An Adaptive Multiobjective Evolutionary Algorithm for Economic Emission Dispatch

Tsung-Che Chiang
Dept. of Computer Science and Information Engineering,
National Taiwan Normal University
Taipei, Taiwan, R.O.C.
tcchiang@ieee.org

Thammarsat Visutarrom
Dept. of Computer Science and Information Engineering,
National Taiwan Normal University
Taipei, Taiwan, R.O.C.
thammarsat@gmail.com

Sadan Kulturel-Konak
Management Information Systems
Penn State Berks
Reading, PA 19610, USA
sadan@psu.edu

Abdullah Konak
Information Sciences and Technology
Penn State Berks
Reading, PA 19610 USA
auk3@psu.edu

<< This paper is included in the Proceedings of IEEE Congress on Evolutionary Computation (CEC), Padua, Italy, Jul. 18–23, 2022. >>

***Abstract*—This paper addresses the economic emission dispatch (EED) problem where the goal is to allocate the power output of power generation units to satisfy power demand and minimize the cost and emissions simultaneously. We propose a multiobjective differential evolution algorithm and a reinforcement learning technique to adaptively control the parameters of differential evolution. Moreover, the proposed approach utilizes mating restriction and preferences in mating selection to improve search effectiveness and a dynamically controlled mutation to increase the exploration ability. The proposed ideas and algorithm were examined using four EED test cases. Experimental results showed positive effects of our proposed methods and the competitive performance of our algorithm.**

Keywords—economic dispatch, emission, multiobjective, evolutionary algorithm, adaptive control, reinforcement learning, parameter control

I. Introduction

Energy consumption and environmental protection are two important contemporary issues. The Economic Emission Dispatch (EED) problem aims to allocate the output of power generation units in a power system to minimize the cost and pollutant emissions simultaneously. In the EED problem, we are given N power generation units, and the output of each unit i ($1 \le i \le N$) is denoted by P_i. The two objectives are to minimize the generation cost and emissions, which are formulated by (1) and (2), respectively.

$$f_1: f_{cost} = \sum_{i=1}^{N} \left[a_i + b_i P_i + c_i P_i^2 + \left|d_i(\sin(e_i(P_i^{min} - P_i)))\right|\right] \tag{1}$$

$$f_2: f_{emission} = \sum_{i=1}^{N} [\alpha_i + \beta_i P_i + \gamma_i P_i^2 + \eta_i \exp(\delta_i P_i)] \tag{2}$$

In (1) and (2), the values of model coefficients, including a_i, b_i, and so on, are fixed and known in advance. In some models, the coefficients d_i and e_i are assumed to be zero, which means that those models do not consider the so-called valve-point effect. In the EED, there are two main types of constraints. First, the power limit constraints define the feasible range of P_i, as (3) shows. Second, the power demand constraint requires that the total power satisfy the power demand P_D and the power loss P_L as given in (4). Some models do not consider loss and set P_L by zero. There are two types of loss models in the literature. This paper adopts Kron's B-coefficients formula to calculate the power loss as a quadratic power generation function, as given in (5).

$$P_i^{min} \le P_i \le P_i^{max} \qquad i = 1, \ldots, N \tag{3}$$

$$\sum_{i=1}^{N} P_i - P_D - P_L = 0 \tag{4}$$

$$P_L = \sum_{i=1}^{N} \sum_{j=1}^{N} P_i B_{ij} P_j + \sum_{i=1}^{N} B_{0i} P_i + B_{00} \tag{5}$$

The EED problem aims to minimize the two objective functions simultaneously. When there is some conflict between the cost and emissions objectives, decision-makers are interested in discovering tradeoffs between the two objectives. A set of solutions representing such tradeoffs can be found based on the concept of Pareto dominance. A solution x is said to dominate another solution y if and only if x is better than y in terms of at least one objective and is not worse than y in terms of all objectives. A solution is Pareto optimal if it is not dominated by any other solution. The projection of all Pareto optimal solutions onto the objective space is called the Pareto front. Our goal in solving the EED problem is to find or approximate the set of Pareto optimal solutions.

Most approaches to the EED problem in the literature are based on metaheuristics, which are a group of stochastic search algorithms and are usually inspired by biological or physical phenomena. In this paper, we use an evolutionary algorithm (EA) to solve the EED problem. The EA is a population-based search algorithm and is thus suitable for simultaneously searching for the set of Pareto optimal solutions. The EED problem is a continuous, constrained, and multiobjective optimization problem. A simple way of understanding our algorithm is through how we deal with continuous optimization, constrained optimization, and multiobjective optimization. The proposed EA is based on differential evolution (DE) [1]. DE has many successful applications, especially in continuous optimization, but its performance could be sensitive to the parameter setting. Thus, in our algorithm, we apply reinforcement learning (RL) to adjust DE's parameters adaptively. As for constraint handling, we adopt a repair mechanism that is commonly used in the EED literature. Only feasible solutions are allowed to participate in the evolutionary process. Lastly, we apply the selection mechanism of NSGA-III [2] to identify and keep promising solutions when the two objectives are considered simultaneously. NSGA-III is the third version of the well-known NSGA algorithm family [3][4]. Using NSGA-III, solutions are ranked by dominance relationship and evaluated by their distribution in the objective space when they have the same rank. Dominance-based selection enables an EA to search for the set of Pareto optimal solutions in a single run without requiring to define objective weights.

The rest of this paper is organized as follows. Section II reviews the related literature, and Section III describes the proposed adaptive multiobjective DE (called MO-RLDE). Experiments and results are presented in Section IV, and conclusions and future work are given in Section V.

II. Literature Review

This section reviews the related literature through how the existing studies address continuous, constrained, and multiobjective optimization problems, which are three aspects of the EED problem.

