[Lenin *, Vol.6 (Iss.4): April 2018] ISSN- 2350-0530(O), ISSN- 2394-3629(P)
(Received: Mar 11, 2018 - Accepted: Apr 30, 2018) DOI: 10.5281/zenodo.1248188

INTERNATIONAL JOURNAL OF RESEARCH – GRANTHAALAYAH
A knowledge Repository

Science

LUCID PARTICLE SWARM OPTIMIZATION ALGORITHM FOR SOLVING OPTIMAL REACTIVE POWER PROBLEM

Dr. K. Lenin [*1]
[*1] Professor, Department of EEE, Prasad V. Potluri Siddhartha Institute of Technology, Kanuru, Vijayawada, Andhra Pradesh -520007, India

Abstract

This paper presents, Lucid Particle Swarm Optimization (LPSO) algorithm for Solving Optimal Reactive Power Problem. Particle swarm method is typically made up of a population of simple agents intermingle locally with one another and with their surroundings, with the aim of locating the optima within the operational environment. In this paper, a robust and Lucid particle swarm optimization framework based on multi-agent system is presented, where learning capabilities are integrated into the particle agents to dynamically fiddle with their optimality behaviours. Self-Sufficiency is achieved by the use of communicators that separate an agent's individual operation from that of the swarm, thereby making the system more robust. The proposed Lucid Particle Swarm Optimization (LPSO) algorithm has been tested on standard IEEE 118 & practical 191 bus test systems and simulation results show clearly about the premium performance of the proposed algorithm in reducing the real power loss.

Keywords: Optimal Reactive Power; Transmission Loss; Lucid Particle Swarm Optimization.

Cite This Article: Dr.K. Lenin. (2018). "LUCID PARTICLE SWARM OPTIMIZATION ALGORITHM FOR SOLVING OPTIMAL REACTIVE POWER PROBLEM." *International Journal of Research - Granthaalayah*, 6(4), 312-324. https://doi.org/10.5281/zenodo.1248188.

1. Introduction

Optimal reactive power problem is to minimize the real power loss and bus voltage deviation. Various numerical methods like the gradient method [1-2], Newton method [3] and linear programming [4-7] have been adopted to solve the optimal reactive power dispatch problem. Both the gradient and Newton methods have the complexity in managing inequality constraints. If linear programming is applied then the input- output function has to be uttered as a set of linear functions which mostly lead to loss of accuracy. The problem of voltage stability and collapse play a major role in power system planning and operation [8]. Evolutionary algorithms such as genetic algorithm have been already proposed to solve the reactive power flow problem [9-11]. Evolutionary algorithm is a heuristic approach used for minimization problems by utilizing nonlinear and non-differentiable continuous space functions. In [12], Hybrid differential evolution algorithm is proposed to improve the voltage stability index. In [13] Biogeography Based algorithm is projected to solve the reactive power dispatch problem. In [14], a fuzzy based method is used to solve the optimal reactive power scheduling method. In [15], an improved evolutionary programming is used to solve the optimal reactive power dispatch problem. In [16], the optimal reactive power flow problem is solved by integrating a genetic algorithm with a nonlinear interior point method. In [17], a pattern algorithm is used to solve ac-dc optimal reactive power flow model with the generator capability limits. In [18], F. Capitanescu proposes a two-step approach to evaluate Reactive power reserves with respect to operating constraints and voltage stability. In [19], a programming based approach is used to solve the optimal reactive power dispatch problem. In [20], A. Kargarian et al present a probabilistic algorithm for optimal reactive power provision in hybrid electricity markets with uncertain loads. This paper presents, Lucid Particle Swarm Optimization (LPSO) algorithm for Solving Optimal Reactive Power Problem. Implementing the Particle Swarm Optimization (PSO) method from conventional supervised approach has a number of setbacks. To begin with, it is deficient in the autonomy of the system which nature deserves; the particles attain their goals by performing a fully detailed program, and they are restricted by the highly unified execution, since they have a consistent algorithm for implementation that prohibits self-sufficiency and intellect. But PSO naturally fits a system where the agents are delegated goals in some elevated level way, and then the agents decide for themselves how best to achieve their goals – the agents here have the capability to choose how best to act so as to achieve their delegated goals. The scalability that is needed in such a system is also lacking since an inert population is usually assumed during optimization procedure. There are variants of PSOs where partisanship of the total swarm population grows or shrinks dynamically [21-24], depending on the strengths and behaviours of members of the structure. Therefore, scalability in the form of population enlargement or reduction is desirable, which is fairly difficult to do in a monolithic and highly unified system. A perfect model will therefore not take for granted fixed neighbourhood of particle agents; agents should be able of moving from one neighbourhood to another, which may have dissimilar sizes. This feature is naturally available in Multi Agent System (MAS), since every agent is treated separately. Multifaceted communication patterns arise among the particles within a typical PSO system, and if not properly implemented, communication can be inherently synchronous. This drastically degrades the overall systems performance especially if the population size is too elevated. With the MAS approach, there is a natural asynchronous communication, because concurrency is natural, as agents implement independently. Implementing a population-based algorithm without regards for isolated behaviour of each candidate of the population makes the entire procedure as a complex one, especially with a large population. Since every particle has divided behaviour from other particles within the system, it is highly desirable to model the system as such. A number of other characteristics of PSO [25] that make it suitable for MAS exist; Natural algorithm: it is based on the behaviour of real birds/fish which are real agents; Parallel and distributed algorithm: the swarm is a population of agents move simultaneously, independently and without a supervisor; Cooperative particles: each agent chooses a new point partly based on the information received from other agents. To address these issues, literatures on MAS-based PSOs exist [25-27], however, the importance in these literatures are based on modelling, implementations, and load balancing/fault tolerance. The proposed Lucid Particle Swarm Optimization (LPSO) algorithm has been evaluated in standard IEEE 57 bus test system & the simulation results shows that the proposed approach outperforms all reported algorithms in minimization of real power loss.

