Research Journal of Applied Sciences, Engineering and Technology 10(3): 315-321, 2015
DOI: 10.19026/rjaset.10.2493
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2015, Maxwell Scientific Publication Corp.
Submitted: December 26, 2014 Accepted: February 8, 2015 Published: May 30, 2015

Research Article

Multiobjective Optimal Reactive Power Dispatch Considering Voltage Stability Using Shuffled Frog Leaping Algorithm

[1]V. Tamilselvan, [1]M. Sathish and [2]T. Jayabarathi
[1]Department of Electrical and Electronics Engineering, Adhiyamaan College of Engineering, Hosur,
[2]School of Electrical Engineering, VIT University, Vellore, Tamilnadu, India

Abstract: This study addresses a shuffled frog leaping algorithm for solving the multi-objective reactive power dispatch problem in a power system. Optimal Reactive Power Dispatch (ORPD) is formulated as a nonlinear, multi-modal and mixed-variable problem. The intended technique is based on the minimization of the real power loss, minimization of voltage deviation and maximization of the voltage stability margin. Generator voltages, capacitor banks and tap positions of tap changing transformers are used as optimization variables of this problem. A memetic meta-heuristic named as shuffled frog-leaping algorithm is intended to solve multi-objective optimal reactive power dispatch problems considering voltage stability margin and voltage deviation. The Shuffled Frog-Leaping Algorithm (SFLA) is a population-based cooperative search metaphor inspired by natural memetics. The algorithm contains elements of local search and global information exchange. The most important benefit of this algorithm is higher speed of convergence to a better solution. The intended method is applied to ORPD problem on IEEE 57 bus power systems and compared with two versions of differential evolutionary algorithm. The simulation results show the effectiveness of the intended method.

Keywords: Multi objective optimization, reactive power dispatch, shuffled frog-leap algorithm

INTRODUCTION

In the present scenario, the load density of the system has increased abnormally and due to which the quality of power has decreased. The quality of power lacks due to the shortage of reactive power during the peak load periods which results in the reduction of overall voltage level. So the Reactive Power Optimization (RPO) and voltage control are the essential topic of research. The reactive power and voltage control improve the economy and security of the power system. The load bus voltages can be maintained within the permissible limits by reallocating reactive power generation in the system which is achieved by adjusting the transformer taps, generator voltages and capacitor banks. So the RPO problem deals with the minimization of the real power loss and improvement of the voltage profile of the system. Mathematically, RPO is a complicated, non-linear programming problem with non-linear objective functions, nonlinear equality and inequality constraints.

Optimization deals with the problem of seeking solution over a set of possible choices to optimal criteria. If the criterion considered is one, it is a single objective optimization problem. If the number of criteria is more than one and if they are treated simultaneously, the problem is a Multi Objective Optimization (MOO) problem. The conventional RPO takes minimum power loss or voltage quality as major objective and concerns little over the voltage stability. So RPO problem is a single objective optimization problem. The conventional RPO problem is solved using the non-linear programming technique, sensitive and gradient based techniques and heuristic techniques. The non-linear programming has various drawbacks like insecure convergence, more execution time and complexity. The sensitive and gradient based techniques get trapped in the local minima which lead to the attainment of a solution which is not optimal. The heuristic technique which is a search based technique has attained great success in solving the RPO problem.

In recent investigation in order to improve the system stability and to minimize loss in transmission lines, the RPO is formulated with multiple objectives like voltage deviation, voltage stability margin and minimization of active power loss.

