ATTRACTIVE AND REPULSIVE PARTICLE SWARM OPTIMIZATION, HYBRID ARTIFICIAL BEE ALGORITHM FOR SOLVING REACTIVE POWER OPTIMIZATION PROBLEM

K. Lenin[1], B. Ravindranath Reddy[2], M. Surya Kalavathi[3]
[1]Research Scholar, [2]Deputy executive engineer & [3]Professor
Electrical and Electronics Engineering, Jawaharlal Nehru Technological University
Kukatpally, Hyderabad 500 085, India.

ABSTRACT

Reactive Power Optimization is a complex combinatorial optimization problem involving non-linear function having multiple local minima, non-linear and discontinuous constrains. This paper presents Attractive and repulsive Particle Swarm Optimization (ARPSO) and Hybrid Artificial Bee Colony (HABC) applied for reactive optimization problem. It is one of the recent additions to the class of swarm intelligence based algorithms that mimics the foraging behaviour of honey bees. Artificial bee colony (ABC) consists of three groups of bees namely employed, onlooker and scout bees. In ABC, the food locations represent the potential candidate solution. In the present study an attempt is made to generate the population of food sources (Colony Size) adaptively and the variant is named as A-ABC. A-ABC is further enhanced to improve convergence speed and exploitation capability, by employing the concept of elitism, which guides the bees towards the best food source. This enhanced variant is called E-ABC. ARPSO and HABC are applied to Reactive Power Optimization problem and are evaluated on standard IEEE 30Bus System. The results show that ABC prevents premature convergence to high degree but still keeps a rapid convergence. It gives best solution when compared to Attractive and repulsive Particle Swarm Optimization (ARPSO) and Particle Swarm Optimization (PSO).

KEYWORDS: *Attractive and repulsive, particle Swarm, Artificial Bee Colony, Reactive Power Optimization, enhanced variant.*

I. INTRODUCTION

The reactive power optimization problem has a significant influence on secure and economic operation of power systems. The reactive power generation, although itself having no production cost, does however affect the overall generation cost by the way of the transmission loss. A procedure, which allocates the reactive power generation so as to minimize the transmission loss, will consequently result on the lowest production cost for which the operation constraints are satisfied. The operation constraints may include reactive power optimization problem. The conventional gradient-based optimization algorithm has been widely used to solve this problem for decades. Obviously, this problem is in nature a global optimization problem, which may have several local minima, and the conventional optimization methods easily lead to local optimum. On the other hand, in the conventional optimization algorithms, many mathematical assumptions, such as analytic and differential properties of the objective functions and unique minima existing in problem domains, have to be given to simplify the problem. Otherwise it is very difficult to calculate the gradient variables in the conventional methods. Further, in practical power system operation, the data acquired by the SCADA (Supervisory Control and Data Acquisition) system are contaminated by noise. Such data may cause difficulties in computation of gradients. Consequently, the optimization could not be carried out in many occasions. In the last decade, many new stochastic search methods have been developed for the global optimization problems such as simulated annealing, genetic algorithms and evolutionary programming.

A major problem with evolutionary algorithms (EAs) in multi-modal optimization is premature convergence (PC), which results in great performance loss and sub-optimal solutions. As far as GAs is concerned, the main reason for premature convergence is a too high selection pressure or a too high

gene flow between population individuals. With PSOs the fast information flow between particles seems to be the reason for clustering of particles.
Diversity declines rapidly, leaving the PSO algorithm with great difficulties of escaping local optima. Consequently, the clustering leads to low diversity with fitness stagnation as an overall result.

Attractive and repulsive algorithm

Recently R. Ursem has suggested a model called the Diversity-Guided Evolutionary Algorithm (DGEA) [1]. He redefines the traditional mutation operator, the Gaussian mutation, to be a directed mutation instead. The important issue is that this directed mutation, in general, increases the diversity, whereas normal Gaussian mutation is not likely to do this, because it simply adds random noise from some distribution with a mean of zero, normally $N(0; \sigma^2)$. Consequently, the DGEA applies diversity-decreasing operators (selection, recombination) and diversity-increasing operators (mutation) to alternate between two modes based upon a distance-to-average-point measure. The performance of the DGEA clearly shows its potential in multi-modal optimization. As [1] rightfully pinpoints, the diversity measure is traditionally used to *analyze the* evolutionary algorithms rather than *guide* them. We are great believers of *adaptive controlling*; that measuring and using different properties of the swarm/population while running, adds significant potential to the algorithm. We have therefore adopted the idea from Ursem with the decreasing and increasing diversity operators used to control the population into the basic PSO model. We find, it is a natural modification of the PSO, and the idea behind it is surprisingly simple. The modified model uses a diversity measure to have the algorithm alternate between exploring and exploiting behavior. We introduce two phases' attraction and *repulsion*. By measuring the diversity we let the swarm alternate between these phases. As long as the diversity is above a certain threshold d_{low} the particles attract each other. When the diversity declines below d_{low} the particles simply switch and start to repel each other until the threshold d_{high} is met. With this simple scheme we obtain our modified model, which we have chosen to call the ARPSO model – the attractive and repulsive PSO. Artificial Bee colony, proposed by Karaboga in 2005 [17]-[18] is the newest algorithm that belongs to the swarm of swarm intelligence algorithms.