2. Problem Formulation

The Optimal reactive power flow problem is measured as a general minimization problem with constraints, and can be mathematically written in the following form:

Minimize f(x, u) (1)

Subject to g(x,u)=0 (2)

And

$$h(x, u) \leq 0 \tag{3}$$

Where f(x,u) is the objective function. g(x.u) and h(x,u) are respectively the set of equality and inequality constraints. x is the vector of state variables, and u is the vector of control variables.

The state variables are the load buses (PQ buses) voltages, angles, the generator reactive powers and the slack active generator power:

$$x = \left(P_{g1}, \theta_2, .., \theta_N, V_{L1}, ., V_{LNL}, Q_{g1}, .., Q_{gng}\right)^T \tag{4}$$

The control variables are the generator bus voltages, the shunt capacitors/reactors and the transformers tap-settings:

$$u = \left(V_g, T, Q_c\right)^T \tag{5}$$

Or

$$u = \left(V_{g1}, \dots, V_{gng}, T_1, .., T_{Nt}, Q_{c1}, .., Q_{cNc}\right)^T \tag{6}$$

Where Ng, Nt and Nc are the number of generators, number of tap transformers and the number of shunt compensators respectively.

3. Objective Function

Active power loss

The objective of the reactive power dispatch is to minimize the active power loss in the transmission network, which can be described as follows:

$$F = PL = \sum_{k \in Nbr} g_k \left(V_i^2 + V_j^2 - 2V_iV_j cos\theta_{ij}\right) \tag{7}$$

Or

$$F = PL = \sum_{i \in Ng} P_{gi} - P_d = P_{gslack} + \sum_{i \neq slack}^{Ng} P_{gi} - P_d \tag{8}$$

Where g_k : is the conductance of branch between nodes i and j, Nbr: is the total number of transmission lines in power systems. P_d: is the total active power demand, P_{gi}: is the generator active power of unit i, and P_{gsalck}: is the generator active power of slack bus.

Voltage profile improvement

For minimizing the voltage deviation in PQ buses, the objective function becomes:

$$F = PL + \omega_v \times VD \tag{9}$$

Where ω_v: is a weighting factor of voltage deviation.

VD is the voltage deviation given by:

$$VD = \sum_{i=1}^{Npq} |V_i - 1| \tag{10}$$

Equality Constraint

The equality constraint g(x,u) of the ORPD problem is represented by the power balance equation, where the total power generation must cover the total power demand and the power losses:

$$P_G = P_D + P_L \tag{11}$$

This equation is solved by running Newton Raphson load flow method, by calculating the active power of slack bus to determine active power loss.

