ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20180525204445
Original scientific paper

Distribution Network Reconfiguration with Large Number of Switches Solved by a Modified Binary Bat Algorithm and Improved Seed Population

Michell J. QUINTERO-DURAN, John E. CANDELO-BECERRA, Katherine CABANA-JIMENEZ

Abstract: The paper presents a methodology based on a Modified Binary Bat Algorithm (MBBA) and Improved Seed Population search that provides nearly optimal solutions to the power loss minimization problem, considering network reconfiguration and a large number of switches. The existence of many switches leads to a very large number of combinations, making it hard for algorithms to find a good solution. The proposed method is based on eliminating non-feasible solutions and defining an initial matrix with improved seed population for searching the optimal solution. This seed is used for the random process of the algorithm to produce new solutions and is continually updated to obtain better results close to the optimal solutions found during the searching process of the metaheuristic algorithm. This algorithm was tested against the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and the Seed Population search alone on the modified versions of the IEEE 13-node test and IEEE 123-node test feeders. From several runs, the proposed method reached the optimal solution more times than the other algorithms and the remainder achieved near-optimal solutions. With this result, the MBBA provides good options to improve the solutions in the network reconfiguration problem with a large number of switches.

Keywords: Bat Algorithm; Modified Binary Bat Algorithm; power loss minimization; power optimization; reconfiguration; Seed Population

1 INTRODUCTION

The reconfiguration of distribution networks continues to be an important technique to reduce power losses [1]. It involves changing the topology of the network with the aid of binary elements located in the network, called "switches". Switches are classified into sectionalizing switches (*normally closed switch*) and tie switches (*normally open switch*). Topology changing is based on the modification of the initial status of a single switch, i.e., if a sectionalizing switch is considered, then it will be opened and vice versa [2]. However, when this technique considers a large number of switches, the binary combination of the possible changes makes it difficult to find the solution quickly. This is because evaluating all possible combinations of switches takes considerable time to search the optimal reduction of power loss, which is important to achieve better network flexibility in the operational mode [3].

Many authors have performed distribution network reconfiguration to minimize power loss [3], [4], [13], [5]–[12]. Most researchers have used metaheuristic techniques to reduce the time evaluation by finding good solutions and redirecting the search process to the best-found solutions [14]; however, these algorithms have some convergence problems and the solutions are not always obtained near the optimal [15]. A detailed state-of-the-art in recent trends concerning reconfiguration and most used metaheuristic techniques is in [16].

This paper presents a method that includes seeds in the searching process and eliminates non-feasible combinations of switches to reduce the search in the Binary Bat Algorithm (BBA) proposed by Mirjalili et al. [17]. The network reconfiguration is performed to reduce active power loss and we consider the commonly used constraints. The proposed method is based on eliminating non-possible solutions and defining an initial matrix of improved seed population for searching the optimal solutions. This seed is used for the random process of the algorithm to produce new solutions close to the optimal solutions found during the searching process. This article also works on the determination of greater network flexibility by means of algorithms that allow finding solutions very close to the optimum. The used technique integrates methods of disposal of non-feasible solutions and defines an inner search for best solutions focused on having higher option rates to achieve the final goal.

The rest of the paper is organized in five sections. Section 2 presents mathematical formulation for the problem. Sections 3 and 4 present the binary bat algorithm and a seed search for avoiding unfeasible combinations that do not comply with the radial network constraints. Section 5 presents the materials and method used in this research. Section 6 presents the results and discussion, and Section 7 presents the conclusions.

2 PROBLEM FORMULATION

Real power loss is obtained by multiplying the square of the current magnitude that flows through the branches and the resistance of those branches [18]. The minimization of real power loss is performed with Eq. (1), where l is the branch number, R_l is the resistance of the l^{th} branch, and I_l is the current flowing through each phase of branch l:

$$\min\left(P_{\text{loss}}\right)=\sum_{l=1}^{nbr} R_l \times I_l^{\,2}. \tag{1}$$

Distribution network reconfiguration has been widely used for minimizing real power loss. In case there are already tie branches installed in the system, it is assumed that the switches are also fixed, which means that there is no need to make high investments on installing new devices. It is normal for a distribution network operator to have switches in the system for several purposes, such as restoration, fault isolation, or even disconnection [18].Nevertheless, when there are no tie branches, the distribution network operator should consider the installation costs to improve reliability.

Switches only present two operating states: normally open (tie-switch) and normally closed (sectionalizing switch). When a switch is open, a "zero" is convenient for

mathematical representation and when it is closed, the number "one" is the correct mathematical representation [2], [3]; thus, the distribution network reconfiguration process becomes a binary approach.

Real distribution systems are large enough to consider them as large-scale designs and the larger the system the bigger the combinatorial problem becomes. This is because the total number of possible combinations is directly dependent on the number of switches installed in the system as presented in Eq. (2), where *comb* is the total number of possible combinations that change the topology of the system and *n* is the number of switches installed in the system:

$$comb = 2^n. \tag{2}$$

Once a switching operation is performed, the topology of the system is changed and a power flow must be achieved for evaluating the state of the system, including radiality, real power loss, node voltage profile, branch current limits, and feeding of all nodes [19].

