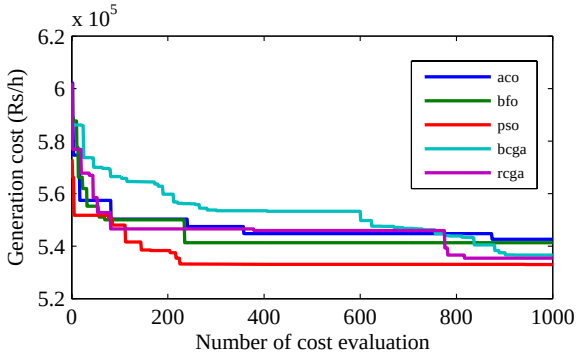
Population size for all the methods were chosen as 10.

C. Convergence characteristics

ACO and BFO evaluate the fitness of each ant/bacterium several times in one iteration. As fitness evaluation is most time consuming, convergence characteristics of all the methods were compared with the number of fitness evaluation. Convergence characteristics with the number of fitness evaluation are shown in Figure 1 for all the methods. Figure 1 shows that all the methods have good convergence characteristics for the selected parameters. ACO and BFO converges quickly than RCGA and BCGA but may not explore good quality solutions as obtained by RCGA and BCGA. Among all the methods,PSO converges quickly and explore good quality solution.

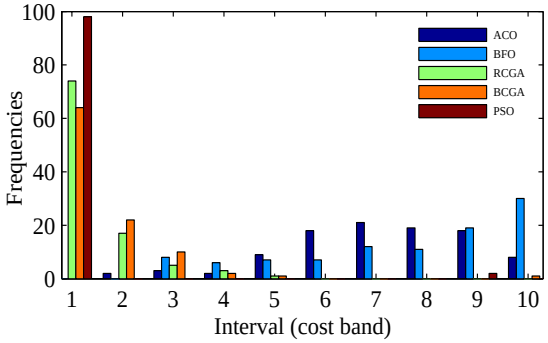
D. Robustness test

Heuristic method may not converge to exactly same solution at each run owing to their randomness. Therefore, their performances could not be judged by the results of a single run. Many trials should be done to reach a useful conclusion



(a) 30 bus

Fig. 1. Convergence characteristics of different EAs



(a) 30 bus

Fig. 2. Comparative solution spreads of different EAs

about the performance of the algorithm. Therefore, to test the robustness of the evolutionary algorithms, 100 independent trials were carried out. The best, worst and average minima obtained by each method are given in Table I. From Table I, it is clear that average minima obtained by PSO is better than the others.

To test the robustness of the evolutionary algorithms in a statistical manner, the interval between worst cost and best cost among all the solutions is subdivided into 10 equal subintervals and the frequencies (number) of solution within the specific cost range are shown in Figure 2. Figure 2 shows that most of the solution for PSO is in band 1. This shows that PSO explore better solutions in most of the time. Complete ED results corrosponding to the best solution obtained by PSO is given in Table II.

TABLE I
TEST RESULTS FOR 30 BUS SYSTEM FOR 100 INDEPENDENT TRIALS

Method	Best Minima	Worst minima	Avg. minima
BCGA	543836	567113	546273
RCGA	544162	553479	545779
PSO	543818	564629	544500
BFO	549025	566789	550829
ACO	546470	567617	551636

TABLE II
COMPLETE ED RESULTS FOR IEEE 30 BUS TEST SYSTEM

Bus	Act. gen. (MW)	React. gen (MVAR)	Volt. mag. (p.u.)	Volt. ang. (deg.)
1	116.89	14.80	1.06000	0.00
2	69.89	16.65	1.04300	-1.97
3	0.00	0.00	1.02539	-3.91
4	0.00	0.00	1.01664	-4.78
5	49.93	16.39	1.01000	-6.65
6	0.00	0.00	1.01363	-5.54
7	0.00	0.00	1.00417	-6.52
8	25.11	20.23	1.01000	-5.67
9	0.00	0.00	1.05382	-6.99
10	0.00	0.00	1.04965	-9.17
11	24.95	15.22	1.08200	-4.38
12	0.00	0.00	1.05858	-9.28
13	2.40	9.50	1.07100	-9.11
14	0.00	0.00	1.04386	-10.08
15	0.00	0.00	1.04003	-10.07
16	0.00	0.00	1.04781	-9.50
17	0.00	0.00	1.04390	-9.49
18	0.00	0.00	1.03135	-10.45
19	0.00	0.00	1.02932	-10.48
20	0.00	0.00	1.03365	-10.21
21	0.00	0.00	1.03715	-9.67
22	0.00	0.00	1.03763	-9.67
23	0.00	0.00	1.02995	-10.31
24	0.00	0.00	1.02485	-10.28
25	0.00	0.00	1.01983	-10.04
26	0.00	0.00	1.00220	-10.46
27	0.00	0.00	1.02529	-9.64
28	0.00	0.00	1.00945	-5.99
29	0.00	0.00	1.00550	-10.86
30	0.00	0.00	0.99405	-11.74