Inequality Constraints

The inequality constraints h(x,u) reflect the limits on components in the power system as well as the limits created to ensure system security. Upper and lower bounds on the active power of slack bus, and reactive power of generators:

$$P_{gslack}^{min} \leq P_{gslack} \leq P_{gslack}^{max} \tag{12}$$

$$Q_{gi}^{min} \leq Q_{gi} \leq Q_{gi}^{max}, i \in N_g \tag{13}$$

Upper and lower bounds on the bus voltage magnitudes:

$$V_i^{min} \leq V_i \leq V_i^{max}, i \in N \tag{14}$$

Upper and lower bounds on the transformers tap ratios:

$$T_i^{min} \leq T_i \leq T_i^{max}, i \in N_T \tag{15}$$

Upper and lower bounds on the compensators reactive powers:

$$Q_c^{min} \leq Q_c \leq Q_C^{max}, i \in N_C \tag{16}$$

Where N is the total number of buses, N_T is the total number of Transformers; N_c is the total number of shunt reactive compensators.

4. Lucid Particle Swarm Optimization

Lucid is connected with the way of thinking &is carried out in order to land at a termination. It refers to the capability to come to accurate conclusions about what is true or real, and about how to solve problems [28]. Reasoning in a general sense is a broad subject matter that refers to the ability to make sense of things, to set up and confirm facts, and to alter or validate practices and beliefs [29]. We use practical way of thinking model to agency [30, 31] to represent intellectual actions. Realistic way of thinking is the capacity for resolving, through reflection, the question of what is to be done. Deliberation of this kind is realistic in the subject matter, insofar as it is concerned with action. But it is also practical in its consequences or its issue, in so far as indication about action itself directly moves an agent to act [32]. The idea of realistic reasoning agent is modelled by looking at an agent as having a set of beliefs which are the discernment of the agent's operating environment, set of requirements which are the various options at the agent's disposal, and set of intents which are selections made from the desires by filtering the most excellent options. Realistic reasoning is comprised of two major components [31]: The intentions here are the future directed intentions, which simply symbolize the state of mind of the agent, with no deeds taken. Deliberation is modelled as choice of generation and filtering processes [31], which are thus described as follows:

i. Option generation function takes the present beliefs and present intentions in order to yield the agent's desire set. Thus,

$$option : 2^{Bel} \times 2^{Int} \rightarrow 2^{Des} \tag{17}$$

ii. Then the intentions to be devoted to obtain by filtering and selecting the best options using the following function:

$$filter : 2^{Bel} \times 2^{Int} \times 2^{Des} \rightarrow 2^{Int} \tag{18}$$

The agent updates its belief through a belief reassess function defined as:

$$brf : 2^{Bel} \times per \rightarrow 2^{Bel} \tag{19}$$

Where "per"is a set of percept in the operating environment.

Practically, a particle agent within the swarm will arrive at a substitute by first deliberating on the available options, and then build decisions by acting on the best alternatives. A Particle agent starts by having a particular set of beliefs which are accumulated in a belief database, and then executed to intentions for actions based on the initial beliefs and desires. As time steps forward, the beliefs about the actual world may be refined, and the agent's desires and intentions may also be redefined to imitate the changes in the belief database. After an agent deliberates and creates intentions to which it is committed, the agent needs to plan how to achieve the intentions based on the current state of the environment (agent's belief) and the actions that are available to it. This is means-ends reasoning. So, a particle agent perceives its atmosphere and then adjusts its belief database appropriately, upon which it derives its intentions, and then uses realistic reasoning to take a deed that alters the actual world, which advances the structure towards a potential solution.

The properties of the traditional Particle Swarm Optimization (PSO) model make it a suitable candidate for Multi-Agent System (MAS) implementation. In the MAS-based PSO, we model into MAS the suitable qualities of the traditional PSO and then initiate concepts that progress on the overall systems performance as an optimization method. Agents within MAS deem problems by weighing conflicting considerations for and against challenging options, where the applicable considerations are provided by what the agent desires or values and what the agent believes. An agent takes deed by first deliberating on what state of affairs to achieve from the available options, which represents its Intentions that modify its state of mind. Then the agent reasons on how to attain the chosen state of affairs, which results in a plan of how best to accomplish the option. By so doing, intellect is built into the system.