In the last decades, Computational intelligence-based techniques have been proposed for the application of reactive power optimization, such as Differential Evolution (DE). Abido (2006) has formulated the optimal VAR dispatch problem as non linear constrained multi-objective optimization problem where the real power loss and voltage deviation are to be simultaneously minimized. Dai *et al*. (2009)

Corresponding Author: V. Tamilselvan, Department of Electrical and Electronics Engineering, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

proposed a Seeker Optimization Algorithm (SOA) based method for ORPD, considering static voltage stability and voltage deviation. Jeyadevi *et al.* (2011) has addressed an application of Modified NSGA-II (MNSGA-II) by incorporating controlled elitism and Dynamic Crowding Distance (DCD) strategies in NSGA-II to multi-objective ORPD problem by minimizing real power loss and maximizing the system voltage stability. Jeyanthy and Devaraj (2010) proposed a hybrid particle swarm optimization algorithm for solving multi-objective real power optimization problem with minimization of loss and maximization of voltage stability margin are considered as objectives. Mancer *et al.* (2012) proposed a new variant of PSO algorithm with varying acceleration co-efficients to solve the MOORPF with power loss and voltage deviation as objective functions. Shi and Liu (2005) developed a fuzzy evaluation based multi objective model for reactive power optimization in power distributed networks. Xiong *et al.* (2008) proposed an Optimal Reactive Power Flow (ORPF) incorporating static voltage stability based on a Multi-Objective Adaptive Immune Algorithm (MOAIA). Zhen *et al.* (2007) proposed the multi-objective optimization problem with real power losses and voltage stabilities to be simultaneously optimized with the Bacterial Swarming Algorithm (BSA).

From the above review, certain limitations of the EP, TS and PSO based algorithms are summarized as follow:

- The convergence is very slow, as these algorithms are random search techniques and they have to handle multiple objectives with enormous decision variables.
- In most of the methods, the optimal solutions are obtained in any way by weakening one or more objectives. So, a diverse optimal solution has to be determined.

It reveals that there exists a need for evolving a simple, effective and faster algorithm for solving Multi Objective optimal Reactive Power Dispatch (MORPO). The algorithm should obtain the optimum results with faster convergence. In this study, an attempt has been made to solve the MORPO by using shuffled frog leaping based algorithm which is able to find diverse solutions.

METHODOLOGY

Problem formulation: The multi-objective functions of the Optimal Reactive Power Dispatch include the technical and economic goals:

- The economic goal is mainly to minimize the active power transmission loss.
- The technical goals are to minimize the load bus voltage deviation from the ideal voltage and to improve the Voltage Stability Margin (VSM).

Hence, the objectives of the Optimal Reactive Power Dispatch model in this study are active power loss (P_{loss}), voltage deviation (ΔV_L) and Voltage Stability Margin (VSM).

The active power loss: The active power loss minimization in the transmission network can be defined as follows:

$$\min P_{loss} = f(\overline{x_1}, \overline{x_2}) = \sum\nolimits_{k \in N_E} g_k (V_i^2 + V_j^2 - 2V_i V_j cos\theta_{ij}) \tag{1}$$

Subject to:

$$\begin{cases} P_{Gi} - P_{Di} = V_i \sum_{j \in N_i} V_j \big(G_{ij} cos\theta_{ij} + B_{ij} sin\theta_{ij}\big) & i \in N_0 \\ Q_{Gi} - Q_{Di} = V_i \sum_{j \in N_i} V_j \big(G_{ij} sin\theta_{ij} - B_{ij} cos\theta_{ij}\big) & i \in N_{PQ} \\ V_i^{min} \le V_i \le V_i^{max} & i \in N_B \\ T_k^{min} \le T_k \le T_k^{max} & i \in N_T \\ Q_{Gi}^{min} \le Q_{Gi} \le Q_{Gi}^{max} & i \in N_G \\ Q_{Ci}^{min} \le Q_{Ci} \le Q_{Ci}^{max} & i \in N_C \\ S_l \le S_l^{max} & i \in N_l \end{cases} \tag{2}$$

