Genetic-based algorithm for power economic load dispatch

C.-L. Chiang

Abstract: An improved genetic algorithm with multiplier updating (IGAMU) to solve practical power economic load dispatch (PELD) problems of different sizes and complexities with non-convex cost curves, where conventional mathematical methods are inapplicable, is developed. The improved genetic algorithm (IGA) provides an improved evolutionary direction operator and a migrating operator, enabling it to efficiently search and actively explore solutions. Multiplier updating (MU) is introduced to avoid deforming the augmented Lagrange function, which is adopted to manage the system constraints of PELD problems. The proposed IGAMU integrates the IGA with the MU. Two practical examples are employed to demonstrate that the proposed algorithm has the benefits of straightforwardness, ease of implementation, better effectiveness than previous methods, better effectiveness and efficiency than the genetic algorithm (GA) with MU (GA–MU), automatic adjustment of the randomly assigned penalty to an appropriate value and the requirement for only a small population when applied to real-life PELD operations.

1 Introduction

The genetic algorithm (GA) has frequently been reported on the power economic load dispatch (PELD) problem, which is one of the most important optimisation systems in a power system for allocating generation among the committed units to satisfy the system constraints imposed and minimise the energy requirements. Improvements in scheduling the unit outputs can lead to significant cost savings. For simplicity, the generator cost function was mostly approximated by a single quadratic function [1]. However, because the cost curve of a fossil fired plant is highly nonlinear, containing discontinuities owing to valve-point loadings [2], the fuel cost function is more realistically denoted as a recurring rectified sinusoidal function [3] rather than a single quadratic function. Moreover, the whole operating range may not always be available. Units may have prohibited operation zones (POZs) because faults in machines or associated auxiliaries, including boilers and feed pumps, causing instabilities in certain ranges. The prohibited zone separates the decision space into disjointed subsets, constituting a non-convex solution space, turning the PELD problem into a non-smooth optimisation problem with complex and non-convex characteristics, which makes the global optimum hard to discover. The PELD problem is traditionally solved using conventional mathematical techniques such as the lambda iteration ($\lambda - \delta$) and gradient schemes. These approaches require that fuel cost curves should increase monotonically to obtain the global optimal solution. Conversely, the units have naturally highly nonlinear input–output properties due to valve-point loadings or POZ effects, causing traditional methods to generate either multiple local minima or infeasible solutions for the PELD problem. Various evolutionary algorithms, such as evolutionary strategy optimisation (ESO) [3], simulated annealing (SA) [4], GA [5], hybrid stochastic search (HSS) [6], hybrid evolutionary programming (EP) and sequential quadratic programming (SQP) method (EP–SQP) [7], hybrid particle swarm optimisation (PSO) technique with the SQP approach (PSO–SQP) [8] and integrated artificial intelligence (ETQ) [9], have recently been presented to solve the PELD problem. The SA is designed to solve the high nonlinear PELD problem without restrictions on the shape of the fuel cost function. Nevertheless, SA cannot easily be utilised to tune the related control parameters of the annealing schedule, and may be too slow for a practical power system. The GA can find a global solution after sufficient iterations, but has a high computational burden. The HSS incorporates SA in the GA selection process. The EP also takes a long computation time to obtain solutions. The PSO converges more quickly than EP, but has a slow fine-tuning ability of the solution. The ETQ integrates EP, Tabu search and quadratic programming methods, and the ESO is based on the classical evolutionary strategy considering Gaussian mutation to solve the PELD problem.

To enhance the GA's computational efficiency, the improved genetic algorithm (IGA) was developed with an improved evolutionary direction operator (IEDO) altered from the work of Yamamoto and Inoue [10], and a migration [11]. The IGA has been successfully applied to constrained optimisation problems, including the design of an induction motor controller for tracking control [12], and nonlinear mixed-integer optimisation problems [13]. IGA needs only a small population, and is more efficient than GA.

