Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

A **new modified teaching-learning algorithm** for **reservation-constrained dynamic financial issuance**

[1*] Mr.Pradosh Ranjan Parida, [2] **Mr.SHYAMALENDU KHUNTIA**
[1*] *Asst. Professor, Dept. Of Electrical Engineering, NIT BBSR,*
Asst. Professor DEPT. of Electrical Engineering, NIT BBSR,
[1*] pradoshranjan@thenalanda.com, shyamalendukhuntia@thenalanda.com,

Abstract - This paper presents a new optimization algorithm, **called a** modified teaching-learning algorithm, **which solves** a more practical **design** of **a reserve-limited** dynamic **economic allocation** of **heat devices, taking into account** network losses and **operational limits of generator units (i.e. valves).). load effect and creep limitations). Unlike previous approaches, the rotary reserve requirements** of the three types of **systems are directly modeled into** the **problem,** and a new **constraint approach** is proposed to satisfy them**. The proposed teaching-learning optimization algorithm is a new population-based optimization method features between the teacher and learners (students). Therefore, this algorithm searches for the global optimal solutionthrough two main phases: 1) the "teacher phase" and 2) the"learner phase". Nevertheless, these two phases are not adequate for learning interaction between the teacher and the learners in the entire search space. Thus, in this paper a new phase named "modified phase" based on a self-adaptive learning mechanismis added to the algorithm to improve the process of knowledge learning among the learners and accordingly generate promising candidate solutions. The proposed framework is applied to 5-, 10-,30-, 40-, and 140-unit test systems in order to evaluate its efficiencyand feasibility.**

***Index Terms*—Dynamic economic dispatch, modified teaching-learning algorithm, ramp rate, reserve constraint, valve-point effects.**

NOMENCLATURE

Indices

m	Learner index.
i, j	Unit index.
k	Iteration index.
t	Time interval index.

Constants

a_i, b_i, c_i, e_i, f_i	Cost coefficients of unit i.
$B_{ij,t}$	Loss coefficient relating the productionsof units and at time (MW).
$B_{0,i,t}$	Loss coefficient associated with the production of unit at time .

Manuscript received September 27, 2011; revised February 12, 2012 and April 28, 2012; accepted June 22, 2012. Date of publication August 28, 2012; date of current version April 18, 2013. Paper no. TPWRS-00915-2011.
The authors are with the Department of Electronics and Electrical Engineering, Shiraz University of Technology (SUTech), Shiraz, Iran (e-mail: r.azizipanah@sutech.ac.ir; niknam@sutech.ac.ir; aghaei@sutech.ac.ir).
Color versions of one or more of the figures in this paper are available onlineat http://ieeexplore.ieee.org.
Digital Object Identifier 10.1109/TPWRS.2012.2208273

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

$B_{00,t}$	Loss coefficient parameter at time(MW).
DR_i	Ramp-down rate of unit
(MW/h). $k_{\max}$	Maximum iteration.
N_{learner}	Number of learners.
NT	Number of time intervals.
NU	Number of units.
$P_{D,t}$	Load demand at time (MW).
$P_i^{\max}$	Capacity of unit (MW).
$P_i^{\min}$	Minimum power output of unit (MW).
$\mathrm{rand}1,\ldots,10(\cdot)$	Random function generator in the range[0, 1].
SR_t	60-min spinning reserve requirementsat time (MW).
SR'_t	10-min spinning reserve requirementsat time (MW).
TF^k	Teaching factor in iteration .
UR_i	Ramp-up rate of unit (MW/h).

Variables

Delta_t	Power mismatch at time t (MW).
$F(\mathbf{P}_G)$	Total fuel cost at time span NT ($).
$G(\mathbf{P}_t)$	Total fuel cost at time t ($).
$\mathbf{ME}^k$	Mean matrix in iteration k
$\mathbf{P}_G$	Unit production matrix.
$P_{\mathrm{Loss},t}$	Total real power losses at time t (MW).
$P_{i,t}$	Generation output of unit i at time t (MW).
$\underline{P}_{i,t}$	Lower limit of the ith unit output power at time t (MW).
$\bar{P}_{i,t}$	Upper limit of the ith unit output power at time t (MW).
$\mathbf{TR}^k$	Teacher matrix in iteration k.

I. Introduction

DYNAMIC economic dispatch (DED) in power systems deals with determining the optimal production levels of the scheduled units over a short-term horizon to meet load demands. It is necessary to note that DED is the extension of the

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

conventional economic dispatch (ED). In DED, it is desirable to minimize the total fuel cost. In practical situations, the model of DED problem may need to consider the spinning reserve requirements (SRRs) in order to incorporate the unit coupling of ramp rates at the system level via unit reserve on the top of the time coupling of ramp rates at the unit level. Traditionally, the valve-point loading effects of the large steam turbines were ignored and a convex quadratic fuel cost function was considered for the thermal units. This leads to a mathematically simple formulation of the problem. However, a more realistic model must take into account the valve-point effects. This makes the fuel cost function non-convex and non-smooth. Moreover, the search space of the DED problem is irregular due to the ramp rate limits and SRRs constraint. Therefore, the DED problem is a complicated optimization problem for which finding the optimal solution is a difficult task.

Currently, the available methods and algorithms for solving DED problem are classified into two categories of classical optimization-based and meta-heuristic methods. The optimization-based methods consist of linear programming (LP) [1], nonlinear programming (NLP) [2], quadratic programming (QP) [3], Lagrangian relaxation (LR) [4], and dynamic programming (DP) [5], which impose no restriction on the non-smooth and non-convex characteristics of the valve-point effects. Nevertheless, these methods suffer from the "curse of the dimensionality" in the case of large-scale power systems. Consequently, these methods cannot guarantee to find the global optimum as well as to manage computational time when the nonlinearity and discontinuous characteristics are considered in the evaluations.

As a result, recently many modern meta-heuristic optimization algorithms have been developed and utilized successfully to solve the DED problem. Some of the most well-known methods are: simulated annealing (SA) [6], differential evolution (DE) [7]–[10], particle swarm optimization (PSO) [11]–[14], artificial immune system (AIS) [15], and improved pattern search based algorithm (PS) [16]. However, similar to the other methods mentioned before, these methods do not guarantee to find the global solution. Correspondingly, hybrid methods based on combined heuristic methods such as hybrid evolutionary programming and sequential quadratic programming (EP-SQP) [17], PSO-SQP [18], modified hybrid EP-SQP (MHEP-SQP) [19], hybrid quantum inspired PSO (HQPSO) [20], etc. were proposed to solve the DED problem by improving the ability of searching the entire search space while using fast computational analysis. Previously available approaches, e.g., [18], solved RCDED that the associated cost of SRRs and other constraints are added as penalty terms to the fuel cost function. However, no DED approach with simultaneous constraints-handling is currently available in the literatures without enforcing any restrictions on the objective function.

The original Teaching-Learning Algorithm (TLA) was firstly proposed by Rao *et al.* to solve a mechanical design optimization problem [21]. In their work, TLA was successfully applied to the test system and it was shown that the performance of TLA was more satisfactory than the other well-known algorithms in the area. In fact, TLA is a new population-based heuristic search algorithm, which considers the teacher role and the learners' interaction for solving optimization problems. Hence, TLA per-

formance has two phases: 1) teacher phase; and 2) learner phase. In the teacher phase, the teacher improves the knowledge of the learners up to the level of his/her own knowledge level. In fact, in this phase, the quality of the learners is affected by the good quality of the teacher as the best individual. In the learner phase, similarly to the other meta-heuristic algorithms, the in- formation is shared between the learners so that the level of their knowledge would be improved. The superiority of TLA in com- parison to the other heuristic methods has been illustrated in
[21] on a benchmark function. Unlike similar optimization algo- rithms, performance of TLA is independent of the initial valuesof parameters.

In this paper, a new modification phase is proposed and added to the original TLA to improve its performance. In the new mod- ification phase, a self-adaptive learning framework is adopted to probabilistically implement four mutation strategies with dif- ferent features in parallel. Indeed, the augmented phase can im-prove the convergence property and enhance the quality of the solution. The new modified TLA (MTLA) is implemented to solve the non-convex and non-smooth complex reserve con- strained dynamic economic dispatch (RCDED) problem using four test cases with five units, ten units, thirty units and one hun- dred units. Simulation results show that the new modified algo-rithm achieves better solutions and improves the convergence rate compared to other methods.

The main contributions of this paper can be summarized as follows: 1) the RCDED problem including ramp rate limits, valve-point effect and three types of the SRRs is formulated. Moreover, an enhanced simultaneous constraints-handling scheme is proposed to bias the optimization toward the feasibleregion without enforcing any restrictions on the objective function; 2) a new modified algorithm is proposed to solvethe RCDED problem; and 3) the performance of the proposed approach is successfully evaluated by numerical simulations.

The remainder of this paper is organized as follows: In Section II, a brief mathematical formulation of the RCDED is provided. In Section III, the new modified algorithm isdescribed. The proposed solution methodology is presentedin Section IV. In Section V, the feasibility and efficiency ofthe proposed method are investigated using four test systems. Finally, the paper concludes in Section VI.

II. Problem Formulation

The objective function and constraints of RCDED are de- scribed as follows:

A. Objective Function

The fuel cost of each thermal unit is characterized in the form of a quadratic function plus the absolute value of a sinu- soidal term corresponding to the valve point effects [22]. Con- sequently, the RCDED problem can be formulated as follows:

$$\blacksquare\mathrm{in}\ F(\mathbf{P}_G) = \sum_{t=1}^{\mathrm{NT}} G(\mathbf{P}_t) = \sum_{t=1}^{\mathrm{NT}} \sum_{i=1}^{\mathrm{NU}} \left(\begin{array}{l} a_i + b_i P_{i,t} + c_i P_{i,t}^2 \\ + \left| e_i \sin\left(f_i \left(P_i^{\min} - P_{i,t} \right) \right) \right| \end{array} \right) \quad (1)$$

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

where

$$\mathbf{P}_G = [\mathbf{P}_1 \quad \mathbf{P}_2 \quad \dots \quad \mathbf{P}_{\mathrm{NT}}]$$

and

$$-\mathbf{P}_t = [P_{1,t} \quad P_{2,t} \quad \dots \quad P_{\mathrm{NU},t}]^T.$$