A. Continuous Optimization

The EED problem includes N real-valued decision variables representing the output of N power generation units to satisfy demand and minimize the cost and emissions. In the literature, almost all metaheuristic algorithms for the EED problem with N power generation units encoded solutions as real vectors and searched in an N-dimensional continuous decision space.

One of the common metaheuristics for continuous optimization is Particle Swarm Optimization (PSO). The PSO treats solutions in a metaphor of flocking birds. Each bird has its own velocity and position. Each bird adjusts its velocity based on its best position in the search history (pbest) and the best of all pbest (i.e., gbest). Then, it changes its position by its velocity. Wang and Singh [5] proposed FMOPSO to solve the EED problem. In addition to the classic PSO process, they added a turbulence operator, which was a kind of mutation operator. They compared their FMOPSO with two other algorithms in solving a 14-unit problem instance, and their algorithm showed the best result. Liu *et al*. [6] proposed a cultural quantum-behaved PSO (CMOQPSO). They incorporated the belief space of the cultural algorithm and the multi-measurement of quantum computing into PSO. Zou *et al*. [7] proposed NGPSO, which is featured by random re-initialization for exploration and ignoring the effect of personal best for exploitation.

Another popular metaheuristic algorithm in the field of continuous optimization is differential evolution (DE). DE searches the decision space by moving solutions according to the difference between solutions directly (without the idea of velocity in PSO). Besides, DE does selection by comparing the current solutions and new solutions. Wu *et al*. [8] and Basu [9] proposed two MODEs, and both adopted classic rand/1/bin operators of DE. Niknam *et al*. [10] proposed Tribe-MDE, in which they divided the population into sub-populations (tribes) and evolved them separately, cooperatively, and wholly in three sequential stages. Two important parameters of DE, F and CR, were encoded into solutions and evolved together during the process. Gong *et al*. [11] combined PSO and DE. They took DE as a local optimizer of a small number of solutions located in the least crowded area of the objective space.

Since 2000, many metaheuristic algorithms have been proposed with different metaphors and algorithmic features. These algorithm were tried to solve the EED, for example, harmony search (HS) [12][13], teaching-learning-based optimization (TLBO) [14], chemical reaction optimization (CRO) [15], backtracking search algorithm (BSA) [16][17], water cycle algorithm (WCA) [18], squirrel search algorithm (SSA) [19], grey prediction evolutionary algorithm (GPEA) [19], and slime mould algorithm (SMA) [21]. They provided various ideas/operators to change/move solutions in the decision space.

B. Constrained Optimization

The EED problem commonly includes two kinds of constraints: the power limit constraints, which are boundary constraints on the decision variables, and the demand constraint, which requires the total generated power to be equal to the total demand plus power loss. There are three main methods to deal with these constraints in the literature: the penalty method, repair method, and hierarchical method.

A feasible solution satisfies all constraints; otherwise, it is infeasible and leads to some constraint violation. The penalty method aggregates the objective value(s) and the amount of constraint violation into a single value as the fitness of a solution. Then, the solution with a better fitness value is preferred. In NGPSO [7], the authors dealt with the demand constraint by fixing values of (N – 1) decision variables and solving the derived quadratic equation to find the value of the last decision variable. Then, the amount of violation of the boundary constraints is calculated. Finally, the fitness of a solution is the sum of the objective value and the constraint violation times a penalty factor.

The repair method attempts to reduce the amount of constraint violations through an iterative procedure. It usually deals with the boundary constraint by setting the infeasible values of variables to the closest boundary values. Then, the difference between the total generated power and the demand is calculated. This difference is compensated by modifying the value of a (randomly selected) decision variable. Next, the boundary constraint is checked for this variable, and the difference in the demand is calculated again. These steps are repeated until a pre-specified number of trials or a pre-specified allowed error (e.g., 10^{-5}) is reached. MO-DE/PSO, Tribe-MDE, and CMOQPSO used this repair method.

The hierarchical method hierarchically minimizes constraint violation and objective values. More specifically, this method intends to minimize constraint violation first and then minimize the objective value when there is no constraint violation. The constrained domination principle (CDP) of NSGA-II follows the above concept and extends it to multiple objectives. The CDP defines the dominance relationship between solutions by: (1) when one solution is feasible and the other one is infeasible, the feasible one dominates the other one; (2) when both solutions are feasible, the standard dominance relationship is applied; (3) when both solutions are infeasible, the one with smaller constraint violation dominates the other. This method was adopted in FMOPSO and SMODE [22].

Kuk *et al*. [23] analyzed the impact of constraint handling methods in four multi-objective EAs (MOEAs). They tested the three methods mentioned above and compared the performance in solving ten test cases. For three MOEAs, the repair method outperformed the other two methods in solving all ten cases. For the last MOEA, the repair method performed the best in solving small-scale cases. They also reported that the penalty method had difficulty in finding feasible solutions. In conclusion, they recommended adopting the repair method.

C. Multiobjective Optimization

The simplest method to deal with multiobjective optimization is to convert multiple objective values into a single one and then solve the transformed single-objective problem. In BSA [16], after the cost (f_1) and emissions (f_2) of all solutions are calculated, the corresponding normalized objective values are calculated by $f_k' = (f_k - f_k^{\min})/(f_k^{\max} - f_k^{\min})$, where $f_k^{\max}$ and $f_k^{\min}$ are the maximum and minimum values of the kth objective function, respectively. Finally, the fitness of a solution is the weighted sum of the normalized objective values. NGPSO also adopts this kind of method but uses a different normalization equation. One disadvantage of this method is that decision-makers need prior knowledge to set the proper value of the objective weights. They may need to try different weights and run the algorithm several times to get a satisfactory solution or investigate the tradeoff between the objectives. SMODE also calculates the fitness of solutions by summing the normalized objective values, but it separates the objective space into grids and collects the solutions with the smallest summation in the grids so that a set of well-spread solutions can be maintained.