The objective function for the combinatorial problem is stated in Eq. (1) and the constraints are listed in Eqs. (3)–(7), where *nbr* is the total number of branches, *N* is the number of nodes, ***A*** is the adjacency matrix that represents the connections between nodes and its determinant states whether all nodes are fed or not, $g(x)$ represents the power flow, and Eq. (5) guarantees the power balance in the system. $U_{i\min}$ is the lower voltage node limit, U_i is the voltage at node *i*, $U_{i\max}$ is the upper voltage node limit, I_l is the current flowing through branch *l*, and $I_{l\max}$ is the upper thermal limit of the conductor material of branch *l*:

$$nbr = N - 1, \tag{3}$$

$$\det(\boldsymbol{A}) = \pm 1, \tag{4}$$

$$g(x) = 0, \tag{5}$$

$$U_{i\min} \le U_i \le U_{i\max}, \tag{6}$$

$$I_l \le I_{l\max}. \tag{7}$$

For larger systems, an exhaustive search for the configuration that allows the lower real power loss cannot be performed in real scenarios as this would be a highly time-consuming task. This is the reason why, over the last two decades, nature-inspired algorithms have become useful tools for finding optimal or near-optimal solutions in a reasonable time.

3 BINARY BAT ALGORITHM

The bat algorithm was first proposed by Yang [20] for solving combinatorial continuous optimization problems. Although it has been used for solving some electrical problems [21], [22], the main application is not useful for discrete or binary approaches. Mirjalili et al. introduced the binary version for facing binary problems [17]. This technique was inspired by the use of echolocation by bats searching for prey, avoiding obstacles, and locating their roosting crevices in the dark [20] while they are flying (bats are mostly blind).

Bats may be mathematically analysed to emulate their behaviour to optimize both continuous and binary problems. This technique avoids clogging in local minima with the aid of random search and works as an iterative approach. For a complete explanation of the continuous Bat algorithm, refer to Yang's work [20]. The general binary Bat formulation is summarized in the current section taken from Mirjalili et al. [17].

Bats perform echolocation by decreasing the loudness and increasing the rate of emitted ultrasonic frequencies that are established within a range from maximum to minimum [17]. In the standard bat algorithm, an artificial bat has an initial frequency, velocity, and position vectors, which are updated within the iteration process. A frequency vector is calculated for every bat and is updated using Eq. (8), where F_i is the i^{th} bat sound frequency and $F_{\min}$ and $F_{\max}$ are the minimum and maximum selected frequencies, respectively. A random value is stated as β to change the value of F_i:

$$F_i = F_{\min} + (F_{\max} - F_{\min})\beta. \tag{8}$$

The velocity is referred to as the probability for a bat to change its position and is presented in Eq. (9), where $v_i^k(t + 1)$ is the i^{th} bat velocity at dimension *k* in iteration *t*+1. $v_i^k(t)$ is the i^{th} bat velocity in the k^{th} dimension in iteration *t*. $x_i^k(t)$ is the i^{th} bat position in the k^{th} dimension in iteration *t*. *Gbest* is the best so far found position:

$$v_i^k(t+1) = v_i^k(t) + \left[x_i^k(t) - Gbest\right]F_i. \tag{9}$$

For binary approaches, the V-shaped transfer function is used to convert the calculated velocity to a binary space as given in Eq. (10), where $V(v_i^k(t + 1))$ is the V-shaped transfer function for velocity:

$$V\left(v_i^k(t+1)\right) = \left|\frac{2}{\pi}\arctan\left(\frac{\pi}{2}v_i^k(t+1)\right)\right|. \tag{10}$$

The position in a binary search space is like a hypercube because "position" is a vector with *n* bits that represent each switch installed in the system and may have only one state at a time from a binary decision. If a switch is closed, then that bit will contain the number 1, otherwise 0. The complete position vector represents a configuration for the distribution network. The binary position per bit is calculated from Eq. (11), where $x_i^k(t+1)$ is the bat position in dimension *k* for the iteration *t*+1. If a random value is smaller than the V-shaped conversion for velocity at the iteration *t*+1 value, then the binary position for a bat *i* in dimension *k* for iteration *t*+1 will be the contrary with respect to the position for the same bat *i* and same dimension, but at iteration *t*. In other words, if the position is binary and represented by 0, then it will be changed to 1. Otherwise, the position keeps the same value:

$$x_i^k(t+1) = \begin{cases} \left(x_i^k(t)\right)^{-1} \xrightarrow{if} rand < V\left(v_i^k(t+1)\right) \\ x_i^k(t) \xrightarrow{if} rand \ge V\left(v_i^k(t+1)\right) \end{cases}. \tag{11}$$

The loudness and pulse emission rate balance a combination between Particle Swarm Optimization (PSO) and intense local search for avoiding local minimal binding. These elements are calculated from Eqs. (12) and (13), respectively; where $A_i(t+1)$ and $A_i(t)$ are the loudness at iteration $t+1$ and t, respectively. $r_i(t+1)$ and $r_i(0)$ are the pulse emission rate at iteration $t+1$ and the initial value, respectively. α is a constant that guarantees that the loudness will return to 0 as the iterations increase and γ is a constant that guarantees that the value of the pulse emission rate will return to its initial value as the iterations increase:

$$A_i(t+1) = \alpha A_i(t), \tag{12}$$

$$r_i(t+1) = r_i(0)\left[1-\exp(-\gamma t)\right]. \tag{13}$$

The pulse emission rate reinforces the local search while the loudness keeps the algorithm making a global search randomly. The process described above is performed by several virtual bats and a stop criterion, which is usually a maximum number of iterations.

