Contents lists available at ScienceDirect

Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Metaheuristics approaches to solve combinatorial optimization problems in distribution power systems. An application to Phase Balancing in low voltage three-phase networks

Gustavo Schweickardt [a,1], Juan Manuel Gimenez Alvarez [b,*], Carlos Casanova [a,1]

[a] CONICET, Universidad Tecnológica Nacional, Concepción del Uruguay, Ing. Pereira 676, 3260 Concepción del Uruguay, Argentina
[b] CONICET, Universidad Nacional de San Juan, Facultad de Ingeniería, Av. Libertador Gral. San Martín 1109 Oeste, Desamparados, San Juan, Argentina

ARTICLE INFO

Article history:
Received 23 February 2014
Received in revised form 16 September 2015
Accepted 19 September 2015

Keywords:
Metaheuristic algorithm
Swarm Intelligence
Fuzzy sets
Electric distribution
Phase Balancing

ABSTRACT

Metaheuristics algorithms are widely recognized as one of most practical approaches for combinatorial optimization problems. One the most interesting areas of application are the power systems. In particular, Distribution Systems planning and operation. This paper presents some metaheuristics approaches to solve a typical combinatorial optimization problem: the Phase Balancing in Low Voltage Electric Distribution Systems. A model supported in Linear Integer-Mixed Programming is presented, to observe and discussing its limitations. From this, is introduced a new metaheuristic, called Fuzzy Evolutionary Particle Swarm Optimization, based in the Swarm Intelligence Principles and Evolution Strategies, which is extended to fuzzy domain to modeling a multi-objective optimization, by mean of a fuzzy fitness function. A simulation on a real system is presented, and advantages of this approach respect to the Classical Simulated Annealing and Particle Swarm metaheuristics, selected between the most representatives, are evidenced.

Introduction

Metaheuristics algorithms are widely recognized as one of more practical and success approaches to solve combinatorial problems [1–3]. However, the original formulations have been oriented to mono-objective optimizations. Many proposal of extensions to multi-objective domain have been established, but each formulation has showed particular advantages and limitations, in general, over certain kind of problems. Such is the case of Phase Balancing in Low Voltage Electric Distribution Systems (LVEDS), when a classic programming approach is not addressed to solve it. The balance is referred to the loads in the feeders of a LV network in an EDS. The classic approach, has demonstrated significant limitations, as will be discussed. For this reason, a metaheuristic approach is an alternative that may produce very good results.

There are a significant number of metaheuristics, such as Tabu Search (TS) [4], Simulated Annealing (SA) [5], Ant Colony Optimization (ACO) [6] and Particle Swarm Optimization (PSO) [7,8], that have been proposed to solve specific combinatorial problems, with a great success. However, there are two aspects that remain in discussion: (a) How model a best multi-objective metaheuristic and (b) How design a self-adaptive metaheuristic, with the less number of external parameters as possible (if were possible, none). This paper presents a new multi-objective metaheuristic, called Fuzzy Evolutionary Particle Swarm Optimization (FEPSO), which results can to drive towards a general model, capable to overcome the two aspects mentioned. The problem of Phase Balancing, under a multi-objective formulation, is, then, considered to simulations with the aim of comparing the performance of FEPSO with respect to a classical metaheuristic. Among the most representative metaheuristics was selected Simulated Annealing (SA), which will be extended to the fuzzy domain to define a multi-objective version of SA, that will be called FSA. The selection of SA is not arbitrary, but respond to fact of to separate two strategies very different that can be observed in the metaheuristics algorithms: in the first, that supported SA, the evolution of the solution algorithm, is simulated using probabilistic sampling techniques, supported by successive generation of states of energy, corresponding to solutions of the combinatorial optimization. In the second, that supported FEPSO, such evolution is based in the denominated Swarm Principles. These concepts will be presented in section ‘The metaheuristics simulated annealing and Particle Swarm Optimization’.

* Corresponding author. Tel./fax: +54 264 4211700.
E-mail addresses: gustavoschweickardt@conicet.gov.ar (G. Schweickardt), jgimenez@unsj.edu.ar (J.M.G. Alvarez), casanovac@utn.frcu.edu.ar (C. Casanova).
[1] Tel./fax: +54 3442 423898.

http://dx.doi.org/10.1016/j.ijepes.2015.09.023
0142-0615/© 2015 Elsevier Ltd. All rights reserved.

The work is organized as follows: in section 'The Phase Balancing problem in Low Voltage Feeder System', the problem of Phase Balancing is presented. It discussing the no desired effects of a high unbalance degree on the LV feeders system. A model supported in Linear Integer-Mixed Programming is presented, to observe and discussing its limitations. Then, a multi-objective formulation for the problem, according with the purpose of simulations by metaheuristics algorithm, is presented. In section 'The metaheuristics simulated annealing and Particle Swarm Optimization', the essential of the metaheuristics SA and PSO, are described. In section 'The metaheuristic Evolutionary Particle Swarm Optimization (EPSO)', the first approach towards the contribution of this paper, the metaheuristic Evolutionary Particle Swarm Optimization (EPSO), is presented. Section 'Static fuzzy decision and fuzzy fitness function' is focused in the formulation of a fuzzy fitness function, supported in the static decision-making in fuzzy environment principle, formulated by Bellman and Zadeh [9]. From this, in section 'Simulation on the real LV feeder system', the multi-objective metaheuristics FSA and FEPSO, in the specific context of Phase Balancing problem considered in section 'The Phase Balancing problem in Low Voltage Feeder System', are introduced. Finally, in section 'Simulation on the real LV feeder system', a simulation of the two fuzzy metaheuristics on the same real LV feeders system is presented, and its results are compared and discussed. In section 'Conclusions', the most important conclusions of this work, are presented.

The Phase Balancing problem in Low Voltage Feeder System

The most LV networks of an EDS are three-phase systems. Feeders loads in a LV network, for low incomes residential areas, are commonly single-phase. Original feeders system design, depends on accuracy of the given load data, there always will be certain unbalance degree and it must be as low as possible.

There are, in general, two options for Phase Balancing: (a) feeders reconfiguration at the system level and (b) phase swapping at the feeder level. The option (b), phase swapping, is a direct and effective way to balance a feeder in terms of phases, and this method will be considered in this work. For the purpose of this paper, LV feeders system will have only single-phase loads. A formulation to the general problem of Phase Balancing, in this context, considering, without loss of generality, only one feeder (principal), can be expressed as follows:

$$Min\left\{Loss_T; I(\Delta u); \left|I^{[o]}\right|_f\right\} \tag{1}$$

Subject to:

$$\left|I^{[R]}\right|_f \leqslant I_{Max} \tag{2}$$

$$\left|I^{[S]}\right|_f \leqslant I_{Max} \tag{3}$$

$$\left|I^{[T]}\right|_f \leqslant I_{Max} \tag{4}$$

where the subindex f, refers the output of substation connected to the principal feeder of the system; $Loss_T$ indicates the total active power loss of system and $I(\Delta u)$ is an index that depends of voltage drops. $\left|I^{[o]}\right|_f$ is the homopolar component of the unbalance intensities system, that satisfy the equation:

$$\left|I^{[R]}\right|_f + \left|I^{[S]}\right|_f + \left|I^{[T]}\right|_f = 3 \times \left|I^{[0]}\right|_f \tag{5}$$

If the system is balanced, then $\left|I^{[0]}\right|_f = 0$.

The subindex $[R]$, $[S]$ and $[T]$, represent each phase of system. In addition, three constraints are imposed: the intensities in each phase at the output, must be less than the phase line capacity, I_{Max}, Eqs. (2)–(4). The problem can be seen as a set of swapping single phase loads or a load assignment to lines. For example, a single phase load can only be assigned to either phase $[R]$, $[S]$ or $[T]$ (see Fig. 1). This assignment should be executed until the objectives (1) are satisfied. This problem is clearly combinatorial: if there are 3 phases and n loads that can be swapping, then the number of states of search space for the solution, is 3^n.