Michalewicz and Schoenauer [14] surveyed and compared several constraint-handling schemes used in evolutionary algorithms, and revealed that the penalty function algorithm is among the most popular methods of managing constraints. Powell [15] observed that classical optimisation schemes with a penalty function have certain disadvantages that become severe when using large penalty parameters,

doi:10.1049/iet-gtd:20060130

Paper first received 13th January and in revised form 1st July 2006

C.-L. Chiang is with the Department of Electronic Engineering, Nan Kai Institute of Technology, Nan-Tou 542, Republic of China

E-mail: t129@nkc.edu.tw

because the penalty function becomes ‘ill-conditioned’, making good solutions difficult to obtain. However, a situation in which the penalty parameters are too small makes it impossible for the constraint violation to contribute a high cost to the penalty function. Therefore selecting suitable penalty parameters is not trivial. This investigation develops the MU technique to manage constrained optimisation problems. The MU method can eradicate the ill-conditioned property of the objective function.

The proposed approach integrates the IGA and MU such that it has the following merits: straightforwardness; ease of implementation; better effectiveness than previous methods; better effectiveness and efficiency than GA–MU; automatic adjustment of the randomly assigned penalty to an appropriate value and the requirement for only a small population when applied to practical PELD problems. Two realistic examples indicate that the proposed algorithm has these benefits mentioned earlier in real-life PELD operations.

2 Problem formulation

The PELD problem can be described as an optimisation process based on the following objective function

$$\text{minimize} \sum_{i=1}^{n_p} F_i(P_i) \tag{1}$$

where $F_i(P_i)$ is the fuel cost function of the unit i, P_i the power generated by unit i and n_p the number of online units. The equality constraint of the power balance is given by

$$\sum_{i=1}^{n_p} P_i = P_\text{d} + P_\text{L} \tag{2}$$

where P_d is the system load demand and P_L is the transmission loss. The generating capacity constraints are written as

$$P_i^{\min} \le P_i \le P_i^{\max}, \quad i = 1, \ldots, n_\text{p} \tag{3}$$

where $P_i^{\min}$ and $P_i^{\max}$ are respectively, the minimum and maximum power outputs of unit i.

This work addresses two practical PELD cases with highly nonlinear cost functions. The first case considers the valve-point loadings of units where cost functions are typically described as the superposition of sinusoidal and quadratic functions. The second case considers the non-convex cost functions of units with POZs.

2.1 PELD problem considering valve-point loadings

Cost functions comprise very different input–output curves, since each generator has multi-valve steam turbines. A sinusoidal function is incorporated into a quadratic function to consider the valve-point loadings. Cost functions addressing the valve-point loadings of generating units [2] are given by

$$\begin{aligned} F_i(P_i) &= a_i + b_i P_i + c_i P_i^2 \\ &\quad + \left| e_i \times \sin\left(f_i \times (P_i^{\min} - P_i)\right) \right| \end{aligned} \tag{4}$$

where a_i, b_i and c_i are cost coefficients of generator i, and e_i and f_i are fuel cost coefficients of unit i with valve-point loadings.

2.2 PELD problem considering POZs effects

Additional constraints on the unit operating range denote the effects of a generator with POZs [5]

$$\begin{aligned} &P_i^{\min} \le P_i \le P_{i,1}^{\text{l}} \quad \text{or} \\ &P_{i,j-1}^{\text{u}} \le P_i \le P_{i,j}^{\text{l}}, \qquad j = 2, \ldots, n_i \quad \text{or} \\ &P_{i,ni}^{\text{u}} \le P_i \le P_i^{\max}, \quad \forall i \in \omega \end{aligned} \tag{5}$$

where $P_{i,j}^{\text{l}}$ and $P_{i,j}^{\text{u}}$, respectively, are the lower and upper bounds of prohibited zone j of unit i; n_i is the number of prohibited zones in unit i and ω is the set of all online units with prohibited zones.

Clearly, the entire operating region of a dispatching unit with n_i prohibited zones is divided into $(n_i + 1)$ disjoint operating sub-regions. The total number of decision subspaces caused by that division may be counted as follows

$$N = \prod_{i \in \omega} (n_i + 1) \tag{6}$$

Equation (6) shows that the total number of decision subspaces rises extremely quickly as the number of units with prohibited zones rises.