While solving a multiobjective optimization problem, a common practice is the dominance-based method. The non-dominated sorting procedure of NSGA-II is widely used. It classifies the solutions into several ranks based on the dominance relationship. Shortly speaking, the solutions that are not dominated by any other solution are classified into the first rank; then, temporarily ignoring the solutions of rank 1, 2, …, (r–1), the solutions that are not dominated by any other are classified into the rth rank. A solution x is regarded as better than another solution y if x has a lower rank. When two solutions have the same rank, they are compared by the crowding distance, which measures the distance between the two neighboring solutions. The method of NSGA-II was adopted in several studies on EED [9][23][24]. Wu *et al*. [8] also used NSGA-II but proposed an entropy-based measure to replace the crowding distance.

For the metaheuristic algorithms without explicit selection steps (e.g., PSO), dealing with multiobjective optimization focuses on collecting the set of non-dominated solutions and identifying leading solutions (e.g., personal/global best). Usually, an external archive is maintained to store the non-dominated solutions that have been found so far during the search process. When the number of solutions exceeds a pre-specified limit, inferior solutions are removed by considering some diversity measures [5]. The local or personal leading solution is usually updated when the new solution is not dominated by the original one. The leading global solution is often (randomly) selected from the archive. Some additional actions such as random expansion [5] and area-based selection [6] may be applied to lead the population to move toward different parts of the objective space.

III. Proposed Algorithm (MO-RLDE)

This paper proposes a multiobjective DE to solve the EED problem. In addition to the standard components of an EA, we apply the RL technique for adaptive control of two important parameters, F and CR, of DE. We call our algorithm MO-RLDE. The details of MO-RLDE will be presented in the following sub-sections.

A. Solution Encoding and Initialization

To solve the EED problem, we aim to set the output of N power generation units in the power system. Intuitively, we encode a solution $X_j = \{P_{j1}, P_{j2}, \ldots, P_{jN}\}$ as a real-valued N-dimensional vector. Each solution X_j in the initial population is randomly initialized

by setting power output P_{ji} of generation unit i to a random value within the feasible range [$P_i^{\min}$, $P_i^{\max}$] specified in the power limit constraints (3).

B. Constraint Handling

Every time an infeasible solution is generated by initialization or mutation/crossover, we perform a repair procedure to make the solution feasible – satisfying both power limit and total demand constraints. To do so, we utilize the repair procedure in [10], which iteratively adjusts the power output of a randomly selected unit to reduce the violation in the demand constraint until the violation is small enough. In our algorithm, we set the error threshold by 10^{-5}.

C. Mating Selection

In an EA, mating selection refers to selecting solutions as parents to produce offspring (new solutions). In the canonical DE, each solution in the population serves as a parent, called the target vector in the terminology of DE, for one time. The most common operator of DE to produce new solutions is rand/1/bin, which generates a mutant solution based on three different solutions and exchanges values of some variables between the target and the mutant to get the trial solution, which is the offspring of the target.

There are two implicit selection actions in the above steps: first, each solution is selected exactly once as a parent; second, all remaining solutions (except the parent) are selected with the same probability of generating the mutant solution. In our algorithm, we have two modifications. Since it is usually more difficult to find solutions close to the two ends of the Pareto front, solutions closer to the ends of the Pareto front have a higher probability of being selected as a target solution. We sort the solutions in ascending order of their cost (f_1). Then, we use 2-tournament to select the target solution by preferring the solution with a smaller value of min(t_j, N_{pop} + 1 – t_j), where N_{pop} is the population size, and t_j denotes the order of solution X_j in the population sorted in the ascending order of f_1. For example, assume that N_{pop} is 10. The solution X_p, whose t_p is 1, is preferred over the solution X_q whose t_q is 2 since min(1, 11 – 1)=1 is smaller than (2, 11 – 2)=2. Similarly, the solution X_p, whose t_p is ten, is preferred over the solution X_q, whose t_q is nine, since min(10, 11 – 10)=1 is smaller than (9, 11 – 9)=2. Our second modification is that only solutions within a neighborhood of a target solution can be selected to produce the mutant solution. This idea of mating restriction was borrowed from MOEA/D [25]. More details of this selection approach are described in the following subsection.

D. Mutation and Crossover

We apply the rand/1 mutation in our algorithm. For each target solution, we generate a mutant solution V by $V = X_{r1} + F\cdot(X_{r2} - X_{r3})$. The basic implementation of rand/1 randomly selects three distinct solutions, X_{r1}, X_{r2}, and X_{r3}, from the whole population. The parameter F, called the scaling factor, is a real value, which is usually in the range [0, 2]. In multiobjective problems, however, generating new solutions based on distant solutions in the objective space might reduce the effectiveness of the search [26]. Therefore, we incorporate the idea of mating restriction; to do so, we use the index value t_j of solution X_j in the sorted population. After we select a target solution, we allow only N_{nb} solutions whose index values are closest to that of the target solution to produce the mutant solution. For example, if the index value of the target solution is 4 and N_{nb} is 5, then only solutions whose index values fall in [2, 3, <u>4</u>, 5, 6] are allowed. After mutation, we do the binomial crossover to the target and the mutant to generate the trial solution. The values of the decision variables of the trial solution are taken from the mutant solution with probability CR or the target solution with probability (1 – CR). The crossover rate, CR, is another important parameter of DE.