An approach from mathematical programming supported in linear Mixed-Integer Programming

The MIP (Mixed-Integer Programming) model, was presented in [10], and its formulation, to the context of this work, can be stated as follows:

$$Min\left\{\sum_j p_j \times U_j\right\}$$

Subject to :

$$U_j = Max\left\{\sum_j \left|I_f^{[R]} - I_f^{[T]}\right|; \left|I_f^{[S]} - I_f^{[T]}\right|; \left|I_f^{[R]} - I_f^{[S]}\right|\right\} \tag{6}$$

$$I_j^{[\Phi]} = \sum_k I_k^{[\Phi]} + \sum_w \delta_i^{[\Phi]}{}_w \times I_i^{[\Phi]}{}_w \tag{7}$$

$$\sum_w \delta_i^{[\Phi]}{}_w = 1; \quad \forall[\Phi] \in \{R, S, T\} \tag{8}$$

$$\sum_w \delta_i^{[\Phi]}{}_w = 1; \quad \forall[w] \in [1 \dots nL] \tag{9}$$

$$I_j^{[\Phi]} \leqslant I_{Max\ j} \tag{10}$$

$$\delta_i^{[\Phi]}{}_w \in \{0, 1\}; \quad \forall[i, \Phi] \tag{11}$$

$$\sum_j p_j = 1; \quad \forall[j] \in [1 \dots nB] \tag{12}$$

From Eq. (6), in the context of this work, it is intend to assign the single-phase loads to the conductors (phases) such that in each branch j, the maximum difference in magnitude between the intensities (taken in pairs of phases), be a minimum. Then U_j becomes in a measure of unbalance degree per branch. To complete the objective, a convex set of weights, indicated as p_j, is externally fixed, by the decision-maker, and, this way, the most simple multi-objective linear programming formulation is proposed. p_j is a subjective value that reflex the importance of unbalance degree, U_j, in the branch j of feeder system. The remaining symbols have the following meaning: Φ is the phase in $\{R, S, T\}$; $\delta_i^{[\Phi]}{}_w$ is a decision variable, (binary) in $\{0, 1\}$, for w-th load connected to phase Φ at node i; nL is the number of loads; nB is the number of branches; I_{Maxj} is the phase line capacity of branch j. In this formulation, Eq. (7) represents the Kirchhoff Law of current at the node i; Eqs. (8), (9) and (11) ensure that each load has assigned (or is connected to) only one phase; the Eq. (5) is the line capacity constraint, and the Eq. (12) is the convex set of branch-weights constraint.

This model has three major limitations: (a) The problem is not linear and its linearization is valid only if the load characteristics are constant-current. In the most real systems, this condition is not observable, (b) the model require the subjective weights, p_j,

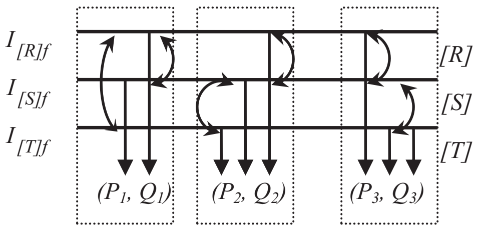

Fig. 1. Single phase loads and balance by phase swapping in some loads.

which is a poor approach to search a global minimum in the objectives functions formulated in (1). Is probably that the "optimal solution" reached from this MIP approach, result very far from real optimum (minimum unbalance degree) and (c) the model ignores the voltage drops, which is an aspect that must be had into account to improve the quality of service (under regulation control).

The metaheuristics simulated annealing and Particle Swarm Optimization

Simulated Annealing (SA)

The concept of Simulated Annealing in combinatorial optimization was introduced by Kirkpatrick [11]. SA appears like a flexible metaheuristic that is an adequate tool to solve a great number of combinatorial optimization problems. It is motivated by an analogy to annealing in solids. The Metropolis algorithm simulated the cooling of material in a heat bath. This is the process know as annealing. Consists on two steps: (a) the temperature is raised to a state of maximum energy and (b) the temperature is slowly lowered until a minimum energy state, equivalent to thermal equilibrium, is reached. The structural properties of system, depends on the rate of cooling. If it is cooled slowly enough, large crystals will be formed. However, if the system is cooled quickly, the crystals will contain imperfections. Metropolis's algorithm simulated the material as a system of particles.

In order to more clearly explain the SA metaheuristic, is possible present an analogy between a physical system, with a large number of particles, and a combinatorial problem. This analogy can be stated as follows:

The solutions of combinatorial problem are equivalent to the physical system states.
The attributes of solutions are equivalent to the energy of different states.
The control parameter in the combinatorial problem is equivalent to the temperature of physical system.

The evolution of the solution algorithm is simulated using probabilistic sampling techniques, supported by successive generation of states. This process begins with an initial state, i, evaluated by an energy function, $E(i)$. After generating and analyzing a second state, $j, E(j)$, it is performed an acceptation test. The acceptance of this new solution, j, depends on a probability computed by:

$$p(accept\ j) = \begin{cases} 1; & \text{if } E(j) \leqslant E(i) \\ e^{\frac{E(j)-E(i)}{c}}; & \text{if } E(j) \geqslant E(i) \end{cases} \tag{13}$$

where c is a positive real number, $c = k_B \times T$; k_B is a constant (called Boltzmann constant, in Metropolis's Algorithm) and T is a temperature of system.

A procedure is defined, in pseudo-code, as follows:

```
Minimize the function f(i) for i on S (Search Space)
Begin Procedure SA
1. set starting point $i = i_0$
2. set starting temperature $\boldsymbol{T} = \boldsymbol{T}_0$ and cooling rate: $0 < \alpha < 1$;
3. set NT (number of trials per temperature);
4. while stopping condition is not satisfied do
5.   for $k \leftarrow 1$ to NT do
6.     generate trial point j from $S_i$ using q(i, j);
7.     accept j with probability p(accept j) (Eq. (13));
8.   end for
9.   reduce temperature by $T \leftarrow T \times \alpha$;
10. end while
End Procedure SA
```

S_i is a *Neighborhood* of solution i: is a set of discrete points j satisfying $j \in S_i \Longleftrightarrow i \in S_j$. The *generation function* of S_i is $q(i,j)$ defined externally.

Swarm Intelligence (SI)

Swarm Intelligence refers a type of artificial intelligence, based on the collective behavior of self-organized systems. The expression was introduced by Gerado Beni, in 1989 [8]. There are five Swarm Intelligence Principles that have been recognized as the fundamental of optimization strategies: (a) Proximity: The swarm should be able to perform simple calculation of space and time. (b) Quality: promotes the swarm ability to respond to factors that improve the fitness of individuals in the space of solutions. (c) Diversity of Response: promotes the possibility of different responses of individuals, to the same stimuli. (d) Stability: The swarm should be stable in the absence of stimuli that induce improvements in the solutions reached. (e) Adaptation: It is a complementary aspect to the Stability, because the swarm should adapt to any change that causes an improvement in the fitness of its solutions.

Classical PSO

The Swarm Intelligence is adopted by Particle Swarm Optimization metaheuristic, (PSO) [7,8] that rely on a following concepts: mimicking zoological behavior, imitating the collective or social behavior of animals swarms, flocks or schools, a set of particles evolves in the search space motivated by three factors, called *habit*, *memory* and *cooperation*. The first factor impels a particle to follow a path along its previous movement direction. It is frequently called the *inertia* factor. The second factor influences the particle to come back on its steps (i.e., to tend to go back to the best position it found during its life). The third factor (related with to information exchange), induces the particle to move closer to the best point in the space, found by the collective of all particles in its family group. Analogy between Particle Swarm and Combinatorial Problem is easy to see, establishing a correspondence between the position of particle and a solution in the search space.

In the Classical PSO, one must have, at a given iteration, a set of solutions or alternatives called "particles".