where,

$f\,(x_1,x_2)$ = The active power loss function of the transmission network

$(V_G \cdot K_T Q_C)^T$ = The control variable vector

(V_L Q_G) = The dependent variable vector
V_L = The generator voltage (continuous)
T_K = The transformer tap (integer)
Q_C = The shunt capacitor/inductor (integer)
V_L = The load bus voltage
Q_G = The generator reactive power
k = (i, j), i E N_B, J E N_i
G_K = The conductance of branch k
ϕ_{ij} = The voltage angle difference between bus i and j
P_{Gi} = The injected active power at bus i
P_{Di} = The demanded active power at bus i
V_i = The voltage at bus i
G_{ij} = The transfer conductance between bus i and j
B_{ij} = The transfer susceptance between bus I and j
Q_{Gi} = The injected reactive power at bus i
Q_{Di} = The demanded reactive power at bus
N_E = The set of numbers of network branches
N_{PQ} = The set of numbers of PQ buses
N_B = Numbers of total buses
N_i = The set of numbers of buses adjacent to bus I (including bus i)
N_0 = The set of numbers of total buses excluding slack bus
Nc = The set of numbers of possible reactive power source installation buses
N_G = The set of numbers of generator buses
N_T = The set of numbers of transformer branches
S_l = The power flow in branch l

The superscripts "min" and "max" denote the corresponding lower and upper limits, respectively.

Voltage deviation: Treating the bus voltage limits as constraints in ORPD often results in all the voltages toward their maximum limits after optimization, which means the power system lacks the required reserves to provide reactive power during contingencies. One of the effective ways to avoid this situation is to choose the deviation of voltage from the desired value as an objective function i.e.:

$$\min \Delta V_L = \sum_{i=1}^{N_L} \left| \frac{V_i - V_i^*}{N_L} \right| \tag{3}$$

where,
ΔV_L = The per unit average voltage deviation
N_L = The total number of the system load buses
V_i and V_i^* = The actual voltage magnitude and the desired voltage magnitude at bus i

Voltage stability margin: Voltage stability problem has a close relationship with the reactive power of the system and the voltage stability margin is inevitably affected in optimal reactive power flow. Hence, the maximal voltage stability margin should be one of the objectives in ORPF. In the literature, the minimal eigen value of the non-singular power flow Jacobian matrix has been used by many researchers to improve the voltage stability margin. Here also, it is employed:

$$\max \text{VSM} = \max (\min|\text{eig (jacobi)}|) \tag{4}$$

where,
Jacobi : The power flow Jacobian matrix
eig (Jacobi) : Returns all the eigen values of the Jacobian matrix
min (eig (Jacobi)) : The minimum value of eig (Jacobi)
max (min (eig (Jacobi))) : To maximize the minimal eigen value in the Jacobian matrix

Multi-objective conversion: Considering different sub-objective functions have different ranges of function values, every sub-objective uses a transform to keep itself within (0, 1). The first two sub-objective functions, i.e., active power loss and voltage deviation, are normalized:

$$f_1 = \begin{cases} 0 & if\ P_{loss} < P_{loss_{min}} \\ \dfrac{P_{loss} - P_{loss_{min}}}{P_{loss_{max}} - P_{loss_{min}}} & if\ P_{loss_{min}} \le P_{loss} \le P_{loss_{max}} \\ 1 & if\ P_{loss} > P_{loss_{max}} \end{cases} \tag{5}$$

$$f_2 = \begin{cases} 0 & if\ \Delta V_L < \Delta V_{L_{min}} \\ \dfrac{\Delta V_L - \Delta V_{L_{min}}}{\Delta V_{L_{max}} - \Delta V_{L_{min}}} & if\ \Delta V_{L_{min}} \le \Delta V_L \le \Delta V_{L_{max}} \\ 1 & if\ \Delta V_L > \Delta V_{L_{max}} \end{cases} \tag{6}$$

where, the subscripts "min" and "max" in Eq. (5) and (6) denote the corresponding expectant minimum and possible maximum value, respectively.

Since voltage stability margin sub-objective function is a maximization optimization problem, it is normalized and transformed into a minimization problem using the following equation:

$$f_3 = \begin{cases} 0 & if\ VSM > VSM_{max} \\ \dfrac{VSM_{max} - VSM}{VSM_{max} - VSM_{min}} & else \end{cases} \tag{7}$$

where, the subscripts "min" and "max" in Eq. (7) denote the possible minimum and expectant maximum value, respectively.