The fundament for the development of PSO is hypothesis that a potential solution to an optimization problem is treated as a bird without quality and volume, which is called a particle, coexisting and evolving simultaneously based on knowledge sharing with neighbouring particles. While flying through the problem search space, each particle modifies its velocity to find a better solution (position) by applying its own flying experience (i.e. memory having best position found in the earlier flights) and experience of neighbouring particles (i.e. best-found solution of the population). Particles update their positions and velocities as shown below:

$$v_{t+1}^{i} = \omega_t . v_t^i + c_1 . R_1 . (p_t^i - x_t^i) + c_2 . R_2 . (p_t^g - x_t^i) \quad (20)$$

$$x_{t+1}^{i} = x_t^i + v_{t+1}^{i} \quad (21)$$

Where x_t^i represents the current position of particle i in solution space and subscript t indicates an iteration count; p_t^i is the best-found position of particle i up to iteration count t and represents the cognitive contribution to the search velocity v_t^i . Each component of v_t^i can be clamped to the range to control excessive roaming of particles outside the search space; p_t^g is the global best-found position among all particles in the swarm up to iteration count t and forms the social contribution to the velocity vector; r_1 and r_2 are random numbers uniformly distributed in the interval (0,1), wherec_1 andc_2 are the cognitive and social scaling parameters, respectively;ω_t is the particle inertia, which is reduced dynamically to decrease the search area in a gradual fashion.

The agent chooses more talented neighbours based on its previous knowledge and experience. The requirements of the agent increases in the direction of more promising neighbours and it updates its beliefs, and subsequently gets attracted towards more promising regions.

Description 1: We define the state of environment in which the agent's search space may be as follows:

$$E = \{P_1, P_2, .., P_N\} \quad (22)$$

Where $P_i = (P_{i1}, P_{i2}, .., P_{in})^T$ are the best positions ever visited by each particle agent, representing the present state (i = 1,2,..,N , N being the population size and n the current iteration counter).

Description 2: Each particle agent has a range of actions at its disposal, which are the consequences of agent's invocation. If we generally define the set of actions as $A_c = \{\alpha, \alpha', ..\}$, then specifically, the ith particle agent in the system has these actions:

$$AC_i = \{V, X, N^*, \sigma, C\} \tag{23}$$

Where V and X are the velocity and position functions respectively of the agent within the main neighbourhood, $N^*$is the prospective neighbourhood vector described earlier, which is an action that computes the values for the belief database and updates same, and C is a communicator which the agent uses to communicate with other agents, thereby separating the social activities of the Multi-Agent System (MAS) from the individual agent's activities. σ is a recap function that permits an agent to appraise its history from the last time stamp in order to decide whether or not to change neighbourhood.

Particle agents in the explore space have single-minded commitments, because an agent continues to maintain an intension of recuperating the fitness values within a particular neighbourhood until it believes either that the intension has been achieved, or else it is no longer a more reasonable option to remain in that neighbourhood, in which case it is sensible for the agent to move away to a more promising neighbourhood. We presume that the size of neighbourhoods can vary because there may be increase or decrease in population, agents can move from one neighbourhood to another, or any other unforeseen factor.

Description 3: A run r , of an agent in an environment is a sequence of interleaved environment states and actions. If we let be the set of all such runs, then we have:

$$R = \{r, r', ..\} \tag{24}$$

Let R^{AC} be the subset of these that end with an action and R^E be the subset of these that end with an environment state.

Description 4: When a particle agent invokes an action on an environment, it transforms the environment state, and this effect is modelled by the state transformer function defined as:

$$\tau: R^{AC} \to 2^E \tag{25}$$

Where 2^E power set of E.

This means that from runs which end with actions taken by a particle agent, the structure will always end up in a particular environment state; taking an action by a particle agent on a previous environment state moves the environment to another state.
Description 5: We define an environment as a tuple:

$$\xi = \langle E, \theta_D, \tau, T, \eta \rangle \tag{26}$$

Where E is the set of environment states, θ_D is an initial state, τ is a state transformer function, T is theactive topology in the environment, and η is a set of neighbourhoods defined as $\eta = \{n_1, n_2, .., n_k\}$, where K is the total number of neighbourhoods.