The canonical DE completes the generation of a new solution after executing the above mutation and crossover operators. From preliminary experimental results, we found that the population could converge quickly, and thus we apply the polynomial mutation [27] to the trial solution. The parameter η_{m} of polynomial mutation controls the distribution of the range of mutated values. In our algorithm, we set the value of $\eta_{\text{m}}(t)$ at generation t by

$$\eta_{\text{m}}(t) = \eta_{\text{m}}(0)\cdot(1 - t/N_{\text{gen}}), \tag{6}$$

where $\eta_{\text{m}}(0)$ is the initial value of η_{m} and N_{gen} is the maximum number of generations. By decreasing the value of η_{m} gradually, the range of the mutated values gets larger and helps explore the search space more effectively.

E. Environmental Selection

In an EA, the environmental selection step determines which solutions survive to the next generation and maintains the population size. The canonical DE employs a one-by-one environmental selection, that is, the better one of the target solution or the trial solution survives. This mechanism is straightforward and reasonable in solving single-objective problems. However, when solving multiobjective problems, it is not easy to decide the better one of two solutions since they usually do not dominate each other. Therefore, we adopt a population-based environmental selection. In other words, we generate N_{pop} trial solutions and then select N_{pop} solutions from $2\cdot N_{\text{pop}}$ solutions (N_{pop} current solutions from the previous generation plus N_{pop} trial solutions) to survive. We use the selection mechanism of NSGA-III. Like NSGA and NSGA-II, NSGA-III classifies $2\cdot N_{\text{pop}}$ solutions into several ranks. The feature of NSGA-III is that it sets evenly distributed reference lines in the objective space. Each solution is associated with the closest reference line. The number of associated solutions serves as a measure of crowdedness. For the solutions of the same rank, those with less crowdedness have a higher priority to survive. For more details, readers are referred to [2].

F. Reinforcement Learning-based Parameter Control

The scaling factor F and crossover rate CR are two important parameters of DE, and their values usually have a significant impact on the performance of DE. Parameter tuning, which refers to running the algorithm with different fixed parameter settings, may be able to find proper parameter values. However, this process takes a lot of computation time. Moreover, sometimes the algorithm may need different parameter values in various stages of the search process, for example, large F at the beginning and small F at the end. Furthermore, the best parameter values are usually problem-specific. Different scales and numbers of decision variables may require different parameter settings. These motivated us to develop a mechanism to control the values of F and CR adaptively.

Reinforcement Learning (RL) is one machine learning technique that helps an agent learn how to respond to the environment through repeated trials. The agent takes an action and then receives a reward from the environment. Recording the rewards obtained by taking actions in different states, the agent gradually learns which action can lead to the maximum reward in each state. Using RL as an adaptive parameter control mechanism of DE, the learning process is like that solutions learn how to choose values of F and CR based on the results of previous trials. Visutarrom *et al*. [28] proposed a DE with RL to solve the economic dispatch problem, which is a single-objective problem. This paper adapted and extended their RL control mechanism to the EED problem with two objectives. In the following, we explain our approach by defining the RL mechanism's states, actions, and rewards embedded in our multiobjective DE.

States: As in [28], we consider each solution as an agent and define the current state of a solution based on the relative quality of solutions in the current population. The idea behind this approach is that the solutions of different levels of quality may need different parameter values. Our algorithm applies the environmental selection mechanism of NSGA-III to decide which solutions survive. In the selection process, the solutions of the lower level of domination rank are selected earlier. For the solutions of the last acceptable level, solutions with less crowdedness are selected first. In other words, the order in which the solutions are selected to survive can represent the quality of the solutions. We divide the solutions into N_{st} groups with roughly equal sizes based on this order of solutions. Solutions in the sth group are said to be in state s. (For example, the best $1/N_{st}$ population are in state 1.)

Actions: The action refers to choosing the values of F and CR. An action a_{jk} means that the value of F is randomly selected from the range $[j\cdot0.2, (j+1)\cdot0.2]$ and the value of CR is randomly selected from the range $[k\cdot0.2, (k+1)\cdot0.2]$, where j and k are in [0, 1, 2, 3, 4]. In other words, there are 5×5 = 25 actions for every state. The Q-table stores the accumulated rewards $Q(s, a_{jk})$ for all pairs of state s and action a_{jk}. When a solution is in state s, it selects the action a^* corresponding to the largest $Q(s, a^*)$ among all 25 actions with probability $(1 - \varepsilon)$ and selects a random action in probability ε, where ε is a hyperparameter of RL.

Rewards: The reward $R(s, a)$ of an action a reflects how beneficial the action in a state s is. Taking action a means that a target solution generates a trial solution by using values of F and CR randomly selected from the range specified by a. The reward of the action depends on the solution quality of the trial solution. If the trial solution survives with state s (s = 1, 2, …, N_{st}), the reward is $10\cdot(N_{st} - s + 1)$; if the trial solution does not survive to the next generation, the reward is zero. The Q-table is updated by the standard Q-learning equation in (7), where α and γ are hyperparameters.

$$Q(s,a) = (1-\alpha)Q(s,a) + \alpha[R(s,a) + \gamma \cdot max_a Q(s',a)] \tag{7}$$

IV. Experiments and Results

A. Test Cases and Experimental Setting

We examined the effects of the proposed ideas and the performance of our MO-RLDE by four test cases with different scales. Unit data of test cases BASU6 and BASU10 were taken from [9]; unit data of test case IEEE14 were taken from [17]; unit data of test cases TW40 were taken from [7]. Table I shows the number of units, power demand, and whether the valve point effect and power loss are considered.

TABLE I. Test Cases

Test Case	Number of Units	Demand	Valve Point Effect	Loss Considered?
BASU6	6	1200	N	Y
BASU10	10	2000	Y	Y
IEEE14	14	950	N	Y
TW40	40	10500	Y	N

Our MO-RLDE has several parameters. The population size (N_{pop}) was fixed by 40 in the experiments in sub-sections B–D. The maximum number of generations (N_{gen}) was 200 for BASU6 and BASU10, 1000 for IEEE14, and 2000 for TW40, respectively. The neighborhood size (N_{nb}) of mating restriction was 10, and the initial value of η_m of polynomial mutation was 5. The number of states (N_{st}) and the initial Q values of the RL-based control mechanism were 4 and 0, respectively.