Artificial bee algorithm

ABC simulates the foraging behaviour of the bee colony and employs this intelligent foraging behaviour to solve numerical and engineering design optimization problems. Like other population based search algorithm, ABC [19] starts with a population of potential candidate solutions (food sources may be flower patch etc.) where the population size (in case of ABC, the number of food sources) are fixed in the beginning of the algorithm. However, practically analyzing it is very rare that the every patch will contain the same number of flowers. Keeping this in mind, in the present study we propose a modified variant of ABC in which the population of food sources change adaptively. The corresponding algorithm is named A-ABC which is further enhanced (E-ABC) by employing the concept of elitism and improving the exploitation capability where the bees are always guided towards the best food source (i.e. the one having the best fitness function value).

II. PROBLEM FORMULATION

The objective of the reactive power optimization problem is to minimize the active power loss in the transmission Network as well as to improve the voltage profile of the system. Adjusting reactive power controllers like Generator bus voltages, reactive Power of VAR sources and transformer taps performs reactive Power scheduling.

$$\textbf{min } P_L = \sum_{i=1}^{NB} P_i(X, Y, \delta) \quad \dots \quad (1)$$

Subject to

i) The control vector constraints

$$X_{min} \le X \le X_{max} \quad \dots \quad (2)$$

ii) The dependent vector constraints

$$Y_{min} \leq Y \leq Y_{max} \quad \ldots \quad (3)$$

and

iii) The power flow constraint

$$F(X, Y, \delta) = 0 \quad \ldots \quad (4)$$

where

$$X = [V_G, T, Q_C] \quad \ldots \quad (5)$$

$$Y = [Qg, V_L, I] \quad \ldots \quad (6)$$

NB	-	Number of buses in the system.
δ	-	Vector of bus phase angles
P^i	-	Real Power injection into the ith bus
V_G	-	Vector of Generator Voltage Magnitudes
T	-	Vector of Tap settings of on load Transformer Tap changer.
Q_C	-	Vector of reactive Power of switchable VAR sources.
V_L	-	Vector of load bus Voltage magnitude.
I	-	Vector of current in the lines.
P_L	-	Vector of current in the lines.

III. BASIC PSO MODEL

The basic PSO model consists of a swarm of particles moving in an n-dimensional, real valued search space of possible problem solutions. For the search space, in general, a certain quality measure, the fitness, is defined making it possible for particles to compare different problem solutions. Every particle has a position vector x and a velocity vector v. Moreover, each particle contains a small memory storing its own best position seen so far p and a global best position g obtained through communication with its fellow neighbor particles. This information flow is obtained by defining a neighborhood topology on the swarm telling particles about immediate neighbors.

The intuition behind the PSO model is that by letting information about good solutions spread out through the swarm, the particles will tend to move to good areas in the search space. At each time step t the velocity is updated and the particle is moved to a new position. This new position is simply calculated as the sum of the previous position and the new velocity:

$$\vec{x}(t+1) = \vec{x}(t) + \vec{\upsilon}(t+1) \quad (7)$$

The update of the velocity from the previous velocity to the new velocity is, as implemented in this paper, determined by:

$$\vec{\upsilon}(t+1) = \omega.\vec{\upsilon}(t) + \phi_1(\vec{p}(t)) + \phi_2(\vec{g}(t) - \vec{x}(t)), \quad (8)$$

where ϕ_1 and ϕ_2 are real numbers chosen uniformly and at random in a given interval, usually [0,2]. These values determine the significance of $\vec{p}(t)$ *and* $\vec{g}(t)$ respectively. The parameter w is the inertia weight and controls the magnitude of the old velocity $\vec{\upsilon}(t)$ in the calculation of the new velocity $\vec{\upsilon}(t+1)$.[2]

3.1 The Modified Model – ARPSO

We define the attraction phase merely as the basic PSO algorithm. The particles will then attract each other, since in general they attract each other in the basic PSO algorithm because of the information flow of good solutions between particles. We define the second phase repulsion, by "inverting" the velocity-update formula of the particles:

$$\vec{\upsilon}(t+1) = \omega.\vec{\upsilon}(t) - \phi_1(\vec{p}(t) - \vec{x}(t)) - \phi_2(\vec{g}(t) - \vec{x}(t)). \quad (9)$$