B. Constraints

Limits associated with RCDED are as follows:

a) Power balance

$$\sum_{i=1}^{\mathrm{NU}} P_{i,t} = P_{D,t} + P_{\mathrm{Loss},t} \quad t = 1, \dots, \mathrm{NT} \tag{2}$$

where the power losses is in the following form [23]:

$$P_{\mathrm{Loss},t} = \sum_{i=1}^{\mathrm{NU}} \sum_{j=1}^{\mathrm{NU}} P_{i,t} B_{ij,t} P_{j,t} + \sum_{i=1}^{\mathrm{NU}} \mathrm{B}_{0,i,t} P_{i,t} + B_{00,t} \quad t = 1, \dots, \mathrm{NT}. \tag{3}$$

b) Up/down ramp rate limits

The power generated at the output of the ith thermal unit at time t may affect its output power in the next time step. This limitation can be expressed as follows:

$$P_{i,t} - P_{i,t-1} \le \mathrm{UR}_i \quad i = 1, \dots, \mathrm{NU};\ t = 1, \dots, \mathrm{NT} \tag{4}$$

$$P_{i,t-1} - P_{i,t} \le \mathrm{DR}_i \quad i = 1, \dots, \mathrm{NU};\ t = 1, \dots, \mathrm{NT}. \tag{5}$$

c) Generation limits

According to the ramp rate, the generation limits will be

$$\underline{P}_{i,t} \le P_{i,t} \le \bar{P}_{i,t} \quad i = 1, \dots, \mathrm{NU};\ t = 1, \dots, \mathrm{NT} \tag{6}$$

$$\bar{P}_{i,t} = \min\left(P_i^{\max}, P_{i,t-1} + \mathrm{UR}_i\right) \quad i = 1, \dots, \mathrm{NU};\ t = 1, \dots, \mathrm{NT} \tag{7}$$

$$\underline{P}_{i,t} = \max\left(P_i^{\min}, P_{i,t-1} - \mathrm{DR}_i\right) \quad i = 1, \dots, \mathrm{NU};\ t = 1, \dots, \mathrm{NT}. \tag{8}$$

d) Spinning reserve requirements

The SRRs should be considered as an additional constraint to avoid an unexpected large load to the system or a failure in a certain large unit. Here, SRRs for the RCDED problem are formulated in three different ways:

$$\left(\Delta_t^{(1)} = \sum_{i=1}^{\mathrm{NU}} P_i^{\max} - \left(P_{D,t} + P_{\mathrm{loss},t} + \mathrm{SR}_t\right)\right) \ge 0 \quad t = 1, \dots, \mathrm{NT} \tag{9}$$

$$\left(\Delta_t^{(2)} = \sum_{i=1}^{\mathrm{NU}} \left(\min\left(P_i^{\max} - P_{i,t}, \mathrm{UR}_i\right)\right) - \mathrm{SR}_t\right) \ge 0 \quad t = 1, \dots, \mathrm{NT}$$

$$\left(\Delta_t^{(3)} = \sum^{\mathrm{NU}} \min\left(P_i^{\max} - P_{i,t}, \frac{\mathrm{UR}_i}{6}\right) - \mathrm{SR}'_t\right) \ge 0 \tag{10}$$

Constraints (9) and (10) are generally applied in the unit commitment and DED problems within 60 min of being required [18], [24]. Using (11) will exactly satisfy the SRRs from the spinning generators in each time within 10 min of being required and its amount is related to the ramp up rate constraint of generating unit. For time interval t to $t+1$ the ramp up rate of unit i is UR_i (MW/h), the corresponding amount for 10 min is $\mathrm{UR}_i/6$ [25].

III. Modified Teaching-Learning Algorithm

As mentioned before, TLA as a novel optimization algorithm does not need to adjust its controlling parameters to reach the optimum solution. The performance of the original TLA depends on two main parts: 1) "teacher phase" or learning from teacher, and 2) "learner phase" or exchange of information between learners.

A. Teacher Phase

In TLA [21], each class consists of a number of learners $(\mathbf{P}_{G,m})$ with different grades. Similar to what happens in reality, the learner with the best grade is selected as the teacher. In TLA, the teacher's task is to improve the mean of the class to a value close to his or her mean value depending on the capabilities of the learners. In fact, a good teacher among his staff is one who brings his/her learners up to his/her level in terms of knowledge. Hence, the mean mark of his/her class, named "$\mathbf{ME}^k$", is improved sufficiently. In each iteration, the learner with the best fitness value among all learners is selected as a new teacher, which can be shown as $\mathbf{TR}^k = [\mathbf{tr}_1^k, \mathbf{tr}_2^k, \dots, \mathbf{tr}_{\mathrm{NT}}^k]$. The structure of each learner and the mean value of the class are defined as

$$\mathbf{P}_{G,m}^k = \left[\mathbf{P}_{m,1}^k, \mathbf{P}_{m,2}^k, \dots, \mathbf{P}_{m,\mathrm{NT}}^k\right] \quad m = 1, \dots, N_{\mathrm{learner}} \tag{12}$$

$$\mathbf{ME}^k = \left[\mathbf{me}_1^k, \mathbf{me}_2^k, \dots, \mathbf{me}_{\mathrm{NT}}^k\right]. \tag{13}$$

In this study, each learner $(\mathbf{P}_{G,m})$ is indicative of the solution which refers to the generation pattern of the generating units (as shown in (12)). The mean value of the class can be calculated as

$$\frac{\mathbf{me}_t^k = \left(\mathbf{P}_{1,t}^k + \mathbf{P}_{2,t}^k + \dots + \mathbf{P}_{N_{\mathrm{learner}},t}^k\right)}{N_{\mathrm{learner}} \quad t = 1, \dots, \mathrm{NT}.} \tag{14}$$

Now, for each learner, a new vector can be defined as follows:

$$\mathbf{P}_{G,m,\mathrm{new1}}^k = \left[\mathbf{P}_{m,1,\mathrm{new1}}^k, \mathbf{P}_{m,2,\mathrm{new1}}^k, \dots, \mathbf{P}_{m,\mathrm{NT},\mathrm{new1}}^k\right] \quad m = 1, \dots, N_{\mathrm{learner}} \tag{15}$$

$$\mathbf{P}_{G,m,\mathrm{new1}}^k = \mathbf{P}_{G,m}^k + \mathbf{DM}^k \quad m = 1, \dots, N_{\mathrm{learner}} \tag{16}$$

where $\mathbf{DM}^k$ as the difference value is defined as

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

(11)

$$\mathbf{DM}^k = \left[\mathbf{dm}_1^k, \mathbf{dm}_2^k, \ldots, \mathbf{dm}_{\mathrm{NT}}^k\right] \tag{17}$$

$$\mathbf{DM}^k = \mathrm{rand1}(\cdot)(\mathbf{TR}^k - \mathrm{TF}^k\mathbf{ME}^k) \tag{18}$$

$$\mathrm{TF}^k = \mathrm{round}(1 + \mathrm{rand2}(\cdot)). \tag{19}$$

In order to calculate each element $(\mathbf{P}_{m,t,\mathrm{new1}}^k)$ of the mth learner $(\mathbf{P}_{G,m,\mathrm{new1}}^k)$, the fitness function $G(\mathbf{P}_{m,t,\mathrm{new1}}^k)$

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

is compared with the fitness function of the target vector $(G(\mathbf{P}^k_{m,t,\text{learner}}))$:

$$\mathbf{P}^k_{m,t,\text{learner}} = \begin{cases} \mathbf{P}^k_{m,t,\text{new1}} & \text{if } G\left(\mathbf{P}^k_{m,t,\text{new1}}\right) \leq G\left(\mathbf{P}^k_{m,t,\text{learner}}\right) \\ \mathbf{P}^k_{m,t,\text{learner}} & \text{otherwise.} \end{cases} \quad (20)$$

B. Learner Phase

In this part, the learners try to increase their knowledge by helping each other. Each learner interacts with other learners randomly via group discussions, presentations, formal communications, etc. [21]. Thus, each learner can gain knowledge if the other ones know more than him or her. This process is simulated as described in the following.

For the mth learner in the class, two of the best individuals (n_1, n_2) are selected in the way that $n_1 \neq n_2 \neq m$. Now the new individual $(\mathbf{P}^k_{G,m,\text{new2}})$ is defined as shown in (21) and (22) at the bottom of the page.

Similar to the teacher phase, the replacement procedure can be implemented as

$$\mathbf{P}^k_{m,t,\text{learner}} = \begin{cases} \mathbf{P}^k_{m,t,\text{new2}} & \text{if } G\left(\mathbf{P}^k_{m,t,\text{new2}}\right) \leq G\left(\mathbf{P}^k_{m,t,\text{learner}}\right) \\ \mathbf{P}^k_{m,t,\text{learner}} & \text{otherwise.} \end{cases} \quad (23)$$

C. Modified Phase

Compared to the other evolutionary algorithms, TLA has major advantages that can be used in solving complex nonlinear optimization problems such as the RCDED problem. Some of these advantages are simple concept, lower computational complexity, easy implementation, higher consistency mechanism, minimal storage requirement and no need to tune algorithm parameters. Despite these characteristics, the interactions in the second phase (learner phase) may lead to inappropriate knowledge exchange between learners in the way that the algorithm may be trapped in local optima. Therefore, a novel self-adaptive learning modification approach is proposed to overcome this deficiency. It is necessary to note that the basic idea behind this approach is to simultaneously select adaptively multiple effective strategies from the candidate strategy pool on the basis of their previous experiences in the generated promising solutions and applied to perform the mutation operation. It means that at different steps of the optimization procedure, multiple strategies may be assigned a different probability based on their capability in generating improved solutions.