From one iteration to the following, each particle, $\boldsymbol{i}$, moves according to a rule that depends on the three factors described (habit, memory and cooperation). In addition, each particle of swarm keep the record of the best point found in its past life, b_i, and the record of the current global best point found by the swarm, b_G. Then, the PSO Movement Rule, sates that (X, V , b_i and b_G are vectors):

$$X_i^{New} = X_i + V_i^{New} \times \Delta t \tag{14}$$

and if Δt is adopted as 1 (t is a discrete variable that indicates the iteration number, and Δt indicates the iterative incremental step):

$$X_i^{New} = X_i + V_i^{New} \tag{15}$$

where V_i is called the velocity of particle i, and is defined by the equation, referred as Canonical PSO:

$$V_i^{New} = V_i + rnd_1 \times w_M \times (b_i - X_i) + rnd_2 \times w_C \times (b_G - X_i) \tag{16}$$

For this version, a typical selection of constants, are: $w_M = w_C = 2$.

An small alteration, and most common to observe, is the form called with Inertial Weight:

$$V_i^{New} = w_I \times V_i + rnd_1 \times w_M \times (b_i - X_i) + rnd_2 \times w_C \times (b_G - X_i) \tag{17}$$

And for this version, a typical selections of constants, are: $0.4 \leqslant w_I \leqslant 0.8$ and $w_M = w_C = 2$.

The dimension of vectors is the number of decision variables. The first term of (17), represents the inertia or habit of the particle: keeps moving in the direction it had previously moved. The second term, represents the memory: the particle is attracted to the best point in its trajectory-past life, b_i. The third term represents the cooperation or information exchange: the particle is attracted to the best point found by all particles, b_G. The parameters w_I, w_M and w_C are weights fixed in the beginning of process; rnd_1 and rnd_2 are random numbers sampled from a uniform distribution $U(0,1)$. The movement of a particle is represented in Fig. 2. The movement rule is applied iteratively, until either of two conditions occurs: there are no changes in the global best, b_G or the number of iterations reaches a limit. It can be seen that the PSO meets, from the equations of Movement (15) and Velocity (16), the five Swarm Intelligence Principles.

Modifications on Classical PSO

There are some variations on Classical or Canonical form of PSO. The most important are discussed briefly in this section.

(A) PSO with Inertia Function: in this form, the velocity operator is modified introducing a *function decreasing with the progress of iterations*, $\delta(t)$, that reduce, progressively, the importance of inertia term. Is expressed as follows:

$$V_i^{New} = \delta(t) \times w_I \times V_i + rnd_1 \times w_M \times (b_i - X_i) + rnd_2 \times w_C \times (b_G - X_i) \quad (18)$$

A typical linear function can be formulated as follows:

$$\delta(t) = w_{Max} - \frac{(w_{Max} - w_{Min})}{nT} \times t \quad (19)$$

where t is the actual iteration and nT is de maximum number de iterations externally imposed; w_{Max} and w_{Min} are two inertial weights whose values typically are 0.9 and 0.4 respectively. The external definition of decreasing function, $\delta(t)$, require some caution, because is intuitive that if inertia term is eliminated at an early stage of the process, the procedure risks to be trapped at some local minimum.

(B) PSO with Constriction Factor:

$$V_i^{New} = \chi \times \{V_i + \varphi_M \times rnd_1 \times (b_i - X_i) + \varphi_C \times rnd_2 \times (b_G - X_i)\} \quad (19)$$

where – is the *Constriction Factor*. Is obtained from:

$$\chi = \frac{2 \times \kappa}{\left|2 - \varphi - \sqrt{\varphi^2 - 4 \times \varphi}\right|} \quad (20)$$

$$\text{with } \varphi_M + \varphi_C = \varphi; \quad \varphi > 4 \text{ and } 0 < \kappa \leqslant 1 \quad (21)$$

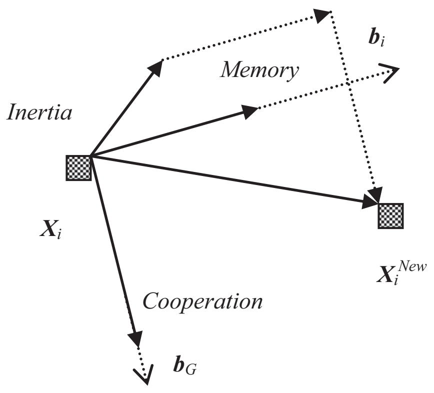

Fig. 2. Movement rule of PSO in two dimensions.

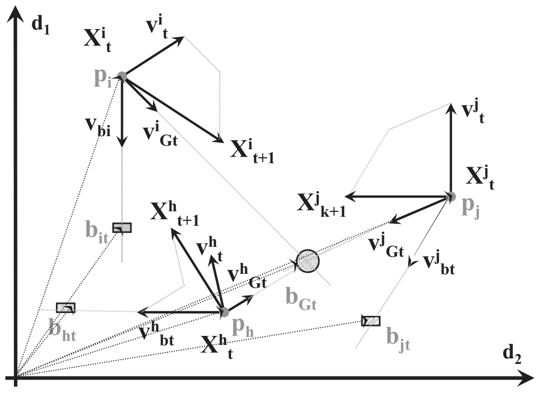

Fig. 3. Three particles of PSO moving in two dimensions.

Two typical configurations for this set of parameters, are: (a) $\varphi_M = \varphi_C = 2.05\varphi_M$, $\varphi = 4.1$, $\kappa = 1$ and $\chi = 0.729$; (b) $\varphi_M = \varphi_C = 2.8$, $\varphi = 4.1$, $\kappa = 1$ and $\chi = 0.729$.

In Fig. 3, it can observe the movements of three particles of PSO, p_i, p_j and p_h, with all components, in a two dimensional space. Notice the influence of the *individual best* of each particle, $b_{i,j,h}$, and the *swarm global best*, b_G, in the components of respective velocities. The transition refers the interval $\Delta t = (t+1) - t$, *from one iteration to the following.*

The strategy of PSO, as an algorithm, will be described in section 'Simulation on the real LV feeder system', where the multi-objective metaheuristic FEPSO, proposed in this work, is explained. This is because FEPSO is an extension of PSO and EPSO.

The metaheuristic Evolutionary Particle Swarm Optimization (EPSO)

There are some striking points in PSO, such as: (a) it depends of a number of parameters defined externally by the user, and most certainly with values that are problem-dependent. This is certainly true for the definition of weights w_I, w_M and w_C: a delicate work of tuning is necessary in the most of practical problems; (b) the external definition of decreasing function, $\delta(t)$, require some caution, because is intuitive that if inertia term is eliminated at an early stage of the process, the procedure risks to be trapped at some local minimum; (c) last, the random factors rnd_1 and rnd_2, while introducing stochastic flavor, only have a heuristics basis and are not sensitive to evolution of the process; and (d) however the modifications introducing on the Canonical form of PSO, with the purpose of confer most stability and adaptability to this metaheuristic, nothing guarantees that the optimization to be trapped in some local optimum.

The introducing of EPSO/self-adaptive metaheuristic, be intend overcome these limitations.

PSO can be observed as a proto-evolutionary process, because exists a mechanism to generate new individuals from a previous set (the movement rule). It is not a explicit selection mechanism in the darwinian sense. However the algorithm exhibits a positive progress rate (evolution) because the movement rule induces such property implicitly. The idea behind EPSO is to grant a PSO scheme with an explicit selection procedure and with self-adapting properties for its parameters. The self-adaptive Evolution Strategies (σA-ES) model, although there are many variants, may be represented by the following procedure:

(a) Each individual of generation is duplicated. (b) The strategic parameters of each individual are undergo mutation. (c) The object parameter of each individual are mutated under a procedure commanded by its strategic parameters (this generates new individuals). (d) A number of individuals are undergo recombination (this also generates new individuals). (e) From the set of parents and sons (the original and the new individuals), the best fit are selected to form a new generation.