In the repulsion phase the individual particle is no longer attracted to, but instead repelled by the best known particle position vector g(t) and its own previous best position vector p(t).
In the attraction phase the swarm is contracting, and consequently the diversity decreases. When the diversity drops below a lower bound, d_{low}, we switch to the repulsion phase, in which the swarm expands due to the above inverted update-velocity formula (9). Finally, when a diversity of d_{high} is reached, we switch back to the attraction phase. The result of this is an algorithm that alternates between phases of exploiting and exploring – attraction and repulsion – low diversity and high diversity. The pseudo-code for the ARPSO algorithm is shown in Fig. 1 and 2.

```
Program PSO
  init();;
 while not done do
 setDirection();          //new!
 update Velocity();
 newPosition ();
 assginFitness();
 calculateDiversity();    // new!
```

Figure 1: The ARPSO algorithm.

```
Function setDirection
     if (dir > 0 && diversity < dLow) dir = -1;
     if (dir > 0 && diversity < dHigh) dir = 1;
```

Figure 2: setDirecton

The first of the two new functions, **setDirection** determines which phase the algorithm is currently in, simply by setting a sign-variable, dir, either to 1 or -1 depending on the diversity. In the second function, **calculateDiversity**, the diversity of the swarm (in the pseudo-code stored in the variable "diversity"), is set according to the diversity-measure:

$$diversity(S) = \frac{1}{|S|.|L|}.\sum_{t-1}^{|S|} \sqrt{\sum_{j-1}^{N} p_{ij} - \bar{p}_j)^2}, \tag{10}$$

where S is the Swarm, [S] is the swarmsize, [L] is the length of longest the diagonal in the search space, N is the dimensionality of the problem, p_{ij} is the j'th value of the I'th particle and pj is the j'th value of the average point p. Note that this diversity measure is independent of swarmsize, the dimensionality of the problem as well as the search range in each dimension.
Finally, the velocity-update formula, eqn. (9) is changed by multiplying the sign-variable direction to the two last terms in it. This decides directly whether the particles attract or repel each other:

$$\vec{\upsilon}(t+1) = \omega.\vec{\upsilon}(t) + dir(\phi_1(\vec{p}(t) - \vec{x}(t)) + \phi_2(\vec{g}(t) - \vec{x}(t))). \tag{11}$$

IV. ARTIFICIAL BEE COLONY

ABC is one of the newest algorithms based on the foraging behavior of insects. It tries to model natural behavior of real honey bees in food foraging. Honey bees use several mechanisms like waggle dance to optimally locate food sources and to search new ones. Waggle dance is a means of communication among bees by which the successful foragers share the information not only about the direction and distance of the food sources but also about the amount of nectar available to the other foragers. This information exchange among bees helps them in detecting the optimal food locations. In ABC, this collective cooperative behavior of bees is simulated as an optimization algorithm. Since ABC algorithm is simple in concept, easy to implement, and has fewer control parameters, it has been widely used in many fields. ABC algorithm has been applied successfully to a large number of various optimization problems [19], [20]-[34]. The colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. A bee waiting on the dance area for making a decision to choose a food source is called onlooker and one going to the food source visited by it before is named employed bee. The other kind of bee is scout bee that carries out random search for discovering new sources. The position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. In the algorithm, the first half of the colony consists of employed artificial bees and the second half constitutes the onlookers. The number of the employed bees or the onlooker bees is equal to the number of solutions in the population. At the first step, the ABC generates a randomly distributed initial population of *NP* solutions (food source positions), where NP denotes the size of population. Each solution *xi* where *i* =1, 2,..., *SN* is a *D*-dimensional vector, where *D* is the number of optimization parameters. After initialization, the population of the positions (solutions) is subjected to repeated cycles, *C* =1, 2,..., MCN of the search processes of the employed bees, the onlooker bees and scout bees. An employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) and tests the nectar amount (fitness value) of the new source (new solution). Provided that the nectar amount of the new one is higher than that of the previous one, the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one in her memory. After all employed bees complete the search process; they share the nectar information of the food sources and their position information with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount. As in the case of the employed bee, she produces a modification on the position in her memory and checks the nectar amount of the candidate source. Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the old one. An artificial onlooker bee chooses a food source depending on the probability value associated with that food source pi, calculated as Eq. (12):

$$p_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \tag{12}$$

Where *fiti* is the fitness value of the solution *i* which is proportional to the nectar amount of the food source in the position *i* and *SN* is the number of food sources which is equal to the number of employed bees. In order to produce a candidate food position from the old one in memory, the ABC uses the following Eq. (13):