Control variables are self-constrained and dependent variables are constrained using penalty terms. Then, the overall objective function is generalized as follows:

$$\min f = w_1 f_1 + w_2 f_2 + w_3 f_3 + \lambda_V \sum_{N_V^{\lim}} \Delta V_L^2 + \lambda_Q \sum_{N_Q^{\lim}} \Delta Q_G^2 \tag{8}$$

where,

ω_i (i = 1, 2, 3) = The user-defined constants which are used to weight the contributions from different sub-objectives

λ_V, λ_Q = The penalty factors

N_V^{lim} = The set of numbers of load buses on which voltage outside limits

N_Q^{lim} = The set of numbers of generator buses on which injected reactive power outside limits

ΔV_L and ΔQ_L are defined as:

$$\Delta V_L = \begin{cases} V_L^{min} - V_L & if\ V_L < V_L^{min} \\ V_L - V_L^{max} & if\ V_L > V_L^{max} \end{cases} \tag{9}$$

$$\Delta Q_G = \begin{cases} Q_G^{min} - Q_G & if\ Q_G < Q_G^{min} \\ Q_G - Q_G^{max} & if\ Q_G > Q_G^{max} \end{cases} \tag{10}$$

SHUFFLED FROG LEAPING ALGORITHM

A memetic meta-heuristic called the Shuffled Frog-Leaping Algorithm (SFLA) has been developed for solving combinatorial optimization problems. The SFLA is a population-based cooperative search metaphor inspired by natural memetics. The algorithm contains elements of local search and global information exchange. The SFLA consists of a set of interacting virtual population of frogs partitioned into different memeplexes. The virtual frogs act as hosts or carriers of memes where a meme is a unit of cultural evolution. The algorithm performs simultaneously an independent local search in each memeplex. The local search is completed using a particle swarm optimization-like method adapted for discrete problems but emphasizing a local search. To ensure global exploration, the virtual frogs are periodically shuffled and reorganized into new memplexes in a technique similar to that used in the shuffled complex evolution algorithm. In addition, to provide the opportunity for random generation of improved information, random virtual frogs are generated and substituted in the population. The algorithm has been tested on several test functions that present difficulties common to many global optimization problems.

Implementation of SFLA:

Global exploration:

Step 0: Initialize: Select m and n, where m is the number of memeplexes and n is the number of frogs in each memeplex. Therefore, the total sample size F in the swamp is given by $F = mn$.

Step 1: Generate a virtual population: Sample F virtual frogs U (1), U (2), ..., U (F) in the feasible space ΩcK^d, where d is the number of decision variables (i.e., number of memotype (s) in a meme carried by a frog). The i^{th} frog is represented as a vector of decision variable values U (i) = (U_i^1, U_i^2, .. U_i^d). Compute the performance value f (i) for each frog U (i).

Step 2: Rank frogs: Sort the F frogs in order of decreasing performance value. Store them in an array X = {U (i), f (i), i = 1... F}, so that i = 1 represents the frog with the best performance value. Record the best frog's position P_X in the entire population (F frogs) (where P_X = U (1)).

Step 3: Partition frogs into memeplexes: Partition array X into m memeplexes (Y_1, Y_2,.. Ym) each containing n frogs, such that:

$$Y_k = [U\,(j)^K, f\,(j)^K] \tag{11}$$

where, $U\,(j\,)^K = U\,(k + m\,(j - 1))$, $f\,(j\,)^K = f\,(k + m\,(j - 1))$, ($j$ = 1, ..., n; k = 1, ..., m) (e.g., for m = 3, rank 1 goes to memeplex 1, rank 2 goes to memeplex 2, rank 3 goes to memeplex 3, rank 4 goes to memeplex 1 and so on).

Step 4: Memetic evolution within each memeplex: Evolve each memeplex Y^K, k = 1... m according to the frog-leaping algorithm outlined below.

Step 5: Shuffle memeplexes: After a defined number of memetic evolutionary steps within each memeplex, replace Y_1, ..., Y_m into X, such that X = {Y^k, k = 1, ..., m}. Sort X in order of decreasing performance value. Update the population best frog's position P_X.

Step 6: Check convergence: If the convergence criteria are satisfied, stop. Otherwise, return to step 3. Typically, the decision on when to stop is made by a pre specified number of consecutive time loops when at least one frog carries the 'best memetic pattern without change. Alternatively, a maximum total number of function evaluations can be defined.