3 Proposed approach

3.1 Improved genetic algorithm

Consider the following general nonlinear programming problem (NLP)

$$\begin{aligned} &\min_x f(x) \\ &x^\text{L} \le x \le x^\text{U} \end{aligned} \tag{7}$$

where x represents an n_C-dimensional variable, and x^L and x^U are the lower and upper limits of variables. The objective function is defined on the feasible domain ψ, which is defined as n_C-dimensional variables in the space R^{n_C}, and $x \in \psi$. Two important operators, the proposed IEDO and migration [11], are balanced in the IGA, which can determine an efficient direction in which to search for a solution and simultaneously maintain a suitably diverse small population. The following IGA operations are proposed

1. Presentation and Initialisation: The IGA is a parallel direct search algorithm that uses N_p individuals of decision variables x in the NLP, that is $x^\text{G} = \{x_i^\text{G}, i = 1, \ldots, N_\text{p}\}$, as a population in generation G. Herein, decision variables, x_i^G, are coded directly as real values within those bounds. Initialisation randomly generates N_p individuals and aims to cover the entire search space uniformly.

2. IEDO: The main weakness of the evolutionary direction operator [10] is that it creates a new chromosome from three arbitrary chromosomes in each generation, making the search operator blind. IEDO improved the operator by selecting three best solutions in each generation to implement the IEDO, and then obtaining a new solution that is superior to the original best one. The IEDO is presented in what follows.

A chromosome that carries a set of solutions with n_C optimising parameters may be expressed as $x_j = \{C_1, C_2, \ldots, C_p, \ldots, C_{n_C}\}$. Each C_p represents a continuous decision variable and is limited by its lower and upper bounds, ($C_p^{\min}$ and $C_p^{\max}$). Three sets of optimal chromosomes can be selected in each generation. These three preferred chromosomes are ascended according to their fitness and are called the ‘low’, ‘medium’ and ‘high’ chromosomes, respectively. Three inputs (preferred) and the output (created) chromosomes are denoted in what follows.

Inputs: 'low' chromosome, $\{C_{l1}, C_{l2}, \ldots, C_{ln_C}\}$, with fitness F_l
'medium' chromosome, $\{C_{m1}, C_{m2}, \ldots, C_{mn_C}\}$, with fitness F_m
'high' chromosome, $\{C_{h1}, C_{h2}, \ldots, C_{hn_C}\}$, with fitness F_h
Output: chromosome, $\{C_{o1}, C_{o2}, \ldots, C_{on_C}\}$, with fitness F_{new}

The IEDO can significantly reduce the effort required to search for the optimal solution, because it enhances the local searching capability for GA. Fig. 1 shows the flow chart of minimum operations for IEDO.

3. Reproduction, Crossover and Mutation: Three preferred individuals created by the IEDO are chosen for reproduction. The reproduction probabilities of these three selected individuals are set as follows: the first preferred unit 35%; the second preferred unit 25% and the third preferred unit 15%. The remaining 25% of population is formed by a randomly created feasible individual. A binomial mutual crossover [13] is used to increase the local diversity of individuals. For a small population (e.g. $N_p = 5$), the probability of crossover is set to 0.3, which is high enough to generate new individuals and to prevent high diversity resulting in divergence of the population. Mutation [13] introduces a small perturbation to increase the diversity of trial individuals after the crossover operation, stopping trial individuals from clustering and causing premature convergence of solutions. The probability of mutation is set to 0.03.