TABLE II. PARAMETER SETTING OF PROPOSED MO-RLDE

Parameter	Initial Value	Parameter	Initial Value
N_{pop}	40	$\eta_m(0)$	5
N_{gen}	200/200/1000/2000	N_{st}	4
N_{nb}	10	initial $Q(s, a)$	0

In sub-sections B, C, and D, we examined the effect of the proposed ideas, including the RL-based parameter control, enhanced mating selection, and adaptive polynomial mutation. We used the inverted generational distance (IGD) to evaluate the quality of the set of solutions obtained by different algorithm variants. The IGD calculates the average distance from the reference front to the obtained front. We used the net set of non-dominated solutions among solutions obtained by the compared algorithm variants plus the solutions obtained by the full version of MO-RLDE with 27 settings of (α, γ, ε) as the reference front. (Details will be given in the following sub-section.)

B. *Effect of RL-based Parameter Control Mechanism*

In this sub-section, we examine the effect of the RL-based parameter control mechanism. The enhanced mating selection mechanism and adaptive polynomial mutation were not added in this experiment. We tested 27 (3×3×3) algorithm variants with each of the hyperparameters α, γ, and ε being set by three values in {0.1, 0.5, 0.9}. For comparison, we also tested 25 algorithm variants with static values of F and CR, where the values of F and CR were set by five values in {0.1, 0.3, 0.5, 0.7, 0.9}. The number of variants with adaptive F/CR values and those with static F/CR values are roughly the same (27 vs. 25). Each algorithm variant solved each test case for ten runs, and the IGD metric assessed the quality of the solutions obtained in each case.

We chose the best variant from the 25 variants with static parameter values and called it Static25Best; we also selected the best variant from the 27 variants with adaptive parameter control, referred to as RL27Best. As mentioned, each algorithm variant solved each test case for ten runs. The IGD values of the best and worst runs of each of Static25Best and RL27Best are presented in Table III. The results show that the two best variants have similar performance.

TABLE III. IGD COMPARISON BETWEEN THE BEST VARIANT OF TWO CR/F CONTROL MECHANISMS

	Static25Best		RL27Best	
Test Case	Best-run IGD	Worst-run IGD	Best-run IGD	Worst-run IGD
BASU6	0.003	0.004	0.003	0.004
BASU10	0.009	0.011	0.010	0.011
IEEE14	0.011	0.016	0.011	0.013
TW40	0.022	0.031	0.021	0.033

Next, we picked up the best and the worst variants from the 25 variants with static parameter values and from the 27 variants with adaptive parameter control in terms of the median of IGD values over ten runs. Results are presented in Table IV. We can see that the best static variant and the best adaptive variant have similar performance, but the worst adaptive variant is better than the worst static variant, especially in larger problems.

TABLE IV. MEDIAN IGD COMPARISON BETWEEN THE BEST AND WORST VARIANTS OF TWO CR/F CONTROL MECHANISMS

	Static		RL	
Test Case	Best variant	Worst variant	Best variant	Worst variant
BASU6	0.003	0.005	0.003	**0.004**
BASU10	0.009	0.013	0.010	**0.012**
IEEE14	0.012	0.020	0.012	**0.014**
TW40	0.027	0.140	0.026	**0.041**

Based on the results of this experiment, we could tune parameters F and CR to find a static algorithm variant with a good performance. However, the disadvantage of parameter tuning is that we need to spend significant time trying different parameter values. If the parameter values were not set properly, the performance worsened significantly. (See the IGD values of the best and worst static variants in Table IV.) We can achieve the same high solution quality using the proposed RL-based adaptive control. Besides, the adaptive variants show strong robustness and are relatively insensitive to the values of the hyperparameters. All MO-RLDE algorithm variants adopted RL-based adaptive control in the following two sub-sections. We fixed the values of α, γ, and ε by 0.9, 0.1, and 0.1, respectively.

C. *Effect of Enhanced Mating Selection Mechanism*

In this sub-section, we examine the effect of the proposed mating selection mechanism, including a 2-tournament selection that prefers solutions closer to the ends of the front and mating restriction based on neighborhood. We tested two algorithm variants with and without the proposed mating selection mechanism. The IGD values of the best and the worst runs of each of the two variants are presented in Table V. Since test cases BASU6 and BASU10 are relatively easy to solve, here we compared the two variants only by using test cases IEEE14 and TW40. We present the best objective values (cost and emissions) found by the two variants over ten runs in Table VI. From Table V, we found that the enhanced mating selection mechanism slightly improved the IGD value in the best case but may have a negative effect in the worst case. From Table VI, we found that the proposed mechanism was helpful in improving the extreme solutions at the two ends of the front.

TABLE V. IGD COMPARISON BETWEEN MATING SELECTION MECHANISMS

	Without enhancement		With enhancement	
Test Case	Best IGD	Worst IGD	Best IGD	Worst IGD
IEEE14	0.011	0.013	0.010	0.012
TW40	0.021	0.031	0.019	0.039

TABLE VI. COST/EMISSIONS COMPARISON BETWEEN MATING SELECTION MECHANISMS

	Without enhancement		With enhancement	
Test Case	Best cost	Best emissions	Best cost	Best emissions
IEEE14	4303.9621	25.9863	**4303.5748**	**25.5331**
TW40	122089.118	178125.765	**121719.109**	**176876.148**

D. *Effect of Adaptive Polynomial Mutation*

In the last experiment, we examined the effect of adaptive polynomial mutation (ADPM), which acts as an extra perturbation in our algorithm. Again, we tested two algorithm variants with and without ADPM. The IGD values of the best and the worst runs of each of the two variants are presented in Table VII. The best objective values found by the two variants over ten runs are shown in Table VIII. We can see that ADPM helps to improve the worst-case performance.