The binary bat algorithm evaluates one dimension of a vector at a time. This is a drawback for the reconfiguration approach because many unfeasible configurations may be obtained. An unfeasible configuration is one that does not accomplish the radiality constraints presented in Eqs. (3) and (4). The next section presents a modified binary bat algorithm for avoiding the above.

4 MODIFIED BINARY BAT ALGORITHM

The distribution network reconfiguration approach for reducing real power loss requires some modifications in the binary version of the bat algorithm mentioned above. In this section, we explain those modifications and how they are inserted in the standard binary bat algorithm. Obtaining the main steps from the flowchart presented in previous literature [17], we proposed a modified binary bat algorithm applied to distribution network reconfiguration as shown in Fig. 1.

4.1 Loading Case of Study

The case of study provides the following information about the distribution network: slack bus, branch data, number of nodes, regulators, and number and locations of switches in the system.

4.2 Determining the Number of Tie Switches

From the case of study and Eq. (3), it is possible to know the required number of branches to keep a radial configuration. The number of branches must be the total number of nodes minus 1. If the number of branches is smaller than that number, then simply subtract from the required number of branches, the number of branches registered in the case of study to know how many switches may be closed. This is expressed in Eq. (14), where *nbr* is the required number of branches calculated from Eq. (3), *num_branches* is the number of branches registered in the case of study, and *ssw* is the number of sectionalizing switches required to reach *nbr*:

$$nbr = num\_branches + ssw. \tag{14}$$

The number of tie switches is obtained from Eq. (15), where *sw* is the total number of switches and *tsw* is the required number of tie switches for the case of study:

$$sw = ssw + tsw. \tag{15}$$

The sectionalizing switches' (*ssw*) and tie switches' (*tsw*) parameters are important to identify only feasible configurations.

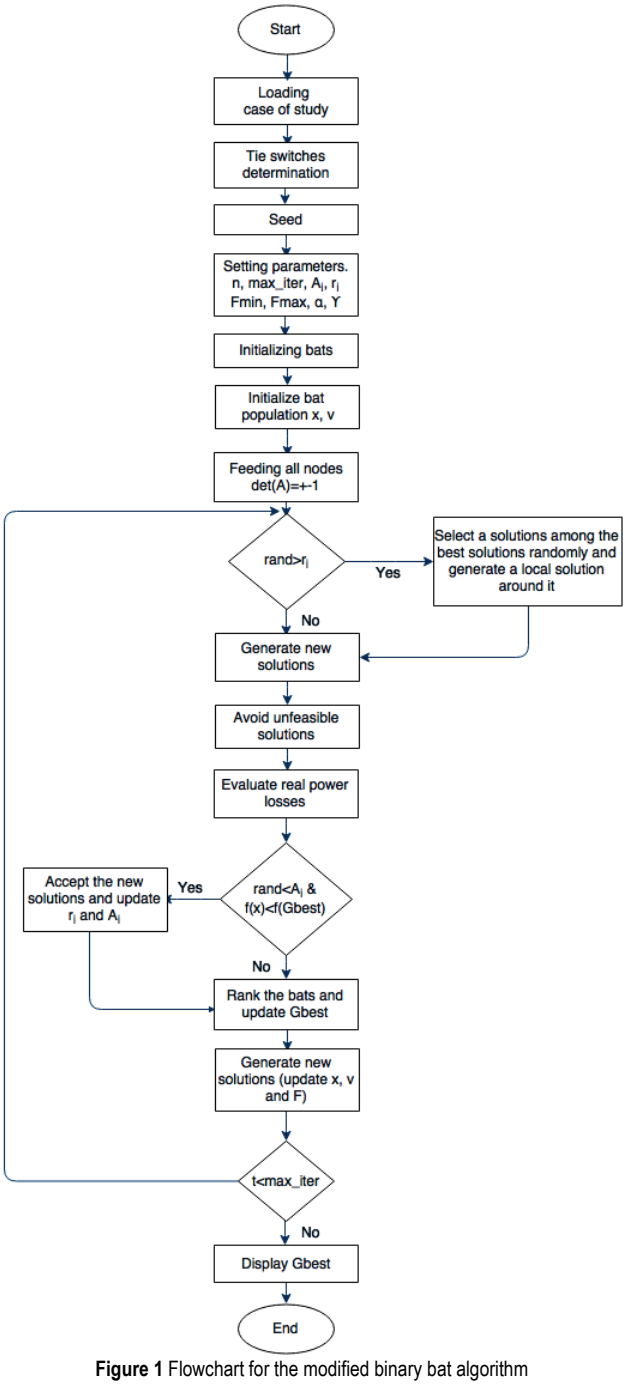

Figure 1 Flowchart for the modified binary bat algorithm

4.3 Avoiding Unfeasible Configurations – Seed

Unfeasible configurations spend valuable time and memory in the PC because they do not accomplish the radiality constraint expressed in Eq. (3). Avoiding these

unfeasible configurations, helps earn extra time to find the optimal solution. Furthermore, this process assures that the algorithm only evaluates configurations where the number of branches is equal to *nbr*.