If both strategies (σSA-ES and PSO) are combined, it is possible to create a such scheme (self-adaptive/evolutionary PSO). At a given iteration, consider a set of solutions that can will keep calling "particles". Then, the general scheme for EPSO is the following:

Replication: each particle is replicated *r* times; **Mutation**: each particle has its weights mutated; **Reproduction**: each mutated particle generates an offspring according to the particle movement rule; **Evaluation**: each offspring has its fitness evaluated; **Selection**: by stochastic tournament, the best particles survive to form a new generation. Then, the Movement Rule of EPSO is not changed respect PSO, and is valid, for example, the Eq. (17). But the new EPSO *velocity operator*, is expressed by:

$$V_i^{New} = w_{Ii}^* \times V_i + w_{Mi}^* \times (b_i - X_i) + w_{Ci}^* \times (b_G^* - X_i) \tag{22}$$

The Movement Rule of EPSO, keeps its terms of *inertia*, *memory* and *cooperation*. However, the symbol $*$ indicates that *the parameters will undergo mutation*:

$$w_{Ii}^* = w_{Ii} + \sigma \times N(0,1) \tag{23}$$

$$w_{Mi}^* = w_{Mi} + \sigma \times N(0,1) \tag{24}$$

$$w_{Ci}^* = w_{Ci} + \sigma \times N(0,1) \tag{25}$$

$$b_G^* = b_G + \sigma' \times N(0,1) \tag{26}$$

where $N(0,1)$ is a random variable with *gaussian distribution, mean 0 and variance 1*; σ and σ' are learning parameters (either fixed or treated also as strategic parameters and therefore also subject to mutation). The global best b_G is randomly perturbed too. In Fig. 4, a new Movement Rule of EPSO, with the perturbed global best, is represented. Notice that *the vector associated with the cooperation factor does not point to the global optimum,* b_G, *but to a mutated location,* b_G^*. An option about randomly disturbed *best global*, is set by the expression:

$$b_G^* = b_G + w_{Gi}^* \times N(0,1) \tag{27}$$

where w_{Gi}^* is the *forth strategic parameter*, associated with the particle *i*. It *control the size of neighborhood of* b_G *where is more likely to find the real global best solution*. Another difference respect to the velocity operator of PSO, *is that the weights are defined for each particle of swarm* (subindex *i*).

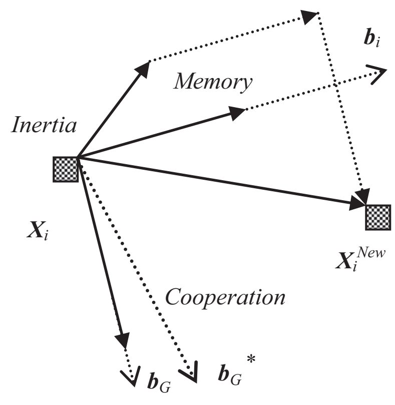

Fig. 4. Movement rule of EPSO in two dimensions.

Static fuzzy decision and fuzzy fitness function

Fuzzy decision

The metaheuristics SA and PSO were designed, originally, to *mono-objective* optimizations problems.

There are many approaches proposed to extend them to *multi-objective* optimizations [8]. In this paper, *a new extension capable to treat with no stochastic uncertainties of value, is proposed*. This kind of uncertainties are present in the preferences between the criteria of *multi-objective* optimization, and in the *satisfaction degree* that certain value of a single objective, produce in the decision-maker.

To represent and introduce such uncertainties in the model, the *static decision-making in fuzzy environments principle* [9] is proposed. It is expressed as follows:

Let a set of *fuzzy objectives (whose uncertainties of value are represented by mean of fuzzy sets)* $\{O\} = \{O_1, O_2, \ldots, O_n\}$ whose *membership functions* are μ_{Oj}, with $j = 1 \ldots n$, and a set of *fuzzy constraints (whose uncertainties of value in the upper and lower limits, are represented by mean of fuzzy sets)* $\{R\} = \{R_1, R_2, \ldots, R_h\}$ whose *membership functions* are μ_{Ri}, with $i = 1 \ldots h$. The Decision fuzzy set, ***D***, results:

$$D = O_1 \langle C \rangle O_2 \langle C \rangle \cdots \langle C \rangle O_n \langle C \rangle R_1 \langle C \rangle R_2 \langle C \rangle \cdots \langle C \rangle R_h \tag{28}$$

where $\langle C \rangle$ is a fuzzy sets operator called "confluence". The most common confluence operator, is the *intersection*. Then, the membership function of *D* is expressed as:

$$\mu_D = \mu_{O1} C \mu_{O2} C \ldots C \mu_{On} C \mu_{R1} C \mu_{R2} C \ldots C \mu_{Rh} \tag{29}$$

where *C* is an operator (called, in general, *t-norm*) between values of *membership functions*. For example, *if the confluence is* $\langle C \rangle \equiv \cap$ *(intersection), then C is the t-norm* $\equiv$ *min (minimum value, at certain instance, of all membership function* in Eq. (29)). Then, the *Maximizing Decision over a set of alternatives, [X], is:*

$$\mu_D^{Max} = MAX^{[X]} \{ \mu_{O1} C \mu_{O2} C \ldots C \mu_{On} C \mu_{R1} C \mu_{R2} C \ldots C \mu_{Rh} \} \tag{30}$$

A *t-norm* is defined as follows:

If $t : [0,1] \to [0,1]$ is a *t-norm*, then: (a) $t(0,0) = 0$; $t(x,1) = x$; (b) $t(x,y) = t(y,x)$; (c) if $x \leqslant \alpha$ and $y \leqslant \beta \Rightarrow t(x,y) \leqslant t(\alpha,\beta)$; and (d) $t(t(x,y),z) = t(x,t(y,z))$.

Notice that all fuzzy sets (*Objectives* and *Constraints*) *are mapping in the same fuzzy set of decision,* ***D****, and are treated the same way*.

This type of *fuzzy decision*, is *static*. Is evaluated at certain instance or values corresponding to memberships functions.

Fuzzy sets of optimization criteria in the Phase Balancing problem

To define a *multi-objective fuzzy function*, will be used these concepts. Develop of expressions, will be oriented to the *objectives* and *constraints* (criteria) of the Phase Balancing problem, but it could be extended to any set of criteria, represented by fuzzy sets.

Will be assume that the LV feeder system is under operation and exhibit a significant unbalance degree. Four criteria/objectives are introduced in the optimization, and all of them *must be minimized*: $Loss_T$, $I(\Delta u)$, $\left| I^{[0]} \right|_f$, from model (1), and *NC* that represent the *number of changes* (swapping) respect to reference system or existing system. A change has associated *a cost* (and it can disturb the normal service). The *constraints* (2)–(4), extended to all branches of system, will be considered *as crisp sets, and any solution that no satisfy that equations, will be discarded*.

All *membership functions* of fuzzy sets, will be construct as *linear functions* and, then, will be affected by exponentials weights, that represent the importance between the preferences of criteria. This is explain below.

Then, let two *limits values* for a given criteria $m : vMax$ and $vMin$, and let p_μ^m the exponential weight of fuzzy set.

The membership function associated to criteria-objective m, is expressed by:

$$\mu_m = 1; \quad \text{if } vMin_m \geqslant vm \tag{31}$$

$$\mu_m = \left[\frac{vMax_m - vm}{vMax_m - vMin_m}\right]^{p_\mu^m}; \quad \text{if } vMin_m \leqslant vm \leqslant vMax_m \tag{32}$$

($p_\mu^m > 1 \rightarrow$ *more importance-Contraction of fuzzy set;* $p_\mu^m < 1 \rightarrow$ *less importance-Expansion of fuzzy set).*

$$\mu_m = 0; \quad \text{if } vMax_m \leqslant vm \tag{33}$$

To observe this effect of exponential weights on a linear fuzzy sets, see Fig. 5.

In this work, such limits values are obtained as follows: (a) the $vMin_m$ will be the result of a PSO *mono-objective* simulation (that *minimize* the criteria vm) with a *deterministic fitness function* and (b) $vMax_m$ is a value that depend on the criteria under analysis. The calculation of *limit values* for each *membership function*, considering the four criteria in the specific Phase Balance problem, is presented to continue.

(1) *Total Active Power Loss* ($Loss_T$)
In this case, a *mono-objective* PSO is simulated to obtain the *minimal power loss* of LV feeder system, $vMin_{Loss}$. The value $vMax_{Loss}$ is obtained by a simulation of a three-phase load flow, in the reference situation (real LV feeder system, before optimization).