TABLE VII. IGD COMPARISON BETWEEN VARIANTS WITH AND WITHOUT ADAPTIVE POLYNOMIAL MUTATION

	Without ADPM		With ADPM	
Test Case	Best IGD	Worst IGD	Best IGD	Worst IGD
IEEE14	0.010	0.012	0.011	0.012
TW40	0.019	0.039	0.017	**0.024**

TABLE VIII. COST/EMISSION COMPARISON BETWEEN VARIANTS WITH AND WITHOUT ADAPTIVE POLYNOMIAL MUTATION

	Without ADPM		With ADPM	
Test Case	Best cost	Best emissions	Best cost	Best emissions
IEEE14	4303.5748	25.5331	4303.6668	25.4101
TW40	121719.109	176876.148	121534.158	177012.457

E. *Comparison with Existing Algorithms*

In this sub-section, we compare the performance of MO-RLDE with that of the existing algorithms. We list the compared algorithms and some related information in Table IX. The first column lists the algorithms and their references. The second column shows how the algorithm solves the EED problem. If it solves EED by solving multiple converted single-objective problems with different objective weights, we mark it by SO; otherwise, we mark it by MO. In the following four columns, we present the population size (top value) and the maximum number of generations (bottom value) that each algorithm used. In [23], Kuk *et al*. only mentioned that the stopping criterion is 300,000 function evaluations.

TABLE IX. COMPARED ALGORITHMS AND COMPUTATIONAL BUDGET

Algorithm	SO/MO	BASU6	BASU10	IEEE14	TW40
NGPSO [7]	SO	40 200	40 200		40 800
BSA [16]	SO	n/a	n/a		
MOHS [12]	MO			20 1000	
BSA-NDA [17]	MO		20 1000	20 1000	
NSGA-II.R [23]	MO		(300000)		(300000)
MO-RLDE	MO	40 200	40 200	20 1000	40 2000

Tables X through XIII present the three best solutions obtained by each of the compared algorithms for four test cases, respectively. The best-cost solution is the solution with the lowest cost, and the best-emissions solution is the solution with the lowest emissions. The best compromise solution is the solution that achieves the best balance between cost and emissions.

In Tables X and XI, all compared algorithms show very similar performance when solving the two smaller test cases with 6 and 10 units. In Table XII, our MO-RLDE achieves the lowest cost and the lowest emissions among the three tested algorithms. In Table XIII, NSGA-II.R achieves the lowest cost and NGPSO achieves the lowest emissions. However, they consumed more computational budget to find the solutions. NSGA-II.R performed 300000 function evaluations, but our MO-RLDE only performed 40×2000 = 80000 function evaluations. NGPSO performed 40×800 = 32000 function evaluations. However, NGPSO is a SO method; hence, it needs to run three times to get the best cost, emissions, and compromise solutions. In total, it performed 32000×3 = 96000 function evaluations.

TABLE X. SOLUTION COMPARISON FOR TEST CASE BASU6

Algorithm	Objective value	Best cost	Best Emissions	Best Compromise
BSA	f_{cost}	63976	65992	64766.8227
	$f_{emissions}$	1360.1	1240.6	1289.5856
NGPSO	f_{cost}	63975.7788	65992.3518	n/a
	$f_{emissions}$	1360.0659	1240.6542	
MO-RLDE	f_{cost}	63975.7782	65993.1048	64729.0025
	$f_{emissions}$	1360.0654	1240.6592	1292.0096

TABLE XI. SOLUTION COMPARISON FOR TEST CASE BASU10

Algorithm	Objective value	Best cost	Best Emissions	Best Compromise
BSA	f_{cost}	111497.6308	116412.4441	113126.7515
	$f_{emissions}$	4572.1940	3932.2433	4146.7286
BSA-NDA	f_{cost}	111498.8712	116395.0552	112807.3733
	$f_{emissions}$	4563.3844	3932.8879	4188.0926
NGPSO	f_{cost}	111497.6308	116412.4440	116179.6487
	$f_{emissions}$	4572.1957	3932.2433	3939.2278
NSGA-II.R	f_{cost}	111497.63	116412.47	112856.15
	$f_{emissions}$	4572.21	3932.24	4181.17
MO-RLDE	f_{cost}	111497.7502	116396.9403	112991.4391
	$f_{emissions}$	4572.4319	3932.3578	4164.4028

TABLE XII. SOLUTION COMPARISON FOR TEST CASE IEEE14

	Objective value	Best cost	Best Emissions	Best Compromise
MOHS	f_{cost}	4311.55	4457.47	4358.26
	$f_{emissions}$	390.2289	72.4220	153.7821
BSA-NDA	f_{cost}	4321.5187	4541.5267	4405.8321
	$f_{emissions}$	248.7270	26.5126	88.8972
MO-RLDE	f_{cost}	**4303.7151**	4545.0061	4359.6870
	$f_{emissions}$	382.4276	**25.5579**	142.2466

TABLE XIII. SOLUTION COMPARISON FOR TEST CASE TW40

	Objective value	Best cost	Best Emissions	Best Compromise
NGPSO	f_{cost}	121513.48	129955.0011	129277.6300
	$f_{\text{emissions}}$	359295.8480	**176682.52**	177325.4405
NSGA-II.R	f_{cost}	**121414.50**	129973.09	125668.93
	$f_{\text{emissions}}$	356705.57	176718.90	195668.93
MO-RLDE	f_{cost}	121534.1584	129852.8894	125290.4988
	$f_{\text{emissions}}$	358456.0930	177012.4571	205709.0054

In Tables X–XIII, we also found that none of the best compromise solutions dominates one another. Actually, this could be a long-lasting problem in the EED literature. The EED problem is a multiobjective problem, and the Pareto front is sought. However, performance comparison in most papers was carried out only based on three solutions (best-cost, best-emissions, and best compromise) instead of the whole set of non-dominated solutions.

