Optimal multiple distributed generation output through rank evolutionary particle swarm optimization

J.J. Jamian [a,*], M.W. Mustafa [a], H. Mokhlis [b]

[a] Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
[b] Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received 10 May 2013
Received in revised form
29 July 2014
Accepted 2 November 2014
Communicated by: A. Abraham
Available online 11 November 2014

Keywords:
Benchmark function
Distributed generation
Particle swarm optimization
Power loss reduction
Re-sizing

ABSTRACT

The total power losses in a distribution network are usually minimized through the adjustment of the output of a distributed generator (DG). In line with this objective, most researchers concentrate on the optimization technique in order to regulate the DG's output and compute its optimal size. In this article, a novel Rank Evolutionary Particle Swarm Optimization (REPSO) method is introduced. By hybridizing the Evolutionary Programming (EP) in Particle Swarm Optimization (PSO) algorithm, it will allow the entire particles to move toward the optimal value faster than usual and reach the convergence value. Moreover, the local best (P_{best}) and global best (G_{best}) values are obtained in simplify manner in the REPSO algorithm. The performance of this new algorithm will be compared to 3 well-known PSO methods, which are Conventional Particle Swarm Optimization (CPSO), Inertia Weight Particle Swarm Optimization (IWPSO), and Iteration Particle Swarm Optimization (IPSO) on 10 mathematical benchmark functions, and solving the optimal DG output problem. From the results, the IWPSO, IPSO and REPSO methods gave the similar "best" value in all functions after being tested 50 times, except for Function 6. However, the REPSO algorithm provided the lowest SD value in all problems. In the power system analysis, the performance of REPSO is similar to IWPSO and IPSO, and better than CPSO, but the REPSO algorithm requires less numbers of iteration and computing time. It can be concluded that the REPSO is a superior method in solving low dimension analysis, either in numerical optimization problems, or DG sizing problems.

1. Introduction

Distribution network is the most important network that delivers electrical power to consumers. In view of the fact that the configuration of the distribution network is in radial connection, it causes the power to flow in only a single direction, and the total power loss will increase with respect to the distance between the load and the main power supply. However, the existence of Distribution Generation (DG) can reduce the total power loss problem in the distribution network [1–3]. Furthermore, the presence of DG units will also increase the reliability of supply and provide numerous benefits, such as relieving transmission and distribution bottlenecks, adding new generation capacity, and can be used as a support to the power system maintenance and restoring operations [4]. However, the connection of DG will result in an opposite effect if the output of the DG is inappropriate [5]. Therefore, many approaches and techniques utilize either analytical or Meta-Heuristic optimization to obtain the appropriate location and size for the single DG [2,3] or multiple DG units [6,7].

In this article, a new Meta-heuristic technique, which is the hybridization between Evolutionary Programming (EP) and Particles Swarm Optimization (PSO), is introduced. The concept of combination, ranking, and selection in the EP will help the particles in PSO move faster toward the global optimal solution compared to the original PSO. Furthermore, the EP algorithm will help simplify the procedure of PSO to obtain the new population set in the next iteration. This new algorithm is known as Rank Evolutionary Particle Swarm Optimization (REPSO). Since the performance of Meta-heuristic optimization can be measured through the capability of the algorithm in providing the best optimal results (global optimal value), the consistence results (small standard deviation), the total computing time, and the number of iteration required in searching for an optimal solution, the performance of REPSO will be validated with 3 other PSO methods, which is the conventional Particle Swarm Optimization (CPSO), Inertia Weight Particle Swarm Optimization (IWPSO), and the Iteration Particle Swarm Optimization (IPSO) on standard mathematical benchmark functions, and in solving the optimal output of multiple DG units in 33-bus distribution network. The results in the "*results and discussion*" section have shown that the REPSO algorithm is capable of providing

* Corresponding author. Tel.: +6013 3854888; fax: +607 5566272.
E-mail addresses: jasrul@fke.utm.my (J.J. Jamian), wazir@fke.utm.my (M.W. Mustafa), hazli@um.edu.my (H. Mokhlis).

http://dx.doi.org/10.1016/j.neucom.2014.11.001
0925-2312/

the global optimal results with the smallest standard deviation, shortest computing time, and the smallest number of iteration in all of the analysis. Thus, for applications that require shorter computing time with the guarantee of consistence optimal results, REPSO seems to be the best option.