E. Computational efficiency

Computational efficiencies of all the methods are compared based on the average CPU time taken to converge the solution. CPU times taken by each algorithm are given in Table III. From Table III, it is clear that average convergence time for PSO is minimum. BCGA takes more time to converge than RCGA because of conversion from genotype to phenotype or viceversa.

TABLE III
COMPARISON OF EXECUTION TIME

Test system	CPU time (sec)				
	BCGA	RCGA	PSO	BFO	ACO
IEEE 30 Bus	0.543	0.334	0.201	0.289	0.319

VI. CONCLUSIONS

In this paper a comparative study of different evolutionary techniques to solve the power system ED problem is investigated. From the simulation results presented in this paper it is clear that solution quality and robustness of RCGA is better than BCGA. Convergence characteristics of ACO and BFO are attractive and they converge quickly but solution quality is not as good as PSO or GA. From all these findings, it can be concluded that PSO outperforms the others for the chosen set of parameters for solving the Economic Dispatch problem.

APPENDIX A

TABLE IV
GENERATORS COST COEFFICIENTS

UNIT SIZE (MW)	p Rs/h	q Rs/MW/h	r Rs/MW^2/h	e Rs/h	f 1/MW
$\leq$25	0	2025	1.50	0.0	0.0
26-50	0	1875	1.425	1.2×10^4	0.126
51-100	0	1800	1.35	2.4×10^4	0.063
101-200	0	1650	1.25	3.2×10^4	0.047
201-250	0	1575	1.50	3.0×10^4	0.05
251-300	0	1575	1.25	3.6×10^4	0.042
301-350	0	1500	1.35	3.35×10^4	0.045
351-400	0	1500	1.25	3.85×10^4	0.039
401-500	0	1200	1.50	4.0×10^4	0.038
$>$500	0	1200	1.00	4.4×10^4	0.034