In Table XII, the best compromise solutions of BSA-NDA and MO-RLDE do not dominate each other. This might lead to the conclusion that the two algorithms behave equally well. However, after we plot the 20 solutions found by both algorithms in Fig. 1, we can see that the solutions of MO-RLDE dominate the solutions of BSA-NDA in some parts. (The 20 solutions of BSA-NDA were taken from Table 30 in [17], and the 20 solutions of MO-RLDE were the solutions of the run with the worst IGD value over ten runs.) We calculated the IGD values of these two solution sets. The IGD value of the solution set of BSA-NDA is 0.034, and the IGD value of the solution set of MO-RLDE is 0.023. MO-RLDE outperformed BSA-NDA. Again, in Table XIII, the best compromise solutions of NGPSO and MO-RLDE do not dominate each other. Nevertheless, the distribution of the obtained solutions by the two algorithms is quite different, as shown in Fig. 2. NGPSO could not find a large part of the set of non-dominated solutions. In all cases, MO-RLDE was able to discover solutions almost uniformly distributed across the whole spectrum of the Pareto font, indicating its effectiveness in discovering the tradeoff between the objectives.

The above two examples show that comparing two multiobjective algorithms based on a single compromise solution cannot reflect the real solution quality of the algorithms. We need the complete list of solutions to apply multiobjective performance indicators such as IGD. To this end, we provide the solutions obtained by MO-RLDE as downloadable files on the first author's website.

V. CONCLUSIONS AND FUTURE WORK

This paper proposed an adaptive EA to solve the EED problem. We used DE to generate new solutions for continuous optimization, NSGA-III for dealing with two objectives simultaneously, and a repair method for constraint handling. In addition, we utilized RL for adaptive control of parameters of DE, enhanced mating selection in DE by imposing some preferences and restrictions, and applied an adaptive mutation operator for better exploration of the solution space. The computational studies using the four test cases from the literature showed that the proposed methods positively improved the robustness and solution quality. The proposed algorithm also provided competitive results when compared with existing algorithms.

In our future work, we plan to investigate more deeply the components of our algorithm. First, we will try different definitions of the state in our RL-based control mechanism. Second, we will perform more studies on the effects of the neighborhood size (N_{nb}) of mating restriction and the number of states (N_{st}) of RL. Finally, we will consider using more than one DE mutation strategy. Dynamic or adaptive control of the above two parameters and mutation strategies will be the main research topic.

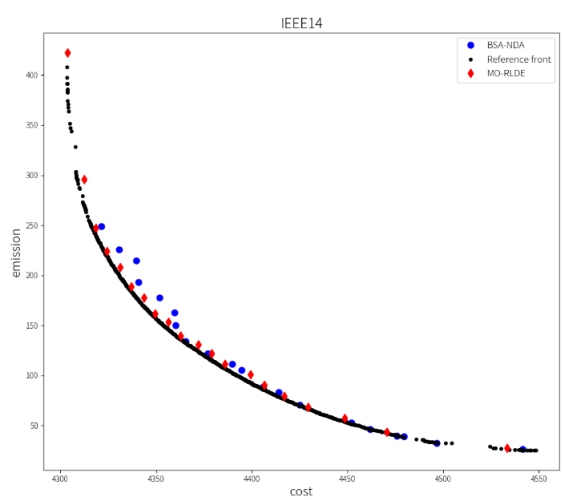
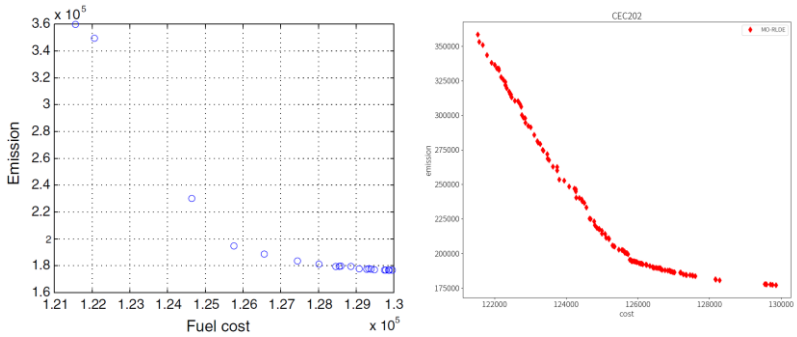


Fig. 1. Solutions obtained by BSA-NDA and MO-RLDE for test case IEEE14



(a) solutions of NGPSO (Fig. 7(e) in [7])

(b) solutions of MO-RLDE

Fig. 2. Solutions obtained by NGPSO and MO-RLDE for test case TW40

ACKNOWLDGEMENT

This research is supported by the Ministry of Science and Technology, Taiwan, R.O.C. under Grant no. 110-2221-E-003-017 and the Pennsylvania State University-National Taiwan Normal University Collaboration Development Fund.

