2020

A Parallel Evolutionary Strategy for the Large-Scale Dynamic Optimal Reactive Power Flow

Chixin Xiao
University of Wollongong, cx472@uowmail.edu.au

Danny Sutanto
University of Wollongong, soetanto@uow.edu.au

Kashem M. Muttaqi
University of Wollongong, kashem@uow.edu.au

Minjie Zhang
University of Wollongong, minjie@uow.edu.au

A Parallel Evolutionary Strategy for the Large-Scale Dynamic Optimal Reactive Power Flow

Abstract

© 2020 IEEE. This paper proposes a parallel evolutionary strategy to solve a large-scale dynamic optimal reactive power flow (ORPF) problem to minimize the transmission losses while ensuring that the power system constraints are met, by varying the voltage magnitude of generators, the transformer taps and all reactive power support, installed in the power system. The existing heuristic algorithms are time-consuming due to the large number of sequential calculations that need to be carried out. This paper proposes the use of a parallel evolutionary strategy to speed up the optimization of a large-scale ORPF problem based on the orthogonal design and the Nash equilibrium. The paper proposes to apply the orthogonal crossover operator for small but good representative combination samples, and then to apply the Nash Equilibrium to refine the combinations (i.e., to achieve the optima). The proposed design is effective to strengthen both the exploration and the exploitation of the decision space to obtain the global optimum. The simulation results validate the effectiveness of the proposed algorithms for a large-scale ORPF when applied on the IEEE 30-and IEEE 118 bus systems when compared with the results from current state-of-art approaches.

Disciplines

Engineering | Science and Technology Studies

Publication Details

C. Xiao, D. Soetanto, K. Muttaqi & M. Zhang, "A Parallel Evolutionary Strategy for the Large-Scale Dynamic Optimal Reactive Power Flow," in 2020 IEEE International Conference on Power Electronics, Smart Grid and Renewable Energy, PESGRE 2020, 2020, pp. 1-6.

A Parallel Evolutionary Strategy for the Large-Scale Dynamic Optimal Reactive Power Flow

Chixin Xiao, *Member, IEEE*, Danny Sutanto, *Senior Member, IEEE,* Kashem M. Muttaqi, *Senior Member*, Minjie Zhang, *Senior Member IEEE*

***Abstract*-- This paper proposes a parallel evolutionary strategy to solve a large-scale dynamic optimal reactive power flow (ORPF) problem to minimize the transmission losses while ensuring that the power system constraints are met, by varying the voltage magnitude of generators, the transformer taps and all reactive power support, installed in the power system. The existing heuristic algorithms are time-consuming due to the large number of sequential calculations that need to be carried out. This paper proposes the use of a parallel evolutionary strategy to speed up the optimization of a large-scale ORPF problem based on the orthogonal design and the Nash equilibrium. The paper proposes to apply the orthogonal crossover operator for small but good representative combination samples, and then to apply the Nash Equilibrium to refine the combinations (i.e., to achieve the optima). The proposed design is effective to strengthen both the exploration and the exploitation of the decision space to obtain the global optimum. The simulation results validate the effectiveness of the proposed algorithms for a large-scale ORPF when applied on the IEEE 30-and IEEE 118 bus systems when compared with the results from current state-of-art approaches.**

***Index Terms*—Optimal Reactive Power Flow, Orthogonal Design, Nash Game, Parallel Computing.**

I. Nomenclature

P_{loss}:	The total active power losses (p.u.).
N_L:	The total number of branches.
N_{Bus}:	The total number of buses excluding the slack node.
N_{PV}:	The total number of PV buses.
N_{PQ}:	The total number of PQ buses.
N_T:	the total number of the transformer branches.
N_C:	the total number of reactive power compensators.
Y_{ij}:	The *i*-row and the *j*-column element of the admittance matrix.
G_{ij}:	The conductance part of Y_{ij}.
B_{ij}:	The susceptance part of Y_{ij}.
θ_{ij}:	The angular difference of buses *i* and *j* (rad).
V_i, V_j:	The voltage magnitudes at the *i*-th, *j*-th buses respectively (p.u.).
P_{G_i}:	The injected active power at the *i*-th bus (p.u.).
Q_{G_i}:	The injected reactive power at the *i*-th bus (p.u.).
P_{D_i}:	The real-time active power demand at the *i*-th bus (p.u.).
Q_{D_i}:	The real-time reactive power demand at the *i*-th bus (p.u.).
$P_{D_i}^0$:	The scheduled active power demand at the *i*-th bus (p.u.).
$d_i(t)$:	The dynamic load disturbance at the *i*-th bus in time *t* (p.u.).
a-bus:	The set of load buses affected by the dynamic disturbance $d_i(t)$.
S_l:	The power flow in in the transmission line (p.u.).
V_{G_i}:	The voltage magnitude at the *i*-th PV bus (p.u.).
Q_{Ci} :	The reactive power source installation at the *i*-th bus (p.u.)
T_i :	The tap ratio of the *i*-th transformer.

This work was supported by the Australian Research Council (ARC) linkage grant under Grant LP0991428.