Accordingly, during the evolution process, with respect to each target solution in the current population which is extracted from the second phase (learner phase), one method will be selected from the strategy pool based on its probability. The more successfully one mutation method behaved in previous iterations to generate promising solutions, the more probably it will be chosen in the current iteration to produce solutions. In this paper, four mutation strategies are implemented in MTLA to optimize the complex non-linear, non-smooth and non-convex RCDED problem. These mutation operators can be described as follows:

$$\textbf{Method1}: \mathbf{P}^k_{G,m,\text{mod1}} = \mathbf{TR}^k + \text{rand4}(\cdot)\left(\mathbf{P}^k_{G,q_1} - \mathbf{P}^k_{G,q_2}\right) \quad m = 1, \ldots, N^k_1 \quad (24)$$

$$\textbf{Method2}: \mathbf{P}^k_{G,m,\text{mod2}} = \mathbf{P}^k_{G,q_1} + \text{rand5}(\cdot)\left(\mathbf{P}^k_{G,q_2} - \mathbf{P}^k_{G,q_3}\right) + \text{rand6}(\cdot)\left(\mathbf{TR}^k - \mathbf{P}^k_{G,q_4}\right) \quad m = 1, \ldots, N^k_2 \quad (25)$$

$$\textbf{Method3}: \mathbf{P}^k_{G,m,\text{mod3}} = \mathbf{P}^k_{G,q_1} + \text{rand7}(\cdot)\left(\mathbf{P}^k_{G,q_2} - \mathbf{P}^k_{G,q_3}\right) \quad m = 1, \ldots, N^k_3 \quad (26)$$

$$\textbf{Method4}: \mathbf{P}^k_{G,m,\text{mod4}} = \mathbf{P}^k_{G,q_1} + \text{rand8}(\cdot)(\mathbf{TR}^k - \mathbf{WR}^k) \quad m = 1, \ldots, N^k_4 \quad (27)$$

where N^k_1, N^k_2, N^k_3 and N^k_4 are the respective number of learners which choose the mutation method 1, 2, 3, and 4 in iteration k. In this regard, four learners (q_1, q_2, q_3, q_4) are randomly selected from the existing population in such a way that $q_1 \neq q_2 \neq q_3 \neq q_4 \neq m$ in order to uniformly cover the algorithm search domain. Also, $\mathbf{WR}^k$ is the worst vector among population in iteration k In order to improve the solutions of the proposed large-scale problem and further increase the population diversity and enhance the globally search capabilities, the mutation method1 and 2 can be used. The $\mathbf{TR}^k$ is used as an attractor to guide the information exchanging between the learners with a better manner. However, the premature convergence may be occurred in solving the problems with enormous local optima. The mutation method3 is able to achieve lower convergence speed but avoids quickly being trapped by local optima on the complex problems and taken from [26]. It is observed that this mutation only relies on the difference of learner information. The mutation strategy4 has

$$\mathbf{P}^k_{G,m,\text{new2}} = \left[\mathbf{P}^k_{m,1,\text{new2}}, \mathbf{P}^k_{m,2,\text{new2}}, \ldots, \mathbf{P}^k_{m,\text{NT},\text{new2}}\right]$$

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 51, Issue 04, April : 2022

$$m = 1, \ldots, N_{\text{learner}} \quad (21)$$

$$\mathbf{P}_{G,m,\text{new2}}^{k} = \begin{cases} \mathbf{P}_{G,m}^{k} + \text{rand3}(\cdot)\left(\mathbf{P}_{G,n_1}^{k} - \mathbf{P}_{G,n_2}^{k}\right) & \text{if } F\left(\mathbf{P}_{G,n_1}^{k}\right) < F\left(\mathbf{P}_{G,n_2}^{k}\right) \\ \mathbf{P}_{G,m}^{k} + \text{rand3}(\cdot)\left(\mathbf{P}_{G,n_2}^{k} - \mathbf{P}_{G,n_1}^{k}\right) & \text{otherwise} \end{cases}$$

$$m = 1, \ldots, N_{\text{learner}} \quad (22)$$

a powerful local search capability and fast convergence speed. This mutation is motivated from nature and human actions. In other words, although all learners in a class are different in many ways but all of them tend to enhance themselves by following the same direction of the elite learner and similarly they try to avoid the direction of the lazy one in competition with others.

Generally speaking, the criteria of selecting these four strategies are that they have different characteristics that cover diverse conditions. The occurrence of mutation is followed from the requirements of the TLA search process. All the learners in the population will have a chance to be mutated based on the probability of their methods of mutating. In this approach, instead of using relatively fixed execution probabilities during the whole optimization procedure, MTLA uses a probabilistic updating mechanism which is described in the following manner. In the probability model, each learner selects one of these four methods. Denote $\mathrm{Prob}_a^1 = 0.25, a = 1, 2, 3, 4$ as the initial probability of implementing ath mutation strategy. Also, a parameter called accumulator is assigned to each of mutation strategies denoted by $\mathrm{Acum}_a^k, (a = 1, 2, 3, 4)$ which have the initial value of zero. In each iteration, a weight factor is assigned to each learner after sorting the population according to (28). It is clear that the best learner gets the larger weight factor. After that the related accumulator of each strategy will be updated based on (29) [27]:

$$w_m^k = \frac{\log(N_{\mathrm{learner}} - m + 1)}{\log(1) + \cdots + \log(N_{\mathrm{lernear}})} \quad m = 1, \ldots, N_{\mathrm{learner}} \qquad (28)$$

$$\mathrm{Acum}_a^{k+1} = \mathrm{Acum}_a^k + \frac{w_m^k}{N_a^k} \quad m = 1, \ldots, N_a^k;\ a = 1, 2, 3, 4 \qquad (29)$$

where $w_m^k \ (m = 1, \ldots, N_a^k)$ are the weight factors corresponding to each strategy in iteration k. After the fixed number of generations LP, the excitation probability is calculated as [27]

$$\mathrm{Prob}_a^{k+1} = (1 - \alpha)\mathrm{Prob}_a^k + \alpha \frac{\mathrm{Acum}_a^{k+1}}{\mathrm{LP}} \quad a = 1, 2, 3, 4;\ \mathrm{LP} = 10 \qquad (30)$$

where α is the learning rate to control the learning speed in the MTLA algorithm and it is considered to be equal to $\alpha = 1/6$ in this paper [27]. Finally, the Roulette Wheel Mechanism (RWM) is applied to choose the ath modification method for each learner based on the normalized probability values as follows:

$$\mathrm{Prob}_a^{k+1} = \frac{\mathrm{Prob}_a^{k+1}}{\sum_{a=1}^{4} \mathrm{Prob}_a^{k+1}} \quad a = 1, 2, 3, 4. \qquad (31)$$

It can be expected that the mutation methods which have generated higher-quality individuals tend to increase their probabilities iteration by iteration. In the MTLA solution technique, with respect to each target solution in the current population which is extracted from the second phase, i.e., learner phase, one trial solution generation method is selected from the strategy pool according to its probability on the basis of (31). The selected method is subsequently applied to the corresponding target solution to generate a trial solution. The details of this procedure are as follows:

For $m = 1$ to N_{learner}

Select the ath mutation strategy by RWM selection based on (31) for the mth learner as follows:

If $\mathrm{rand}_m(\cdot) < \mathrm{Prob}_1^{k+1}$

Select mutation method1 for target solution m.

Elseif $\mathrm{rand}_m(\cdot) < \mathrm{Prob}_1^{k+1} + \mathrm{Prob}_2^{k+1}$

Select mutation method2 for target solution m.

Elseif $\mathrm{rand}_m(\cdot) < \mathrm{Prob}_1^{k+1} + \mathrm{Prob}_2^{k+1} + \mathrm{Prob}_3^{k+1}$

Select mutation method3 for target solution m.

Else

Select mutation method4 for target solution m.

Endif

Endfor (refers to index m)

After the above process, the new solution is generated for each learner m as $\mathbf{P}_{G,m,\mathrm{new3}}^k$. Then modified individual is mixed with $\mathbf{P}_{G,m}^k$, which generates $\mathbf{P}_{G,m,\mathrm{new3}}^k$ as

$$\mathbf{P}_{G,m,\mathrm{moda}}^k = \left[\mathbf{P}_{m,1,\mathrm{moda}}^k, \mathbf{P}_{m,2,\mathrm{moda}}^k, \ldots, \mathbf{P}_{m,\mathrm{NT},\mathrm{moda}}^k\right] \qquad (32)$$

$$\mathbf{P}_{G,m,\mathrm{new3}}^k = \left[\mathbf{P}_{m,1,\mathrm{new3}}^k, \mathbf{P}_{m,2,\mathrm{new3}}^k, \ldots, \mathbf{P}_{m,\mathrm{NT},\mathrm{new3}}^k\right]. \qquad (33)$$

Each element of $\mathbf{P}_{G,m,\mathrm{new3}}^k$, denoted by $\mathbf{P}_{m,t,\mathrm{new3}}^k$, is calculated as

$$\mathbf{P}_{m,t,\mathrm{new3}}^k = \left[P_{m,t,1,\mathrm{new3}}^k \quad P_{m,t,2,\mathrm{new3}}^k \quad \ldots \quad P_{m,t,\mathrm{NU},\mathrm{new3}}^k\right]^T \qquad (34)$$

$$P_{m,t,i,\mathrm{new3}}^k = \begin{cases} P_{m,t,i,\mathrm{moda}}^k & \text{if } (\mathrm{rand9}(\cdot) \le \mathrm{rand10}(\cdot)) \\ P_{m,t,i,\mathrm{learner}}^k & \text{otherwise} \end{cases} \qquad (35)$$

where $P_{m,t,i,\mathrm{new3}}^k$, is the power generated at the output of the ith unit in the tth time interval for the mth learner of the kth iteration. For the replacement operation, the fitness function of the mixed vector $G(\mathbf{P}_{m,t,\mathrm{new3}}^k)$ should be compared with the fitness function of the existing vector, $G(\mathbf{P}_{m,t,\mathrm{learner}}^k)$, as follows:

$$\mathbf{P}_{m,t,\mathrm{learner}}^k = \begin{cases} \mathbf{P}_{m,t,\mathrm{new3}}^k & \text{if } G\left(\mathbf{P}_{m,t,\mathrm{new3}}^k\right) \le G\left(\mathbf{P}_{m,t,\mathrm{learner}}^k\right) \\ \mathbf{P}_{m,t,\mathrm{learner}}^k & \text{otherwise.} \end{cases} \qquad (36)$$

The final results are the MTLA output of the kth iteration and the input population for the next iteration.

It should be noted that similar to other evolutionary algorithms such as GA [22] and PSO [11], the TLA and MTLA try to find the optimal solution through populations that are randomly generated. Although effectiveness of the algorithm changes with its parameter values, unlike other optimization techniques, TLA do not require any parameter tuning process [21]. This is the attractive aspect of the proposed approach. In original TLA, solutions are more likely to cluster together in similar groups, while in MTLA, solutions do not have propensity to clump due to the added modified phase. This is the advantage of MTLA in comparison to TLA. As in PSO [11], TLA uses the teacher of the iteration to improve the existing solution so as to increase the convergence rate. GA [22] uses selection, crossover and mutation processes to develop itself, but TLA implement the mean value of the class to improve itself. However, Elitism operation improves the algorithm's efficiency in this respect.

IV. Solution Methodology

In this section, MTLA is applied to the RCDED problem, the pseudo-code of the proposed approach is presented, and some relevant tools are discussed. The flowchart of the whole process of the MTLA technique is given in Fig. 1 in order to depict the order of the proposed algorithm. It is clear that the first phase (teacher phase), the second phase (learner phase) and the third phase (modified phase) are applied on the population, consecutively. The output solutions of the modified phase are as the input population for the next iteration.

A. Application of MTLA to the RCDED Problem

The decision variables of the RCDED problem are the generation pattern of the NG thermal units through the NT time intervals. Therefore, each learner is associated with $\mathrm{NU} \times \mathrm{NT}$ variables. The process of the MTLA can be summarized as follows:

Step 1: *Input the required information of the RCDED problem*.

Step 2: *Representation of the learner*; each learner indicates a solution for the power generation of the units for the NT time intervals as in (12).