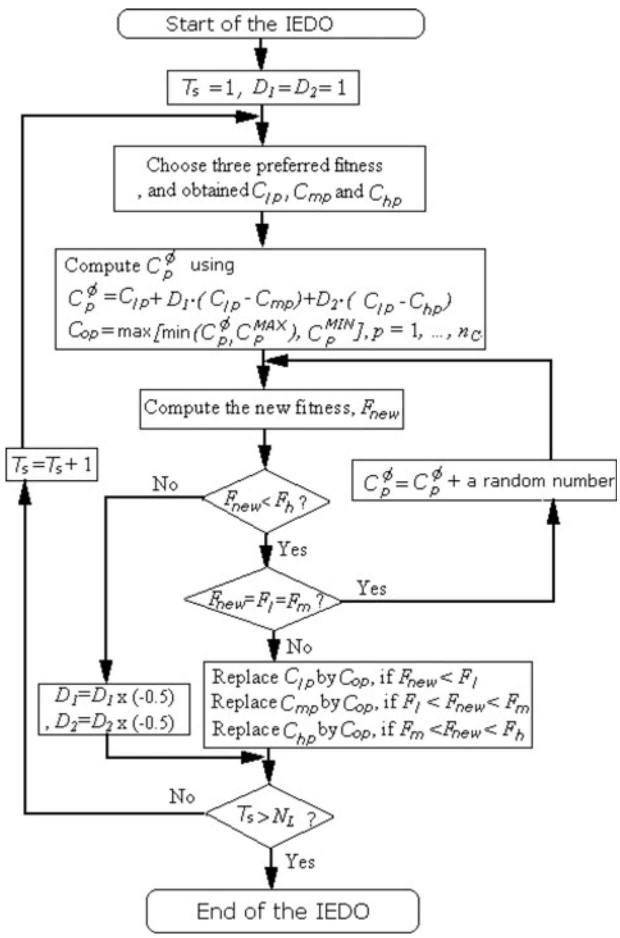

Fig. 1 *Flow chart of operations for the IEDO*

4. Natural or enforced migration if necessary: A migration is included in the IGA to regenerate a newly diverse population, preventing individuals from gradually clustering and thus significantly increasing the amount of search space explored for a small population. The migrant individuals are generated from the best individual, $x_b^{G+1} = (x_{1b}^{G+1}, x_{2b}^{G+1}, \ldots, x_{nb}^{G+1})$, by non-uniform random choice. Genes of individual i are regenerated according to

$$x_{ki}^{G+1} = \begin{cases} x_{kb}^{G+1} + \rho_i (x_k^L - x_{kb}^{G+1}), & \text{if } r_i < \dfrac{x_{kb}^{G+1} - x_k^L}{x_k^U - x_k^L} \\ x_{kb}^{G+1} + \rho_i (x_k^U - x_{kb}^{G+1}), & \text{otherwise} \end{cases} \tag{8}$$

where $k = 1, \ldots, n_C$, $i = 1, \ldots, N_p$ and r_i is a random number in the range [0, 1]. The migrant population not only becomes a set of newly promising solutions, but also has high possibility of escaping the local extreme value trap. The natural migration is executed only if a measure fails to match the desired tolerance of population diversity. The measure (ε) is defined as follows

$$\varepsilon = \frac{\left(\sum_{\substack{i=1 \\ i \neq b}}^{N_p} \sum_{k=1}^{n_C} \eta_{ki} \right)}{[n_C (N_p - 1)] < \varepsilon_1} \tag{9}$$

where

$$\eta_{ki} = \begin{cases} 1, & \text{if } \left| (x_{ki}^{G+1} - x_{kb}^{G+1}) / x_{kb}^{G+1} \right| > \varepsilon_2 \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where ε_1 and ε_2, respectively, are the desired tolerance of the population diversity and the gene diversity with respect to the best individual. Here, η_{ki} is defined as an index of gene diversity and its value of zero indicates that gene k of individual i is close to gene k of the best individual. The diversities of ε_1 and ε_2 are both set to 0.05 for naturally migrating operation.

The aforementioned natural migration is performed only if individuals closely cluster. Conversely, all individuals may be simultaneously trapped in a completely flat region of solution space, implying that the measure (ε) is not less than ε_1, but that fitness of all individuals are nearly identical, that is $f(x_1) \simeq f(x_2) \simeq \cdots \simeq f(x_{N_p})$. Hence, the natural migration cannot be actuated, and the algorithm remains locked in this region indefinitely, and an enforced migration using (8) must be performed to escape from it. The migration is enforcedly executed while the best fitness has not been improved for 500 consecutive generations.