From Eqs. (14) and (15), we know both the number of switches (*sw*) and the number of sectionalizing switches (*ssw*). A permutation with no repetitions is performed for obtaining all the configurations that contain the required number of branches (*nbr*). Equation (16) presents the permutations obtained from two numbers, in this case *sw* and *ssw* [23] and *binary* represents the feasible configurations:

$$binary = \frac{sw!}{ssw!\left(sw - ssw\right)!}. \tag{16}$$

As this algorithm was prepared in MATLAB©, the function *permpos* was used for obtaining all the permutations calculated from Eq. (16). Every configuration attained from this step fulfils with Eqs. (3), (14), and (15).

4.4 Setting Binary Bat Algorithm Parameters

Initial parameters are established to develop the optimization process via the modified binary bat algorithm. Those parameters are the number of bats (*n*), maximum iterations (*max_iter*), loudness (*A*), pulse emission rate (*r*), maximum and minimum frequencies (F_{max} and F_{min}), and constants α and γ.

4.5 Initializing Bats

For a single bat, a configuration is selected randomly from the *binary* matrix calculated in Eq. (16). For real purposes, the number of bats is set and a matrix for all initial bat configurations is described for each bat. This matrix contains *n* rows, which is the number of bats, and *sw* columns, which is the number of switches installed in the system. For developing the random selection, the MATLAB© function *randi* was used.

It is important to emphasize that the random selection process to determine the initial configuration, calculated with Eq. (16), is only used to obtain the initial bat positions and not as a decision variable in the optimization of the tie switches status.

4.6 Testing the Nodes Feeds

Before performing a power flow, the corresponding initial position (configuration) for each bat must be tested to check if all nodes are fed. This step is possible with the evaluation of the adjacency matrix built by the modified three-phase unbalanced backward/forward sweep-based power flow method [24]. Eq. (4) calculates the determinant of the adjacency matrix. Whether the answer is ±1, all nodes are fed, otherwise they are not. When a configuration does not comply with this constraint, it is randomly changed for another combination from the matrix *binary* obtained in Eq. (16). Once a new position is obtained, it must be evaluated in the same manner. The three-phase unbalanced backward/forward sweep-based power flow method is made for every admissible position (configuration) to calculate real power loss. The best so far encountered position is saved as *Gbest*.

The above procedure guarantees that the initial bats (seeds) have feasible configurations, and allows the algorithm to find a near-optimal solution in each run.

4.7 Binary Bat Algorithm Process

When all initial positions are feasible, the binary bat algorithm is performed as presented in Eqs. (8)–(11) and Fig. 1. The steps are as follows:

- Frequency updating: Eq. (8).
- Velocity calculation for each dimension of all bats: Eq. (9).
- V-shaped transfer function: Eq. (10).
- Position updating: Eq. (11).
- Local search: if a random number is greater than *r*, then the position vector is crossed with the best so far encountered solution.

Positions obtained in this process can be unfeasible. To avoid them, go to the next step.

4.8 Evaluation of Updated Positions

The new positions obtained from Eq. (11) may be unfeasible. The simplest form of eliminating these positions (configurations) is adding up all bits of the new position vector. As an example, consider the vector *x* in Eq. (17):

$$x = \left[10011\right]. \tag{17}$$

This example provides three sectionalizing switches and two tie switches. Thus, when adding all dimensions of vector *X*, a total of three closed switches are obtained. For fulfilling radiality constraints, the sum of all dimensions must be equal to *nbr* as presented in Eq. (18), where *k* is the number of total dimensions of vector x_i, and x_i is the position vector for the i^{th} bat:

$$nbr = \sum_{i=1}^{k} x_i. \tag{18}$$

If a configuration does not comply with Eq. (18), Eqs. (8)–(11) are calculated again to get a new position vector. This new vector must be evaluated in the same form until a feasible solution is obtained.

4.9 *Gbest* Updating

A power flow evaluation is performed for each bat to calculate real power loss. A position is considered as better than others are, when power loss is smaller than the calculated from other positions. Whether a new position is better than *Gbest* and a random number is lower than the loudness (*A*), *Gbest* is updated with the new position.

4.10 *A* and *r* Updating

Update loudness and pulse emission rate using Eqs. (12) and (13). Start all over again until the maximum number of iterations is achieved.

5 MATERIALS AND METHODS

5.1 Test Design

The evaluation of the proposed algorithm was done considering the following scenarios:

- Low-scale distribution network: the modified IEEE 13-node test feeder was used to achieve a low-scale system.
- Large-scale distribution network: the modified IEEE 123-node test feeder was selected to demonstrate the accuracy of the modified binary bat algorithm.

Moreover, the proposed algorithm was tested against the standard binary particle swarm optimization (BPSO) [25]. A modified BPSO, which consists of the standard BPSO plus three first steps of the modified binary bat algorithm. The standard genetic algorithm (GA) [26]. A modified GA, which comprises the standard GA plus the three first steps of the modified binary bat algorithm. The standard binary bat algorithm; and the proposed seed alone, which comprises the three first steps of the modified binary bat algorithm plus random selection of feasible configurations. The original algorithms are modified as explained in subsections 4.1 to 4.3, to ensure that the optimization is performed under similar conditions.

