ISSN : 2348 - 9065 (Online)
ISSN : 2349 - 3143 (Print)
Vol. 3, Issue 4 Oct. - Dec. 2016
International Journal of Research in Electronics and Communication Technology (IJRECT 2016)

Active Power Loss Reduction by Allure & Revulsion Type Particle Swarm Optimization, Arbitrary Virus Algorithm

Dr.K.Lenin
Researcher, JNTU, Hyderabad, India

Abstract
This paper presents Allure& Repulsion Particle Swarm Optimization (ARPSO), Arbitrary Virus Algorithm (AVA) are applied to Reactive Power Optimization problem and evaluated on in standard IEEE 30Bus System. The results show that bothARPSO, AVA prevents premature convergence to extraordinary level but still keeps a quick convergence. Proposed ARPSO, AVA algorithms has been compared with other standard algorithms and simulation study indicates the efficiency of the proposed algorithm in reducing the real power loss.

Key words
Allure & Revulsion, particle Swarm,Arbitrary virus algorithm, Reactive Power Optimization.

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.

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 Arbitrary 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' Allureand *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.Arbitrary Virus Algorithm (AVA) [6] a new, evolutionary type algorithm is proposed to the multidisciplinary optimization. The algorithm is based on the simulation of spreading of biological viruses, which results in a boom-like propagation in the number of entities and quick improvement of the objective function in subsequent iterations. A virus is always a very simple construction, contains only the most important information necessary for life and reproduction. This simplicity gives a very high flexibility to a virus in changing and mutation, therefore they can accommodate to several conditions very easily. Therefore the efficiency of a virus is very high in point of view of behaviour, construction, and life reproduction and changing. Both ARPSO, AVA has been applied to solve the reactive power problem & evaluated in standard IEEE 30 bus system.

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

ISSN : 2348 - 9065 (Online)
ISSN : 2349 - 3143 (Print)

reactive power controllers like Generator bus voltages, reactive Power of VAR sources and transformer taps performs reactive Power scheduling.

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

Subject to

i) The control vector constraints

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

ii) The dependent vector constraints

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

and

iii) The power flow constraint

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

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 i [th]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 Arbitraryin a given inteAVAl, 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]