The remainder of the article is organized as follows. Section 2 present related researches about the implementation of the optimization methods in DG sizing analysis, followed by the briefing on the optimization methods that are used in this study in Section 3. The detail on new REPSO algorithm is discussed in Section 4. Section 5 describes the performance of the PSOs' methods in solving the mathematical function and the optimal multiple DG output. Finally, Section 6 concludes the study.

2. Related research

Since the incompatibility size and location of DG will affect the performance of the network, many researchers have applied optimization techniques, such as Genetic Algorithm (GA) [8,9], Particle Swarm Optimization (PSO) [10–12], Evolutionary Programming (EP) [13], Ant Colony Optimization (ACO) [14,15], and others to obtain the optimal sizing and location for the DG. The analysis in [16] is one of the most recent studies on optimal placement and sizing of DG units in a distribution system. The author used two heuristic methods to determine the location and the size of the DG. The GA is used to determine the suitable location, while the PSO is used to optimize the size of DG. The result of the combination method is compared to the PSO and GA methods used to determine both the location and size of the DG. From their analysis, the GA/PSO combination resulted in the best optimal location and sizing in 33-and 69-bus systems. However, since the GA and PSO analysis was done separately, the computing time for this combination method will exceed the original algorithm. The used of improved GA technique with self-organizing learning proposed by Qiau Cai *et al.* [17] can help in reducing the computing time by GA and increase the ability to achieve global optimal solution.

Rather than combining two different optimization methods, most of the researchers modified the original optimization method to obtain better results in the analysis [18–21]. T. Niknam [22] modified the honeybee optimization method in solving the multi-objective problem for DG allocation and sizing. In their research, the DG unit is presented as renewable energy resources. The authors tried to obtain the optimal size of DG by accounting for the impacts of DG size to the power losses value, voltage profile, total generation cost, and emission. However, it became very difficult to validate the optimal results when too many objective functions were taken into account during the optimization process. The Pareto front results will give many possible solutions, which make it more difficult for a user to decide. Furthermore, by improving the power losses in the system with optimal DG output, the voltage profile and stability of the system will also improve (in most cases).

The analysis on the impacts of the load model to the DG sizing and location is among the popular research being discussed [23–27]. Since different loads type exists in the network, the impacts of optimal location and size of DG will be different compared to the results that are obtained during constant power load model analysis. Thus, the results on power losses for different load models will be different as well. However, as long as the optimization method that is used to analyze the system can give optimal results, the different load model in the system is not a big issue for the method in providing the optimal DG value. Therefore, this article focuses on the performance of the proposed method to determine the optimal multiple DG units' output in the 33-bus distribution network in order to reduce the power losses in the system. The algorithm will be tested 30 times with different initial random values, so that the robustness and the efficiency of the proposed algorithm can be proved.

3. Overview on optimization methods

The original Particle Swarm Optimization (PSO) algorithm introduced by Ebehart and Kennedy [28] is adapted from the food searching behavior of groups of animals, such as birds. The birds will move in a group toward the foods' location in a specific speed until they arrive at the destination. In the optimization concept, these birds are known as particles that move in the searching space toward the optimal solution (food location). Two parameters are introduced in the optimization process, which are position (x_i) and speed (v_i) of the particles. The position and velocity of i^{th} particle in d-dimensional search space is represented as $x_i=(x_{i1},x_{i2},\ldots,x_{id})$ and $v_i=(v_{i1},v_{i2},\ldots,v_{id})$, respectively. Furthermore, the speed for the particle depends on the current position (x_i), Local best value ($P_{best\text{-}i}$), and Global best value ($G_{best\text{-}i}$). The $P_{best\text{-}i}$ is defined as the best position that has been visited by i^{th} particle until the current (k) iteration $P^k_{best\text{-}i}=(P^k_{best\text{-}i1}, P^k_{best\text{-}i2},\ldots, P^k_{best\text{-}id})$, while $G_{best\text{-}i}$ is defined as the best position among the $P^k_{best\text{-}i}$ in a population (group). Thus, the new position (x_i^{k+1}) for the particle in the Particle Swarm Optimization will be updated based on the velocity (v_i^{k+t}) value, using the formula