For each scenario, all algorithms were used considering the following results: minimal real power loss (best trial), maximum real power loss (worst trial), mean, standard deviation, ranking of the best trial, ranking of the worst trial, the number of times the best trial was reached, and the average time spent by each algorithm.

The ranking above was established by the evaluation of all possible combinations.

Table 1 Parameters used for simulation – IEEE 123-node test feeder

Trials = 30 Population = 30 Max_iterations = 50							
MBBA	A_i	r_i	α	Γ	$F_{\min}$	$F_{\max}$	
	0,9	0,9	0,95	0,15	0	2	
BBA	A_i	r_i	α	Γ	$F_{\min}$	$F_{\max}$	
	0,9	0,9	0,95	0,15	0	2	
MBPSO	V	Up	Low	$W1$	$C1$	$C2$	$V_{\max}$
	1	50	−50	0,5	1	1	4
BPSO	V	Up	Low	$W1$	$C1$	$C2$	$V_{\max}$
	1	50	−50	0.5	1	1	4
MGA	pc	pm	mu				
	0,8	0,3	0,02				
GA	pc	pm	mu				
	0,8	0,3	0,02				
Seed	Does not apply						

Tab. 1 presents the parameters used for each algorithm considering the IEEE 123-node test feeder. For the IEEE 13-node test feeder, the only differences are in the number of trials, which is five, population, which is five, and the maximum number of iterations, which also is five. The remaining parameters keep the same values as those presented in Tab. 1.

From Tab. 1, subscripts are identified as modified binary bat algorithm (MBBA), binary bat algorithm (BBA), modified binary particle swarm optimization (MBPSO), binary particle swarm optimization (PSO), modified genetic algorithm (MGA), and genetic algorithm (GA). Additionally, parameter values were tested and those presented in Tab. 1 are the best adapted to this problem.

5.2 Test Feeders – Cases of Study

The modifications to both the IEEE 13- and 123-node test feeders may be consulted in [24]. For example, the IEEE 13-node test feeder has 4 tie switches and at least 2 of them must be closed to guarantee radiality. In the case of the IEEE 123-node test feeder, there are 22 tie switches and 13 of them must be closed to guarantee radiality.

The size of the two systems can be calculated using Eq. (2), and the space of solutions becomes 16 for the smaller case, and 4,194,304 for the larger one.

6 RESULTS AND DISCUSSION

This section is divided in two subsections. The first part presents the results and discussion concerning the application of all algorithms on the modified IEEE 13-node test feeder. The second part shows the results and discussion about the modified IEEE 123-node test feeder.

We performed simulations in an AMD A10-5745M APU with Radeon(tm) HD Graphics 2.10 GHz, 8GB RAM computer.

6.1 Small-scale Evaluation

A modified IEEE 13-node test feeder is chosen to demonstrate the usefulness of the MBBA and to compare it with the most used PSO and GA techniques. Tab. 2 presents the results for all the algorithms applied to this scenario. This table contains 10 rows of results. Rows 1 and 3 are referred to as the minimum real power loss, expressed in kW, found in the five evaluations performed by each algorithm and the maximum real power loss found by each algorithm, respectively. Rows 2 and 4 present the tie switches for the minima and maxima configuration, respectively. Rows 5 and 6 state the means and standard deviations of all evaluations performed by each algorithm, respectively. The minima and maxima ranking from the evaluation of all feasible solutions are presented in rows 7 and 8. Row 9 presents the number of times that each algorithm found the minimum real power loss value. The average time per algorithm is in row 10.

From Tab. 2, a small-scale system is not enough to establish differences between these techniques as they always find the best configuration, producing a real power loss of 73,61 kW and keeping the switches between nodes 646–684 and 671–692 open. As this is the best configuration, its rank position is number 1. The statistical results are the same for all algorithms and the time average is very close, considering a standard deviation between those times of 0.095, and the slowest algorithm (MGA) has a 5.42% increment compared with the faster algorithm (BPSO). The MBBA does not outperform the other algorithms as its results are almost the same and the execution time average is 3.01% greater than BPSO.

6.2 Large-scale Evaluation

The modified IEEE 123-node test feeder is selected to demonstrate the accuracy of the proposed MBBA. The results of all algorithms are presented in Tab. 3. The description of this table is similar to Tab. 2. The only differences are in rows 3 and 6, which are the first iteration, where each algorithm found the minima and maxima. The rank value is organized from the best to the worst feasible configuration obtained from permutations Eq. (16) between the total number of switches and required sectionalizing switches, which is 13 in this case. Only the best 2,408 configurations were saved for memory resource reasons.