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{13}$$

Where *k* ϵ{1, 2, …, *NP*} and *j*ϵ{1, 2, …, *D*} are randomly chosen indexes. Moreover, *k ≠* i. *Øij* is a random number between [-1, 1]. It controls the production of neighbor food sources around xij and represents the comparison of two food positions visible to a bee. This can be seen from Eq. (13), as the difference between the parameters of the *xij* and *xkj* decreases, the perturbation on the position *xij* decreases, too. Thus, as the search approaches to the optimum solution in the search space, the step length is adaptively reduced. After each candidate source position is produced and evaluated by the artificial bee, its performance is compared with that of its old one. If the new food source has equal or better quality than the old source, the old one is replaced by the new one. Otherwise, the old one is retained. If a position cannot be improved further through a predetermined named "limit", then that food source is assumed to be abandoned. The corresponding employed bee becomes a scout. The abandoned position will be replaced with a new food source found by the scout. Assume that the abandoned source *xi*, then the scout discovers a new food source to be replaced with *xi*. This operation can be defined as in Eq. (14): bee becomes a scout. The abandoned position will be replaced with a new food source found by the scout. Assume that the abandoned source *xi*, then the scout discovers a new food source to be replaced with *xi*. This operation can be defined as in Eq. (14):

$$x_i^j = x_{\min}^j + rand()(x_{\max}^j - x_{\min}^j) \qquad (14)$$

Pseudo code of the ABC Algorithm

```
1. Initialize the population of solutions xij (i=1,2,…,SN,
j=1,2,…,D).
2. Evaluate the population.
3. cycle (represented as G) = 1
repeat
4. Produce new solutions (food source positions) using equation (13).
5. Apply the greedy selection process between xij,G  and vij,G.
6. Calculate the probability values pi for the solutions xij,G using the equation (12).
In order to calculate the fitness values of solutions the following equation is employed:
```

$$fit_i = \begin{cases} \dfrac{1}{1+f_i} & if \ f_i \geq 0 \\ 1+abs(f_i) & if \ f_i < 0 \end{cases}$$

```
Normalize pi values into [0, 1].
7. Produce the new solutions (new positions) vij,G for the onlookers from the solutions xij,G , selected depending on pi, and evaluate them.
8. Apply the greedy selection process for the onlookers between xij,G and vij,G.
9. Determine the abandoned solution (source), if exists, and replace it with a new randomly produced solution xi for the scout using the equation (14).
10. Memorize the best food source position (solution) achieved so far.
11. Cycle = cycle+1.
12. Until cycle = Maximum Cycle Number (MCN).
```

4.1 Proposed Hybrid ABC Algorithms & its variants

In the proposed hybrid ABC algorithm an attempt is to generate adaptive Colony Size. First of all we randomly generate some initial population of solutions, which at a later stage keep on changing adaptively. The Food Source position (*SN*) of subsequent generations is taken as the average of the population size attribute from all individuals in the current population as follows:

```
1 Initialize the population of solutions xi,j,G. Then declare p as variable (Initialize p=0)
2 for (i=0; i<FoodNumber; i++)
3 int p = p + Foods[i][D];
4 FoodNumber = (int) (p/FoodNumber) + 0.5);
5 (Check population size for even as number of food sources equals the half of the colony size)
6 Then follow the steps 2 – 12 described above in the Pseudo code.
```

It can be observed from the above code that the population of food source (*SN*) depend on p and may vary in every generation. The only thing to be kept in mind is that the population size should be even in number as the number of the employed bees or the onlooker bees is equal to the number of solutions in the population. This is a profitable situation as it may reduce the number of function evaluations leading to a faster convergence. The proposed hybrid ABC is further modified by including in it a concept of elitism, due to which the bees are always guided towards the best food source. Here, the food source is generated as follows:

$$v_{i,j} = x_{best,j} + \phi_{ij}(x_{r1,j} - x_{k,j}) \qquad (15)$$

Where *xbest,j* indicates the best food location. The remaining notations have the same meaning as defined in the previous section. This particular modification will further aid in getting a faster convergence. The bees always look for the best solution. In the other variant of the proposed HABC algorithm is further implemented to improve the exploitation of ABC by using the equation given below:

$$v_{i,j} = x_{i,j} + \phi_{ij}(x_{i,j} - x_{k,j}) + C(x_{best,j} - x_{r1,j}) \qquad (16)$$

Where C is taken as 1.5. This variant is inspired from the Particle Swarm Optimization (PSO) [35] (though the equation is not exactly same) which takes care of global as well as local exploration so that the domain of the problem is thoroughly is explored.