Local exploration:

Frog-leaping algorithm: In step 4 of the global search, evolution of each memeplex continues independently N times. After the memeplexes have been evolved, the algorithm returns to the global exploration for shuffling. Below are details of the local search for each memeplex.

Step 0: Set im = 0 where im counts the number of memeplexes and will be compared with the total number m of memeplexes. Set iN = 0 where iN counts the number of evolutionary steps and will be compared with the maximum number N of steps to be completed within each memeplex.

Step 1: Set $im = im$+1.

Step 2: Set $iN = iN$+1.

Step 3: Construct a submemeplex: The frogs' goal is to move towards the optimum ideas by improving their memes. As stated earlier, they can adapt the ideas from the best frog within the memeplex Yim or from the global best. Regarding the selection of the best memeplex, it is not always desirable to use the best frog because the frogs' tendency would be to concentrate around that particular frog which may be a local optima. So, a subset of the memeplex called a submemeplex is considered. The submemeplex selection strategy is to give higher weights to frogs that have higher performance values and less weight to those with lower performance values. The weights are assigned with a triangular probability distribution, i.e., $p_j = 2 (n + 1 - j) /n (n + 1)$, $j = 1, ..., n$, such that within a memeplex the frog with the best performance has the highest probability $p_1 = 2/ (n + 1)$ of being selected for the submemeplex and the frog with the worst performance has the lowest probability $p_n = 2/n (n + 1)$. Here, q distinct frogs are selected randomly from n frogs in each memeplex to form the submemeplex array Z. The submemeplex is sorted so that frogs are arranged in order of decreasing performance ($i_q = 1, ..., q$). Record the best ($i_q = 1$) frog's position and worst ($i_q = q$) frog's position in the submemeplex as P_B and P_W, respectively.

Step 4: Improve The Worst Frog's Position. The step and new position are computed for the frog with worst performance in the submemeplex by step size:

$S = \min \{\text{int} [\text{rand} (P_B - P_W)], S_{\max}\}$
for a positive step
$S = \max \{\text{int} [\text{rand} (P_B - P_W)], -S_{\max}\}$
for a negative step

where, rand is a random number in the range (0, 1) and $S_{\max}$ is the maximum step size allowed to be adopted by a frog after being infected. Note that the step size has dimensions equal to the number of decision variables. The new position is then computed by:

$$U (q) = P_W + S \tag{12}$$

If $U (q)$ is within the feasible space $\Omega c K^d$, compute the new performance value $f (q)$. Otherwise go to step 5. If the new $f (q)$ is better than the old $f (qi$: i.e., if the evolution produces a benefit, then replace the old $U (q)$ with the new $U (q)$ and go to step 7. Otherwise go to step 5.

Step 5: If step 4 cannot produce a better result, then the step and new position are computed for that frog by step size:

S = $\min \{\text{int} [\text{rand} (P_X - P_W)], S_{\max}\}$ for a positive step
S = $\max \{\text{int} [\text{rand} (P_X - P_W)], -S_{\max}\}$ for a negative step
And the new position is computed by Eq. (11). If $U (q)$ is within the feasible space $\Omega c K^d$, compute the new performance value $f (q)$; otherwise go to step 6. If the new $f (q)$ is better than the old $f (q)$, i.e., if the evolution produces a benefit, then replace the old $U (q)$ with the new $U (q)$ and go to step 7. Otherwise go to step 6.

Step 6: **Censorship:** If the new position is either infeasible or not better than old position, the spread of defective meme is stopped by randomly generating a new frog r at a feasible location to replace the frog whose new position was not favorable to progress. Compute $f (r)$ and set $U (q) = r$ and $f (q) = f (r)$.

Step 7: **Upgrade the memeplex:** After the memetic change for the worst frog in the submemeplex, replace Z in their original locations in Y^{im}. Sort Y^{im} in order of decreasing performance value.

Step 8: If $iN<N$, go to step 2.

Step 9: If $im<m$, go to step 1. Otherwise return to the global search to shuffle memeplexes.