Table 2 Modified IEEE 13-node test feeder evaluations

	BBA	MBBA	BPSO	MBPSO	GA	MGA	Seed
Min (kW)	73,61	73,61	73,61	73,61	73,61	73,61	73,61
Tie-sw Min	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692
Max (kW)	73,61	73,61	73,61	73,61	73,61	73,61	73,61
Tie-sw Max	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692	646-684; 671-692
Mean (kW)	73,61	73,61	73,61	73,61	73,61	73,61	73,61
Std dev	0	0	0	0	0	0	0
Min rank	1	1	1	1	1	1	1
Max rank	1	1	1	1	1	1	1
Num_best	5	5	5	5	5	5	5
Aver. Time (s)	5	5,13	4,98	5,03	5,02	5,10	5,25

Table 3 Modified IEEE 123-node test feeder evaluations

	BBA	MBBA	BPSO	MBPSO	GA	MGA	Seed
Min (kW)	204,16	204,16	204,16	207,01	207,92	204,89	208,10
Tie-sw Min	1;2;3;4;5; 8;9;10;11;12; 13;14;17.	1;2;3;4;5; 8:9;10;11;12; 13;14;17.	1;2;3;4;5; 8;9;10;11;12; 13;14;17.	1;2;3;4;5; 7;8;9;10;11; 12;13;14.	1;2;3;4;5; 8;9;10;11;12; 16;17;19.	1;2;3;4;5; 8;9;10;11;12; 14;17;19.	1;2;3;4;5; 7;8;10;11;12; 13;14;19.
Iter-Min	27	16	23	24	7	22	1
Max (kW)	273,28	210,21	237,45	227,93	265,98	254,06	233,43
Tie-sw Max	1;4;8;10;11; 12;13;15;16;17; 18;19;20.	1;2;3;4;5; 8;9;10;11;12; 14;17;21.	1;3;4;6;8; 9;10;11;12;14; 17;18;21.	1;3;4;5;6; 7;8;9;10;11; 12;18;21.	1;2;4;5;6; 7;8;9;10;12; 18;19;21.	1;2;3;4;8; 9;10;11;13;14; 15;17;20.	2;4;5;6;8; 10;11;12;13;15; 17;18;19.
Iter-Max	19	14	17	20	23	1	1
Mean (kW)	218,34	206,78	212,60	215,89	223,85	219,86	219,68
Std dev	15,68	1,76	6,79	4,90	14,60	12,32	5,99
Min rank	1	1	1	7	10	2	11
Max rank	>2408	26	1150	616	>2408	2397	923
Num_best	1	4	1	0	0	0	0
Aver Time (s)	30,89	1747,67	38,88	85,23	44,73	169,94	148,58

Each row of Tab. 3 is analysed for comparison purposes.

In row 1, only BBA, MBBA, and MPSO reached the best configuration that brings the minimal real power loss of 204.16 kW; however, MGA is close, with its real power loss of 204,89 kW, which is only 0,36% greater. The seed only achieved the worst scenario with 208.10 kW in power loss, which is 1.93% above. In row 2, all the tie switches that must be closed for the minimal found solution are presented.

Row 3 presents the iteration in which the best solution was found for each algorithm. MBBA encountered the optimal solution in 16 of 50 permissible, whereas other techniques did the same in fewer iterations.

The worst encountered configuration is in row 4. The MBBA was the best in this aspect because its worst solution was 210,21 kW; i.e., approximately 6 kW, or 2,96% higher than the best solution. The closer solution is obtained from MBPSO, which is 227,93 kW and 8,43% greater than that obtained by MBBA and 11,64% above the optimal solution. The worst scenario is registered from BBA with real power loss of 273,28 kW and 30% above the worst solution achieved by MBBA. In this aspect, MBBA outperforms the other algorithms. Row 5 presents the tie switches that are closed for worst configurations obtained by each algorithm.

Row 6 presents the iteration in which the worst solution was found for each algorithm. MBBA encountered this in 14 of 50 permissible iterations. The only two algorithms that reached the worst solution in fewer iterations were MGA and Seed, but with worse solutions than the one obtained by MBBA.

Row 7 presents the means of all 30 runs, the 30-size population, and the 50 iterations, which is a total of 45000 trials per algorithm. Again, the proposed MBBA accomplished the best figure, with a mean of 206,78 kW; i.e., only 2,62 kW and 1,28% higher than the best solution. The second best is BPSO with 212,60 kW and 4,13% higher than the optimal. In third place is MBPSO with 215,89 kW and 5,74% higher than the optimal and in fourth place is BBA with 218,34 kW and 6,95% higher than the optimal. In fifth place is Seed with 219,68 kW and 7,60% higher than the optimal. MGA takes sixth place with 219,86 kW and 7,69% higher than the optimal, and the worst performer is GA, with 223,85 kW and 9,64% higher than the optimal.

Row 8 presents the standard deviations calculated from all solutions obtained per algorithm. The MBBA attains the lowest standard deviation with 1.76. This brings

a range of 205,02–208,54 kW, or from 0,42% to 2,14% higher the optimal solution, which is a small limit when considering a 5% threshold. The remaining algorithms reached standard deviations higher than 70% above the MBBA figure. This is a crucial topic, as the MBBA overcomes the rest of techniques far enough.

Rows 9 and 10 contain the best and worst obtained solution rankings. From the evaluation of all feasible combinations, an ascending rank order is organized. The best and worst solutions per algorithm are compared with the rank for determining how far they are from the optimal solution. Again, the MBBA is the best algorithm because the best solution is ranked in the first position and the worst solution is ranked in the 26th position. This is the lowest range; therefore, this technique guarantees topologies very close to the optimal solution. The rest of algorithms present good best solutions, but all the worst solutions are ranked in positions higher than 600.