V. ALGORITHM FOR REACTIVE OPTIMIZATION PROBLEM

Step 1. Initial searching points and velocities of agents are generated.

Step 2. Ploss to the searching points for each agent is calculated using the load flow calculation. If the constraints are violated, the penalty is added to the loss (evaluation value of agent).

The fitness function of each particle is calculated as:

$$f_n = P^n{}_L + \alpha \sum_{j=1}^{NG} Q_{G,j}^{\lim,n} + \beta \sum_{j=1}^{NL} V_{L,j}^{\lim,n} \;;\; n = 1,2..,N_n \qquad \ldots \qquad (17)$$

α, β	=	penalty factors
P^n_L	=	total real power losses of the n-th particle

$$Q_{G,j}^{\lim,n} = \begin{cases} Q_{G,\min} - Q_{G,j}^{n} & \text{if } Q_{G,j}^{n} < Q_{G,\min} \\ Q_{G,j}^{n} - Q_{G,\max} & \text{if } Q_{G,j}^{n} > Q_{G,\max} \end{cases} \qquad \ldots \qquad (18)$$

and

$$V_{L,j}^{\lim,n} = \begin{cases} \left| V_{L,j}^{n} \right| - V_{L,\max}, & \text{if } \left| V_{L,j}^{n} \right| > V_{L,\max} \\ 0 & \text{otherwise} \end{cases} \qquad \ldots \qquad (19)$$

Step 3. Pbest is set to each initial searching point. The initial best evaluated value (loss with penalty) among pbests is set to gbest.

Step 4. New velocities are calculated using eqn. (7).

Step 5. Update the velocity from previous velocity to the new velocity using eqn. (8)**.**

Step 6. To new function applied.

i. ***setdirection***

ii. calculateDiversity to control swarm.

Step 7. Ploss to the new searching points and the evaluation values are calculated.

Step 8. If the evaluation value of each agent is better than the previous pbest, the value is set to pbest. If the best pbest is better than gbest, the value is set to gbest. All of gbests are stored as candidates for the final control strategy.

Step 9. If the iteration number reaches the maximum iteration number, then stop. Otherwise, go to Step 4. If the voltage and power flow constraints are violated, the absolute violated value from the maximum and minimum boundaries is largely weighted and added to the objective function (1) as a penalty term.

VI. SIMULATION RESULTS

NB = 30, ***NL*** = 41, ***NG*** = 6, ***NTR*** = 4, Population size = 50
d_{low} = 0.01, d_{high} = 0.1

Table 1 Optimal Control values

	ARPSO	**PSO**	**HABC**
VG1	1.05	1.06	1.04
VG2	1.03	1.04	1.03
VG3	1.01	1.01	1.01
VG4	1.01	1.02	1.01
VG5	1.07	1.09	1.07
VG6	1.08	1.08	1.09
T_1	0.99	0.98	0.93
T_2	0.95	0.95	0.99
T_3	1.00	1.00	1.04
T_4	0.94	0.93	0.99

Table 2 Parameter sensitivity analysis of IEEE 30 (100 trails)

Method	Compared item	IEEE 30 bus	Time (sec)	Iterations
ARPSO	Min. loss	9.4769	8.227	178
	Avg. loss value	9.4792		
PSO	Min. loss	9.4911	12.425	225
	Avg. loss value	9.5001		
HABC	Min. loss	9.4689	7,897	148
	Avg. loss value	9.4700		

Table 3 Parameter sensitivity analysis of IEEE 30 (100 trails)

W_{max} W_{min}		C_i				
		1.0	1.5	2.0	2.5	3.0
0.9 0.4	Avg.	9.4799	9.4827	9.4827	9.4827	9.4827
	Min	9.4769	9.4818	9.4817	9.4817	9.4817
2.0 0.9	Avg.	9.4827	9.4826	9.4826	9.4823	9.4825
	Min	9.4818	9.4818	9.4818	9.4817	9.4817
2.0 0.4	Avg.	9.4827	9.4827	9.4827	9.4827	9.4827
	Min	9.4819	9.4819	9.4819	9.4819	9.4819

W = Weight function for velocity of agent
C_i = Weight co-efficient for each term