(2) *Drop Voltage Index* ($I(\Delta u)$)
A LV feeder system is the radial operation. This mean that one option to evaluate the maximum voltage drop, is from voltages in the terminal nodes of each feeder of system. Then, setting two values *uint* (voltage *in tolerance*) and *uoutt* (voltage *out of tolerance*), assigning it pertinent values per unit of nominal voltage (for example: *uint = 0.95 [pu]* and *uoutt = 0.92*) and applying it to the terminal nodes (the worst situation of voltage drops), is possible to define the limits as follow: $vMin_u = 1/uint$ and $vMax_u = 1/uoutt$. For each terminal node of LV feeder system, is considered a membership function (31)–(33), with this limits. Let nt the number of terminal nodes, then there will be $\mu(vu)_1, \mu(vu)_2, \ldots, \mu(vu)_{nt}$ membership functions related to the voltage drops in the feeders system. Then, it's proposal the following index $I(\Delta u)$:

$$I(\Delta u) = \mu(vu) = \sqrt[nt]{\prod_{i=1}^{nt} \mu(vu)_i} \tag{34}$$

(3) *Homopolar Component Current* $\left(\left|I^{[0]}\right|_f\right)$
In this case, a *mono-objective* PSO is simulated to obtain the minimal $\left|I^{[0]}\right|_f$ in LV feeder system, $vMin_{|I[0]|f}$.

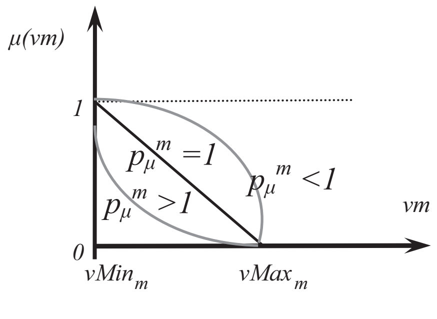

Fig. 5. Contraction ($p_\mu^m > 1$) and expansion ($p_\mu^m < 1$) of a linear fuzzy set $\mu(vm)$.

The value $vMax_{|I[0]|f}$ is obtained by a simulation of a three-phase load flow in the reference situation (real feeder systems, before optimization).

(4) *Number of Phase Changes- Swapping* (*NC*)
To determine $vMax_{NC}$, is proposed the expression:

$$vMax_{NC} = MAX\left\{NC_{PSO\ LossT}; NC_{PSO\ I(\Delta u)}; NC_{PSO\ |I[0]|f}\right\} \tag{34}$$

And to determine $vMin_{NC}$ is proposed:

$$vMin_{NC} = MAX\left\{NC_{PSO\ LossT}; NC_{PSO\ I(\Delta u)}; NC_{PSO\ |I[0]|f}\right\} - NC0 \tag{35}$$

where *NC0* is a number externally fixed (can be selected as 0, if is convenient).

Fuzzy fitness function

To extend the Fitness Function of a metaheuristic to multi-objective domain, two fundamentals requirement must be imposed to the eligible *t-norm*: (1) *it must satisfy the Pareto-Dominance scale that may to exist between solutions* and (2) *it must have Compatibility Metric with the Space of Search of problem to solve*. The first requirement is clearly necessary to any measure of merit associated to solutions, in a multi-objective optimization: *must be identified the better solutions and the non-dominated solutions (non dominated frontier)*. To explain the second requirement, must be considered the type of mechanism (heuristic) that produce the different individuals in the evolution of metaheuristic algorithm. For example, in SA metaheuristic, these evolution is simulated using probabilistic sampling techniques, and the fitness multi-objective function should be work together with the Function of Generation of Neighborhood for each solution, $q(i,j)$, to promote that a stimuli produced from q, be capable to impels the search in the space of solutions, to better global optimum approximation. In this case, *the fitness function, for the SA, is said metric compatible respect to Search Space*. In PSO/EPSO metaheuristics, the five principles of the Swarm Intelligence, constitute the reference toward *metric compatibility*: in this case, *the fitness function is said metric compatible respect to Search Space if promote the satisfaction of five Swarm Intelligence Principles*. The*t-norm* proposed in this work, that satisfy the two requirements (Pareto-Dominance and Metric Compatibility) mentioned, is the Einstein Product, defined as:

$$t_{PE} = \frac{x \times y}{2 - (x + y - x \times y)} \tag{36}$$

where x and y are two generic membership functions. From the properties of a *t-norm*, presented in section 'Fuzzy decision', is possible construct the *membership function of Decision fuzzy set*, as follows:

$$t_{PE}^1 = \frac{\mu(Loss_T) \times \mu\left(\left|I^{[0]}\right|_f\right)}{2 - \left(\mu(Loss_T) + \mu\left(\left|I^{[0]}\right|_f\right) - \mu(Loss_T) \times \mu\left(\left|I^{[0]}\right|_f\right)\right)} \tag{37}$$

$$t_{PE}^2 = \frac{t_{PE}^1 \times \mu(vu)}{2 - (t_{PE}^1 + \mu(vu) - t_{PE}^1 \times \mu(vu))} \tag{38}$$

and then:

$$\mu_D = \frac{t_{PE}^2 \times \mu(NC)}{2 - (t_{PE}^2 + \mu(NC) - t_{PE}^2 \times \mu(NC))} \tag{40}$$

Finally, the Fuzzy fitness function, Fff, as stated as follows:

$$\mu_D = t_{PE}\left\{\mu(Loss_T); \mu(vu); \mu\left(\left|I^{[0]}\right|_f\right); \mu(NC)\right\} = Fff \tag{41}$$

applied on each individual in the metaheuristic algorithm. The set of alternatives, [*X*], for the static fuzzy decision in the Eq. (30), depending on metaheuristic considered. For example, in PSO/EPSO, if the number of individuals or particles of swarm were 150, and at a given iteration, each particle has a position vector X_i, with $i = 1 \ldots 150$, then should be 150 alternatives to selection. However, if the metaheuristic were SA, the number of alternatives should be stated by the total energy states of system, evaluated in the whole algorithm, until the stopping condition is reached. This multi-objective approach is valid to the fitness function of any meta-heuristic: the SA is extended to FSA, and the EPSO to FEPSO with the same fuzzy fitness function.

Simulation on the real LV feeder system

The simulation of the two metaheuristics, FSA and FEPSO, is applied on the same real LV feeder system, represented in Fig. 6. This system exists in the city of Bariloche, province of Río Negro, Argentina. It corresponds to one of the six output of a Medium Voltage (MV)–Low Voltage (LV) substation, located in a low-incomes suburban area. For this reason there are only single phase loads in the feeders. This system is adopted as reference. It can observed the connection of loads to phases [*R*], [*S*] an [*T*]. The conductors of feeders have the followings parameters: **Pr**: 3×95 [mm^2], $(r = 0.372 + i\ xl = 0.089)$ [Ω]/[km]; **SI**, **SII**, **SIII**, **SIV**, **SV**, **SVI**, **TI**, **TII**, **TIII** and **TIV**: 3×35 [mm^2], $(r = 1.390 + i\ xl = 0.097)$ [Ω]/[km]. The number of loads is *nL = 115*.