Step 3: *Generation of the initial population with constraint-handling*; the candidate solution of each individual (generating units' output) is randomly initialized in the feasible range, which would satisfy the constraints given by (2)–(11) as follows:

Step 3.1: *Simultaneous handling of the SRRs and ramp rate constraints*: for each hour, the feasibility of constraints (6)–(11) is checked. If these constraints are violated, the algorithm returns to previous hours and modifies them in the way that it can reach the desired solution according to the following backward and forward procedure:

For $m = 1$ to N_{learner}

For $t = 1$ to NT

$H = t$;

Generate $\mathbf{P}_{m,H}$ randomly subject to constraint (6). To satisfy power balance, go to the step 3.2 and return. Then, calculate the value of violation to the SRRs constraints as

$$\Delta'_{m,H} = \min\left(\Delta^{(1)}_{m,H}, \Delta^{(2)}_{m,H}, \Delta^{(3)}_{m,H}\right) \qquad (37)$$

if $\Delta'_{m,H} < 0$.

Backward procedure: go to the previous time and subtract $\Delta'_{m,H}$ from each $\bar{P}_{m,H-1,i}$ which are fixed to their maximum values. Generate $\mathbf{P}_{m,H-1}$ randomly subject to constraints (6), (7), and (8) then, compute $\Delta'_{m,H-1}$. This procedure continues until the time reached in which the violation is greater than or equal to zero. Save this time in h

Forward procedure: Generate $\mathbf{P}_{m,t}$, $t = h+1$ to H randomly subject to constraint (6). Check the power balance according to step 3.2 and calculate the total fuel cost $G(\mathbf{P}_{m,t})$ using (1). Then the value of violation $\Delta'_{m,H}$ is calculated again based on the power output of the $\mathbf{P}_{m,H}$. The backward and forward procedures continue until $\Delta'_{m,H} \geq 0$.

Else

Calculate the total fuel cost $G(\mathbf{P}_{m,t})$ using (1).

Endfor (refers to index t)

Calculate $F(\mathbf{P}_{G,m})$ from (1).

Endfor (refers to index m)

Step 3.2: *Power balance handling*: for satisfying the constraint (2) the value of power mismatch is calculated for each $\mathbf{P}_{m,t}$ of matrix $\mathbf{P}_{G,m}$ as follows:

$$\mathrm{Delta}_{m,t} = \sum_{i=1}^{\mathrm{NU}} P_{m,t,i} - P_{D,t} - P_{m,\mathrm{Loss},t} \qquad t = 1, 2, \ldots \mathrm{NT}. \qquad (38)$$

If $\mathrm{Delta}_{m,t} = 0$, return.

If $\mathrm{Delta}_{m,t} \neq 0$, select one unit $P_{m,t,i}$ of $\mathbf{P}_{m,t}$ randomly and subtract $\mathrm{Delta}_{m,t}$ from it, subject to (6). This procedure continues to reach the zero value of $\mathrm{Delta}_{m,t}$ by selecting different units to repair power mismatch [12]. The flowchart for the proposed constraints handling is shown in Fig. 1.

Step 4: *Teacher phase*; in the current iteration (k), the best solution is selected for the teacher $(\mathbf{TR}^k)$ and the mean value of the class $(\mathbf{ME}^k)$ is calculated using (13) and (14). This step is implemented as described in Section III-A.

Step 5: *Learner phase*; learners try to improve themselves via the interaction process described in Section III-B.

Step 6: *Modified phase*; this step is implemented as described in Section III-C. The modification process can be expressed as shown in the next subsection.

Step 7: *Update procedure*; the initial population is updated based on the new improved learners.

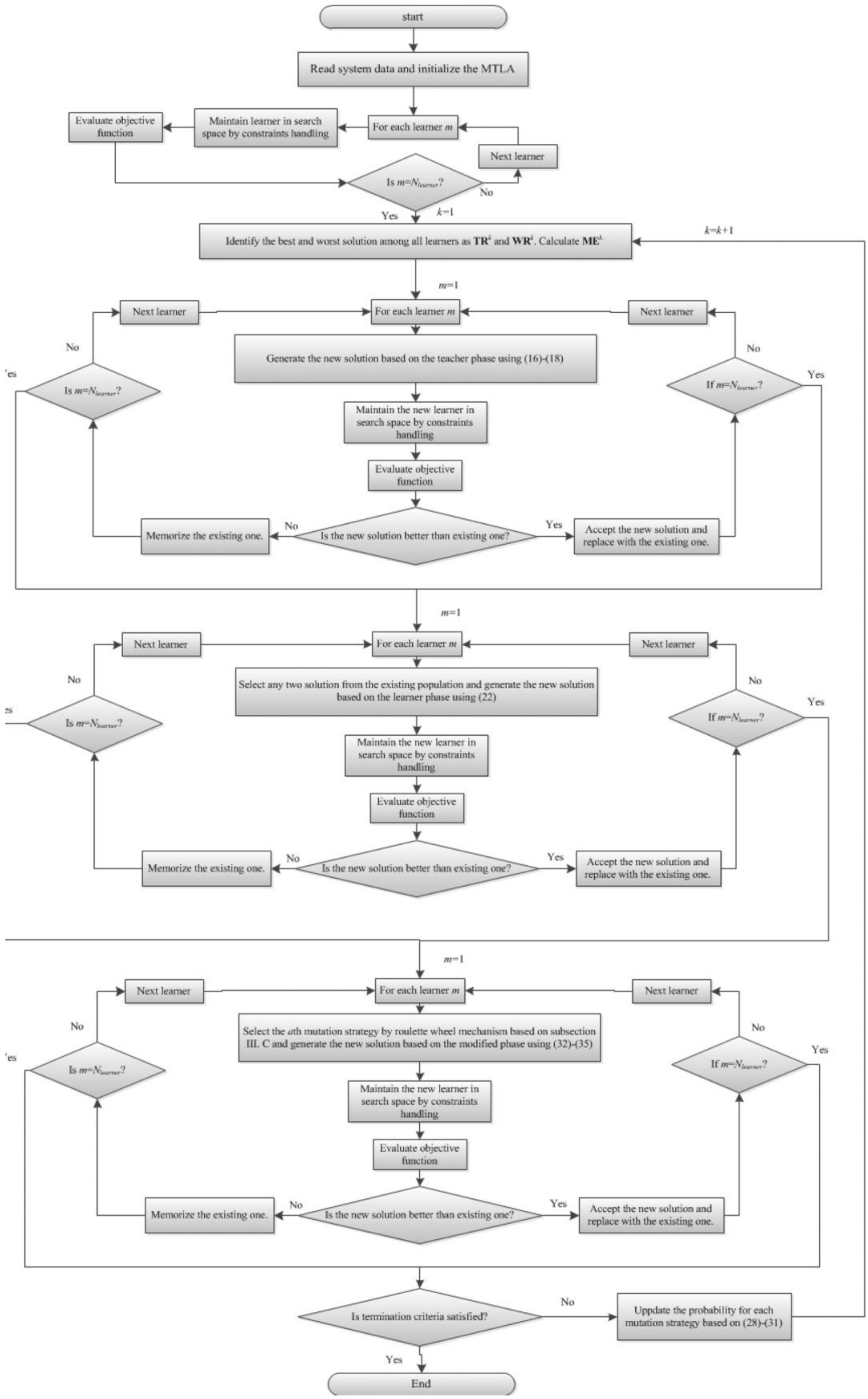

Fig. 1. Flowchart of the proposed MTLA method.

Step 8: *Checking the convergence criteria*; if the convergence criteria are satisfied, terminate the optimization process and select the best learner denoted by $\mathbf{TR}^{k_{\max}}$ as the optimal solution, otherwise return to step 4 and repeat the process.

B. Pseudo-Codes of the Proposed Algorithm

Input all required data.

1. **Initialization:**

For $m = 1$ **to** N_{learner}

For $t = 1$ **to** NT

Generate $\mathbf{P}^1_{m,t}$ randomly while satisfying(2)–(11).

End For (it refers to index t)

Calculate $F(\mathbf{P}^1_{G,m})$ from (1).

End For (it refers to index m)

Initialize:

$\mathbf{TR}^1$: The learner with the best fitness value among all $\mathbf{P}^1_{G,m}$;

$\mathbf{WR}^1$: The learner with the worst fitness value among all $\mathbf{P}^1_{G,m}$;

$k = 1$;

2. **While** $k \leq k_{\max}$

Update the teaching factor TF^k using (19).

Update the mean matrix $\mathbf{ME}^k$ of all existing learners using (13), (14).

Teacher phase

For $m = 1$ **to** N_{learner}

Adapt learner based on the teacher matrix $\mathbf{TR}^k$ using (16)–(18) to generate$\mathbf{P}^k_{G,m,\text{new1}}$.

If the new solution is better than the existing one;

Accept $\mathbf{P}^k_{G,m,\text{new1}}$ and replace $\mathbf{P}^k_{G,m}$;

Else

Memorize $\mathbf{P}^k_{G,m}$.

End If

End For (it refers to index m)

Learner phase

For $m = 1$ **to** N_{learner}

Select the two best learners $n_1 \neq n_2 \neq m$ from existing class.

Adapt learner using (22) to generate $\mathbf{P}^k_{G,m,\text{new2}}$.

If the new solution is better than the existing one;

Accept $\mathbf{P}^k_{G,m,\text{new2}}$ and replace $\mathbf{P}^k_{G,m}$;

Else

Memorize $\mathbf{P}^k_{G,m}$.

End If

End For (it refers to index m)

Modified phase

For $m = 1$ **to** N_{learner}

Select the ath mutation strategy by roulette wheel mechanism based on Section III-C and calculate the modification operator using (32)–(35) to generate $\mathbf{P}^k_{G,m,\text{new3}}$.

If the new solution is better than the existing one;

Accept $\mathbf{P}^k_{G,m,\text{new3}}$ and replace $\mathbf{P}^k_{G,m}$;

Else

Memorize $\mathbf{P}^k_{G,m}$.

End If

End For (it refers to index m)

Update the learner for the next iteration.

Update $Acum^k_a$ and Prob^k_a for the next iteration based on (28)–(31).

Determine $\mathbf{TR}^k$ and $\mathbf{WR}^k$

$k = k + 1$;

End While (it refers to index k)

3. **Return** the final teacher found.

C. Tool Usage

The proposed tool can be used at the beginning of each period based on the rolling window information system. Thus, the impact of all equality and inequality constraints on meeting load demand, transmission losses and SRRs are mitigated for practical systems in real-time applications. For illustrative and comparative purposes, consider the time period of one day with an hourly time step. In each time horizon, for each time interval, the system demand, SRRs and$B$-loss coefficient should be updated and a new RCDED should be run while taking into account the power outputs in the previous hour and the ramping rate limits. Consequently, to handle the aforementioned equality and inequality constraints, implementing the proposed tool by the user to cope with the RCDED problem, which result in the optimal dispatching matrix of units over the 24-h, is of vital importance.