1. The Modified Model – ARPSO

We define the Allure 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 Allure 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 Allure phase. The result of this is an algorithm that alternates between phases of exploiting and exploring – Allure 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!
```

Fig. 1: The ARPSO algorithm.

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

Fig. 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:

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

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:

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

IV. Algorithm for Reactive Power Optimization using ARPSO

The proposed Reactive Power Optimization algorithm using the ARPSO can be expressed as follows:

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:

$$f_n = P^n{}_L + \alpha \sum_{j=1}^{NG} Q_{G,j}^{lim,n} + \beta \sum_{j=1}^{NL} V_{L,j}^{lim,n} \; ; n = 1,2.., \; N_n \quad \ldots(12)$$

α, β = penalty factors

P^n_L = total real power losses of the n^{-th} particle

$$Q_{G,j}^{lim,n} = \begin{cases} Q_{G,min} - Q_{G,j}^n & \text{if } Q_{G,j}^n < Q_{G,min} \\ Q_{G,j}^n - Q_{G,max} & \text{if } Q_{G,j}^n > Q_{G,max} \end{cases} \quad \ldots \quad (13)$$

and

$$V_{L,j}^{lim,n} = \begin{cases} |V_{L,j}^n| - V_{L,max}, & \text{if } |V_{L,j}^n| > V_{L,max} \ldots \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

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. The maximum iteration number should be determined by pre-simulation. As mentioned below, PSO requires less than 100 iterations even for large-scale problems.

V. Arbitrary Virus Algorithm

A new evolutionary type algorithm is proposed to the multidisciplinary optimization. The algorithm is based on the simulation of spreading of biological viruses, which results in a boom-like propagation in the number of entities and quick improvement of the objective function in subsequent iterations. It is possible to control the number of entities if in each iteration step 20- 30% of the entities (e.g. the ones having the worst objective function value or those having child's and grandchild's all with worst objective function values than themselves) are eliminated.

1. The concept of the Arbitrary Virus Algorithm (AVA)

Let us investigate a very efficient biological construction: a virus. The efficiency of this biological system is in his very fast reproduction capacity. If the circumstances and conditions (temperature, light, oxygen, and food) are good, viruses can reproduce themselves in a very high speed and they will cover almost all the possible places in the area of investigation. If the life conditions are not good, or the conditions show a non-uniform distribution, higher number of viruses can be found in an area of better conditions than in some areas giving poor conditions of life for viruses.

Therefore these biological structures are very efficient in finding the best conditions of life, the highest number of virus entities will be found in the area having the best life conditions. Another very important thing is that a virus is always a very simple construction, contains only the most important information necessary for life and reproduction. This simplicity gives a very high flexibility to a virus in changing and mutation, therefore they can accommodate to several conditions very easily. Therefore the efficiency of a virus is very high in point of view of behaviour, construction, and life reproduction and changing. This efficiency is applied during the development of computer viruses, which show several similarities to real biological viruses (simple structures, very fast reproduction, easy changing). Mainly these characteristics give that a computer virus is also a very efficient system. Applying this multi-form efficiency in development of an optimization algorithm could result in high efficiency in the optimum searching process, too. The algorithm presented here destroys 30% of points having the worst objective function value, after the third generation step. This gives high efficiency in the starting phase of the searching process and later the total number of entities will be limited. This is also good in point of view of small number of objective function evaluations and constraint checking to reach the final optimum. On the basis of the above considerations, it is possible to build the algorithm. First step is to find the starting points fulfilling all the explicit and implicit constraints. It is possible to generate point coordinates inside of explicit constraints limits and check the generated points against the implicit constraints. In order to keep the simplicity of the virus algorithm, the number of the starting points is proposed to be set at a very low value. The coordinates of the starting points can be denoted by x_i,i = 1,2,…,n where n is the number of design variables. In this case the points can be denoted as P_j, j= 1,2,…,m , where m is the number of the starting points. The implicit constraints of the design variables:

$l_i \le x_i \le h_i$ i = 1,2,..,n (15)

l_i are the lower limits, h_i are upper limits of the constrains. The implicit constraints can be written in the following form:

$u_k \le f_k(x_i) \le v_k$ I = 1,2,..,n , k=1,2,….,p (16)

Where p is the number of implicit constraints. The starting points are vectors in the design space:

$P_j = \{x_i\}j$ (17)

They can be found in the feasible region of explicit constraints

ISSN : 2348 - 9065 (Online)
ISSN : 2349 - 3143 (Print)

by using Arbitrarynumbers and after they are checked against the implicit constraints. If a point is unfeasible, a new one should be generated.

The goal of the optimization process is to find the extreme value (maximum or minimum) of the objective function:

$\Omega = F(xi)$ (18)

F is an arbitrary nonlinear function of the design variables. Once the starting points generated, the reproduction procedure is starting:

$y_i = x_i + R_i\, q(h_i - l_i)$ (19)

Here y_i are the coordinates of the new point generated, R_i are Arbitrary numbers between the values of 0 and 1 and α is the number of new entities generated in the reproduction procedure, q is the spreading parameter. Proposed value of spreading parameter is between 0.5 and 0.8 in case of the first three generations and between 0.2 and 0.4 afterwards. The reproduction step is executed for each starting point.The next generation (we can call it generation β, after γ and so on) can be created using the reproduction formula of equation (19) for each point of the previous generation. In order to prevent the overwhelmingnumber of points controlled at the same time, it is necessary to select the points having the best objective function value and destroy the points having the worst objective function value. This procedure can be continued until a given number of generations are reached or the procedure can be ended if the maximum difference in objective function values regarding the last generation created will be under a given small value.In some cases, where the evaluation of the objective function needs a large amount of calculations, (for example solution of a large finite element model) it is possible to insert a security upper limit for the total number of calculations of the objective function or a given constraint checking, in order to prevent too much amounts of calculations. It is recommended to consider all these parameters before ending the optimization procedure.

VI. Simulation Results

Validity of proposed ARPSO,AVA algorithms has been verified by testing in IEEE 30-bus, 41 branch system and it has 6 generator-bus voltage magnitudes, 4 transformer-tap settings, and 2 bus shunt reactive compensators. Bus 1 is taken as slack bus and 2, 5, 8, 11 and 13 are considered as PV generator buses and others are PQ load buses. Optimal Control variables limits are given in Table 1.*NB* = 30, *NL* = 41, *NG* = 6, *NTR* = 4 Population size = 50,d_{low} = 0.01, d_{high} = 0.1.Table 2 gives the parameter analysis of application of the proposed algorithm. Finally Table 3 gives the overall comparison of proposed algorithms along with other standard reported algorithms.

Table 1 : Optimal Control values

	ARPSO	**AVA**
VG1	1.04	1.03
VG2	1.02	1.03
VG3	1.03	1.04
VG4	1.02	1.01
VG5	1.06	1.07
VG6	1.07	1.06
T_1	0.90	0.92
T_2	0.94	0.93
T_3	1.00	1.00
T_4	0.92	0.90

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

Method	Compared item	IEEE 30 bus	Time (sec)	Iterations
ARPSO	Min. loss	4.1686	7.092	22
	Avg. loss value	4.1965		
AVA	Min. loss	4.1695	7.902	31
	Avg. loss value	4.1974		

Table 3 : Comparison of results

Techniques	**Real power loss (MW)**
SGA(Wu et al., 1998) [17]	4.98
PSO(Zhao et al., 2005) [18]	4.9262
LP(Mahadevan et al., 2010) [19]	5.988
EP(Mahadevan et al., 2010) [19]	4.963
CGA(Mahadevan et al., 2010) [19]	4.980
AGA(Mahadevan et al., 2010) [19]	4.926
CLPSO(Mahadevan et al., 2010) [19]	4.7208
HSA (Khazali et al., 2011) [20]	4.7624
BB-BC (Sakthivel et al., 2013) [21]	4.690
MCS(Tejaswini sharma et al.,2016) [22]	4.87231
Proposed ARPSO	**4.1965**
Proposed AVA	**4.1974**

VII. Conclusion

In this paper ARPSO, AVA algorithms havebeen successfully solved the Reactive power optimization problem. The performance of the proposed algorithms demonstrated through its evaluation on IEEE 30 bus power system&from the simulation study it has been found that both the ARPSO,AVA efficiently reduces the real power loss when compared to other standard reported algorithms.

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 .*

[6] 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.

[7] M.K. Mangoli, and K.Y. Lee, "Optimal real and reactive power control using linear programming" , Electr.PowerSyst. Res, Vol.26, pp.1-10,1993.

[8] S.R.Paranjothi ,andK.Anburaja, "Optimal power flow using refined genetic algorithm", Electr.PowerCompon.Syst , Vol. 30, 1055-1063,2002.

[9] D. Devaraj, and B. Yeganarayana, "Genetic algorithm based optimal power flow for security enhancement", IEE proc-Generation.Transmission and. Distribution; 152, 6 November 2005.

[10] 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)

[11] Carmelo J.A.BastosFilho , 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

[12] 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

[13] B.Gao ,G.K Morison P.Kundur 'voltage stability evaluation using modal analysis ' Transactions on Power Systems ,Vol 7, No .4 ,November 1992.

[14] D. Dasgupta, Artificial Immune Systems and Their Applications, Springer, Berlin, 1999

[15] Cayzer S and Aickelin U (2002), A Recommender System based on the ImmuneNetwork, in Proceedings CEC2002, pp 807-813, Honolulu, USA.

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

[17] Wu.Q.H,Y.J.Cao,andJ.Y.Wen,(1998),"Optimal reactive power dispatch using an adaptive genetic algorithm", Int.J.Elect.Power Energy Syst. Vol 20. Pp. 563-569.

[18] Zhao.B,C.X.Guo,andY.J.CAO,(2005),"Multiagent-based particle swarm optimization approach for optimal reactive power dispatch",IEEE Trans. Power Syst. Vol. 20, no. 2, pp. 1070-1078.

[19] Mahadevan.K,KannanP.S,(2010)"Comprehensive Learning Particle Swarm Optimization for Reactive Power Dispatch", Applied Soft Computing, Vol. 10, No. 2, pp. 641–52.

[20] Khazali.A.H,M.Kalantar,(2011),"Optimal Reactive Power Dispatch based on Harmony Search Algorithm", Electrical Power and Energy Systems, Vol. 33, No. 3, pp. 684–692.

[21] Sakthivel.S,M.Gayathri,V.Manimozhi,(2013),"A Nature Inspired Optimization Algorithm for Reactive Power Control in a Power System", InternationalJournalofRecentTechnologyandEngineering,pp29-33Vol.2,Issue-1.

[22] TejaswiniSharma,LaxmiSrivastava,Shishir Dixit (2016). "Modified Cuckoo Search Algorithm For Optimal Reactive Power Dispatch", Proceedings of 38 th IRF International Conference,pp4-8. 20th March, 2016, Chennai, India, ISBN: 978-93-85973-76-5.