Chixin Xiao, Prof. Danny Sutanto & Prof. Kashem M. Muttaqi are with School of Electrical, Computer and Telecommunications Engineering, Prof. Minjie Zhang is with School of Computer Science and Software Engineering, University of Wollongong, Wollongong, NSW 2522, Australia. (e-mail: chixinxiao@gmail.com, soetanto@uow.edu.au, kashem@uow.edu.au, minjie@uow.edu.au).

II. Introduction

There is an increasing need for a fast method to solve the real time large-scale optimal reactive power flow [1] (ORPF) problems in modern power systems. The ORPF problem aims to minimize the transmission losses while ensuring that the power system constraints are met, by varying the voltage magnitude of generators, the transformer taps and all reactive power support, such as the shunt capacitors and reactors and the static VAr compensators, installed in the power system. The reduction of the reactive power losses is an effective way to increase the amount of the active power that can be transferred.

The ORPF problem is often deemed as a non-convex, non-linear and overdetermined global optimization problem [2] with simultaneous continuous and discrete decision variables [3]. The major difficulties in solving a real-time large-scale ORPF problem are (i) the need to cope with a high-dimensional decision space because of the many decision variables that need to be considered in the large-scale systems, and (ii) the need to obtain the optimal solutions in the shortest time possible because of the of the real time requirement and the dynamic uncertainties in the system variables, such as the load demand fluctuation, the wind power variation, and the inaccuracies in the dispatch schedule based on prediction.

The ORPF problem can be solved by (i) using the mathematical optimization methods [4] and (ii) using the heuristic approaches. Mathematical Optimization methods can be computationally fast, but they are often gradient-based and the solutions can be trapped into the local optima. Further, they may not be able to process the non-convex problems and discrete decision variables.

The heuristic approaches [5] use random probing and can have good performance by using global searching. However, these approaches are often population-based and rely on the decision space environment, which cause the approaches to be computationally time-consuming and unreliable. To be practical, the heuristic methods need to have a reliable strategy to find the optima in the shortest time possible.

One of the main drawbacks of the heuristic approaches is the need to carry out a large number of sequential evaluations, and the higher dimensionality of the decision space is, the more computation time [6] will be required.

Fortunately, there are abundant distributed parallel computing resources [7], e.g., a cluster of PCs, which can be accessed conveniently with relatively low cost. The ORPF problems can be easily divided into decentralized sub-problems, which can be optimized in parallel.

This paper proposes the use of a parallel evolutionary algorithm to optimize quickly a real-time large-scale ORPF problem. First, this paper proposes the selection step in the

crossover procedure by sampling actions in the form of "experiments". Thus, the sophisticated experimental design methods, e.g., the orthogonal design [8], can be used to help to find a small but good representative combination samples to test, rather than using all the combinations to test.

This is not only useful to save the computation time, but it can also help in the parallel implementation. Based on the proposed method for the crossover, the selected combination samples can be dispatched to the distributed computing unit to be refined further by adopting the distributed evolutionary operators based on the Nash Equilibrium [9], used in the game theory, that is able to optimize a multi-criteria problem by modifying the sub-sets of the variables (hence the discrete variables can be optimized independently). The proposed design can not only provide a better performance on both the solution quality and speed, but also can be statistically based to determine robust and reliable solutions.

To the best of our knowledge, the application of the orthogonal design and the Nash equilibrium for solving the ORPF problem has never been reported in the literature.

The main contributions of the proposed approach are:

1) The proposed approach proposes for the first time the use of parallel algorithms using parallel CPUs and a novel parallel searching strategy to solve a real-time large-scale ORPF problem;

2) The proposed approach applies the orthogonal crossover operator for small yet good representative combination samples, then use Nash Equilibrium to refine the combinations (i.e., the optima). The proposed design is effective to cope with both the exploration and the exploitation of the decision space to obtain the global optimum;

3) The proposed approach splits the global ORPF optimization problem into decentralized sub-optimization problems and dispatched to distributed CPUs for parallel computing through the communication links.

4) The proposed approach applies the Nash Equilibrium, to optimize independently each sub-set (e.g., the discrete variables) such that the exploitation is reinforced.

The paper is organized as follows, Section III discusses the problem formulation and some related backgrounds; Section IV explains the proposed approach in details; Section V presents the simulations and discusses the results; and finally, the Conclusion is presented in Section VI.

III. BASIC CONCEPTS

According to [3], the optimal reactive power flow (ORPF) aims to minimize the transmission losses while ensuring that the power system constraints are met, by varying the following controlled variables: the voltage magnitudes of the generators, the reactive power compensations from the reactive power compensators, and the transformer tap ratios.

A. Dynamic Optimal Reactive Power Flow

1) The objective to minimize the total active power losses P_{loss} can be formulated as (1)

$$Minimize\ P_{loss} = \sum_{k=1}^{N_L} G_{ij}[V_i^2 + V_j^2 - 2V_iV_jcos\theta_{ij}] \quad (1)$$

$$Y_{ij} = G_{ij} + jB_{ij} \quad (1.1)$$

The symbols are given in the Nomenclature section.