Row 11 shows how many times the optimal solution is reached by all algorithms. BBA and BPSO achieved it once, whereas the MBBA achieved it four times and the rest of algorithms did not reach the optimal solution. The MBBA overcomes again the other algorithms in this issue.

Row 12 shows the average spent CPU time for reaching solutions per algorithm. All algorithms are capable of performing 30-size population and 50 iterations in less than 150 seconds (2,5 minutes). The only exclusion is traced with the MBBA as it spends about 1747,67 seconds in every run process, or 29,13 minutes (slightly less than a half hour). Besides, this appears to be a very high drawback, it is necessary to consider that load profiles are normally reported in the system on an hourly basis. In addition, reconfiguration operation cannot be done every hour because this could reduce the reliability of the system, causes non-desirable oscillations in the system, and users could experience many disturbances or even transient blackouts [18].

From the results, it can be observed that MBBA is a promising technique for reconfiguration to be considered by system operators, as the results were obtained inside a range of near-optimal configurations for power loss minimization. However, the reconfiguration process not necessarily has to be done in real-time operation, but in an hourly-basis framework from available records, since the load holds an overall shape all the year.

7 CONCLUSIONS

A modified binary bat algorithm was presented considering an improved seed population search. This technique was tested in two modified IEEE test systems (13- and 123-node test feeders) to determine how it responds in a small and a large system. The results were also compared with well-known techniques, namely, PSO and GA, and the proposed Improved seed population search, showing the following very good solutions:

- Where evaluated in a small system as the modified IEEE 13-node test feeder, there are no differences among any of the techniques cited above because the results are very close in all the analysed variables.
- Where evaluated in a large system as the modified IEEE 123-node test feeder, MBBA guarantees that the worst solution is inside a range of near-optimal solutions, which is below 5% of deviation from the optimal solution, whereas the other algorithms could not achieve similar results.

Considering the above, MBBA is a promising option to perform distribution network reconfiguration; however better computational machines must be used to obtain faster results. For future work, the execution time must be reduced to deal with real-time simulations.

8 REFERENCES

[1] Nguyen, T. T., Nguyen, T. T., Truong, A. V., Nguyen, Q. T., & Phung, T. A. (2017). Multi-objective electric distribution network reconfiguration solution using runner-root algorithm. *Applied Soft Computing*, 52, 93–108. https://doi.org/10.1016/j.asoc.2016.12.018

[2] Abdelaziz, A. Y., Osama, R. A., & Elkhodary, S. M. (2013). Distribution Systems Reconfiguration Using Ant Colony Optimization and Harmony Search Algorithms. *Electric Power Components and Systems, 41*(5), 537–554. https://doi.org/10.1080/15325008.2012.755232

[3] Herazo, E., Quintero, M., Candelo, J., Soto, J., & Guerrero, J. (2015). Optimal power distribution network reconfiguration using Cuckoo Search. In *The 4th International Conference on Electric Power and Energy Conversion Systems (EPECS)* (pp. 1–6). IEEE. https://doi.org/10.1109/EPECS.2015.7368548

[4] Farahani, V., Vahidi, B., & Abyaneh, H. A. (2012). Reconfiguration and Capacitor Placement Simultaneously for Energy Loss Reduction Based on an Improved Reconfiguration Method. *IEEE Transactions on Power Systems, 27*(2), 587–595. https://doi.org/10.1109/TPWRS.2011.2167688

[5] Garcia-Martinez, S. & Espinosa-Juarez, E. (2011). Reconfiguration of power systems by applying Tabu search to minimize voltage sag indexes. In *2011 North American Power Symposium* (pp. 1–6). IEEE. https://doi.org/10.1109/NAPS.2011.6025100

[6] García-Martínez, S. & Espinosa-Juárez, E. (2013). Optimal Reconfiguration of Electrical Networks by Applying Tabu Search to Decrease Voltage Sag Indices. *Electric Power Components and Systems, 41*(10), 943–959. https://doi.org/10.1080/15325008.2013.801053

[7] Glover, F. (1989). Tabu Search—Part I. *ORSA Journal on Computing, 1*(3), 190–206. https://doi.org/10.1287/ijoc.2.1.4

[8] Graditi, G., Di Silvestre, M. L., La Cascia, D., Riva Sanseverino, E., & Zizzo, G. (2016). On multi-objective optimal reconfiguration of MV networks in presence of different grounding. *Journal of Ambient Intelligence and Humanized Computing, 7*(1), 97–105. https://doi.org/10.1007/s12652-015-0304-9

[9] Gu, C., Ji, J., & Liu, L. (2014). Research of immune algorithms for reconfiguration of distribution network with distributed generations. In *The 26th Chinese Control and Decision Conference (2014 CCDC)* (pp. 2156–2160).IEEE. https://doi.org/10.1109/CCDC.2014.6852524

[10] Gupta, N., Swarnkar, A., & Niazi, K. R. (2014). Distribution network reconfiguration for power quality and reliability improvement using Genetic Algorithms. *International Journal of Electrical Power & Energy Systems*, 54, 664–671. https://doi.org/10.1016/j.ijepes.2013.08.016

[11] Abazari, S. & Soudejani, M. H. (2015). A new technique for efficient reconfiguration of distribution networks. *Scientia Iranica, 22*(6), 2516–2526.