3.2 Managing system constraints

Considering the NLP with general constraints as follows

$$\begin{aligned} & \min_x f(x) \\ & \text{subject to } h_k(x) = 0, \quad k = 1, \ldots, m_e \\ & \qquad g_k(x) \le 0, \quad k = 1, \ldots, m_i \end{aligned} \tag{11}$$

where $h_k(x)$ and $g_k(x)$ stand for equality and inequality constraints, respectively.

The penalty function method transforms the primal constrained problem into an unconstrained problem by penalising constraint violations. This technique is simple in concept and implementation. However, its primal limitation is the degree to which each constraint is penalised. These penalty terms have certain weaknesses that become fatal when penalty parameters are large, so the penalty function tends to be ill-conditioned near the boundary of the feasible domain where the optimum point is usually located.

The Lagrange method can significantly eradicate the disadvantages of the penalty method. The augmented Lagrange function (ALF) [13] for constrained optimisation problems is defined as

$$L_a(x,\nu,\upsilon) = f(x) + \sum_{k=1}^{m_e} \alpha_k \{[h_k(x) + \nu_k]^2 - \nu_k^2\} + \sum_{k=1}^{m_i} \beta_k \left\{ \langle g_k(x) + \upsilon_k \rangle_+^2 - \upsilon_k^2 \right\} \quad (12)$$

where α_k and β_k are the positive penalty parameters, and the corresponding Lagrange multipliers $\nu = (\nu_1, \ldots, \nu_{m_e})$ and $\upsilon = (\upsilon_1, \ldots, \upsilon_{m_i}) \geq 0$ are associated with equality and inequality constraints, respectively.

The contour of the ALF does not change shape between generations while constraints are linear. Therefore the contour of the ALF is simply shifted or biased in relation to the original objective function, $f(x)$. Consequently, small penalty parameters can be applied in the MU. However, the shape of the contour of L_a is changed by penalty parameters while the constraints are nonlinear, revealing that large penalty parameters still create computational difficulties. This study adopts adaptive penalty parameters to eliminate the above difficulties. Table 1 presents the computational procedures of the MU. The original objective function can be scaled to prevent ill-conditioning by updating penalty parameters and multipliers. Steps 3, 4 and 6 are executed to reform constraint violations and update penalty parameters. In step 4, if constraint violations have not been improved, such that $\hat{\varepsilon}_k \geq \varepsilon_k$, the penalty parameters are increased by a factor ω_2(e.g. $\omega_2 = 10$), and the multipliers are reduced by the same factor, maintaining the product of penalty parameters time multipliers unchanged. Step 6 applies the factor ω_1(e.g. $\omega_1 = 4$) to update penalty parameters and multipliers while constraint violations are still in existence. The benefit of MU is that the ALF can be scaled to prevent ill-conditioning, which can make PELD problems hard to solve.

3.3 Solution procedures of the proposed algorithm

Fig. 2 displays the flow chart of the proposed algorithm, which has two iterative loops. The ALF is used to obtain a minimum value in the inner loop with the given penalty parameters and multipliers, which are then updated in the outer loop to produce an upper limit of L_a. When both inner and outer iterations become sufficiently large, the ALF converges to a saddle-point of the dual problem [13]. The advantages of the proposed IGAMU are that the IGA efficiently searches the optimal solution in the economic dispatch process, and the MU effectively tackles system constraints.