V. Case Studies

In this section, the proposed method is applied to four case studies to comprehensively investigate the RCDED problem.

A. Description of the Case Studies

Case I: The first case consists of five thermal units considering the transmission losses. Here, the cost coefficients, generation limits, ramp-rate limit of units, forecasted load demand for 24 h and the B loss coefficient of the system considering valve-point loading effect are considered [6].

Case II: The second case is a ten-unit network, which is investigated with and without transmission losses. Here, the system data are mainly derived from [19].

Case III: The third case is obtained by tripling the number of units in the previous case.

Case IV: In order to measure better, the performance of the proposed approach, the scalability study is conducted.
The large-scale 40-unit and 140-unit Korean test system are selected for this goal. The systems data are taken from [28].
It should be noted that the B loss coefficients are assumed to stay unchanged over the time horizon. Moreover, the 60 minute SRRs is set to 5% of load demand in each hour for all of the above case studies with time period 24-hour.
In addition, the 10-min SRRs must be set to (1/6) $\times$ (5%) load demand. In order to show the capability of the proposed approach this quantity is set to (2/6) $\times$ (5%) load demand.

B. Parameter Setting

It should be noted that the simulations are carried out on a Pentium P4, Core 2 Duo 2.4-GHz personal computer with 1 GB of RAM memory. Also, the setup for the proposed algorithm is as following. The numbers of the population are equal to 10, 20, 50, 30, and 30 for the test cases 5, 10, 30, 40, and 140 units, respectively. Maximum number of the iterations is 200 for all of the aforementioned test systems.

It should be noted that the performance of the other meta-heuristic optimization algorithms highly depends on tuning their different parameters. A small change in the parameters may result in a large change in the solution obtained by these algorithms. For instance, PSO [11] requires learning factors, variation of weight, and maximum value of velocity. As mentioned before, TLA is a powerful algorithm, which is free from adjusting the parameters. TLA works in such a way that it only requires the population size and the maximum number of iterations. In other words, this algorithm reaches the optimal solution with adjusting two parameters and this feature is the major advantage and superiority of the algorithm.

C. Computational Result and Comparison

Firstly, in order to show the satisfying performance of MTLA over the other renowned algorithms, the complete comparison and empirical studies between the algorithms' convergence is carried out. For each case study, the value of the total fuel cost is extracted in 30 independent runs and the statistical information including the best, the worst and the average of the solutions as well as the Average CPU time are evaluated. It should be emphasized that the worst value of the total fuel cost is better than the best solution of all other methods in all test cases. This is another advantage of the proposed method that illustrates the superiority of the suggested MTLA over other techniques. Comparing the best solutions and the mean values obtained by different methods, it can be inferred that MTLA is a more powerful algorithm than other ones for finding the optimal solution. Moreover, in this study, the successful percentage and error are implemented to show the robustness of the proposed approach. The successful percentage can be defined as the number of successful runs which converge to the best solution divided by all runs (30 runs). Beside, the error is the average difference between the obtained best solution and the global solution, which indicates the ability of each technique to reach the global optimum solution.

In order to investigate the effectiveness of the self-adaptive learning mechanism of MTLA, the execution probabilities of all four separate mutation strategies on the case study IV are plotted. To obtain each probability curves, the MTLA have been run for 30 trials. The most suitable strategy during the search procedure should yield the largest probability.

The convergence graphs are also plotted to inspect the quality of the best solution over the evolution process.

The average CPU time is highly dependent on the computer systems used for other experiments, which have been reported in the literatures. Hence, the scaled CPU time is calculated by the per-unit CPU speed multiplied by the given Average CPU time for each of the mentioned techniques [8]. The per-unit base speed is 2.4 GHz and the scaled CPU time is as follows:

$$\text{scaled CPU time} = \frac{\text{given CPU speed}}{2.4\ \text{GHz}} \times \text{given average CPU time}. \quad (39)$$

The average CPU time of each optimization method is very important for its application to real problems. Comparing the scaled CPU times for different methods, it can be illustrated that the proposed technique is faster than other methods. This is another advantage of the proposed MTLA.

In each algorithm, the optimum population size is found to be related to the problem dimension and complexity. A change in the population size, affects the performance of the MTLA algorithm but its effect is not noticeable since this method achieves the optimum solution with a few number of population. The results are presented in the following.

Case I: RCDED Problem for 5-Unit Test System: As mentioned before, in this case, the valve-point loading effect is considered. The problem is solved for both conditions (considering and neglecting losses). The complete comparison between the performance of the proposed method and those of other well-known algorithms are shown in Tables I and II, for with and without losses, respectively. It can be seen that the best, the worst and the average value of the total fuel cost are ($43 084.4 and $42 688.2), ($43 199.5 and $42 688.2) and ($43 167.6 and $42 688.2) for with and without losses, respectively. The self-adaptive probabilistic characteristics of the proposed approach are analyzed using the candidate strategies in the pool separately to solve the problem.

The proposed MTLA with the aid of multiple mutation strategies in a parallel way can benefit from both global and local search characteristics. So, in each generation, the MTLA can produce diverse solution even with a small population and less maximum iteration number. Besides, the MTLA with self-adaptive probabilistic mutation operators can better manage transition from each generation to the next one in comparison with

TABLE I
RESULTS OBTAINED BY DIFFERENT METHODS FOR CASE I, II, AND III WITHOUT LOSS

Solution technique	Total fuel cost ($) Best value	Mean value	Worst value	Scaled CPU time (min)
Case I				
TLA	42,821.6	42,900.4	42,963.3	0.038
TLA-method1	42,719.2	42,733.2	42,756.0	0.059
TLA-method4	42,711.6	42,729.7	42,751.1	0.058
TLA-method3	42,701.9	42,709.4	42,722.6	0.055
TLA-method2	42,696.4	42,700.8	42,713.4	0.054
MTLA	**42,688.2**	**42,688.2**	**42,688.2**	**0.052**
Case II				
SQP [17]	1,051,163	NA	NA	0.421
EP [17]	1,048,638	NA	NA	15.049
EP-SQP [17]	1,031,746	1,035,748	NA	7.264
MDE [8]	1,031,612	1,033,630	NA	4.417
HQPSO [20]	1,031,559	1,033,837	1,036,681	0.773
PSO-SQP [18]	1,030,773	1,031,371	1,053,983	6.364
MHEP-SQP [19]	1,028,924	1,031,179	NA	21.23
DGPSO [11]	1,028,835	1,030,183	NA	4.809
PSO-SQP (C) [18]	1,027,334	1,028,546	1,033,983	7.219
IPSO [12]	1,023,807	1,026,863	NA	0.050
AIS [15]	1,021,980	1,023,156	1,024,973	25.346
AHDE [9]	1,020,082	1,022,476	NA	1.10
CDE method3 [10]	1,019,123	1,020,870	1,023,115	0.32
ICPSO [14]	1,019,072	1,020,027	NA	0.350
TLA	1,019,925	1,020,411	1,021,118	0.049
TLA-method1	1.017,820	1,018,243	1,018,803	0.091
TLA-method4	1,017,697	1,018,052	1,018,418	0.084
TLA-method3	1,017,164	1,017,331	1,017,536	0.076
TLA-method2	1,017,050	1,017,225	1,017,395	0.072
MTLA	**1,016,935**	**1,016,972**	**1,017,091**	**0.065**
Case III				
EP [19]	3,164,531	3,200,171	NA	NA
EP-SQP [19]	3,159,204	3,169,093	NA	NA
MHEP-SQP [19]	3,151,445	3,157,738	NA	NA
DGPSO [11]	3,148,992	3,154,438	NA	22.816
IPSO [12]	3,090,570	3,071,588	NA	0.142
CDE method 3 [10]	3,083,930	3,090,542	NA	0.67
ICPSO [14]	3,064,497	3,071,588	NA	0.773
TLA	3,089,802	3,096,729	3,101,300	0.108
TLA-method1	3,058,702	3,060,996	3,066,644	0.153
TLA-method4	3,057,980	3,060,633	3,065,721	0.150
TLA-method3	3,050,834	3,051,569	3,052,065	0.139
TLA-method2	3,050,098	3,051,548	3,051,994	0.136
MTLA	**3,048,609**	**3,049,871**	**3,051,113**	**0.127**

NA: Not available in the literature

TABLE II
RESULTS OBTAINED BY DIFFERENT METHODS FOR CASE I AND II WITH LOSS

Solution technique	Total fuel cost ($) Best value	Mean value	Worst value	Scaled CPU time (min)
Case I				
SA [6]	47,356	NA	NA	4.395
APSO [13]	44,678	NA	NA	NA
AIS [15]	44,385.4	44,758.8	45,553.8	5.333
TLA	43,645.2	43,800.4	43,897.4	0.060
TLA-method1	43,132.9	43,209.4	43,326.3	0.096
TLA-method4	43,105.2	43,192.7	43,288.4	0.091
TLA-method3	43,091.5	43,179.8	43,248.1	0.081
TLA-method2	43,084.3	43,167.6	43,199.5	0.078
MTLA	**43,048.4**	**43,077.9**	**43,128.5**	**0.071**
Case II				
EP [19]	1,054,685	1,057,323	NA	47.23
EP-SQP [19]	1,052,668	1,053,771	NA	27.53
IPSO [12]	1,046,275	1,048,154	NA	0.150
AIS [15]	1,045,715	1,047,050	1,048,431	30.973
TLA	1,045,327	1,046,432	1,047,679	0.083
TLA-method1	1,038,547	1,038,913	1,039,736	0.155
TLA-method4	1,038,126	1,038,587	1,038,965	0.147
TLA-method3	1,037,943	1,038,177	1,038,511	0.131
TLA-method2	1,037,898	1,038,060	1,038,199	0.124
MTLA	**1,037,489**	**1,037,712**	**1,038,090**	**0.111**

NA: Not available in the literature

each separate strategy. It is necessary to note that in each separate method, i.e., TLA-method1, TLA-method2, TLA-method3, and TLA-method4, one mutation technique is implemented for all of the output solutions of the second phase (learner phase). In an attempt to improve the modification process more effectively, a new self-adaptive mutation strategy is used such that the output solutions of the second phase (learner phase) would be improved in the third phase, i.e., modified phase. The added phase gets use of four mutation operations in parallel to enhance the ability of the algorithm for both local and global search exploration adequately. It is obvious that the proposed method has provided better results in terms of the total fuel cost and the scaled CPU time. In Table III, the best dispatch result found by MTLA is shown to consider transmission losses condition and to compare with that of AIS [15] as the earlier report. All the changed generation dispatches have been bolded. One can observe that the generation outputs of many units by MTLA are quite different from those of AIS [15]. This implies that the global searching capability has been improved extensively by the proposed MTLA mechanism. The corresponding total loss is 194.5457 MW which is slightly larger than the transmission losses of AIS [15], i.e., 193.4334 MW. To demonstrate simultaneous handling of three types of the SRRs by MTLA, $\Delta_{m,H}^{(1)}$, $\Delta_{m,H}^{(2)}$, and $\Delta_{m,H}^{(3)}$ are also computed and added to this Table. It should be noted that to satisfy the SRRs constraints, the backward procedure occurs at the peak load demand hours in the next day, i.e., hours 11 am, 12, and 20 pm.