Implementation of SFLA for reactive power optimization: The intended approach to solve multi objective ORPD problem is described in the following steps:

Step 1: Read the parameters of power system and the proposed algorithm and specify the lower and upper limits of each variable.

Step 2: Initialize Number of generation (iter) = 30, Population size (p) = 50, Number of memeplexes (m) = 5 and Iterations within each memeplex = 5.

Step 3: Take iter = 0.

Step 4: Generate population (p) randomly.

Step 5: Calculate the fitness values of (p) using the objective function in (8) based on the results of Newton-Raphson power flow analysis.

Step 6: Find out "personal best (Pbest)" of all frog and "global best (Gbest)" frog from their fitnesses.

Step 7: Sort P in order of decreasing performance value and Partition p into m memeplexes

Step 8: Shuffle the memeplexes and Update the population best frog's position P_x.

Step 9: If the convergence criteria are satisfied, stop. Otherwise, return to step 6.

Step 10: Let iter = iter+1

Step 11: Update the new position of P.

Step 12: Calculate the fitness values of the new positions using the objective function based on the Newton-Raphson power flow analysis results.

Step 13: For each frog if current fitness (P) is better than Pbest then Pbest = P.

Step 14: Set best of Pbest as Gbest.

Step 15: Go to step no. 10, until maximum no of iterations is completed.

Step 16: Coordinate of Gbest particle gives optimized values of control variables and its fitness gives minimized value of losses.

Table 1: Values of control variable after optimization by various methods for IEEE 57-bus system (p.u.)

Variables	Base case	SFLA	L-DE	L-SACP-DE
V_{G1}	1.040	1.036817	1.03970	0.9884
V_{G2}	1.010	1.026660	1.04630	1.0543
V_{G3}	0.985	1.030133	1.05110	1.0278
V_{G6}	0.980	1.033903	1.02360	0.9672
V_{G8}	1.005	1.048680	1.05380	1.0552
V_{G9}	0.980	1.021121	0.94518	1.0245
V_{G12}	1.015	1.041640	0.99078	1.0098
T_{4-12}	0.970	0.961250	1.02000	1.0500
T_{4-18}	0.978	1.006298	0.91000	1.0500
T_{21-20}	1.043	0.972440	0.97000	0.9500
T_{24-26}	1.043	0.966793	0.91000	0.9800
T_{7-29}	0.967	0.974333	0.96000	0.9700
T_{34-32}	0.975	0.955757	0.99000	1.0900
T_{11-41}	0.955	0.992520	0.98000	0.9200
T_{15-45}	0.955	0.962765	0.96000	0.9100
T_{14-46}	0.900	0.943377	1.05000	1.0800
T_{10-51}	0.930	0.978943	1.07000	0.9900
T_{13-49}	0.895	0.986010	0.99000	0.9100
T_{11-43}	0.958	0.969491	1.06000	0.9400
T_{40-56}	0.958	1.019358	0.99000	0.9900
T_{39-57}	0.980	1.027978	0.97000	0.9600
T_{9-55}	0.940	1.028938	1.07000	1.1000
Q_{C18}	0	0.044356	0	0
Q_{C25}	0	0.038841	0	0
Q_{C53}	0	0.018498	0	0