4 System applications

This section demonstrates the effectiveness of the proposed IGAMU with respect to the quality of the solution obtained for realistic PELD problems using two practical systems. Both the IGAMU and the GA–MU were directly coded using real values, and were implemented on a personal computer (PIII-700) in FORTRAN-90. Chiang and Su [12] and Chiang *et al.* [13] recommend setting factors for the proposed algorithm. The following setting factors employed in these examples: the iteration number of the IEDO operation N_L was set to 4; the population size N_p was set to 5 and 20 for the IGAMU and the GA-MU, respectively, and the iteration numbers of the outer loop and inner loop were set to (outer, inner) as (30, 3000) in each example. Each unit's generation of examples 1 and 2 was limited

Table 1: Computational procedures of MU

Step 1. Set the initial iteration = 0. Set initial multipliers, $\nu_k^l = \nu_k^0 = 0$, $k = 1, \ldots, m_e$, and $\upsilon_k^l = \upsilon_k^0 = 0$, $k = 1, \ldots, m_i$, and the initial penalty parameters, $\alpha_k > 0$, $k = 1, \ldots, m_e$ and $\beta_k > 0$, $k = 1, \ldots, m_i$. Set tolerance of the maximum constraint violation, δ_k (e.g. $\delta_k = 10^{32}$), and the scalar factors, $\omega_1 > 1$ and $\omega_2 > 1$.

Step 2. Using the IGA, to solve $L_a(x,\nu^l,\upsilon^l)$ Let x_b^l be a minimum solution to the problem $L_a(x,\nu^l,\upsilon^l)$.

Step 3. Evaluate the maximum constraint violation as $\hat{\delta}_k = \max\{\max_k|h_k|, \max_k|\max(g_k, -\upsilon_k)|\}$, and establish the following sets of equality and inequality constraints whose violation is not modified by the factor ω_1:

$$I_E = \left\{ k : |h_k| > \frac{\delta_k}{\omega_1}, \quad k = 1, \ldots, m_e \right\}$$

$$I_I = \left\{ k : |\max(g_k, -\upsilon_k)| > \frac{\delta_k}{\omega_1}, \quad k = 1, \ldots, m_i \right\}$$

Step 4. If $\hat{\delta}_k \geq \delta_k$, let $\alpha_k = \omega_2 \alpha_k$ and $\nu_k^{l+1} = \nu_k^l / \omega_2$ for all $k \in I_E$, and $\beta_k = \omega_2 \beta_k$ and $\upsilon_k^{l+1} = \upsilon_k^l / \omega_2$ for all $k \in I_I$, and go to step 7; otherwise, go to step 5.

Step 5. Update the multipliers as follows

$$\nu_k^{l+1} = h_k\left(x_b^l\right) + \nu_k^l$$

$$\upsilon_k^{l+1} = \left\langle g_k\left(x_b^l\right) + \upsilon_k^l \right\rangle_+ = \upsilon_k^l + \max\left\{ g_k\left(x_b^l\right), -\upsilon_k^l \right\}$$

Step 6. If $\hat{\delta}_k \leq \delta_k / \omega_1$, let $\delta_k = \hat{\delta}_k$ and go to step 7; otherwise, let $\alpha_k = \omega_2 \alpha_k$ and $\nu_k^{l+1} = \nu_k^l / \omega_2$ for all $k \in I_E$, and $\beta_k = \omega_2 \beta_k$ and $\upsilon_k^{l+1} = \upsilon_k^l / \omega_2$ for all $k \in I_I$. Let $\delta_k = \hat{\delta}_k$ and go to step 7.

Step 7. If the maximum iteration is reached, stop; otherwise, repeat steps 2–6.