2) The constraints are discussed below:

The Nodal Power Balance Constraints: These constraints are typical load flow equations, and they include the active and reactive power balance as expressed below:

$$\Delta P_i = P_{G_i} - P_{D_i} - V_i \sum_{j=1}^{N_{bus}} V_j\,(G_{ij}\cos\theta_{ij} + B_{ij}sin\theta_{ij}) = 0 \quad (2)$$

$$\Delta Q_i = Q_{G_i} - Q_{D_i} - V_i \sum_{j=1}^{N_{bus}} V_j\,(G_{ij}sin\theta_{ij} - B_{ij}cos\theta_{ij}) = 0 \quad (2.1)$$

According to [3], the dynamic power system can be simulated as in (2.2).

$$P_{D_i} = P_{D_i}^0 + d_i(t),\ \ i \in N_{bus} \quad (2.2)$$

and

$$d_i(t) = \begin{cases} \gamma \cdot P_{D_i}^0, & \gamma \in [-0.2,0.2],\ \ if\ i \in abus \\ 0, & if\ i \notin abus \end{cases} \quad (2.3)$$

where γ is the disturbance coefficient (uniformly random number in $[-0.2,0.2]$).

The balance between the supply and demand is established in such a way that the total generator power outputs must equal the total load demand plus losses as shown in (3), from which the P_{loss} can be obtained.

$$\sum_{i=1}^{N_{PV}+slack} P_{G_i} = \sum_{i=1}^{N_{Bus}} P_{D_i} + P_{loss} \quad (3)$$

Generating Unit Constraints: the active power outputs and the reactive power outputs of the generating units are constrained by their minimum and maximum capacities, which are represented as below:

$$P_{G_i}^{min} \le P_{G_i} \le P_{G_i}^{max},\quad i \in N_{PV}\text{+slack} \quad (4)$$

$$Q_{G_i}^{min} \le Q_{G_i} \le Q_{G_i}^{max},\quad i \in N_{PV}\text{+slack} \quad (4.1)$$

Security constraints: the power flow in the transmission lines and transformers are constrained by their maximum capacity as below:

$$|S_l| \le S_l^{max},\quad l \in N_L \quad (4.2)$$

The voltage limits in the load bus are represented by,

$$V_i^{min} \le V_i \le V_i^{max},\quad i \in N_{bus} \quad (4.3)$$

The transformers have minimum/maximum tap setting limits,

$$T_i^{min} \le T_i \le T_i^{max},\ i \in N_T \quad (4.4)$$

The reactive power source installtion

$$Q_{Ci}^{min} \le Q_{Ci} \le Q_{Ci}^{max},\ i \in N_C \quad (4.5)$$

The decision variables (DV, or controlled variables) can be expressed as a vector as shown in (5).

$$DV = [V_{G_1} \cdots V_{G_{N_{PV}}}, Q_{C_1} \cdots Q_{C_{N_C}}, T_1 \cdots T_{N_T}] \quad (5)$$

That is, the decision vector consists of three kinds of variables: the variables corresponding to voltage magnitudes at PV buses; the variables corresponding to reactive power compensators and the variables corresponding to transformer tap ratios.

B. Orthogonal Design

Each solution of the decision vector is actually a value-combination of its variables. It is not cost effective to test all combinations to obtain the optimal solution. The use of orthogonal design (OD) [10], a representative experimental design method, can be helpful to enhance the exploring efficiency. The philosophy of OD is to test a small number of representative factor-combinations scattered uniformly over the whole possible-combination-space to find out the promising factor-combinations quickly. Such kind of representative factor-combinations are often provided by a series of orthogonal arrays.

Usually, an orthogonal array for N factors and Ql levels is denoted by $L_M(Ql^N)$, [8] where L is a Latin square and M is the representative combinations of the levels , that is, $L_M(Ql^N)$ is a

M -row and N -column matrix, in which each row represents a combination of the levels. For instance, $L_9(3^4)$ as shown in (6),

$$L_9(3^4) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 3 & 3 & 3 \\ 2 & 1 & 2 & 3 \\ 2 & 2 & 3 & 1 \\ 2 & 3 & 1 & 2 \\ 3 & 1 & 3 & 2 \\ 3 & 2 & 1 & 3 \\ 3 & 3 & 2 & 1 \end{bmatrix} \quad (6)$$

where 3^4 means an experiment is decided by 4 factors (variables) and each variable has 3 optional levels (values) denoted by $\{1,2,3\}$. The number of the whole possible combinations (solutions) is $3^4 = 81$, and the right part in (6) row-by-row consists of 9 representative combinations.

C. Genetic Algorithm based on Nash Equilibrium

Among representative parallel frameworks, the Nash game [11] was first introduced by J.F.Nash and can be described as follows: For an *m*-objective optimization problem, a Nash game consists of *m* non-cooperative players, each of them is in charge of one objective by managing a corresponding sub-set within the entire decision variables which are similar as shown in (5). The optimal solution of the Nash game is called Nash Equilibrium.

Refs[9] proposed a parallel genetic algorithm framework based on the Nash game. Suppose each player is performed by using an independent genetic algorithm (GA), to play the Nash game means that each GA should optimize its own objective by adjusting its corresponding sub-set among the decision variables, while assuming the values of the other objectives and the other sub-sets controlled by other GAs to be fixed. When no objective can be improved further by its GA, the Nash equilibrium is reached.