Description 6: In MAS, the agents drive the system. The state of the environment emerges as a result of the agents' actions – based on the behaviours and interactions among the agents. Since actions are produced by agents when they execute in the system, we model agents as function of execution, which yield an action (whose effect is the state transformer function).
Thus, a particle agent is defined as:

$$A: R^E \to A_c \tag{27}$$

So if an action, say the position update function $X \in Ac$, A' is desired of a particle agent, the agent produces this action by executing on an existing run ending with environment state, say r' , which is its current position, as follows:

$$X = A'(r') \tag{28}$$

This leaves the run to end with an action. The effect of taking this action, which is modelled by the state transformer function, τ is to produce a new environment state.

Description 7: We define Swarm S to be the set of all agents, as follows:

$$S = \{A_1, A_2, .., A_N\} \tag{29}$$

And the Swarm System is thus defined as (S, ξ) , where R is the set of all runs.

Having established the definitions and the relationships above, we now explain the algorithms that aid practical reasoning particle agents to perform within the swarm. Particle agents require planning ahead and envisaging better fitness values with other neighbourhoods within the same iteration by sharing information with agents outside the main locality. The particle agents thus uphold single-minded commitments of achieving better fitness values by making realistic reasoning and dynamically alternating neighbours in the explore process. Within every iteration, an agent calculate several fitness values in parallel and keeps the history for future reference, and as time progresses, the agent sticks to neighbours that capitulate improved fitness values. So the best global behaviour emerges as the agents interact. As earlier explained, Practical reasoning = deliberation + means-ends reasoning Deliberation = option generation function and Filtering function.

Means-end reasoning = planning

These components of the practical reasoning are obtained below. The environment state as initially perceived by particle agents, denoted by equation (22), represents the initial belief of the agents. So $\mathrm{B} = \mathrm{B_o} = \{\mathrm{P_1}, \mathrm{P_2}, .., \mathrm{P_N}\}$ The agents will initially look at the state of affairs to achieve as their initial intentions. The desired state of affair at the beginning is to apply the velocity function of equation (20) and then take a move using the position function of equation (21).

These are initialized in $I = I_o$.

Lucid Particle Swarm Optimization (LPSO) algorithm for solving optimal reactive power problem

```
B = Bo; // Initial beliefs
I = Io; //initial intentions
While true do
Get next percept ρ of the swarm by making a call to communicator C;
B ← brf (B, ρ);
D ← option (B, I);
I ← filter (B, D, I);
π ← plan(B, I, Ac) ;
Whi1e not (empty (π) or succeeded (I, B) or impossible (I, B)) do
α ← head(π);
Execute (α);
π ← tail(π);
Get next percept ρ of the swarm by
Making a call to communicator C;
B ← brf (B, ρ);
If reconsider (I, B) then
D ← option (B, I);
I ← filter (B, D, I);
End-if
If not sound (π, I, B) then
π ← plan(B, I, Ac) ;
End-if
End-whi1e
End-whi1e
```

5. Simulation Results

At first Lucid Particle Swarm Optimization (LPSO) algorithm has been tested in standard IEEE 118-bus test system [33]. The system has 54 generator buses, 64 load buses, 186 branches and 9 of them are with the tap setting transformers. The limits of voltage on generator buses are 0.95 -1.1 per-unit., and on load buses are 0.95 -1.05 per-unit. The limit of transformer rate is 0.9 -1.1, with the changes step of 0.025. The limitations of reactive power source are listed in Table 1, with the change in step of 0.01.

Table 1: Limitation of reactive power sources

BUS	5	34	37	44	45	46	48
QCMAX	0	14	0	10	10	10	15
QCMIN	-40	0	-25	0	0	0	0
BUS	74	79	82	83	105	107	110

QCMAX	12	20	20	10	20	6	6
QCMIN	0	0	0	0	0	0	0

The statistical comparison results have been listed in Table 2 and the results clearly show the better performance of proposed LPSO approach.

Table 2: Comparison results

Active power loss (p.u)	**BBO [34]**	**ILSBBO/ strategy1 [34]**	**ILSBBO/ strategy1 [34]**	**Proposed LPSO**
Min	128.77	126.98	124.78	102.48
Max	132.64	137.34	132.39	108.26
Average	130.21	130.37	129.22	104.02

Then the Lucid Particle Swarm Optimization (LPSO) algorithm has been tested in practical 191 test system and the following results have been obtained. In Practical 191 test bus system – Number of Generators = 20, Number of lines = 200, Number of buses = 191 Number of transmission lines = 55. Table 3 shows the optimal control values of practical 191 test system obtained by LPSO method. And table 4 shows the results about the value of the real power loss by obtained by Lucid Particle Swarm Optimization (LPSO) algorithm.