[12] Abdelaziz, A. Y., Mohamed, F. M., Mekhamer, S. F., & Badr, M. A. L. (2010). Distribution system reconfiguration using a modified Tabu Search algorithm. *Electric Power Systems Research, 80*(8), 943–953.

https://doi.org/10.1016/j.epsr.2010.01.001
[13] Abdelaziz, A. Y., Osama, R. A., & El-Khodary, S. M. (2012). Reconfiguration of distribution systems for loss reduction using the hyper-cube ant colony optimisation algorithm. *IET Generation, Transmission & Distribution, 6*(2), 176. https://doi.org/10.1049/iet-gtd.2011.0281
[14] Asrari, A., Lotfifard, S., & Ansari, M. (2016). Reconfiguration of Smart Distribution Systems with Time Varying Loads Using Parallel Computing. *IEEE Transactions on Smart Grid*, 1–11. https://doi.org/10.1109/TSG.2016.2530713
[15] Liu, L. H., Wang, Y., Yao, S. J., Ma, L. Y., & Yang, J. (2012). Distribution Network Reconfiguration with Distributed Generation Based on Cloud Genetic Algorithm. *Advanced Materials Research*, 529, 306–310. https://doi.org/10.4028/www.scientific.net/AMR.529.306
[16] Quintero-Duran, M., Candelo, J. E., & Sousa, V. (2017). Recent Trends of the Most Used Metaheuristic Techniques for Distribution Network Reconfiguration. *Journal of Engineering Science and Technology Review, 10*(5), 159–173. https://doi.org/10.25103/jestr.105.20
[17] Mirjalili, S., Mirjalili, S. M., & Yang, X.-S. (2014). Binary bat algorithm. *Neural Computing and Applications, 25*(3–4), 663–681. https://doi.org/10.1007/s00521-013-1525-5
[18] Amanulla, B., Chakrabarti, S., & Singh, S. N. (2012). Reconfiguration of Power Distribution Systems Considering Reliability and Power Loss. *IEEE Transactions on Power Delivery, 27*(2), 918–926. https://doi.org/10.1109/TPWRD.2011.2179950
[19] Alonso, F. R., Oliveira, D. Q., & Zambroni de Souza, A. C. (2015). Artificial Immune Systems Optimization Approach for Multiobjective Distribution System Reconfiguration. *IEEE Transactions on Power Systems, 30*(2), 840–847. https://doi.org/10.1109/TPWRS.2014.2330628
[20] Yang, X.-S. (2010). A New Metaheuristic Bat-Inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2011)* (Vol. 284, pp. 65–74). https://doi.org/10.1007/978-3-642-12538-6_6
[21] Peres, W., Silva Júnior, I. C., & Passos Filho, J. A. (2018). Gradient based hybrid metaheuristics for robust tuning of power system stabilizers. *International Journal of Electrical Power & Energy Systems*, 95, 47–72. https://doi.org/10.1016/j.ijepes.2017.08.014
[22] Niu, T., Wang, J., Zhang, K., & Du, P. (2018). Multi-step-ahead wind speed forecasting based on optimal feature selection and a modified bat algorithm with the cognition strategy. *Renewable Energy*, 118, 213–229. https://doi.org/10.1016/j.renene.2017.10.075
[23] Montgomery, D. C. (2006). Design and Analysis of Experiments. *Technometrics* (Vol. 48). https://doi.org/10.1198/tech.2006.s372
[24] Quintero-Duran, M., Candelo-Becerra, J. E., & Soto-Ortiz, J. D. (2019). A Modified Backward/Forward Sweep-based Method for Reconfiguration of Unbalanced Distribution Networks. *International Journal of Electrical and Computer Engineering, 9*(1), 85-101. https://doi.org/10.11591/ijece.v9i1.pp.85-101
[25] Kennedy, J., & Eberhart, R. C. (1997). Discrete binary version of the particle swarm algorithm. In Proceedings of the *IEEE International Conference on Systems, Man and Cybernetics* (Vol. 5, pp. 4104–4108). Retrieved from https://www.scopus.com/inward/record.uri?eid=2-s2.0-0031352450&partnerID=40&md5=a713161d139c8afba89f6b67c67696c7
[26] Nara, K., Shiose, A., Kitagawa, M., & Ishihara, T. (1992). Implementation of genetic algorithm for distribution systems loss minimum re-configuration. *IEEE Transactions on Power Systems, 7*(3), 1044–1051. https://doi.org/10.1109/59.207317

Contact information:

Michell J. QUINTERO-DURAN, MSc, PhD Student, (Corresponding author)
Universidad Nacional de Colombia, Sede Medellín
Carrera 80 No. 65-223, Medellín, Colombia
E-mail: mjquinterod@unal.edu.co

John E. CANDELO-BECERRA, PhD, Associate Professor
Universidad Nacional de Colombia, Sede Medellín
Carrera 80 No. 65-223, Medellín, Colombia
E-mail: jecandelob@unal.edu.co

Katherine CABANA-JIMENEZ, MSc, Assistant Professor
Universidad de la Costa
Calle 58 No. 55-66, Barranquilla, Colombia
E-mail: kcabana@cuc.edu.co