Case II: RCDED Problem for 10-Unit Test System: As mentioned before, this case is the ten-unit network, which is investigated in two conditions. Firstly, the transmission losses are neglected and the best total fuel cost is evaluated as $1 016 935. To check whether the constraints of the problem are satisfied or not, the detailed dispatch results of the MTLA solution are given in Table IV and compared with that of ICPSO [14] as the earlier report. All the changed generation dispatches of units have been bolded. One can observe that the generation outputs of many units by MTLA are quite different from those of ICPSO [14]. To demonstrate simultaneous handling of three types of the SRRs by MTLA, $\Delta_{m,H}^{(1)}$, $\Delta_{m,H}^{(2)}$, and $\Delta_{m,H}^{(3)}$ are also calculated and added to this Table. It is necessary to note that in this case study, the backward procedure occurs at the peak load demand hours in the next day, i.e., hours 10 am, 11 am, 12, and 20 pm to satisfy the SRRs constraints. Secondly, the effect of the transmission losses is considered. The best value of the total fuel cost evaluated by MTLA is $1 037 489. The corresponding total loss is 823.2054 MW which is greatly smaller than the transmission losses of recently reported approach AIS [15], i.e., 835.6200 MW.

In Tables I and II, the comparison between the results evaluated in two different conditions are shown. In Fig. 2, the best convergence performance for TLA, TLA with mutation method1, TLA with mutation method2, TLA with mutation method3, TLA with mutation method4 and MTLA neglecting losses are depicted. This figure indicates that the MTLA consistently converges faster than other separated methods. The computational cost of the proposed approach is compared with

TABLE III
COMPARISON OF BEST DISPATCH FOUND BY MTLA (FIRST ROWS IN EACH HOUR) AND AIS [15] (SECOND ROWS IN EACH HOUR) FOR CASE I—WITH LOSS

Hour	Load (MW)	P_1 (MW)	P_2 (MW)	P_3 (MW)	P_4 (MW)	P_5 (MW)	P_{loss} (MW)	$\Delta_t^{(1)}$ (MW)	$\Delta_t^{(2)}$ (MW)	$\Delta_t^{(3)}$ (MW)
1	410	**20.6029**	**98.5427**	**30.0002**	124.9100	**139.7598**	3.8156	490.6844	175.9573	26.5000
		12.5000	42.5000	95.2344	124.2884	138.9718	3.4946			
2	435	**10.0000**	**97.9633**	**66.4963**	**124.9069**	**139.7599**	4.1264	464.1236	175.2867	26.0833
		42.27800	20.0040	112.2875	125.7028	138.6054	3.8777			
3	475	**10.0362**	**98.5259**	**106.4963**	**124.9518**	**139.7721**	4.7823	421.4677	172.7241	25.4168
		72.2780	26.3907	114.2580	126.1687	140.5437	4.6390			
4	530	**10.0017**	**98.5814**	112.7089	**174.9518**	**139.7705**	6.0143	362.4857	169.9186	24.5000
		75.0000	31.1875	112.7300	176.1687	140.7643	5.8504			
5	558	**10.0000**	**92.5463**	**112.6655**	**209.8149**	**139.7288**	6.7555	332.3445	162.2851	24.0333
		74.9988	24.3688	113.5320	211.1969	140.4977	6.5942			
6	608	**10.0000**	**98.5409**	**112.6747**	**209.8169**	**184.9591**	7.9916	278.6084	156.2422	23.2000
		75.0000	25.0925	114.9903	210.2393	190.4977	7.8198			
7	626	**10.0000**	**72.4503**	**112.6740**	**209.8158**	**229.5194**	8.4595	259.2405	158.8842	22.9000
		64.1335	20.0615	111.7384	210.5317	227.9034	8.3686			
8	654	**12.7044**	**98.5437**	**112.6743**	209.8158	229.5195	9.2577	229.0423	153.9405	22.4333
		75.0000	34.7330	114.2589	209.7960	229.3024	9.0903			
9	690	**42.7044**	**105.4542**	**112.6743**	**209.8160**	**229.5195**	10.1684	190.3316	145.2298	21.8333
		74.9199	64.7330	117.5007	211.4532	231.4786	10.0853			
10	704	**64.0108**	**98.5398**	112.6735	209.8158	**229.5196**	10.5595	175.2405	132.4336	21.6000
		70.5373	94.7330	112.2591	209.5795	227.4356	10.5444			
11	720	**75.0000**	**104.0359**	**112.6735**	**209.8158**	229.5196	11.0448	157.9552	115.1483	16.3333
		74.9984	100.6899	115.5328	210.1738	229.6279	11.0228			
12	740	75.0000	124.7111	**112.6735**	209.8158	229.5196	11.7200	136.2800	93.4731	11.2889
		75.0000	124.0393	113.5686	209.7999	229.3043	11.7122			
13	704	**64.0108**	**98.5398**	112.6735	**209.8158**	**229.5196**	10.5595	175.2405	132.4336	21.6000
		67.8957	97.5603	112.6091	207.6484	228.8343	10.5477			
14	690	**49.6196**	**98.5398**	112.6735	209.8158	**229.5196**	10.1683	190.3317	147.5248	21.8333
		47.0756	100.5859	112.5156	209.1798	230.8228	10.1798			
15	654	**19.6196**	**91.5856**	**112.6734**	209.8158	229.5196	9.2140	229.0860	157.4842	22.4333
		17.0756	98.5570	108.4764	209.7412	229.4111	9.2613			
16	580	10.0000	75.1865	**112.6734**	159.8158	229.5196	7.1953	308.8047	171.0000	23.6667
		10.0000	75.9723	111.8815	159.8231	229.5225	7.1994			
17	558	10.0000	**87.5823**	112.6735	124.9078	229.5189	6.6825	332.4175	172.1000	24.0333
		10.0574	88.1689	112.1409	124.7947	229.5235	6.6853			
18	608	**10.0000**	**98.5403**	**112.6759**	**165.2142**	**229.5196**	7.9500	278.6500	166.0597	23.2000
		40.0562	106.1047	113.4525	125.7050	230.5613	7.8796			
19	654	**12.7080**	**98.5407**	**112.6735**	**209.816**	229.5196	9.2578	229.0422	153.9433	22.4333
		70.0562	124.9989	113.6349	125.1984	229.2400	9.1284			
20	704	**42.7078**	**119.9405**	112.6735	**209.8158**	229.5196	10.6572	175.1428	130.0437	21.6000
		75.0000	122.5644	112.5402	175.1463	229.3045	10.5554			
21	680	**39.3528**	**98.5399**	**112.6735**	209.8158	229.5196	9.9016	201.0984	152.6443	22.0000
		45.0000	94.7048	111.0462	209.7771	229.3600	9.8881			
22	605	**10.0001**	98.5399	**112.6735**	**162.1377**	**229.5196**	7.8708	281.8792	166.2101	23.2500
		15.0000	98.5583	111.1507	159.8228	228.3217	7.8535			
23	527	10.0000	98.5386	**112.6733**	**124.9077**	**186.7862**	5.90580	365.7442	170.1114	24.5500
		10.0000	98.8304	71.1507	123.6987	229.4886	6.1684			
24	463	10.0000	**80.1510**	**112.6711**	124.9056	**139.7596**	4.4873	434.3627	176.8500	25.6167
		10.0000	73.6784	31.1507	124.8491	228.3089	4.9872			

those of other methods such as ICPSO [14] and HQPSO [20] in Table V. This table shows that the proposed method reaches a lower total fuel cost than [14] ($1 016 935 opposed to $1 019 072) in lower maximum number of iterations, population size, algorithm parameters, power mismatch and scaled CPU time. Moreover, although [20] is in the list of latest hybrid evolutionary algorithms which solve the DED problem, but the result of the proposed MTLA approach is much superior than [20].

To evaluate the effect of population size on the performance of MTLA, different population sizes are selected and the RCDED problem is solved in 30 independent runs on the 10-unit neglecting losses. Table VI shows the statistical information and frequency of convergence for 10, 20, 50, and 100 population sizes. The population size of 20 achieves optimal solutions more consistently for the 10-unit test system.

The successful percentage for all the methods and the proposed MTLA technique to solve RCDED problem considering losses is listed in Table VII. When the complexity of the problem increases, its overall successful percentage decreases, especially for non-convex and non-smooth problems. But the results of Table VII show that the MTLA is more successful in finding satisfactory solution in comparison to the other methods.

Case III: RCDED Problem for 30-Unit Test System: In order to show the efficiency of MTLA in solving medium-scale nonlinear problems, the 30-unit test system is produced by tripling the number of units in the system and the load demand is also tripled for the next 24 hours. It can be seen from Table I that the best total fuel cost is $3 048 609, which is much better and superior than the results of other methods. This is also important because as the size of the system increases, the differences between the methods seem to decrease, but the solution obtained by the MTLA is far better than other methods. It should be pointed out that the backward procedure occurs at the peak load demand hours in the next day similar to the previous case study.