For example, in a 2-objective and 2-player Nash game, each player is acted by two independent GAs, e.g., GA-1, GA-2. The decision vector consisting of all decision variables is $DV = XY$, in which X represents the subset handled by GA-1, and optimized along with the objective-1, likewise, Y denotes the subset handled by GA-2 and optimized along with the objective-2. During the Nash game, GA-1 optimizes DV with respect to the objective-1by modifying X, while considering Y is fixed with the current optimum provide by the GA-2. Likewise, GA-2 optimizes DV with respect to the objective-2 by adjusting Y while assuming X is fixed with the current optimum provided by the GA-1.

D. Differential Evolution

Differential evolution (DE) [12][13] is a population-based real coding evolutionary algorithm, which is more convenient in practical application than the binary coding GA. DE maintains a NP -individual population, $\{DV_{1,Gth}, DV_{2,Gth}, \dots, DV_{NP,Gth}\}$, where NP represents the population size and Gth is the generation index. Each individual of the population represents a candidate solution which is probed randomly among the searching space by using three evolutionary operators such as, mutation, crossover and selection.

1) Mutation: To product a mutant vector (MV) based on the target vector, $DV_{i,Gth}$, (i.e., the current decision vector being processed), the randomly selected individuals and one of mutation formulas. The three widely used DE mutation formulas adopted in the rest sections are shown as follows,

DE/rand/1,

$$MV_{i,Gth} = DV_{i1,Gth} + F \cdot (DV_{i2,Gth} - DV_{i3,Gth}) \quad (7)$$

DE/best/1,

$$MV_{i,Gth} = DV_{best,Gth} + F \cdot (DV_{i2,Gth} - DV_{i3,Gth}) \quad (7.1)$$

DE/current-to-best/1,

$$MV_{i,Gth} = DV_{i,Gth} + F \cdot (DV_{best,Gth} - DV_{i,Gth}) + F \cdot (DV_{i1,Gth} - DV_{i2,Gth}) \quad (7.2)$$

where $i1, i2$ and $i3$ are three distinct indices randomly chosen from the set $\{1,2,\dots,NP\} - \{i\}$; F is the scale factor, a control parameter, belonging to the domain between 0 and 1; $DV_{best,Gth}$ is the best solution among the Gth-generation.

2) Binomial crossover: To product a trail vector, $TV_{i,Gth}$, by recombining the target vector, $DV_{i,Gth}$, and the mutant vector, $MV_{i,Gth}$, the strategy is shown as follows,

$$TV_{i,j,Gth} = \begin{cases} MV_{i,j,Gth} & if\ (rand \le CR)\ or\ (j = j_{rand}) \\ DV_{i,j,Gth} & Otherwise \end{cases} \quad (8)$$

where $TV_{i,j,Gth}$ represents the j-th decision variable in the decision vector $TV_{i,Gth}$, as well in the $MV_{i,j,Gth}$ and the $DV_{i,j,Gth}$; $rand$ is an uniformly random number generated instantly in [0,1]; j_{rand} is a random integer number among {1,2,…, Dim($DV_{i,j,Gth}$)}, where the function Dim(.) returns the dimensionality of the input vector; $CR \in [0,1]$ is the crossover probability.

3) Selection: To survive the better candidate between the target vector $DV_{i,Gth}$ and the trail vector $TV_{i,Gth}$ into the next generation as the new target vector $DV_{i,Gth+1}$.

$$DV_{i,Gth+1} = \begin{cases} TV_{i,Gth} & if\ f(TV_{i,Gth}) < f(DV_{i,j,Gth}) \\ DV_{i,j,Gth} & Otherwise \end{cases} \quad (9)$$

where the fitness objective function $f(\cdot)$ is assumed to be minimized.

IV. The Proposed Algorithm

For a large scale ORPF problem, it is necessary to decrease the computational time to acquire a new acceptable solution while tracking the load changes. For this purpose, two aspects of endeavors based on the use of orthogonal design and Nash game are given in this section.

A. The Main Framework

As shown in Fig. 1, the proposed algorithm can be illustrated as follows,

Input: the case file including all constraints and parameters.

Output: the optimal decision vector.

Step 1: to initialize a candidate (decision vector) population with the size of μ by taking advantage of orthogonal design, and to run a power flow [14] operator to evaluate each candidate;

Step 2: to produce an offspring population with the size of λ still by making use of the orthogonal design and the parent population (in this paper $\lambda = 3\ \mu$);

Step 3: to distribute each candidate of the offspring population to the player set consisting of multiple central processing units (CPU), (e.g., from CPU_1 to CPU_N);

Step 4: to exploit the Nash equilibrium for each offspring

candidate based on the use of the Nash game framework and the parallel cooperation of all players;

Step 5: to collect λ Nash equilibriums into the buffer, and to survive the first μ candidates to update the parent population by sorting the combination (the size is $\mu+\lambda$) of the parents and the Nash equilibriums in the buffer;

Step 6: to decide the stop criterion (in this paper the criterion is the maximum generation): if it has reached the exit else go step 2.