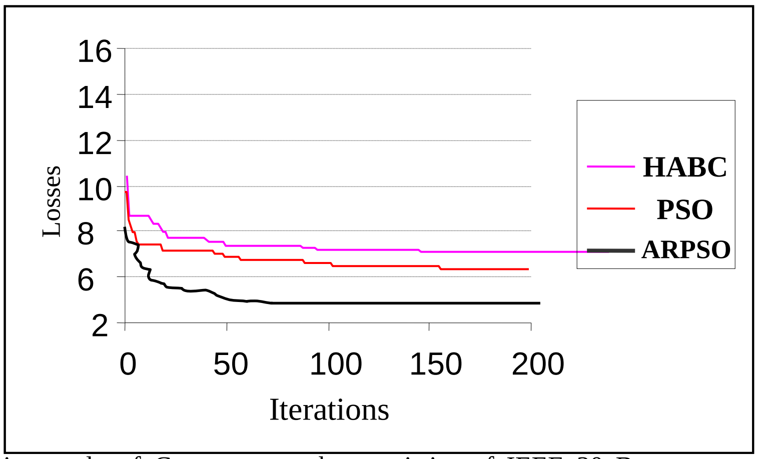

Fig. 3 Comparative study of Convergence characteristics of IEEE 30 Bus system with ARPSO, PSO and HABC.

VII. NOMENCLATURE

NB = total no. of buses, ***NL*** = total no. of load buses, ***NG*** = total no. of generator buses
TR = total no. of transformers, ***VG*** = generator voltage $\boldsymbol{V_g}$ is a vector of generator bus voltages.
$\boldsymbol{Q_s}$ is a vector of switchable VAR sources and
T is a vector of tap settings of on-load tap changing (OLTC) of transformers.
Q_g is a vector of reactive power generations of the generator buses and V_L is a vector of load bus voltages.

VIII. CONCLUSION

In this paper ARPSO and HABC algorithm has been developed for determination of global optimum solution for reactive power optimization problem. The performance of the proposed algorithm demonstrated through its evaluation on IEEE 30 bus power system shows that HABC is able to undertake global search with a fast converges rate and a future of robust computation. From the simulation study it has been found that HABC converges to the global optimum than PSO and ARPSO.

IX. FUTURE WORK

In future proposed algorithm can be applied to practical utility system and also HABC algorithm can be used in solving other power system problems like unit commitment, economic dispatch problem, and voltage curtailment problem.

REFERENCES

[1]. R.K.Ursem, ‘Diversity guided evolutionary algorithms’, in submission for the problem solving from nature conference, (PPSN 7)
[2]. Y.Fukuyama, ‘A Particle Swarm Optimization for Reactive Power and voltage control in Electric Power System’, IEEE Trans. on Power Systems, pp. 87 – 93, 2001.
[3]. Jakob S. Vesterstrom and Jacques Riget, ‘ A Diversity-Guided Particle Swarm Optimizer the ARPSO’, EVALife Technical report no. 2002.
[4]. Y. Fukuyama, et al., "Practical Distribution StateEstimation Using Hybrid Particle Swarm Optimization",Proc. of IEEE Power Engineering Society Winter Meeting, Columbus, 2001.
[5]. R. C. Eberhart and Y. Shi, "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization", Proc. of CEC 2000.
[6]. Youxin Luo , “Optimization for PID control parameters on hydraulic servo control system based on random virus algorithm” advance computer control (ICACC) 2 international conference vol 3 pp 432-435 march 2010 .
[7]. K.Y Lee ,Y.M Park , and J.L Oritz, “Fuel –cost optimization for both real and reactive power dispatches” , IEE Proc; 131C,(3), pp.85-93.
[8]. M.K. Mangoli, and K.Y. Lee, “Optimal real and reactive power control using linear programming” , Electr.Power Syst.Res, Vol.26, pp.1-10,1993.
[9]. S.R.Paranjothi ,and K.Anburaja, “Optimal power flow using refined genetic algorithm”, Electr.Power Compon.Syst , Vol. 30, 1055-1063,2002.
[10]. D. Devaraj, and B. Yeganarayana, “Genetic algorithm based optimal power flow for security enhancement”, IEE proc-Generation.Transmission and. Distribution; 152, 6 November 2005.
[11]. Luo, F.F., Chen, G.L., Guo, W.Z.: An improved ‘fish-search’ algorithm for information retrieval. In: Proceedings of IEEE International Conference on Natural Language, Processing and Knowledge Engineering (NLP-KE 2005), Wuhan, China, pp. 523–528 (2005)
[12]. Carmelo J.A.Bastos Filho , Fernando B. de Lima Neto , Anthony J.C.C.Lins , Antonio I.S. Nascimento , Marilia P.Lima , Fish school search nature – inspired algorithms for optimization studies in computational intelligence vol 193 , 2009 , pp 261-277
[13]. C.A. Canizares , A.C.Z.de Souza and V.H. Quintana , “ Comparison of performance indices for detection of proximity to voltage collapse ,’’ vol. 11. no.3 , pp.1441-1450, Aug 1996
[14]. B.Gao ,G.K Morison P.Kundur ’voltage stability evaluation using modal analysis ‘ Transactions on Power Systems ,Vol 7, No .4 ,November 1992.
[15]. D. Dasgupta, Artificial Immune Systems and Their Applications, Springer, Berlin, 1999
[16]. Cayzer S and Aickelin U (2002), A Recommender System based on the ImmuneNetwork, in Proceedings CEC2002, pp 807-813, Honolulu, USA.