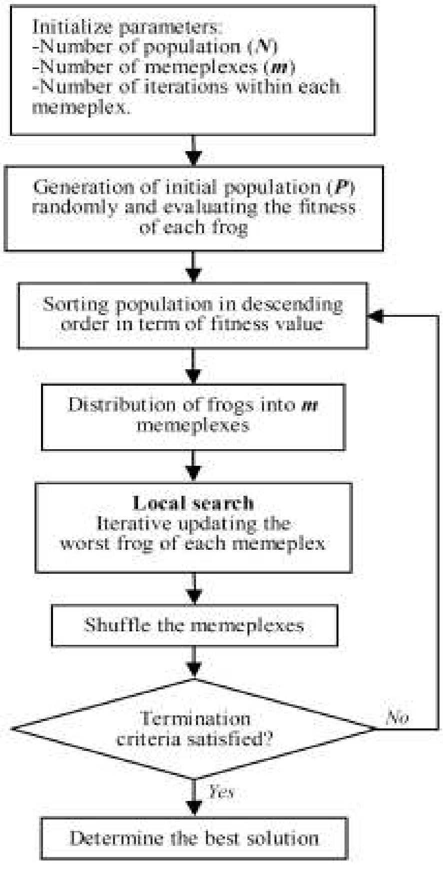
Initialize parameters:
-Number of population (N)
-Number of memeplexes (m)
-Number of iterations within each memeplex.
Generation of initial population (P) randomly and evaluating the fitness of each frog
Sorting population in descending order in term of fitness value
Distribution of frogs into m memeplexes
Local search
Iterative updating the worst frog of each memeplex
Shuffle the memeplexes
Termination criteria satisfied?
No
Yes
Determine the best solution

Fig. 1: Flowchart of the SFLA

Table 2: The computing time for various algorithms on IEEE 57-bus system over 30 runs

Algorithm	SFLA	L-DE	L-SACP-DE
Shortest time (sec)	177.46	1210.73	1212.95
Longest time (sec)	246.49	1239.86	1235.03
Average time (sec)	225.58	1224.27	1221.51

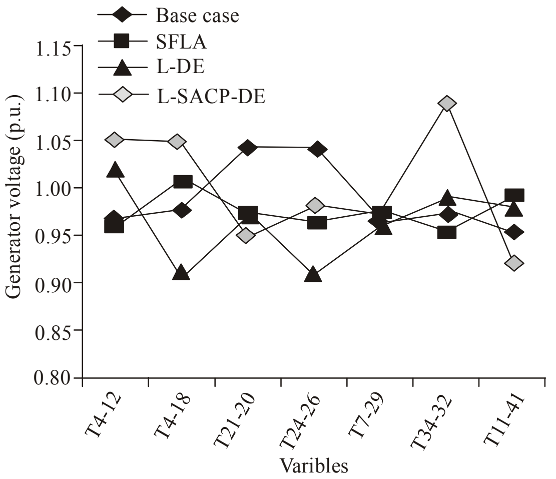

Fig. 2: Generator voltage profiles for various algorithms on IEEE 57-bus

SIMULATION RESULTS

To evaluate the effectiveness and efficiency of the proposed SFLA based reactive power optimization approach, the standard IEEE 57-bus power systems are used as the test systems.

IEEE 57-bus power system: The IEEE 57-bus system consists of seven generators, 80 lines in which 15 lines are tap setting transformers with discrete operating values and three capacitor banks. Seven buses are selected as PV-buses and Vθ-bus as follows: PV-buses: bus 2, 3, 6, 8, 9, 12; Vθ-bus: bus 1, respectively. The others are PQ-buses. At initial operating condition, active power loss is 0.285654 p.u. The search space of this case study has 25 dimensions, including seven generator voltages, 15 transformer taps and three capacitor banks.

Table 1 indicates the values of control variable obtained by different algorithms over 30 independent runs. Furthermore, average computing time of SFLA is better than the two versions of DE. Also, it can be seen that the shortest, average and longest computing time of SFLA is less than that of other algorithms are shown in Table 2. Figure 1 to 3 depict control variables profile of different algorithms. The results are given in Table 3. From Table 3, it can be seen that SFLA converges in 30 iterations achieving the least real power loss of 27.1446 MW in less time than all the other listed algorithms (Fig. 4).

Table 3: The best dispatch solutions for various algorithms on IEEE 57-bus system (p.u.)