The FSA algorithm, follows the procedure described in section 'Simulated Annealing (SA)', by replacing the Energy function *E* to *Fff*. The FEPSO algorithm, is described in the flow-chart of Fig. 7. *NIterMax* is the maximum number of iterations externally defined. It possible observe the scheme corresponding to PSO, by eliminating the processes called Evolutionary Operators and MultiObjective. By eliminating only the process called MultiObjective, it observed the scheme corresponding to EPSO. The parameters used in FSA, are listed below:

(a) Initial temperature: $T_0 = 1.0$. *(b) Number of iteration to the same Temperature: NT = 100. (c) Maximum number of iterations without improvement of fitness function (stopping condition): nMaxI* $= 300$. *(d) Cooling rate:* $\alpha = 0.8$. *(e) Function of generation of*

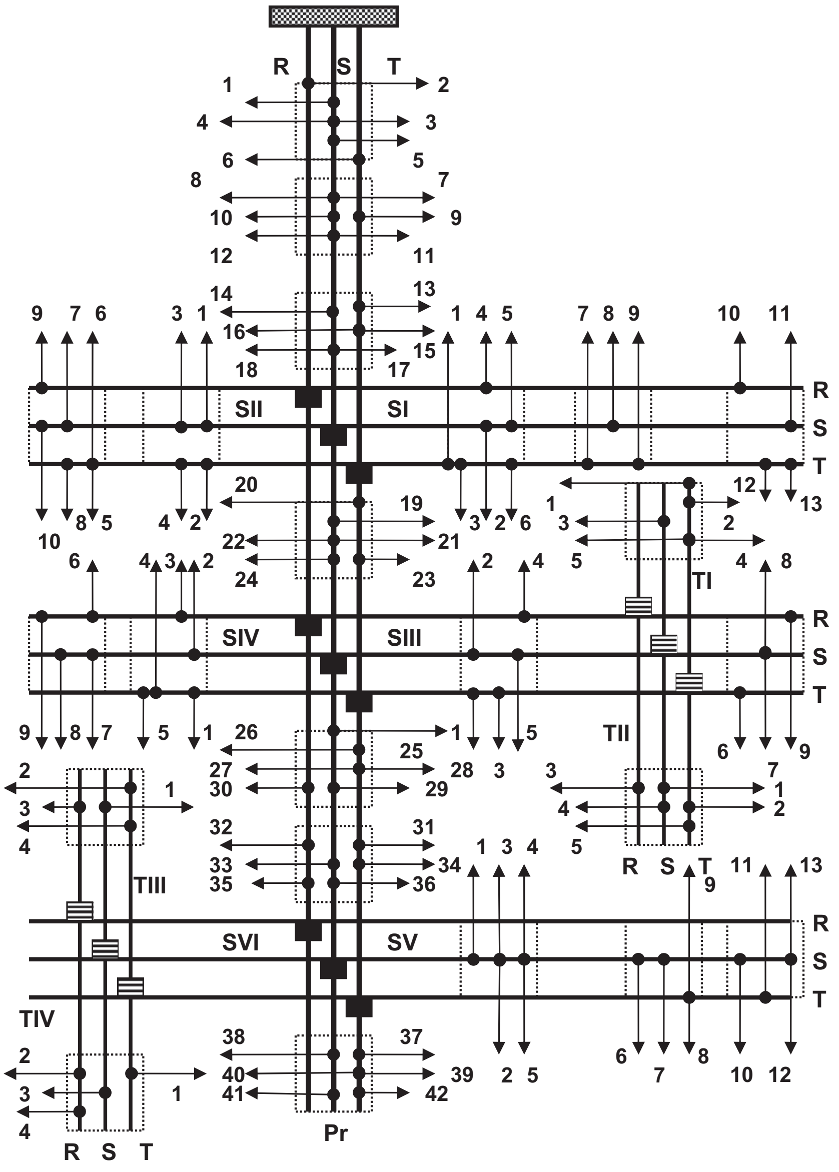

Fig. 6. Real LV feeder system to simulation.

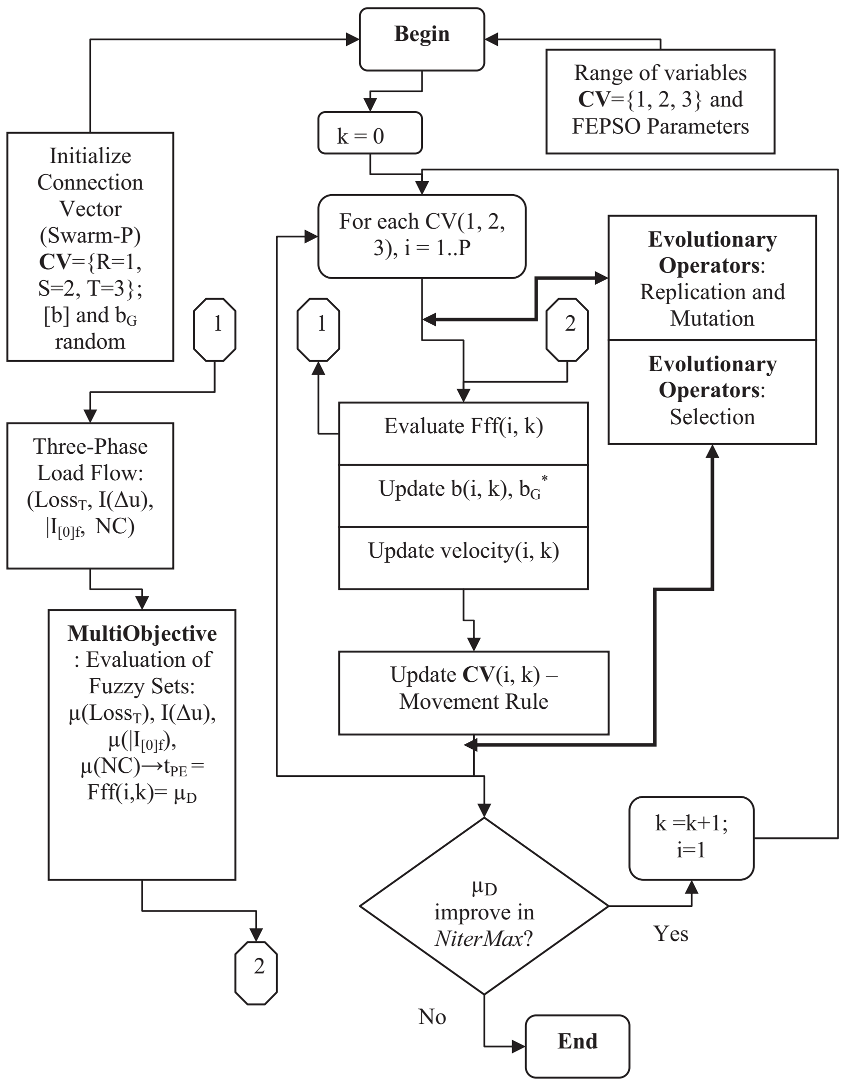
Begin
Range of variables CV={1, 2, 3} and FEPSO Parameters
k = 0
Initialize Connection Vector (Swarm-P) CV={R=1, S=2, T=3}; [b] and b_G random
For each CV(1, 2, 3), i = 1..P
Evolutionary Operators: Replication and Mutation
Evolutionary Operators: Selection
1
1
2
Three-Phase Load Flow: (Loss_T, I(Δu), |I_[0]f, NC)
Evaluate Fff(i, k)
Update b(i, k), b_G*
Update velocity(i, k)
MultiObjective : Evaluation of Fuzzy Sets: μ(Loss_T), I(Δu), μ(|I_[0]f), μ(NC)→t_PE = Fff(i,k)= μ_D
Update CV(i, k) – Movement Rule
k =k+1; i=1
μ_D improve in NiterMax?
Yes
No
2
End

Fig. 7. Flow chart of FEPSO in Phase Balancing.

neighborhood, q(i, j): this identification is making by selection, randomly, of one single phase load, and connecting it in a changed phase. (f) *k_B Constant:* $k_B = 0.00025$.

The Fuzzy fitness function, *Fff*, is evaluated from the results of a three-phase load flow.

Table 1 shows the results corresponding to application of two metaheuristics (FSA and FEPSO). Table 2 shows the complete results of swapping Phase Balancing for mono-objective PSOs and FEPSO. [*S*] is the load power vector [kVA] and [*d*] the nodes distances to substation output [km]. The *power factor* was considered *fp = cos Fi = 0.85* and the *simultaneity factor* of feeder system, *fs = 0.6*.

The *exponential weights*, can be obtained from the preference matrix between the optimization criteria. These preferences, are certainly subjective, and are expressed of contraction/expansion of fuzzy sets. The exponential weights were adopted as: $p\mu(Loss_T) = p\mu(|I_{[0]}|_f) = p\mu(NC) = 3$; $p\mu(\upsilon u) = 4$.