[17]. Cayzer S and Aickelin U (2002b), On the Effects of Idiotypic Interactions for Recommendation Communities in AIS, Proceedings 1st International Conference on AIS, pp 154-160, Canterbury, UK.

[18]. D. Karaboga, An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Kayseri, Turkey: Erciyes University; 2005.

[19]. D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, Journal of Global Optimization 39(2007) 171–459.

[20]. D. Karaboga, B. Basturk, On the performance of artificial becolony (ABC) algorithm, Applied Soft Computing, 8(2008) 687- 697.

[21]. D. Karaboga, C. Ozturk, N. Karaboga, B. Gorkemli, Artificial bee colony programming for symbolic regression, Information Sciences (2012), http:// dx.doi.org/10.1016/j.ins.2012.05.002.

[22]. M Ma, J. Liang, M. Guo, Y. Fan, Y. Yin, SAR image segmentation based on Artificial Bee Colony algorithm, Applied Soft Computing, (In Press) doi:10.1016/j.asoc.2011.05.039, in press.

[23]. WC. Yeh, TJ. Hsieh, Artificial bee colony algorithm-neural networks for s-system models of biochemical networks approximation. Neural Comput Appl. (2012) doi:10.1007/s00521- 010-0435-z.

[24]. F. Gao, Feng-xia Fei, Q. Xu, Y. fang Deng, Yi-bo Qi, I. Balasingham, A novel artificial bee colony algorithm with space contraction for unknown parameters identification and time-delays of chaotic systems, Appl. Math. Comput. (2012),http://dx.doi.org/10.1016/j.amc.2012.06.040.

[25]. H. Zhang, Y. Zhu, W. Zou, X. Yan, A hybrid multi-objective artificial bee colony algorithm for burdening optimization of copper strip production, Applied Mathematical Modelling, 36:6(2012) 2578-2591.

[26]. Liao, J. Zhou, R. Zhang, Y. Zhang, An adaptive artificial bee colony algorithm for long-term economic dispatch in cascaded hydropower systems, International Journal of Electrical Power & Energy Systems, 43:1(2012) 1340-1345.

[27]. H. Gozde, M. Cengiz Taplamacioglu, İ. Kocaarslan, Comparative performance analysis of Artificial Bee Colony algorithm in automatic generation control for interconnected reheat thermal power system, International Journal of Electrical Power & Energy Systems 42:1(2012) 167-178.

[28]. Ali R. Yildiz, A new hybrid artificial bee colony algorithm for robust optimal design and manufacturing, Applied Soft Computing, In Press, 2012.

[29]. O. Kisi, C. Ozkan, B. Akay, Modeling discharge–sediment relationship using neural networks with artificial bee colony algorithm, Journal of Hydrology, 428–429(2012) 94-103.

[30]. S. K. Mandal, Felix T.S. Chan, M.K. Tiwari, Leak detection of pipeline: An integrated approach of rough set theory and artificial bee colony trained SVM, Expert Systems with Applications, 39:3(2012) 3071-3080.

[31]. T.K. Sharma, M. Pant, Enhancing the food locations in an artificial bee colony algorithm, in: IEEE Swarm Intelligence Symposium (SIS), Paris, France, 2011, pp. 119-123.

[32]. T.K. Sharma, M. Pant, Enhancing different phases of artificial bee colony for continuous global optimization problems, in: International Conference on Soft Computing for Problem Solving, SocProS 2011, AISC of Advances in Intelligent and Soft Computing, Roorkee, India, Vol. 130, 2011, pp. 715–724.

[33]. T.K. Sharma, M. Pant, J.C. Bansal, Artificial Bee Colony with Mean Mutation Operator for Better Exploitation, in: IEEE World Congress on Computational Intelligence (CEC), Brisbane, Australia, 2012, pp. 3050 - 3056.

[34]. T.K. Sharma, M. Pant, J.C. Bansal, Some Modifications to Enhance the Performance of Artificial Bee Colony, in: IEEE World Congress on Computational Intelligence (CEC), Brisbane, Australia, 2012, pp. 3454 - 3461.

[35]. Tarun Kumar Sharma, Millie Pant, V.P. Singh, Improved Local Search in Artificial Bee Colony using Golden Section Search, Journal of Engineering, 1:1(2012) 14-19.