Table 3: Optimal Control values of Practical 191 utility (Indian) system by LPSO method

VG1	1.100		VG 11	0.900
VG 2	0.720		VG 12	1.000
VG 3	1.010		VG 13	1.000
VG 4	1.010		VG 14	0.900
VG 5	1.100		VG 15	1.000
VG 6	1.100		VG 16	1.000
VG 7	1.100		VG 17	0.900
VG 8	1.010		VG 18	1.000
VG 9	1.100		VG 19	1.100
VG 10	1.010		VG 20	1.100

T1	1.000		T21	0.900		T41	0.900
T2	1.000		T22	0.900		T42	0.900
T3	1.000		T23	0.900		T43	0.910
T4	1.100		T24	0.900		T44	0.910
T5	1.000		T25	0.900		T45	0.910
T6	1.000		T26	1.000		T46	0.900
T7	1.000		T27	0.900		T47	0.910
T8	1.010		T28	0.900		T48	1.000
T9	1.000		T29	1.010		T49	0.900
T10	1.000		T30	0.900		T50	0.900
T11	0.900		T31	0.900		T51	0.900
T12	1.000		T32	0.900		T52	0.900

T13	1.010		T33	1.010		T53	1.000
T14	1.010		T34	0.900		T54	0.900
T15	1.010		T35	0.900		T55	0.900
T19	1.020		T39	0.900			
T20	1.010		T40	0.900			

Table 4: Optimum real power loss values obtained for practical 191 utility (Indian) system by LPSO method

Real power Loss (MW)	**LPSO**
Min	134.082
Max	138.246
Average	136.864

6. Conclusion

In this paper, Lucid Particle Swarm Optimization (LPSO) algorithm has been successfully solved optimal reactive power problem. In the proposed algorithm Self-Sufficiency is achieved by the use of communicators that separate an agent's individual operation from that of the swarm, thereby making the system more robust. The proposed Lucid Particle Swarm Optimization (LPSO) algorithm has been tested on standard IEEE 118 & practical 191 bus test systems and simulation results show clearly about the premium performance of the proposed algorithm in reducing the real power loss.

References

[1] O.Alsac, and B. Scott, "Optimal load flow with steady state security", IEEE Transaction. PAS -1973, pp. 745-751.

[2] Lee K Y, Paru Y M, Oritz J L –A united approach to optimal real and reactive power dispatch, IEEE Transactions on power Apparatus and systems 1985: PAS-104: 1147-1153

[3] A.Monticelli , M .V.F Pereira ,and S. Granville , "Security constrained optimal power flow with post contingency corrective rescheduling" , IEEE Transactions on Power Systems :PWRS-2, No. 1, pp.175-182.,1987.

[4] Deeb N, Shahidehpur S.M, Linear reactive power optimization in a large power network using the decomposition approach. IEEE Transactions on power system 1990: 5(2): 428-435

[5] A. Hobson, "Network constrained reactive power control using linear programming," IEEE Transactions on power systems PAS -99 (4), pp 868=877, 1980

[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.Power Syst.Res, Vol.26, pp.1-10,1993.

[8] 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.