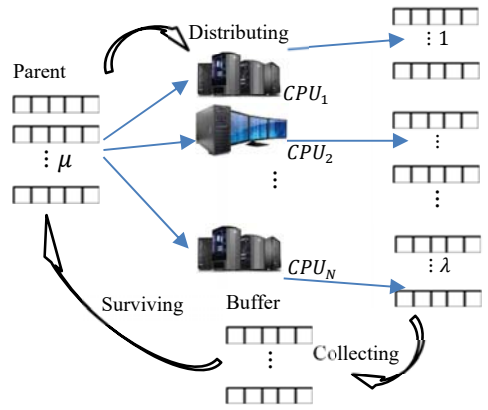

Fig.1 The proposed parallel framework

B. Utilization of the Orthogonal Design

In order to decrease the problem size, as show in (10), the decision vector defined in (5) in section III-A can be simplified as three factors: one factor, denoted as the vector $\vec{V}$, corresponds to the voltage-variables of the PV buses; one factor, denoted as the vector $\vec{C}$, corresponds to the compensator-variables and one factor, denoted as the vector $\vec{T}$, corresponds to the tap-variables.

$$DV=[\vec{V},\vec{C},\vec{T}]=\begin{cases}\vec{V}: & [V_{G_1}\cdots V_{G_{N_{PV}}}]\\ \vec{C}: & [Q_{C_1}\cdots Q_{C_{N_C}}]\\ \vec{T}: & [T_1\cdots T_{N_T}]\end{cases} \tag{10}$$

The upper and lower boundaries of $\vec{V},\vec{C}$,and $\vec{T}$ are given respectively in (10.1), (10.2) and (10.3).

$$\vec{V}:\begin{cases}UB=\left[V_{G_1}^{max},\cdots,V_{G_{N_{PV}}}^{max}\right]\\ LB=[V_{G_1}^{min},\cdots,V_{G_{N_{PV}}}^{min}]\end{cases} \tag{10.1}$$

$$\vec{C}:\begin{cases}UB=\left[Q_{C_1}^{max},\cdots,Q_{C_{N_C}}^{max}\right]\\ LB=[Q_{C_1}^{min},\cdots,Q_{C_{N_C}}^{min}]\end{cases} \tag{10.2}$$

$$\vec{T}:\begin{cases}UB=\left[T_1^{max},\cdots,T_{N_T}^{max}\right]\\ LB=[T_1^{min},\cdots,T_{N_T}^{min}]\end{cases} \tag{10.3}$$

Furthermore, each of the vectors can be treated as an individual factor and the domain of each factor can be quantized evenly into Ql levels, $\{L_1,L_2,\ldots,L_{Ql}\}$, , as shown in (11) ,

$$\begin{cases}L_1: & [(LB+0*\frac{(UB-LB)}{Ql}),(LB+1*\frac{(UB-LB)}{Ql})]\\ \vdots & \vdots\\ L_i: & [(LB+\frac{(i-1)*(UB-LB)}{Ql}),(LB+i*\frac{(UB-LB)}{Ql})]\\ \vdots & \vdots\\ L_{Ql}: & [(LB+\frac{(Ql-1)*(UB-LB)}{Ql}),(LB+Ql*\frac{(UB-LB)}{Ql})]\end{cases},\quad (1\le i\le Ql) \tag{11}$$

In this paper Ql=3, thus, based on (11) and the orthogonal array given in (6), there are nine promising combinations of the three factors, $\vec{V},\vec{C}$,and $\vec{T}$, and their three corresponding levels are listed in Table I.

Because three only factors in this study, the last factor (i.e., column) in (6) has been removed.

Besides, the domains of $\vec{C}$ and $\vec{T}$ are two sets of discrete values respectively between their lower and upper boundaries. Every two consecutive elements in each set have a fixed step size (e.g., Δ). Hence, the domain levels, $\{L_1,L_2,L_3\}$, of the $\vec{C}$ and $\vec{T}$ respectively may consist of invalid values which need to be modified by the use of (12),

$$\vec{X}=LB+\Delta\cdot\left\lfloor\frac{(\vec{X}-LB)}{\Delta}\right\rceil \tag{12}$$

where $\vec{X}$ denotes a variable-vector to be modified.

TABLE I THE NINE PROMISING COMBINATIONS SELECTED BY MAKING USE OF THE ORTHOGONAL ARRAY $L_9(3^4)$

Combination	Decision vector DV		
	Factor 1 - ($\vec{V}$)	Factor 2 - ($\vec{C}$)	Factor 3 - ($\vec{T}$)
1st	L_1	L_1	L_1
2nd	L_1	L_2	L_2
3rd	L_1	L_3	L_3
4th	L_2	L_1	L_2
5th	L_2	L_2	L_3
6th	L_2	L_3	L_1
7th	L_3	L_1	L_3
8th	L_3	L_2	L_1
9th	L_3	L_3	L_2

1)Statistical Sound Initialization:

This section explains the details of Step 1 in Section IV-A. In the initialization stage, unlike the traditional of the population-based algorithm to generate candidates (i.e., decision vectors) randomly, the proposed algorithm produces the candidate-population based on the use of the promising combinations shown in TABLE I and the use of formula in (13)

$$\vec{X}=Lb(L_i)+Rand\cdot(Ub(L_i)-Lb(L_i)) \tag{13}$$

where $\vec{X}$ denotes a variable-vector to be produced based on the use of TABLE I; $Rand$ represents a coefficient vector consisting of random numbers between 0 and 1; $Lb(L_i)$ and $Ub(L_i)$ represent to get the lower and upper boundaries of the domain at the level L_i.