Case IV: ED Problem for 40-Unit and 140-Unit Test Systems: From the point of view of meta-heuristic optimization and similar approaches, the test system of 5, 10, or 30 units is not large

TABLE IV
COMPARISON OF BEST DISPATCH FOUND BY MTLA (FIRST ROWS IN EACH HOUR) AND ICPSO [14] (SECOND ROWS IN EACH HOUR) FOR CASE II—WITHOUT LOSS

Hour	Load (MW)	P_1 (MW)	P_2 (MW)	P_3 (MW)	P_4 (MW)	P_5 (MW)	P_6 (MW)	P_7 (MW)	P_8 (MW)	P_9 (MW)	P_{10} (MW)	$\Delta_t^{(1)}$ (MW)	$\Delta_t^{(2)}$ (MW)	$\Delta_t^{(3)}$ (MW)
1	1036	150.0000	**309.5330**	**73.0000**	60.0000	73.0000	**118.8766**	129.5904	47.0000	20.0000	55	1270.2000	389.7330	58.14293
		150.0000	135.0000	243.9597	60.0000	73.0000	122.4498	129.5904	47.0000	20.0000	55			
2	1110	150.0000	**316.8232**	**86.2700**	60.0000	**122.8666**	122.4498	129.5904	47.0000	20.0000	55	1192.5000	382.4598	56.9096
		150.0000	135.0000	317.9780	60.0000	73.0000	122.4329	129.5891	47.0000	20.0000	55			
3	1258	**150.0000**	**396.7992**	**154.2938**	60.0000	122.8666	122.4500	129.5904	47.0000	20.0000	55	1037.1000	358.2604	54.4429
		226.6175	215.0000	259.4354	60.0000	122.8624	122.4944	129.5904	47.0000	20.0000	55			
4	1406	**226.6242**	**396.8021**	**225.6667**	60.0000	**122.8666**	122.4500	129.5904	47.0000	20.0000	55	881.7000	350.8575	51.9763
		303.2483	295.0000	300.7107	60.0000	73.0000	122.4493	129.5917	47.0000	20.0000	55			
5	1480	**226.6242**	**396.8186**	**299.6503**	60.0000	**122.8666**	122.4499	129.5904	47.0000	20.0000	55	804.0000	307.4908	50.7429
		379.8391	309.5242	283.6138	60.0000	73.0000	122.4493	129.5736	47.0000	20.0000	55			
6	1628	**303.2484**	**396.8007**	**321.1755**	60.0000	**172.7350**	122.4500	129.5904	47.0000	20.0000	55	648.6000	278.5834	48.2763
		456.4968	309.5329	305.0356	60.0000	122.8636	122.4773	129.5937	47.0000	20.0000	55			
7	1702	**379.8726**	**396.8005**	**318.5524**	60.0000	**172.7331**	**122.4510**	129.5904	47.0000	20.0000	55	570.9000	277.5057	47.0429
		456.4864	309.5328	297.3992	60.0000	172.7196	154.2715	129.5904	47.0000	20.0000	55			
8	1776	**379.8726**	**396.7994**	297.3857	**105.3023**	**222.5996**	122.4500	129.5904	47.0000	20.0000	55	493.2000	265.3749	45.8096
		456.4962	389.5328	297.4101	85.7665	172.7358	122.4681	129.5904	47.0000	20.0000	55			
9	1924	456.4968	396.7994	**297.6793**	**146.3842**	222.5997	**122.4502**	129.5904	**77.0000**	20.0000	55	337.8000	191.1842	43.3429
		456.5000	396.8009	305.0732	131.4385	222.5957	160.0000	129.5918	47.0000	20.0000	55			
10	2072	456.4968	**396.7994**	**324.2324**	**191.9169**	222.5997	160.0000	129.5904	**85.3644**	**50.0000**	55	182.4000	119.6813	32.5429
		456.4968	460.0000	340.0000	181.2430	222.6697	160.0000	129.5905	47.0000	20.0000	55			
11	2146	**456.4968**	**396.7994**	340.0000	**241.9169**	**228.8052**	160.0000	129.5904	**85.3342**	**52.0571**	55	104.7000	91.9511	17.9763
		460.1662	460.0000	340.0000	231.2430	243.0000	160.0000	129.5907	47.0000	20.0000	55			
12	2220	456.4968	460.0000	**308.0258**	**291.0550**	**222.5997**	160.0000	129.5904	**85.3095**	**51.9225**	55	27.0000	22.3095	16.7429
		456.4971	460.0000	340.0000	281.2422	240.6727	160.0000	129.5912	47.0000	49.9967	55			
13	2072	456.4968	**396.7994**	**303.0122**	241.2669	222.5997	**160.0000**	129.5904	**85.3121**	**21.9225**	55	182.4000	140.9015	32.5429
		456.4979	460.0000	317.6239	241.2363	222.5997	122.4512	129.5910	47.0000	20.0000	55			
14	1924	456.4968	396.7994	**294.3514**	191.2669	172.7331	122.4499	129.5904	**85.3121**	20.0000	55	337.8000	224.1121	43.3429
		456.4922	396.7994	332.6988	191.2363	172.7330	122.4498	129.5905	47.0000	20.0000	55			
15	1776	379.8726	396.7994	**290.8021**	**180.7354**	122.8657	122.4499	129.5904	**77.8845**	20.0000	55	493.2000	301.5582	45.8096
		379.8725	396.8003	337.7821	164.5865	122.8727	122.4911	129.5930	47.0000	20.0016	55			
16	1554	303.2484	396.7994	**275.2920**	**130.7354**	73.0000	122.4499	129.5904	47.8845	20.0000	55	726.3000	328.1683	49.5096
		303.2262	396.7978	292.3923	114.5865	73.0000	122.4066	129.5906	47.0000	20.0000	55			
17	1480	**226.6242**	**396.7994**	**289.1154**	**120.4206**	73.0000	122.4500	129.5904	47.0000	20.0000	55	804.0000	318.0448	50.7429
		379.8725	316.7978	271.7026	64.5865	73.0000	122.4499	129.5906	47.0000	20.0000	55			
18	1628	**303.2485**	396.7994	**310.6300**	**120.4153**	122.8666	122.4498	129.5904	47.0000	20.0000	55	648.6000	289.1304	48.2763
		379.8726	396.7978	294.4352	60.0000	122.8543	122.4497	129.5903	47.0000	20.0000	55			
19	1776	**379.8726**	396.7994	**299.6038**	**122.9507**	**172.7331**	122.4500	129.5904	**77.0000**	20.0000	55	493.2000	292.7564	45.8096
		456.4309	396.7993	316.0057	60.0001	172.7232	122.4483	129.5926	47.0000	20.0000	55			
20	2072	**459.6143**	**396.7994**	340.0000	**172.9507**	222.7331	160.0000	129.5904	**85.3121**	50.0000	55	182.4000	100.6628	16.2620
		467.8096	460.0000	340.0000	110.0001	222.6000	160.0000	129.5905	77.0000	49.9999	55			
21	1924	456.4968	**390.0205**	**319.6069**	**122.9507**	222.5996	**122.4230**	129.5904	**85.3121**	20.0000	55	337.8000	176.0628	43.3429
		456.5001	389.5686	323.3630	120.3910	222.5866	160.0000	129.5907	47.0000	20.0000	55			
22	1628	379.8726	**310.0205**	**280.0708**	**72.9507**	172.7331	122.4498	129.5904	**85.3121**	20.0000	55	648.6000	336.4890	48.2763
		379.8725	309.5692	321.4019	70.3910	172.7264	122.4486	129.5904	47.0000	20.0000	55			
23	1332	303.2484	**230.0205**	**203.5025**	60.0000	122.8666	122.4498	129.5904	**85.3218**	20.0000	55	959.4000	371.3598	53.2096
		303.2484	229.5731	242.2786	60.0000	122.8594	122.4500	129.5905	47.0000	20.0000	55			
24	1184	226.6242	**222.2665**	**189.6787**	60.0000	73.0000	122.4498	129.5904	**85.3904**	20.0000	55	1114.8000	378.7598	55.6763
		226.6243	149.5731	300.7614	60.0000	73.0006	122.4503	129.5903	47.0000	20.0000	55			

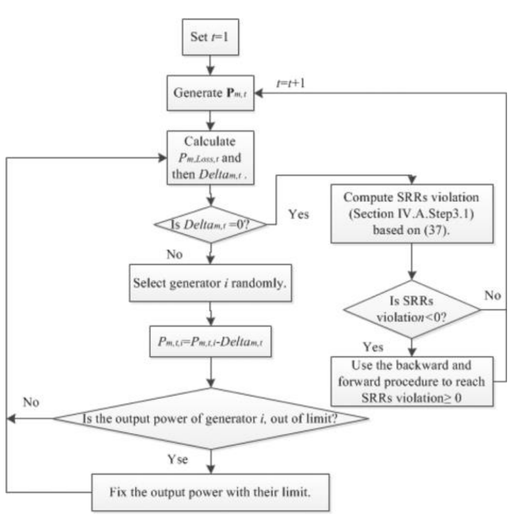

Fig. 2. Flowchart of the constraints handling.

TABLE V
COMPARISON OF ICPSO [14], HQPSO [20] WITH MTLA FOR CASE II WITHOUT LOSSES

Method	No. of algorithm parameters	Population size	Maximum iteration	Power mismatch (MW)	Scaled CPU time (min)
ICPSO [14]	9	100	1,200	0.00178	0.350
HQPSO [20]	7	50	150	0.50000	0.773
MTLA	**2**	**20**	**150**	**0.00000**	**0.065**

TABLE VI
EFFECT OF POPULATION SIZE ON CASE II—WITHOUT LOSS

Population size	No. of hits to ($1016900-$1017400)	Total fuel cost ($)			Average CPU time (min)
		Best value	Mean value	Worst value	
10	23	1,017,155	1,017,278	1,017,464	0.051
20	**30**	**1,016,935**	**1,016,972**	**1,017,091**	**0.065**
50	30	1,016,935	1,017,123	1,017,314	0.102
100	30	1,016,935	1,017,144	1,017,335	0.189

enough to demonstrate the scalability of the proposed approach. Therefore, the 40-unit test system is selected while valve-point effects are considered. In this test case, load demand is 10 500 MW. The best fuel cost for this test case reported until now is $121 403.5362 [28] while it seems to be wrong. The true value based on the results of this paper is equal to $121 412.5483. The static ED (with study horizon 1h) is handled by the proposed approach and the results have been shown in Table VIII. The

TABLE VII
SUCCESSFUL PERCENTAGE (%) OF DIFFERENT METHODS FOR CASE II—WITH LOSS OUT OF 30 TRAIL RUNS

Solution technique	1,037,489-1,037,889	1,037,889-1,038,289	1,038,289-1,038,689	1,038,689-1,039,089	1,039,089-1,039,489
TLA	0	0	0	0	0
TLA-method1	0	0	10%	56.7%	33.3%
TLA-method4	0	13.3%	46.7%	40%	0
TLA-method3	0	60%	40%	0	0
TLA-method2	0	100%	0	0	0
MTLA	**66.7%**	**33.3%**	**0**	**0**	**0**

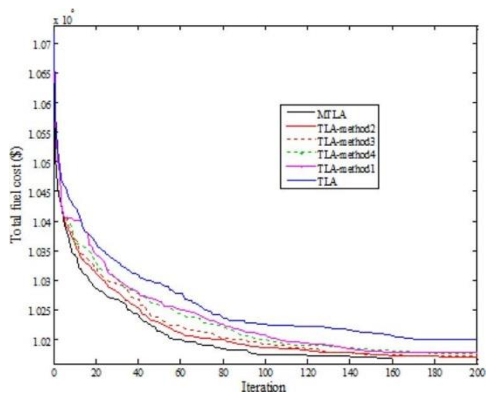

Fig. 3. Convergence graphs of MTLA, TLA-method1, TLA-method2, TLA-method3, TLA-method4, and original TLA for case II.