$$v_i^{k+1}=v_i^{k+1}+c_1r_1(P^k_{best-i}-x_i^k)+c_2r_2(G^k_{best}-x_i^k) \tag{1}$$

$$x_i^{k+1}=v_i^{k+1}+x_i^k \tag{2}$$

where k is the number of iteration, v_i is the velocity of i^{th} particle, c_1 and c_2 are the acceleration coefficients for cognitive and social components, respectively; r_1 and r_2 are independently uniform distributed random number between 0 and 1. In this study, the PSO is known as Conventional Particle Swarm Optimization (CPSO).

However, the proper control on global search and local search ability is needed for the algorithm to avoid being trapped in the local optimal results. Therefore, Shi and Eberhart [29] adopted a new parameter in the velocity formula called inertia weight (w). In this study, this optimization method is known as the Inertia Weight Particle Swarm Optimization (IWPSO). The inertia weight can be used to balance the global search and local search ability to reach a global optimum point. Thus, the velocity formula for the IWPSO has change to (3). Furthermore, from the equation, the IWPSO algorithm consists of 3 controllable parameters that influence the exploration and exploitation of the optimization process, which are the cognitive acceleration coefficient (c_1), the social acceleration coefficient (c_2), and the weight coefficient (w). The updating formula for new positions in IWPSO is still similar to (2)

$$v_i^{k+1}=\omega^k V_i+c_1r_1({P_{best}}^k-x_i^k)+c_2r_2({G_{best}}^k-x_i^k) \tag{3}$$

The other improvements on the PSO optimization method have been proposed by Tsung Y.L and Chu L.C. [30]. By introducing a new parameter known as I_{best}, it can help the original PSO to improve the solutions' quality and the computing time of the algorithm. This I_{best} is defined as the best value of the fitness function that has been achieved by any particles in the present iteration. In other words, I_{best} value is the P_{best} value that is randomly selected among the existing particles in the current population. In addition, the authors also introduced a dynamic acceleration constant parameter, c_3 value for the I_{best} value, as shown in (4). This improved PSO method is known as Iteration Particle Swarm Optimization (IPSO). Therefore, the velocity formula in for IPSO algorithm will become (5).

$$c_3=c_1(1-e^{-c_1k}) \tag{4}$$

$$v_i^{k+1}=wv_i^{k+1}+c_1r_1(P^k_{best-i}-x_i^k)+c_2r_2(G^k_{best}-x_i^k)+c_3(I^k_{best}-x_i^k) \tag{5}$$

The authors claim that the fourth term in the velocity formula helped the IPSO have faster converged solutions compared to the original PSO.