REFERENCES

[1] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, vol. 11, pp. 341–359, 1997. [DE]

[2] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based non-dominated sorting approach, part I: solving problems with box constraints," IEEE Transactions on Evolutionary Computation, vol. 18, no. 4, pp. 577–601, 2014. [NSGA-III]

[3] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting genetic algorithms," Evolutionary Computation, vol. 2, no. 3, pp. 221–248, 1994. [NSGA]

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002. [NSGA-II]

[5] L. Wang and C. Singh, "Environmental/economic power dispatch using a fuzzified multi-objective particle swarm optimization algorithm," Electric Power Systems Research, vol. 77, pp. 1654–1664, 2007. [FMOPSO]

[6] T. Liu, L. Jiao, W. Ma, J. Ma, and R. Shang, "Cultural quantum-behaved particle swarm optimization for environmental/economic dispatch," Applied Soft Computing, vol. 48, pp. 597–611, 2016. [CMOQPSO]

[7] D. Zou, S. Li, Z. Li, and X. Kong, "A new global particle swarm optimization for the economic emission dispatch with or without transmission losses," Energy Conversion and Management, vol. 139, pp. 45–70, 2017. [NGPSO]

[8] L. H. Wu, Y. N. Wang, X. F. Yuan, and S. W. Zhou, "Environmental/economic power dispatch problem using multi-objective differential evolution algorithm," Electric Power Systems Research, vol. 80, pp. 1171–1181, 2010. [MODE10]

[9] M. Basu, "Economic environmental dispatch using multi-objective differential evolution," Applied Soft Computing, vol. 11, pp. 2845–2853, 2011. [BASU-MODE]

[10] T. Niknam, H. D. Maojarrad, and B. B. Firouzi, "A new optimization algorithm for multiobjective economic/emission dispatch," Electrical Power and Energy Systems, vol. 46, pp. 283–293, 2013. [Tribe-MDE]

[11] D. W. Gong, Y. Zhang, and C. L. Qi, "Environmental/economic power dispatch using a hybrid multi-objective optimization algorithm," Electrical Power and Energy Systems, vol. 32, pp. 607–614, 2010. [MO-DE/PSO]

[12] S. Sivasubramani and K. S. Swarup, "Environmental/economic dispatch using multi-objective harmony search algorithm," Electric Power Systems Research, vol. 81, pp. 1778–1785, 2011. [MOHS]

[13] B. Jeddi and V. Vahidinasab, "A modified harmony search method for environmental/economic load dispatch of real-world power systems," Energy Conversion and Management, vol. 78, pp. 661–675, 2014. [MHSA]

[14] P. K. Roy and S. Bhui, "Multi-objective quasi-oppositional teaching learning based optimization for economic emission load dispatch problem," Electrical Power and Energy Systems, vol. 53, pp. 937–948, 2013. [QOTLBO]

[15] K. Bhattacharjee, A. Bhattacharya, and S. H. nee Dey, "Solution of economic emission load dispatch problems of power systems by real coded chemical reaction algorithm," Electrical Power and Energy Systems, vol. 59, pp. 176–187, 2014. [RCCRO]

[16] K. Bhattacharjee, A. Bhattacharya, and S. H. nee Dey, "Backtracking search optimization based economic environmental power dispatch problems," Electrical Power and Energy Systems, vol. 73, pp. 830–842, 2015. [BSA]

[17] M Modiri-Delshad and N. A. Rahim, "Multi-objective backtracking search algorithm for economic emission dispatch problem," Applied Soft Computing, vol. 40, pp. 479–494, 2016. [BSA-NDA]

[18] M. A. Elhameed and A. A. El-Fergany, "Water cycle algorithm-based economic dispatcher for sequential and simultaneous objectives including practical constraints," Applied Soft Computing, vol. 58, p. 145–154, 2017. [WCA]

[19] V. P. Sakthivel, M. Suman, and P. D. Sathya, "Combined economic and emission power dispatch problems through multi-objective squirrel search algorithm," Applied Soft Computing, vol. 100, 2021. [MOSSA]

[20] Z. Hu, Z. Li, C. Dai, X. Xu, Z. Xiong, and Q. Su, “Multiobjective grey prediction evolution algorithm for environmental/economic dispatch problem,” IEEE Access, 2020. [MOGPEA]

[21] M. H. Hassan, S. Kamel, L. Abualigah, and A. Eid, “Development and application of slime mould algorithm for optimal economic emission dispatch,” Expert Systems with Applications, vol. 182, 2021. [ISMA]

[22] B. Y. Qu, J. J. Liang, Y. S. Zhu, Z. Y. Wang, and P. N. Suganthan, “Economic emission dispatch problems with stochastic wind power using summation based multi-objective evolutionary algorithm,” Information Sciences, vol. 351, pp. 48−66, 2016. [SMODE]

[23] J. N. Kuk, R. A. Gonçalves, L. M. Pavelski, S. M. G. S. Venski, C. P. de Almeida, A. T. R. Pozo, “An empirical analysis of constraint handling on evolutionary multi-objective algorithms for the environmental/economic load dispatch problem,” Expert Systems with Applications, vol. 165, 2021. [NSGA-II.R]

[24] T. C. Bora, V. C. Mariani, and L. dos S. Coelho, “Multi-objective optimization of the environmental-economic dispatch with reinforcement learning based on non-dominated sorting genetic algorithm,” Applied Thermal Engineering, vol. 146, pp. 688−700, 2019. [NSGA-RL]

[25] Q. Zhang and H. Li, “MOEA/D: A multiobjective evolutionary algorithm based on decomposition,” IEEE Transactions on Evolutionary Computation, vol. 11, no. 6, pp. 712−731, 2007. [MOEA/D]

[26] A. Konak and A. E. Smith, “Efficient optimization of reliable 2-node connected networks: a bi-objective approach,” INFORMS Journal on Computing, vol. 23, no. 3, pp. 430 − 445, 2011.

[27] K. Deb and S. Agrawal, “A niched-penalty approach for constraint handling in genetic algorithms,” Artificial Neural Nets and Genetic Algorithms, 1999. [PM]

[28] T. Visutarrom, T. C. Chiang, A. Konak, and S. Kulturel-Konak, “Reinforcement learning-based differential evolution for solving economic dispatch problems,” In: Proceedings of IEEE International Conference on Industrial Engineering and Engineering Management, 2020. [RLDE]