For example, to produce a candidate (decision vector) based on the use of the 6th combination in Table I includes three steps: (i) to produce the part $\vec{V}$ by substituting the domain L_2 of $\vec{V}$ into (13); (ii) to produce the part $\vec{V}$ by substituting the domain L_3 of $\vec{C}$ into (13); and (iii) to produce the part $\vec{T}$ by substituting the domain L_1 of $\vec{T}$ into (13).

For each combination in Table I, $\left\lfloor\frac{\mu}{9}\right\rceil$ candidates can be produced, and the steps continues untill the size μ is reached.

2) Orthogonal Crossover:

This section explains the details of Step 2 in Section IV-A. In (14), suppose DV_1 and DV_2 are two candidates (decision vectors) selected randomly out of the parent population as shown in Fig.1,

$$\begin{cases}DV_1=[V_{G_{1,1}}\cdots V_{1,G_{N_{PV}}},Q_{1,C_1}\cdots Q_{1,C_{N_C}},T_{1,1}\cdots T_{1,N_T}]\\ DV_2=[V_{G_{2,1}}\cdots V_{2,G_{N_{PV}}},Q_{2C_1}\cdots Q_{2,C_{N_C}},T_{2,1}\cdots T_{2,N_T}]\end{cases} \tag{14}$$

Then, a temporary domain of exploring can be defined by (14.1)

$$\begin{cases}UB=[\max(V_{G_{1,1}},V_{G_{2,1}}),\cdots,\max(T_{1,N_T},T_{2,N_T})]\\ LB=[\min(V_{G_{1,1}},V_{G_{2,1}}),\cdots,\min(T_{1,N_T},T_{2,N_T})]\end{cases} \tag{14.1}$$

Similarly, the domain in (14.1) can be quantized evenly into

3 levels, $\{L_1, L_2, L_3\}$, and further quantized into 9 promising combinations.

Thus, for each pair of parents, 9 offspring can be generated, and so on till size λ is reached.

C. Nash Equilibrium based on DE

As discussed above, the use of orthogonal design can locate promising domains for the optimal solution. This section explains to exploit further in these domains for the Nash Equilibrium of each offspring candidate by using the Nash game framework with differential evolution, which corresponding to Steps 3-4 of Section IV-A.

Due to only one objective function (1) has been taken into account in this paper, the ORPF problem can be equal to a 1-objective and 3-player Nash game. Each player is an individual DE algorithm, as mentioned in Section III-D. Each DE runs on a distinct CPU which can communicate with each other during the evolutionary process.

As mentioned in (10), each decision vector consists of three sub-vectors such as $\vec{V}, \vec{C}$,and $\vec{T}$, in 1-objective and 3-player Nash game. One player focuses only on exploiting the voltage-vector $\vec{V}$ while keeping the other two sub-vectors $\vec{C}$ and $\vec{T}$ to be fixed till receiving better values from the other two players. In the case of finding better $\vec{V}$, the player will broadcast the new optimal $\vec{V}$ to the other two players.

As shown in the top in Fig.2, in the event of receiving a new offspring candidate (e.g., $DV = \overrightarrow{V_0^* C_0^* T_0^*}$) as mentioned in step 3 in Section IV-A, the Nash game operator divides the $\overrightarrow{V_0^* C_0^* T_0^*}$ into three sub-vectors such as $\overrightarrow{V_0^*}, \overrightarrow{T_0^*}$ and $\overrightarrow{C_0^*}$ respectively, in which the subscript represents the index of evolutionary generation (e.g., '0' means in the initialization stage of the Nash game); the superscript '*' represents the optimal value so far present generation.

For example, in the 1st generation, the Player1 exploits on its proprietary population Pop1($\vec{V}$) which focusing only to find out better $\vec{V}$. For each candidate among Pop1($\vec{V}$), the rest parts excluding the part $\vec{V}$ keep the values inherited from the preceded generation (i.e., $\overrightarrow{T_0^*}, \overrightarrow{C_0^*}$). When it comes to the 2nd generation, Player1 will broadcast the best $\vec{V}$ so far to the other two players, in the meanwhile, all parts of $\overrightarrow{T_0^*}, \overrightarrow{C_0^*}$ will be updated by using the new values $\overrightarrow{T_1^*}, \overrightarrow{C_1^*}$ received from the broadcast from the other two players as shown in Fig.2. In parallel, Player2 and Player3 act in the same way.

Such kind of process continues till there is no significant improvement which is taken at the best solution, that has reached the Nash Equilibrium. In this paper, the maximum generations for each DE (i.e., player) is set to10.

Finally, the Nash game operator as shown in Fig. 2 sends the collected Nash Equilibrium to the buffer as shown in Fig.1, and starts to exploit the next offspring candidate till size λ is reached.

V. Simulation and Analysis

In this section, two benchmark systems, IEEE 30-bus, 118-bus, are adopted into case 1 and case 2 respectively to verify the proposed approach (O_Nash_DE).

The parameters inside of the main framework of the proposed approach are set in, $\mu = 45, \lambda = 135,$ in case 1, and $\mu = 90, \lambda = 370$ in case 2. The Maximum generation of the main loop is 80, and the Maximum generation for exploiting each Nash Equilibrium is 10. Each DE (i.e., player) has two fixed parameters, CR=0.9, F=0.7. All tests were processed on a personal computer with a CPU of Intel Core i7 3.4 GHz, 16 GB of RAM, and with programming Matlab 2016b.