4. Rank evolutionary particle swarm optimization

In this study, the novel Rank Evolutionary Particle Swarm Optimization (REPSO) is introduced to optimize the multiple units of DGs' output in the distribution network. As mention earlier in Section 1, the REPSO is produced from the hybridization process between Evolutionary Programming (EP) and Particle Swarm Optimization (PSO) in order to have faster and accurate solutions. Fig. 1(a) and (b) illustrate the differences between the movement concept in the PSO and REPSO, respectively. In the original PSO, the particles will move based on their own velocity toward to the optimal solution. However, if there is a particle that is located far from the optimal solution, the process to "bring" the particle to reach the optimal point will require several iterations. However, in the REPSO, the "blank" position left by the "in front" particle, which is close to the optimal value, will be filled up by other particles. This "filling" concept can be done using the EP concept, where combination, rank, and selection. Table 1 show the example of REPSO process that will generate the "filling" concept. Let the number in the table be the fitness value for the particles, and columns 1 and 2 is the previous and new set of particles' population, respectively (Solid circle and Doted circle in Fig. 1(a) respectively). Both population is combined and sorted based on their respective fitness values. Next, the selection concept caused the high potential solution to survive in the selection, while the other is terminated. These three steps in the EP concept caused the high potential solution from the combination process to remain as a new population for the current iteration, and render the population closer to the optimal point. It should also be pointed out that the REPSO algorithm only adopted the combination, rank, and selection (which one of the EP algorithm) process in the original PSO. Therefore, there is no mutation process for the EP algorithm in REPSO.

Table 2 summarizes the PSO and REPSO algorithms steps in determining the optimal results. The processes between steps 1 to 4 are similar for both methods (the shaded part). The differences began in the process of determining the new population (x_{i+1}), $P_{best\text{-}i}$ and G_{best}, where the REPSO method is simpler and require less comparison. In the original PSO, the P_{best} value is obtained by comparing each individual particle between the previous set of population and the new set of population, whilst the G_{best} is determined based on comparison of previous G_{best} value and the new G_{best} value. It requires 2 processes of comparison in order to find the new position of the particles. On the other hand, the values of P_{best} and G_{best} for REPSO can be obtained just after the ranking and selection process between the previous set of population and the new set of population, as shown in Table 1 (columns 5 and 6). After the selection process, all survival particles is set to be the P_{best} value in the REPSO, and the particle located at the top of P_{best} is equal to the G_{best} value. In order words, both parameters is obtained just after the selection process. Furthermore, since the P_{best} value for REPSO is always equal to the selected particles, the formula for the new velocity in REPSO is simplified into the form shown in (6).

$$V_{i+1} = w_i V_i + c_2 r_2 (G_{best} - x_i) \tag{6}$$

Thus, the calculation for the new velocity in REPSO has become simpler, but the particles still move to new positions based on the global optimal value references. The new position formula, x_{i+1}, for the REPSO remain similar to the original PSO stated in (4). Fig. 2 shows the pseudo code for the REPSO algorithm.

5. Result and discussion

In order to test the robustness and effectiveness of the proposed method, the REPSO algorithm will be compared with

Table 1
Example of REPSO process to obtain optimal results.

set 1 (Parent)	set 2 (Children)	Combination set1+set2	Sort	Selection / P_{best}		G_{best}
7.432	8.001	7.432	2.054	**2.054**	→	**2.054**
6.281	5.964	6.281	3.001	**3.001**		
3.297	4.315	3.297	3.211	**3.211**		
7.441	6.879	7.441	3.297	**3.297**		
3.211	4.216	3.211	4.216	**4.216**		
2.054	3.001	2.054	4.315	**4.315**		
		8.001	5.964			
		5.964	6.281			
		4.315	6.879			
		6.879	7.432			
		4.216	7.441			
		3.001	8.001			
		Evolutionary Programming Implementation				

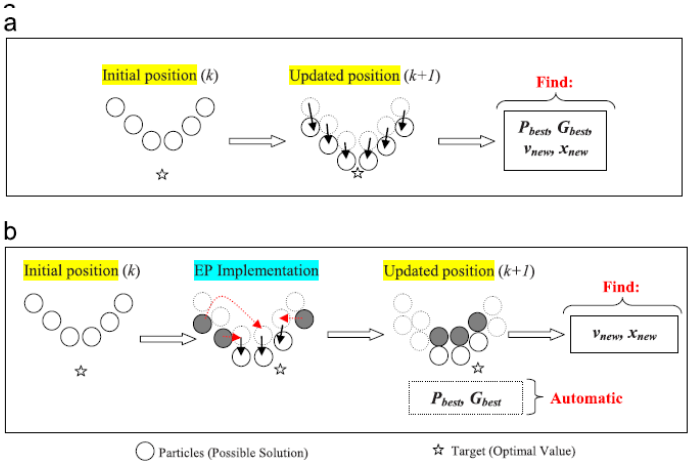

Fig. 1. Comparison Between Original PSO and REPSO particles' movement.