Algorithms	$\sum P_G$	$\sum Q_G$	P_{loss}	Q_{loss}	$\%P_{loss}$	VSM	ΔV_L
SFLA	12.7794	3.3272	0.271446	-1.1816	4.6055	0.21650	0.1584
L-DE	12.7999	3.3656	0.291864	-1.2158	-1.2380	0.17012	2.8865
L-SACP-DE	12.7812	3.2085	0.273183	-1.1868	4.0185	0.18300	4.2829

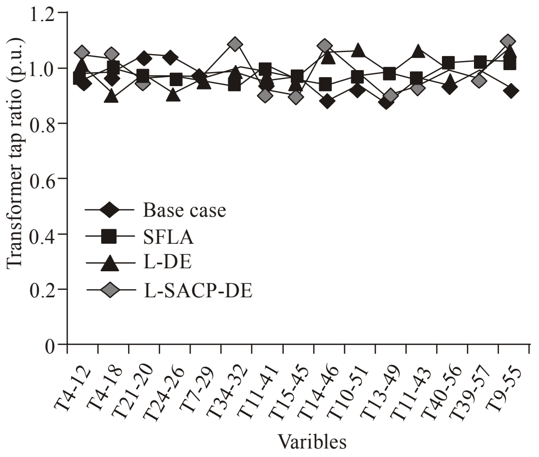

Fig. 3: Transformer tap ratio profiles for various algorithms on IEEE 57-bus

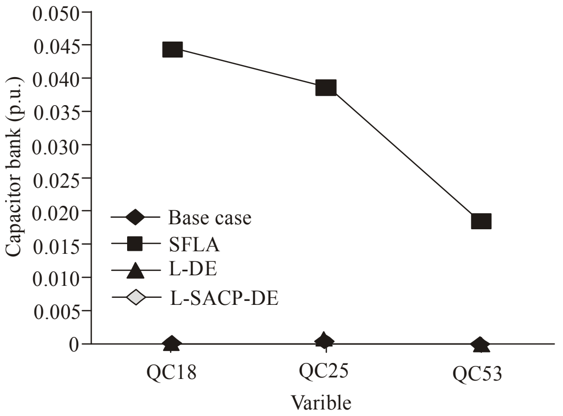

Fig. 4: Capacitor bank profiles for various algorithms on IEEE 57-bus

CONCLUSION

In this study, a shuffled frog leaping algorithm to solve optimal reactive power dispatch problem, considering various generator constraints, has been successfully applied. The proposed method formulates constrained problem with competing objectives namely; minimization of the real power loss, minimization of voltage deviation and maximization of the voltage stability margin. This method shows good performance for the voltage stability enhancement of large complex power system networks. Experimental results indicate that the shuffled frog leaping algorithm is more effective in global search exploration.

REFERENCES

Abido, M.A., 2006. Multiobjective optimal var dispatch using strength pareto evolutionary algorithm. Proceeding of IEEE Congress on Evolutionary Computation (CEC'2006), pp: 2745-2751.

Dai, C., W. Chen, Y. Zhu and X. Zhang, 2009. Reactive power dispatch considering voltage stability with seeker algorithm. Electr. Pow. Syst. Res., 79(10): 1462-1471.

Jeyadevi, S., S. Baskar, C.K. Babulal and M. Willjuice Iruthayarajan, 2011. Solving multi objective optimal reactive power dispatch using modified NSGA-II. Int. J. Elec. Power, 33(2): 219-228.

Jeyanthy, P.A. and D. Devaraj, 2010. Hybrid particle swarm optimization for multi-objective reactive power optimization with voltage stability enhancement. Int. J. Electr. Power Eng., 1(2): 16-21.

Mancer, N., B. Mahdad and K. Srairi, 2012. Multi objective optimal reactive power flow based STATCOM using three variant of PSO. Int. J. Energ. Eng., 2(2): 1-7.

Shi, J. and Y. Liu, 2005. Fuzzy evaluation based multi-objective reactive power compensation in distributed network. In: Wang, L. and Y. Jin (Eds.), FSKD 2005. LNAI 3613, Springer-Verlag, Berlin, Heidelberg, pp: 11-19.

Xiong, X., H. Cheng and H. Li, 2008. Optimal reactive power flow incorporating static voltage stability based on multi-objective adaptive immune algorithm. Energ. Convers. Manage., 49(5): 1175-1181.

Zhen, L., L. Mengshi, W. Tan and Q.H. Wu, 2007. Multi-objective optimization of reactive power dispatch using a bacterial swarming algorithm. Proceeding of Chinese Control Conference (CCC, 2007), pp: 460-464.