The base in this study is 100 MVA. The variable limits of

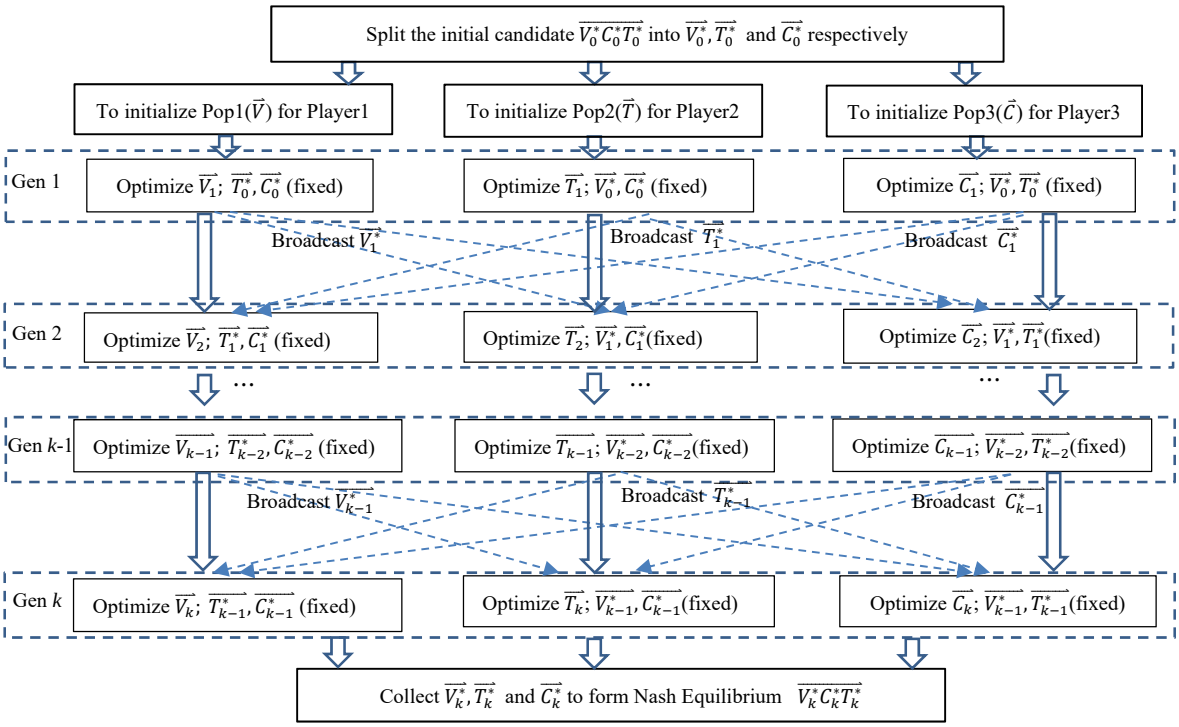

Fig. 2. Nash game operator by taking advantage of parallel cooperation of three players

the decision vector in (5) are taken from [5] [6] and listed in TABLE II, in which the transformer taps and the reactive power source installation are discrete variables.

TABLE II VARIABLES LIMITS (p.u.)

	V_{G_i}	Q_{C_i}	T_i	Step Δ (p.u.)	
min	0.95	0	0.9	ΔQ_{C_i}	0.01
max	1.1	0.05	1.05	ΔT_i	0.01

A. Case 1 (IEEE 30-bus system)

In this case, the total real load and reactive load are 2.834 (p.u.), and 1.262 (p.u.). Under the default setting the real and reactive losses are 0.17557 (p.u.) and 0.6769 (p.u.) respectively.

The results are collected based on 50-time distinct simulations to avoid the issue of randomness.

Table III compares the results of the proposed O_Nash_DE such as the best, worst active power losses of the reactive power dispatch and the CPU time cost, with the results obtained by another two different methods, PSO [5][6], and MAPSO [5][6]. These comparisons show that the proposed approach not only leads to lower active power losses than those found by the other two methods but also requires significantly less computational time, which confirms that the proposed approach has promising performance to obtaining fast solutions to achieve global optimum.

TABLE III COMPARISON OF THE IEEE 30-BUS (P.U.)

	PSO [5]	MAPSO [5]	O_Nash_DE
Best P_{loss} (p.u.)	0.049262	0.048747	**0.02479**
Worst P_{loss} (p.u.)	0.050769	0.048759	**0.02482**
Average P_{loss} (p.u.)	0.049973	0.048751	**0.02480**
Mean CPU time (sec)	59.21	41.93	**19.501**

Table IV summarizes the best decision vectors found by the above three methods in the 30 run times.

TABLE IV VALUES IN THE OPTIMAL DECISION VECTOR BY THREE METHODS (P.U.)