REFERENCES

[1] B. H. Chowdhury and S. Rahman, "A review of recent advances in economic dispatch," *IEEE Trans. Power Systems*, vol. 5, no. 4, pp. 1248–1259, Nov. 1990.
[2] A. J. Wood and B. F. Wollenberg, *Power generation operation and control*, 2nd ed. John Willy and Sons, 1996.
[3] A. I. Selvakumar and K. Thanushkodi, "A new particle swarm optimization solution to nonconvex economic dispatch problems," *IEEE Trans. Power Systems*, vol. 22, no. 1, pp. 42–51, Feb. 2007.
[4] M. A. Abido, "A novel multiobjective evolutionary algorithm for environmental economic power dispatch," *Electric Power Systems Research*, vol. 65, no. 1, pp. 71–81, April 2003.
[5] K. P. Wong and Y. W. Wong, "Genetic and genetic/simulated -annealing approaches to economic dispatch," *Proc. Inst. Elect. Eng., Gen., Transm., Distrib.*, vol. 141, no. 5, pp. 507–513, Sept. 1994.
[6] N. Sinha, R. Chakrabarti, and P. K. Chattopadhyay, "Evolutionary programming techniques for economic load dispatch," *IEEE Trans. Evolutionary Computations*, vol. 7, no. 1, pp. 83–94, Feb. 2003.
[7] W. M. Lin, F. S. Cheng, and M. T. Tsay, "An improved tabu search for economic dispatch with multiple minima," *IEEE Trans. Power Systems*, vol. 7, no. 1, pp. 83–94, Feb. 2003.
[8] J. Cai, X. Ma, L. Li, Y. Yang, H. Peng, and X. Wang, "Chaotic ant swarm optimization to economic dispatch," *Electric Power System Research*, vol. 77, no. 10, pp. 1373–1380, Aug. 2007.
[9] J. B. Park, K. Lee, J. Shin, and K. Y. Lee, "A particle swarm optimization for economic dispatch with nonsmooth cost functions," *IEEE Trans. Power Systems*, vol. 20, no. 1, pp. 34–42, Feb. 2005.
[10] Z. Gaing, "Particle swarm optimization to solving the economic dispatch considering the generator constraints," *IEEE Trans. Power Systems*, vol. 18, no. 3, pp. 1187–1195, Aug. 2003.
[11] D. N. Jeyakumar, T. Jayabarathi, and T. Raghunathan, "Particle swarm optimization for various types of economic dispatch problems," *Int. J Electr. Power and Energy system*, vol. 28, no. 1, pp. 36–42, Jan. 2006.
[12] M. R. Alrashidi and M. E. El-Hawary, "A survey of particle swarm optimization applications in power system operations," *Electric Power Components and Systems*, vol. 34, no. 12, pp. 1349 – 1357, Dec. 2006.
[13] D. Goldberg and K. Deb, *A Comparative Analysis of Selection Schemes Used in Genetic Algorithms*. Morgan Kaufmann, San Mateo, California, 1991, ch. Foundations of Genetic Algorithms, pp. 69–93.
[14] W. M. Spears and K. A. De Jong, "On the virtues of parameterized uniform crossover," in *Proceedings of the Fourth International Conference on Genetic Algorithms*, R. K. Belew and L. B. Booker, Eds., Morgan Kaufmann, 1991.
[15] F. Herrera, M. Lozano, and J. L. Verdegay, "Tackling realcoded genetic algorithms:operators and tools for behavioural analysis," *Artificial Intelligence Review*, vol. 12, no. 4, pp. 265–319, 1998.
[16] D. E. Goldberg, "Realcoded genetic algorithms, virtual alphabets, and blocking," *Complex Systems*, vol. 5, pp. 139–167, 1991.
[17] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Intl. Conf. on Neural Networks*, Perth, Australia, 1995, pp. 1942–1948.
[18] Y. Shi and R. Eberhart, "Parameter selection in particle swarm optimization," in *Proceedings of* 7^{th} *Annual Conference on Evolution Computation*, 1998, pp. 591–601.
[19] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*. Morgan Kaufmann,San Francisco, 2001.
[20] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and contro," *IEEE. Control System Magazine*, pp. 52–67, June 2002.
[21] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation, Dipartimento di Elettronica, Politecnico di Milano, IT, 92.
[22] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in *proc. of the First European Conf. on Artificial Life*. Elsevier Science, 92, pp. 134–142.
[23] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system:optimization by a colony of cooperating agents," *IEEE Trans. System, Man, and Cybernetics-Part B*, vol. 26, no. 1, pp. 1–13, Feb. 1996.
[24] L. M. Gambardella and M. Dorigo, "An ant colony system hybridized with a new local search for the sequential ordering problem," *INFORMS Journal on Computing*, vol. 12, no. 3, pp. 237–255, July 2000.
[25] S. Kamali and J. Opatrny, "Posant: A position based ant colony routing algorithm for mobile ad-hoc networks," in *Proc. Third International Conference on Wireless and Mobile Communications,ICWMC 07*, March 2007.
[26] L. Chen, J. Shen, L. Qin, and J. Fan, *A Method for Solving Optimization Problem in Continuous Space Using Improved Ant Colony Algorithm*, Y. Shi, W. Xu, and Z. Chen, Eds. Springer Berlin / Heidelberg, 2004, vol. 3327/2005.
[27] M. Kong and P. Tian, *Ant Colony Optimization and Swarm Intelligence*. Springer-Verlag, 2006, vol. 4150/2006, ch. A Direct Application of Ant Colony Optimization to Function Optimization Problem in Continuous Domain, pp. 324–331.
[28] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," 2006, article in press: Eur. J. Oper. Res.
[29] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization artificial ants as a computational intelligence technique," *IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE*, pp. 28–39, 2006.
[30] http://www.ee.washington.edu/research/pstca/.
[31] O. Alsac and B. Stott, "Optimal load flow with steady-state security," *IEEE Trans. Power Apparatus Systems*, vol. PAS-93, no. 3, pp. 745–751, May 1974.
[32] http://www.pserc.cornell.edu/matpower/.
[33] J. Hazra and A. K. Sinha, "Congestion management using multi objective particle swarm optimization," *IEEE Trans. Power Systems*, vol. 22, no. 4, pp. 1726–1734, Nov. 2007.
[34] Y. Liu and K. M. Passino, "Biomimicry of social foraging bacteria for distributed optimization: Models,principles, and emergent behaviors1," *Journal of Optimization Theory and Applications*, vol. 115, no. 3, pp. 603–628, Dec. 2002.