[36]. Zhu G., Kwong S.: Gbest-Guided Artificial Bee Colony Algorithm for Numerical Function Optimization, Appl. Math. Comput. (2010).

APPENDIX

Table 4. Line data – 30-bus system

Branch No.	From	To	R (p.u)	X (p.u)	Y/2 (p.u)	Line phase angle limit (deg.)
1	2	1	0.0192	0.0575	0.0264	5.0
2	1	3	0.0452	0.1852	0.0204	17.0

3	2	4	0.0570	0.1737	0.0184	7.0
4	3	4	0.0132	0.0379	0.0042	3.5
5	2	5	0.0472	0.1983	0.0209	15.0
6	2	6	0.0581	0.1763	0.0187	7.0
7	4	6	0.0119	0.0414	0.0045	2.5
8	5	7	0.0460	0.1160	0.0102	5.5
9	6	7	0.0267	0.0820	0.0085	6.0
10	6	8	0.0120	0.0420	0.0045	2.0
13	9	11	0.0000	0.2080	0.0000	4.0
14	9	10	0.0000	0.1100	0.0000	4.0
16	12	13	0.0000	0.1400	0.0000	4.0
17	12	14	0.1231	0.2559	0.0000	5.0
18	12	15	0.0662	0.1304	0.0000	3.0
19	12	16	0.0945	0.1987	0.0000	4.0
20	14	15	0.2210	0.1997	0.0000	2.5
21	16	17	0.0824	0.1932	0.0000	2.0
22	15	18	0.1070	0.2185	0.0000	2.5
23	18	19	0.0639	01292	0.0000	1.5
24	19	20	0.0340	0.0680	0.0000	3.0
25	10	20	0.0360	0.2090	0.0000	4.0
26	10	17	0.0324	0.0845	0.0000	2.0
27	10	21	0.0348	0.0749	0.0000	2.0
28	10	22	0.0727	0.1499	0.0000	2.0
29	21	22	0.0116	0.0236	0.0000	1.5
30	15	23	0.1000	0.2020	0.0000	3.0
31	22	24	0.1150	0.1790	0.0000	3.5
32	23	24	0.1320	0.2700	0.0000	3.0
33	24	25	0.1885	0.3292	0.0000	2.0
34	25	26	0.2544	0.3800	0.0000	2.0
35	25	27	0.1093	0.2087	0.0000	2.5
37	27	29	0.2198	0.4153	0.0000	3.5
38	27	30	0.3202	0.6027	0.0000	5.0
39	29	30	0.2399	0.4533	0.0000	4.5
40	8	28	0.0636	0.2000	0.0214	4.0
41	6	28	0.0169	0.0599	0.0065	3.0

Table 5. Transformer data – 30-bus system

Branch No.	From	To	R (p.u)	X (p.u)	Tap	Tap max	Tap min	Tap step
11	6	9	0.0000	0.2080	1.0155	1.1000	0.9000	0.0250
12	6	10	0.0000	0.5560	0.9629	1.1000	0.9000	0.0250
15	4	12	0.0000	0.2560	1.0129	1.1000	0.9000	0.0250
36	28	27	0.0000	0.3960	0.9581	1.1000	0.9000	0.0250

AUTHORS

K. Lenin has received his B.E., Degree, electrical and electronics engineering in 1999 from university of madras, Chennai, India and M.E., Degree in power systems in 2000 from Annamalai University, TamilNadu, India. At present pursuing Ph.D., degree at JNTU, Hyderabad, India.

Bhumanapally. RavindhranathReddy, Born on 3rd September,1969. Got his B.Tech in Electrical & Electronics Engineering from the J.N.T.U. College of Engg., Anantapur in the year 1991. Completed his M.Tech in Energy Systems in IPGSR of J.N.T. University Hyderabad in the year 1997. Obtained his doctoral degree from JNTUA, Anantapur University in the field of Electrical Power Systems. Published 12 Research Papers and presently guiding 6 Ph.D. Scholars. He was specialized in Power Systems, High Voltage Engineering and Control Systems. His research interests include Simulation studies on Transients of different power system equipment.

M. Surya Kalavathi has received her B.Tech. Electrical and Electronics Engineering from SVU, Andhra Pradesh, India and M.Tech, power system operation and control from SVU, Andhra Pradesh, India. She received her Phd. Degree from JNTU, hyderabad and Post doc. From CMU – USA. Currently she is Professor and Head of the electrical and electronics engineering department in JNTU, Hyderabad, India and she has Published 16 Research Papers and presently guiding 5 Ph.D. Scholars. She has specialised in Power Systems, High Voltage Engineering and Control Systems. Her research interests include Simulation studies on Transients of different power system equipment. She has 18 years of experience. She has invited for various lectures in institutes.