	Bus	PSO [5]	MAPSO [5]	O_Nash_DE
V_1	1	1.0725	1.0780	1.06
V_2	2	1.0633	1.0689	1.045
V_5	5	1.0410	1.0468	1.01
V_8	8	1.0410	1.0468	1.01
V_{11}	11	1.0648	1.0728	1.08
V_{13}	13	1.0597	1.0642	1.07
Q_3	3	0.00	0.00	0.00
Q_{10}	10	0.16	0.16	0.19
Q_{24}	24	0.12	0.12	0.04
T_1	6~9	1.03	1.04	0.98
T_2	6~10	0.95	0.95	0.97
T_3	4~12	0.99	0.99	0.93
T_4	28~27	0.97	0.97	0.97

B. Case2 (IEEE 118-bus system)

In this case, the proposed approach is investigated based on the use of the 118-bus system, in which the total real load and the reactive loads are 42.42 (p.u.), and 14.38 (p.u.). Under the default setting the real and reactive losses are 1.32863 (p.u.) and 7.8379 (p.u.) respectively.

TABLE V COMPARISON OF THE IEEE 118-BUS (P.U.)

	PSO [5]	MAPSO [5]	O_Nash_DE
Best P_{loss} (p.u.)	1.310471	1.26513	**0.6051197**
Worst P_{loss} (p.u.)	1.348792	1.30147	**0.630297**
Average P_{loss} (p.u.)	1.321843	1.28215	**0.629148**
Mean CPU time (sec)	144.46	119.35	**82.22**

Table V summarizes the best and worst loss values and computational time based on 50 individual runs. According to the comparison, the average loss is almost half than the PSO [5][6] and MAPSO [5][6], and the computational time is much faster than the other two methods.

VI. Conclusion

The paper presents a new dynamic ORPF to minimize the real power transmission loss and the performance of the algorithms are verified using the IEEE 30-bus and IEEE 118-bus systems where the results are compared with those simulated using PSO [5] and MAPSO [5].

The experimental results show that the proposed approach performs significantly better than PSO [5] and MAPSO [5], respectively, and the fast performance of the proposed method indicates that the proposed approach with the orthogonal design and the Nash game strategy can be a promising practical optimization method for the large-scale dynamic ORPF.

References

[1] J. Yu, W. Dai, W. Li, X. Liu, and J. Liu, "Optimal reactive power flow of interconnected power system based on static equivalent method using border PMU measurements," IEEE Transactions on Power Systems, vol. 33, no. 1, pp. 421–429, 2017.

[2] M. A. Medina, C. A. C. Coello, and J. M. Ramirez, "Reactive power handling by a multi-objective teaching learning optimizer based on decomposition," IEEE Transactions on power Systems, vol. 28, no. 4, pp. 3629–3637, 2013.

[3] T. Zhu, W. Luo, C. Bu, and L. Yue, "Accelerate population-based stochastic search algorithms with memory for optima tracking on dynamic power systems," IEEE transactions on power systems, vol. 31, no. 1, pp. 268–277, 2015.

[4] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "MATPOWER: Steady-State Operations, Planning and Analysis Tools for Power Systems Research and Education," Power Systems, IEEE Transactions on, vol. 26, no. 1, pp. 12–19, Feb. 2011.

[5] B. Zhao, C. X. Guo, and Y. J. Cao, "A multiagent-based particle swarm optimization approach for optimal reactive power dispatch," IEEE transactions on power systems, vol. 20, no. 2, pp. 1070–1078, 2005.

[6] W. Villa-Acevedo, J. López-Lezama, and J. Valencia-Velásquez, "A novel constraint handling approach for the optimal reactive power dispatch problem," Energies, vol. 11, no. 9, p. 2352, 2018.

[7] F. Guo, C. Wen, J. Mao, J. Chen, and Y.-D. Song, "Hierarchical decentralized optimization architecture for economic dispatch: A new approach for large-scale power system," IEEE Transactions on Industrial Informatics, vol. 14, no. 2, pp. 523–534, 2017.

[8] J.-T. Tsai, T.-K. Liu, and J.-H. Chou, "Hybrid Taguchi-genetic algorithm for global numerical optimization," IEEE Transactions on evolutionary computation, vol. 8, no. 4, pp. 365–377, 2004.

[9] J. F. Wang, J. Periaux, and M. Sefrioui, "Parallel evolutionary algorithms for optimization problems in aerospace engineering," Journal of Computational and Applied Mathematics, vol. 149, no. 1, pp. 155–169, 2002.

[10] Q. Zhang and Y.-W. Leung, "An orthogonal genetic algorithm for multimedia multicast routing," IEEE Transactions on Evolutionary Computation, vol. 3, no. 1, pp. 53–62, 1999.

[11] J. Periaux, L. F. Gonzalez, E. J. Whitney, and K. Srinivas, "MOO methods for multidisciplinary design using parallel evolutionary algorithms, game theory and hierarchical topology: theoretical background (I/III)," LECTURE SERIES-VON KARMAN INSTITUTE FOR FLUID DYNAMICS, vol. 3, p. 2, 2006.

[12] K. Price, R. M. Storn, and J. A. Lampinen, Differential evolution: a practical approach to global optimization. Springer Science & Business Media, 2006.

[13] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," IEEE Transactions on evolutionary computation, vol. 13, no. 5, pp. 945–958, 2009.

[14] R. D. Zimmerman, C. E. Murillo-Sánchez, and D. Gan, "MATPOWER: A MATLAB power system simulation package," Manual, Power Systems Engineering Research Center, Ithaca NY, vol. 1, 1997.