The reference values to form the membership functions to each optimization criteria were results:

$$[\upsilon MinLoss = 6.94\ [\text{kW}];\ \upsilon MaxLoss = 13.02\ [\text{kW}]];$$

$$[\upsilon Min_{|I[0]|f} = 0.1\ [\text{A}];\ \upsilon Max_{|I[0]|f} = 47.6\ [\text{A}]];$$

$$[\upsilon NCMin = 45;\ \upsilon NCMax = 85].$$

The parameters used in FPSO, are listed below:

Table 1
Results of metaheuristics.

Reference system results:
$Loss_T$ [kW] =13.02 – $|I_f^{[0]}|$ [*A*] = 47.6 – $I(\Delta u)$ = 0 – *NC* = 0

	Size	Time [min]	$Loss_T$	$\lvert I_f^{[0]}\rvert$	$I(\Delta u)$	*NC*
PSO						
$Loss_T$	150	45	6.94	18.93	0.32	81
$\lvert I_f^{[0]}\rvert$	150	37	10.16	0.100	0.00	79
$I(\Delta u)$	150	43	7.02	13.80	0.34	85
FSA						
t_{PE}	–	63	9.35	1.73	0.11	72
FEPSO						
t_{PE}	200	48	7.21	0.4	0.27	59

(a) Initial Weights: $w_I = 0.5$; $w_M = w_C = 2$. *In PSO are constants. (b)* σ *and* σ' = 0.2. *(c) Number of replication for each particle: r* = 5. *(d) Maximum number of iterations without improvement of fitness function (stopping condition): nMaxI* = 400.

It can observe in Table 1, the best results reached for the metaheuristic FEPSO, respect to FSA. There are some reasons for this: (a) FSA exhibit a poor ability to "escape" from local optimal, when the

Table 2
Complete swapping Phase Balancing results of metaheuristics PSO and FEPSO.

Feeder Pr [All elements follows the index order of Fig. 7]
[S] = [1.8 1.15 1.15 1.95 1.15 1.15 1.13 1.14 1.15 1.15 1.14 1.15 2.93 1.12 1.13 1 1.15 1.15 1.15 1.15 1.15 1.17 1.16 1.15 1.13 1.13 1.12 1.13 1.15 1.15 1.15 1.18 1.16 1.15 1.15 1.17 1.15 1.18 1.36 1.36 1.36 1.36]
[d] = [0.035 0.035 0.035 0.035 0.035 0.035 0.045 0.045 0.045 0.045 0.045 0.045 0.065 0.065 0.065 0.065 0.065 0.065 0.095 0.095 0.095 0.095 0.095 0.095 0.16 0.16 0.16 0.16 0.16 0.16 0.45 0.45 0.45 0.45 0.45 0.45 0.7 0.7 0.7 0.7 0.7 0.7]
[PSO($Loss_T$)] = [T S R S S S S T T S R S R S S S S T S T T R T T T S T R T T T S T R T T R S R R R S]
[PSO($|I_f^{[0]}|$)] = [S S S T S S S R T R R R S R S T T S S R S S S R T R S T S R T R S T T T R S R R T R]
[PSO($I(\Delta u)$)] = [R S R T R R T R R S R R R S R R R T T T R R S T T R T S T T S T T R R T R S S S S S]
[*FEPSO*] = [*R R S T R R T R R S R R R S R R R T R R R R T S R R T S T T R T S R R T S S T T S S*]

Feeder SI [All e elements follows the index order of Fig. 7]
[S] = [1 1.15 1 1.15 1.155 1 1.15 1.17 1.15 1.15 1.18 1.125 1.125]
[d] = [0.1 0.1 0.1 0.1 0.1 0.1 0.25 0.25 0.25 0.3 0.3 0.3 0.3]
[PSO($Loss_T$)] = [T R R T S R R T R T S T R] – [PSO($|I_f^{[0]}|$)] = [T T T T S R S R T S R S R]
[PSO($I(\Delta u)$)] = [T T S S R R T S T S S T S] – *[FEPSO] = [T S T R S R T S T R S T T]*

Feeder SII [All the elements follows the index order of Fig. 7]
[S] = [1.15 1.158 1.125 1.125 1.118 1.125 1 1.15 1.15 1.18]
[d] = [0.15 0.15 0.15 0.15 0.3 0.3 0.3 0.3 0.3 0.3]
[PSO($Loss_T$)] = [T T R T T R R T R R] – [PSO($|I_f^{[0]}|$)] = [T T T R S T R R S T] – [PSO($I(\Delta u)$)] = [S T T T S S S R T R] – *[FEPSO] = [R T T T S S S R T R]*

Feeder SIII [All the elements follows the index order of Fig. 7]
[S] = [1 1.15 1 1.155 1.155 1.18 1.15 1.155 1.15] – [d] = [0.25 0.25 0.25 0.25 0.25 0.35 0.35 0.35 0.35]
[PSO($Loss_T$)] = [T S R S S T R T R] – [PSO($|I_f^{[0]}|$)] = [T T S T T S R T R] – [PSO($I(\Delta u)$)] = [R R R S T T T S S] – *[FEPSO] = [R S R R T T S S T]*

Feeder SIV [All the elements follows the index order of Fig. 7]
[S] = [1.15 1.148 1.125 1.125 1.15 1.18 1.125 1 1.15] – [d] = [0.1 0.1 0.1 0.1 0.1 0.2 0.2 0.2 0.2]
[PSO($Loss_T$)] = [S R R T R T R T T] – [PSO($|I_f^{[0]}|$)] = *[S R T R R T S T R]* – *[PSO($I(\Delta u)$)] = [T T S S T S T S T]* – *[FEPSO] = [T S R R T R S S T]*

Feeder SV [All the elements follows the index order of Fig. 7]
[S] = [1 1.15 1 1.12 1.125 1 1.18 1.12 1.155 1.15 1.128 1.125 1.125]
[d] = [0.1 0.1 0.1 0.1 0.1 0.15 0.15 0.15 0.15 0.2 0.2 0.2 0.2]
[PSO($Loss_T$)] = [S R T T T R S R R R S S R] – [PSO($|I_f^{[0]}|$)] = [R T R R S T T S R S S R R] – [PSO($I(\Delta u)$)] = [R S T T R S T R R S S S S] – *[FEPSO] = [R S T T R S T R T S S S S]*

Feeder TI [All the elements follows the index order of Fig. 7]
[S] = [1 1.15 1 1.128 1.125] – [d] = [0.15 0.15 0.15 0.15 0.15]
[PSO($Loss_T$)] = [R T T S R] – [PSO($|I_f^{[0]}|$)] = [T S R S T] – [PSO(I(Δu))] = [S T T S S] – *[FEPSO] = [T T S S T]*

Feeder TII [All the elements follows the index order of Fig. 7]
[S] = [1 1.122 1 1.124 1.15] – [d] = [0.095 0.095 0.095 0.095 0.095]
[PSO($Loss_T$)] = [R T R T R] – [PSO($|I_f^{[0]}|$)] = [S T S R T] – [fUft] = [T T S S T] – *[FEPSO] = [T S S S T]*

Feeder TIII [All the elements follows the index order of Fig. 7]
[S] = [1 1.152 1.123 1.725] – [d] = [0.135 0.135 0.135 0.135]
[PSO($Loss_T$)] = [R S S R] – [PSO($|I_f^{[0]}|$)] = [R T R R] – [PSO($I(\Delta u)$)] = [T R S S] – *[FEPSO] = [S R S T]*