[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] A.Berizzi, C. Bovo, M. Merlo, and M. Delfanti, "A ga approach to compare orpf objective functions including secondary voltage regulation," Electric Power Systems Research, vol. 84, no. 1, pp. 187 – 194, 2012.
[12] C.-F. Yang, G. G. Lai, C.-H. Lee, C.-T. Su, and G. W. Chang, "Optimal setting of reactive compensation devices with an improved voltage stability index for voltage stability enhancement," International Journal of Electrical Power and Energy Systems, vol. 37, no. 1, pp. 50 – 57, 2012.
[13] P. Roy, S. Ghoshal, and S. Thakur, "Optimal var control for improvements in voltage profiles and for real power loss minimization using biogeography based optimization," International Journal of Electrical Power and Energy Systems, vol. 43, no. 1, pp. 830 – 838, 2012.
[14] B. Venkatesh, G. Sadasivam, and M. Khan, "A new optimal reactive power scheduling method for loss minimization and voltage stability margin maximization using successive multi-objective fuzzy lp technique," IEEE Transactions on Power Systems, vol. 15, no. 2, pp. 844 – 851, may 2000.
[15] W. Yan, S. Lu, and D. Yu, "A novel optimal reactive power dispatch method based on an improved hybrid evolutionary programming technique," IEEE Transactions on Power Systems, vol. 19, no. 2, pp. 913 – 918, may 2004.
[16] W. Yan, F. Liu, C. Chung, and K. Wong, "A hybrid genetic algorithminterior point method for optimal reactive power flow," IEEE Transactions on Power Systems, vol. 21, no. 3, pp. 1163 – 1169, aug. 2006.
[17] J. Yu, W. Yan, W. Li, C. Chung, and K. Wong, "An unfixed piecewiseoptimal reactive power-flow model and its algorithm for ac-dc systems," IEEE Transactions on Power Systems, vol. 23, no. 1, pp. 170 –176, feb. 2008.
[18] Z.Capitanescu, "Assessing reactive power reserves with respect to operating constraints and voltage stability," IEEE Transactions on Power Systems, vol. 26, no. 4, pp. 2224–2234, nov. 2011.
[19] Z. Hu, X. Wang, and G. Taylor, "Stochastic optimal reactive power dispatch: Formulation and solution method," International Journal of Electrical Power and Energy Systems, vol. 32, no. 6, pp. 615 – 621, 2010.
[20] A.Kargarian, M. Raoofat, and M. Mohammadi, "Probabilistic reactive power procurement in hybrid electricity markets with uncertain loads," Electric Power Systems Research, vol. 82, no. 1, pp. 68 – 80, 2012.
[21] Bell, W.J., Roth, L., and Nalepa, C. (2007). Cockroaches: ecology, behavior, and natural history, The Johns Hopkins University Press.
[22] Halloy, J., Sempo, G., Caprari, G., Rivault, C., Asadpour, M., Tache, F., Said, I., Durier, V., Canonge, S., Amé, J.M., Detrain, C., Correll, N., Martinoli, A., Mondada, F., Siegwart, R., and Deneubourg, J.L. (2007). Social integration of robots into groups of cockroaches to control selforganized choices, Science, November, 318(5853), pp.1155–1158.
[23] Obagduwa, I.C. (2012). Cockroaches optimization algorithms, seminar paper presented at the Progress Seminar, March 3rd, UKZN, Durban.
[24] Parpinelli, R.S., and Lopes, H.S. (2011). New inspirations in swarm intelligence: a survey, International Journal of Bio-Inspired Computation, 3(1), pp.1–16.
[25] Shangxiong, S. (2008). A Particle Swarm Optimization (PSO) algorithm based on multiagent system, In IEEE International Conference on Intelligent Computation Technology and Automation, 978-0-7695-3357-5/08.
[26] Shangxiong, S. (2008). A Particle Swarm Optimization (PSO) algorithm based on multiagent system, In IEEE International Conference on Intelligent Computation Technology and Automation, 978-0-7695-3357-5/08
[27] Lorion, Y., Bogon, T., Timm, I.J., and Drobnik, O. (2009). An Agent based Parallel Particle Swarm Optimization – APPSO, In IEEE Publications, 978-1-4244-2762-8/09.
[28] Davidson, H. (1992). Alfarabi, Avicenna, and Averroes, on Intellect, Oxford University Press, pp.6.
[29] Kompridis, N. (2000). So we need something else for reason to mean, International Journal of Philosophical Studies, 8(3), pp.271-295.
[30] Petrie, H.G. (1971). Philosophy & Rhetoric © 1971, Penn State University Press.

[31] Wooldridge, M. (2009), An introduction to multiagent systems. 2nd ed., John Wiley & Sons Ltd, Chichester.
[32] Wallace, R.J. (2009). Practical Reason, The Stanford Encyclopedia of Philosophy, Edward N. Zalta (ed.), URL = http://plato.stanford.edu/archives/sum2009/entries/ practical-reason.
[33] IEEE, "The IEEE 30-bus test system and the IEEE 118-test system", (1993), http://www.ee.washington.edu/trsearch/pstca/.
[34] Jiangtao Cao, Fuli Wang and Ping Li, "An Improved Biogeography-based Optimization Algorithm for Optimal Reactive Power Flow", International Journal of Control and Automation Vol.7, No.3 (2014), pp.161-176

*Corresponding author.
E-mail address: gklenin@ gmail.com