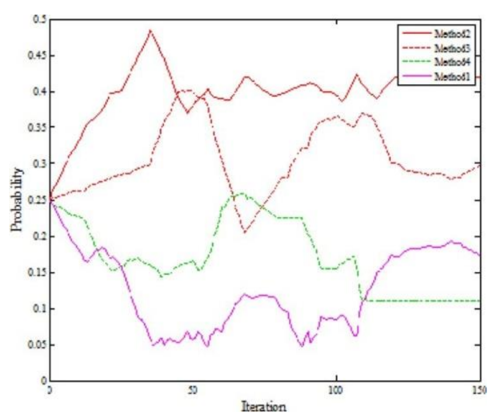

Fig. 4. Evolution of probability for mutation strategies 1, 2, 3, and 4 in Case IV—40 unit.

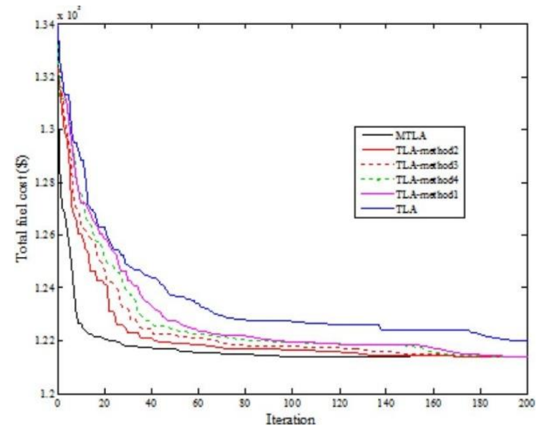

Fig. 5. Convergence graphs of MTLA, TLA-method1, TLA-method2, TLA-method3, TLA-method4, and original TLA for case IV—40 unit.

TABLE VIII
RESULTS OBTAINED BY DIFFERENT METHODS FOR CASE IV

Solution technique	Total fuel cost ($)			Scaled CPU time (min)
	Best value	Mean value	Worst value	
	40unit			
DABFA [32]	123,027.9674*	-	-	0.074
HQPSO [20]	122,318.6058*	-	-	NA
CTPSO [28]	121,703.5133*	-	-	0.264
QPSO [31]	121,448.2100	122,225.0700	NA	NA
CSPSO [28]	121,444.9621*	-	-	0.264
BBO [29]	121,426.9530	121,508.0325	121,688.6634	1.757
RCGA [30]	121,424.4374*	-	-	1.818
COPSO [28]	121,420.9086*	-	-	0.267
DE/BBO [29]	121,420.8948	121,420.8952	121420.8963	0.958
CCPSO [28]	121,412.5483*	-	-	0.268
TLA	122,009.7664	122,074.9032	122,171.5600	0.027
TLA-method1	121,420.8958	121,427.6669	121,443.2010	0.041
TLA-method4	121,416.7159	121,421.2654	121,431.9027	0.038
TLA-method3	121,412.5364	121,412.5843	121,412.6504	0.036
TLA-method2	121,412.5355	121,412.5359	121,412.5365	0.035
MTLA	**121,412.5355**	**121,412.5355**	**121,412.5355**	**0.032**
	140unit			
CCPSO [28]	1,657,962.7300	1,657,962.7300	1,657,962.7300	2.083
COPSO [28]	1,657,962.7300	1,657,962.7300	1,657,962.7300	2.083
CSPSO [28]	1,657,962.7300	1,657,962.7400	1,657,962.8500	1.250
CTPSO [28]	1,657,962.7300	1,657,964.0600	1,658,002.7900	1.389
TLA	1,660,362.2313	1,661,997.8770	1,664,432.5688	0.034
TLA-method1	1,657,951.9053	1,657,952.8345	1,657,953.3559	0.044
TLA-method4	1,657,951.9053	1,657,952.4066	1,657,952.8802	0.043
TLA-method3	1,657,951.9053	1,657,951.9137	1,657,952.0118	0.041
TLA-method2	1,657,951.9053	1,657,951.9053	1,657,951.9053	0.041
MTLA	**1,657,951.9053**	**1,657,951.9053**	**1,657,951.9053**	**0.038**

NA: Not available in the literature

* Exact total fuel costs from the schedule of [20], [28], [30] and [32] are used in this paper which reported in the lower value in these literatures.

proposed method is compared with other methods in the area such as HQPSO [20], CCPSO [28], COPSO [28], CSPSO [28], CTPSO [28], BBO [29], DE/BBO [29], RCGA [30], QPSO [31], and DABFA [32] to illustrate the superiority of the proposed MTLA. Also, the system with 140 units considering ramp rate limits and valve-point effects are selected as another large scale test case. The total load demand is 49 342 MW. The results are reported in Table VIII. The evolution trend of the probability of each strategy and convergence graph for 40-unit test systems have been illustrated in Figs. 3–5, respectively. It is observed that different strategies of MTLA are working together to obtain higher performance in the final results. Fig. 4 indicates that MTLA is able to adaptively select the most suitable strategy among all of them, without any prior knowledge.

It can converge rapidly to a strategy with higher probability. It is necessary to note that the mutation method2 obtains the highest probability in comparison with other strategies, specially the implemented mutation in [26] and plays a central role in the performance of the MTLA to reach to better results.

To show the superiority of the proposed approach, the error index is implemented and calculated for each optimization methods. The corresponding error of original TLA, TLA-method1, TLA-method4, TLA-method3, TLA-method2 and MTLA are $662.3677, $15.1314, $8.7300, $0.0308, $0.0004, and $0.0000 for 40-unit test system.

According to the results of Tables I, II, V, VI, VII, and VIII,the following conclusions can be observed:

1) When the dimensions of the problems increase, their com-plexity increase consistently, especially for the non-smooth and non-convex function. Thus, the overall successful rateof each algorithm decreases for large-scale problem but the proposed approach obtains the best overall results.
2) For large-scale problem, the higher population diversityis required and, hence, the strategy which provides higherdiversity is able to obtain better performance.
3) Regarding the advantages of the MTLA, which are illus- trated via simulations, it would be a proper choice for real-time applications in practical power systems.

D. Performance of the Inclusion SRRs Constraints

Inclusion of SRRs constraints increases complexity and av- erage CPU time burden to satisfy it. In this paper, three types of the SRRs constraints incorporating the ramp rate limits and power balance considering losses are handled simultaneously without any restriction on the objective function. As it can be seen from (9), the reserve capacity has been included in the constraint as an extra term, i.e., SR_t , in addition to load and loss in each hour. Therefore, this constraint will affect the re- sults of the DED problem. Also, if the constraints (10) and (11) are not satisfied in each hour, the algorithm should be returned to the pervious hours and modified them until satisfaction of these equations. Thus in all of the cases with the fixed number of units, inclusion of the additional SRRs constraints consider-ably affect the solution search domain and the performance of the solution procedure. These lead to a more execution time and total fuel cost. To demonstrate this, the DED is solved for the case study II without considering SRRs constraints. The best production cost and the average CPU time taken by the MTLA method decreases to $1 016 748 and 0.065 min, respectively. It is clear that the SRRs constraints increase the complexity, the number of computations and the total fuel cost. The same justi-fication can be implemented for other case studies.

Only two types of SRRs have been considered in the previous work [18]. Indeed, [18] considers the RCDED problem while the reserve have been added as a penalty term to the fuel cost function. In the presence of the SRRs constraints, the best totalfuel costs are $1 083 973 and $1 071 236 for SRRs constraints (9) and (10), respectively, in 10 unit test system in 12.506 and 16.374 min time. Besides, in the viewpoint of optimization is- sues, increasing the number of constraints will significantly de-crease solution space. From the experiments in Tables I and II,it is clear that for all test cases, the MTLA method is better com- pared to the other methods, in terms of producing better solu- tion and computation time. According to the above tables, the results of MTLA are more comparable with other approaches, since there is a considerable computation time and cost saving by using this method.

REFERENCES

[1] R. A. Jabr, A. H. Coonick, and B. J. Cory, "A homogeneous linear programming algorithm for the security constrained economic dispatch problem," *IEEE Trans. Power Syst.*, vol. 15, no. 3, pp. 930–936, Aug.2000.

[2] C. Chen, "Non-convex economic dispatch: A direct search approach," *Energy Convers. Manage.*, vol. 48, no. 1, pp. 219–225, Jun. 2007.

[3] L. G. Papageorgiou and E. S. Fraga, "A mixed integer quadratic pro- gramming formulation for the economic dispatch of generators with prohibited operating zones," *Elect. Power Syst. Res.*, vol. 77, no. 10, pp. 1292–1296, Aug. 2007.

[4] K. S. Hindi and M. R. A. Ghani, "Dynamic economic dispatch for large-scale power systems: A Lagrangian relaxation approach," *Elect. Power Syst. Res.*, vol. 13, no. 1, pp. 51–56, Feb. 1991.

[5] D. L. Travers and R. J. Kaye, "Dynamic dispatch by constructive dy- namic programming," *IEEE Trans. Power Syst.*, vol. 13, no. 1, pp. 72–78, Feb. 1998.

[6] C. Panigrahi, P. Chattopadhyay, R. Chakrabarti, and M. Basu, "Sim- ulated annealing technique for dynamic economic dispatch," *Elect. Power Compon. Syst.*, vol. 34, no. 5, pp. 577–586, 2006.

[7] R. Balamurugan and S. Subramanian, "Differential evolution-based dynamic economic dispatch of generating units with valve-point ef- fects," *Elect. Power Compon. Syst.*, vol. 36, no. 8, pp. 828–843, 2008.

[8] X. Yuan, L. Wang, Y. Yuan, Y. Zhang, B. Cao, and B. Yang, "A mod-ified differential evolution approach for dynamic economic dispatch with valve point effects," *Energy Convers. Manage.*, vol. 49, no. 12, pp. 3447–3453, Dec. 2008.

[9] Y. Lu, J. Zhou, H. Qin, Y. Li, and Y. Zhang, "An adaptive hybrid differential evolution algorithm for dynamic economic dispatch with valve-point effects," *Expert Syst. Appl.*, vol. 37, no. 7, pp. 4842–4849,Jul. 2010.

[10] Y. Lu, J. Zhou, H. Qin, Y. Wang, and Y. Zhang, "Chaotic differentialevolution methods for dynamic economic dispatch with valve-point effects," *Eng. Appl. Artif. Intell.*, vol. 24, no. 12, pp. 378–387, Feb. 2011.