Feeder TIV [All the elements follows the index order of Fig. 7]
[S] = [1.12 1.15 1 1.8] – [d] = [0.125 0.125 0.125 0.125]
[PSO($Loss_T$)] = [S R R R] – [PSO($|I_f^{[0]}|$)] = [T S R S] – [PSO($I(\Delta u)$)] = [T S R S] – *[FEPSO] = [T S S S]*

search space is discrete and the good solutions are very dispersed. In fact, a *bootstrapping* procedure (not in a statistics sense, but computing) was necessary to change the membership function of $I(\Delta u)$ because this index *is strict*, and the algorithm FSA reached the stopping condition with *Fff = 0*. The *bootstrap*, *begins with another more flexible membership function*, $\mu(\nu u)^{\#}$, expressed as $I(\Delta u) = \mu(\nu u)^{\#} = e^{-[\xi \times Nnot]}$; $0 < \xi \leqslant 1$, where *Nnot* is the number of terminal nodes with out of tolerance voltage. If, in certain iteration, some solution that satisfy the Eq. (34), with $I(\Delta u) > 0$ is reached, then it change to $I(\Delta u) = \mu(\nu u)$. (b) Identical situation was presented when simulation of PSO mono-objective to minimize only $I(\Delta u)$, whose results are shown in Table 1 using the same *bootstrap* procedure. This confirm that PSO nor has able enough to "escape" that local optimum in this type of search space. (c) However, this ability is inherent to EPSO structure, because the self-adaptation introduced by the Evolutionary Operators, *allows a self-tuning of weights in the velocity operator*. Consequently, this avoid that algorithm to be trapped at some local minimum, or, even worst, at fitness 0. (d) In the FSA, even when a *bootstrap* procedure was introduced, after several simulations, *always was reached a local minimum*. The solution FSA shown in Table 2, is the best reached in 22 executions of algorithm. In this aspect PSO is a better option than SA, for this problem of Phase Balancing. This conclusion can be extended to any combinatorial optimization where the search space is discrete, the variables have narrow intervals as domain, and goods solutions are very much dispersed.

Conclusions

(A) A new metaheuristic, FEPSO, produce very good results in multi-objective combinatorial optimization problems, such as Phase Balancing with only single phase loads in a LV feeders system, with multiple objectives. It would be not possible to solve this problem with mathematical programming techniques.
(B) A comparison FEPSO vs. FSA methaheuristics, allow observing the advantages of swarm self-adaptive approach. The swarm intelligence principles, such as cooperation, combined with Evolution Strategies, seem like and address of metaheuristics toward solution for any combinatorial optimization problem.
(C) The introduction of PSO metaheuristic to obtain the lower limits in each criteria individually considered, at the same time that a comparison respect SA metaheuristic is pre-

sented. In particular, for the search space of Phase Balancing problem, a bootstrap procedure to minimize the strict Drop Voltage Index is required in SA and PSO, having a better performance in PSO. However this procedure is not required in EPSO/FEPSO, because its self-adaptive capacity to adjust the weights in the velocity operator.

(D) The extension of mono-objective metaheuristic EPSO to multi-objective FEPSO, is supported on the *static decision-making in fuzzy environments principle*, by mean of Einstein Product *t-norm*. This selection, had demonstrated very good results, satisfying the two requirements imposed to fitness function: *Pareto-Dominance scale* and *Metric Compatibility respect to search space*. However, other *t-norms* would be analyzed and proposed.

(E) The Phase Balancing problem, under the approach proposed in this work, is general and applicable to any LV feeder system. More objectives would be introduced in the optimization, under the same considerations presented in section 'Fuzzy sets of optimization criteria in the Phase Balancing problem'.

Appendix: Nomenclature

Acronyms	
EDS	Electric Distribution System
EPSO	Evolutionary Particle Swarm Optimization
FEPSO	Fuzzy Evolutionary Particle Swarm Optimization
FSA	Fuzzy Simulated Annealing
LV	Low Voltage
LVEDS	Low Voltage Electric Distribution System
MIP	Mixed-Integer Programming
MV	Medium Voltage
PSO	Particle Swarm Optimization
SA	Simulated Annealing
σA-ES	Self- Adaptative Evolution Strategies
Symbols	
$Loss_T$	Total Loss of Active Power in the Low Voltage Feeder System
$[R]$, $[S]$, $[T]$	Each phase of three-phase system
$I_f^{[0]}$	Homopolar component of three-phase currents system, referred to the substation output, f
$I(\Delta u)$	Drop Voltage Index (indicated as $\mu(vu)$ in the associated fuzzy set)
NC	Number of Swappings (Chages of Phases)
NC0	Offset to calculate the lower limit of swappings/changes of phases
$I_f^{[R,S,T]}$	Phase intensities refers to the substation output, f
I_{Maxj}	Phase line capacity of branch j
$E(i)$	Simulated Annealing – energy function evaluated in state i
T	Simulated Annealing – temperature
k_B	Simulated Annealing – Boltzmann constant (Metropoli's algorithm)
α	Simulated Annealing – cooling rate
NT	Simulated Annealing – number of trials per temperature
S_i	Simulated Annealing – neighborhood of state i
$q(i,j)$	Simulated Annealing – neighborhood Generation Function to state j from state i
p(accept j)	Simulated Annealing – acceptation probability of state j
X_i	Particle Swarm Optimization – position vector of particle i
V_i	Particle Swarm Optimization – velocity vector of particle i
t	Particle Swarm Optimization – iteration number
Δt	Particle Swarm Optimization – iterative incremental step
$w_{I,M,C}$	Particle Swarm Optimization – weights (I: Inertia; M: Memory; C: Cooperation)
b_i	Particle Swarm Optimization – individual best of particle i *(at given iteration t)*
b_G	Particle Swarm Optimization – global best of Swarm *(at given iteration t)*
$\delta(t)$	Particle Swarm Optimization – inertia function
χ	Particle Swarm Optimization – constriction factor
r	Evolutionary Particle Swarm Optimization – number of replications of each particle
$*$	Evolutionary Particle Swarm Optimization – evolutive operator of mutation
σ	Evolutionary Particle Swarm Optimization – learning parameter to weights
σ'	Evolutionary Particle Swarm Optimization – learning parameter to global best
w_{Gi}	Evolutionary Particle Swarm Optimization – Strategic Parameter to Control the Size of neighborhood of global best
$\langle C \rangle$	Operator of confluence between fuzzy sets
C	Generic *t-norm*, solidary to $\langle C \rangle$, between membership functions of fuzzy sets
μ	Generic membership function of a fuzzy set
μ_D^{Max}	Maximizing decision over certain set of alternatives
$vMax_m$	Upper limit value of criteria m
$vMax_m$	Lower limit value of criteria m
vm	Generic variable to criteria m
t_{PE}	*t-norm* Einstein product
μ_D	Maximizing decision in the fuzzy fitness function
Fff	Fuzzy fitness function

References

[1] Segura S, Romero R, Rider MJ. Efficient heuristic algorithm used for optimal capacitor placement in distribution systems. Int J Electr Power Energy Syst 2010;32():71–8.

[2] Barin A, Pozzatti LF, Canha LN, et al. Multi-objective analysis of impacts of distributed generation placement on the operational characteristics of networks for distribution system planning. Int J Electr Power Energy Syst 2010;32:1157–64.

[3] Chung TS, Lee KK, Chen GJ, et al. Multi-objective transmission network planning by a hybrid GA approach with fuzzy decision analysis. Int J Electr Power Energy Syst 2003;25:187–92.

[4] Tabu search, part I. ORSA J Comput 1989;1(3):190–206.

[5] Smith K, Everson J, Fieldsend R, Misra R, Murphy C. Dominance-based multi-objective simulated annealing. IEEE Trans Evolution Comput 2008;12 (3):323–42.

[6] Dorigo M, Stützle T. Ant colony optimization. MIT Press; 2004. ISBN 0-262-04219-32004.

[7] Miranda V, Keko H, Duque Jaramillo A. EPSO: evolutionary particle swarms. In: Jain L, Palade V, Srinivasan D, editors. Advances in evolutionary computing for system design, Springer series in computational intelligence, vol. 66; 2008. p. 139–68.

[8] Beni G, Wang J. Swarm intelligence in cellular robotic systems. In: Proceedings NATO advanced workshop on robots and biological systems, Tuscany, Italy, June 26–30, 1989.

[9] Bellman R, Zadeh L. Decision-making in a fuzzy environment. Manage Sci 1970;17:141–64.

[10] Zhu J, Griff B, Chow M. Phase balancing using mixed-integer programming. IEEE Trans Power Syst 1998;13(4).

[11] Kirkpatrick S, Gellat C, ecchi M. Optimization by simulated annealing. Science 1995;220(4598):671–80.