3 other PSO methods, which are Conventional Particle Swarm Optimization (CPSO), Inertia Weight Particle Swarm Optimization (IWPSO), and the Iteration Particle Swarm Optimization (IPSO). All these PSO algorithms are used to solve 10 fix dimension standard benchmark functions and real application analysis, which is finding the suitable output for multiple DG units in a 33 distribution network, so that the power losses in the system is reduced. In order to make a fair comparison, the parameters for the PSO algorithms are being set to the same value. The population size that is used in this study is 20 (N=20), with the cognitive and social coefficients (c_1 and c_2) values being 0.5. The analysis was performed using MATLAB 7.8 on Core2Duo processor, 2.00 GHz, and 2GB RAM.

5.1. Comparison the performance CPSO, IWPSO, IPSO and REPSO algorithms in mathematical standard test function

Table 3 shows the ten benchmarked mathematical functions used in this study to investigate the performance of REPSO. The functions consist of unimodal (U), multimodal (M), separate (S), or non-separable (N) types, and the variables ranges for the possible solution are different depending on the problems. All PSO algorithms

Table 2
Differences between PSO and REPSO steps.

Step	PSO	REPSO
1	Generate n-Number of Particles, x_i	Generate n-Number of Particles, x_i
2	Calculate All Particles Fitness, y_i	Calculate All Particles Fitness, y_i
3	Set: P_{best}=all initial particle G_{best}=the best fitness among population	Set: P_{best}= all initial particle G_{best}=the best fitness among population
4	Calculate new position x_{i+1} [Eq. (2)], new velocity v_{i+1} [Eq. (1)] and new fitness value y_{i+1}	Calculate new position x_{i+1} [Eq. (2)], new velocity v_{i+1} [Eq. (6)] and new fitness value y_{i+1}
5	Determine new P_{best} set by comparing the previous $P_{best(i)}$ fitness with new $P_{best\ (i+1)}$ fitness on each particles	Combine previous population and new population and sort base on fitness value [*Example: Table 1 – column 3 & 4*]
6	Determine new G_{best} set by comparing the previous G_{best} fitness with new G_{best} fitness	Select the "half"-best particles as new set of population with P_{best} and G_{best} automatically generated. [*Example: Table 1 – column 5 & 6*]
7	Repeat step 4	Repeat step 4

Step 1 Randomize N number of particles (x)
Check either the random number fulfils all the constraints or not.
if yes **then**
Save the particles value
if number of "Save" particles == N **then**
Proceed to Step 2
else
continue to Step 1 (line 2).
end if
else
Delete and re-randomize new particles
end if
Step 2 Calculate the fitness value for the particles
Step 3 **Set** P_{best}, = all particles and determine G_{best} based on fitness value.
Step 4 Find new particles (x^{new}) from the velocity value (v_i)
Check the x^{new}
if $x^{new}<x_{max}$ **or** $x^{new}>x_{min}$ **then**
Proceed with the updated particle value
else
Assigned the $x^{new}=x_{max}$ for $x^{new}>x_{max}$
Assigned the $x^{new}=x_{min}$ for $x^{new}<x_{min}$
end if
Step 5 Combined the previous population (x) with the new population (x^{new})
Sort the population based on the fitness value
Select the top N particles as the successful population
Assign the top position of successful particles as G_{best} value.
Step 6 Check the stopping criteria
If fulfil stopping criterion **then**
Stop and show the optimal results
else
Continue to **Step 4**
end if

Fig. 2. Pseudo code of REPSO algorithm.