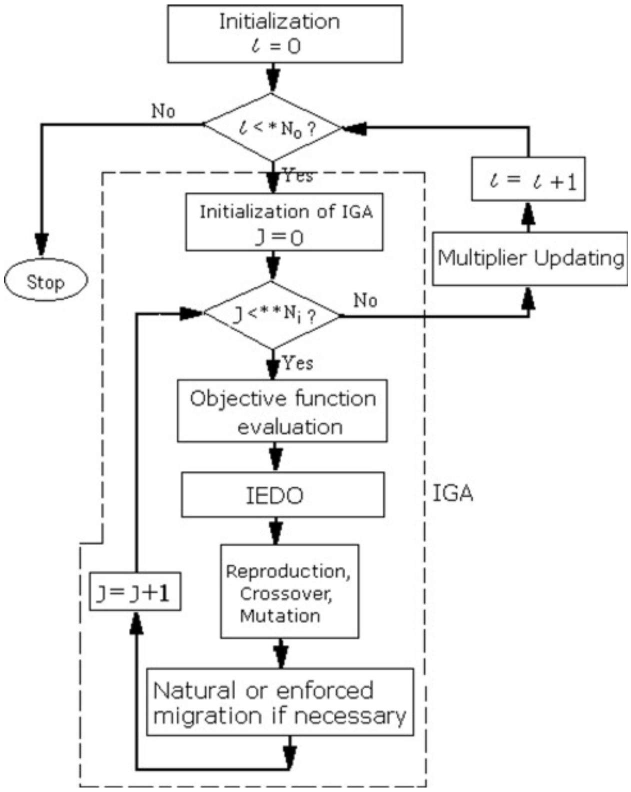

Fig. 2 *Flow chart of the IGAMU*

*N_o represents maximum number of iterations of outer loop
**N_i represents maximum number of iterations of inner loop

between its minimum and maximum capacities, which have been set in the program of the proposed algorithm.

4.1 Example 1

1. Simulation results: The first example consists of a 13-unit system considering valve-point loadings and using the cost functions (4) on the basis of system data from Wong and Wong [4]. The proposed IGAMU with one genetic-simulated-annealing-based algorithm (GAA2) [6], SA-based economic dispatch algorithm (SABED) [6], HSS [6], EP-SQP [7], PSO-SQP [8], ESO [3] and the GA-MU were compared with a load demand of 2520 MW in terms of the best dispatch solution. For the purpose of comparing previous methods with the same situations, examples 1 and 2 do not consider the transmission loss. The proposed algorithm is straightforward to understand and implement, and the implementation of this example can be described as follows

$$\mathrm{L_a}(x, \nu, \upsilon) = f(x) + \alpha_1\{[h_1(x) + \nu_1]^2 - \nu_1^2\} \qquad (13)$$

$$\text{objective}: \min_{x=P_1,P_2,\ldots,P_{13}} f(x) = \sum_{i=1}^{13} F_i(P_i) \qquad (14)$$

$$\text{subject to } h_1: \sum_{i=1}^{13} P_i - P_d = 0 \qquad (15)$$

Table 2 compares the results of previous methods, the GA-MU and the proposed algorithm for this example, and reveals that the proposed IGAMU not only has

Table 2: Compared results of previous methods, GA-MU and the proposed IGAMU for example 1

	Methods							
	GAA2	SABED	HSS	EP–SQP	PSO–SQP	ESO	GA–MU	IGAMU
unit z1	628.32	668.40	628.23	628.3136	628.3205	628.31	628.3179	628.3182
unit z2	356.49	359.78	299.22	299.1715	299.0524	299.12	299.1198	299.1955
unit z3	359.43	358.20	299.17	299.0474	298.9681	299.00	299.1746	299.1982
unit a1	159.73	104.28	159.12	159.6399	159.4680	159.32	159.7269	159.7327
unit a2	109.86	60.36	159.95	159.6560	159.1429	159.32	159.7269	159.7229
unit a3	159.73	110.64	158.85	158.4831	159.2724	159.15	159.7302	159.7330
unit a4	159.63	162.12	157.26	159.6749	159.5371	159.75	159.7320	159.7330
unit b1	159.73	163.03	159.93	159.7265	158.8522	159.61	159.7287	159.7305
unit b2	159.73	161.52	159.86	159.6653	159.7845	159.69	159.7073	159.7323
unit b3	77.31	117.09	110.78	114.0334	110.9618	112.89	73.2978	77.3671
unit c1	75.00	75.00	75.00	75.0000	75.0000	76.20	77.2327	77.3981
unit c2	60.00	60.00	60.00	60.0000	60.0000	92.38	92.2598	87.7394
unit c3	55.00	119.58	92.62	87.5884	91.6401	55.00	92.2454	92.3991
TP (MW)	2519.96	2520.00	2519.99	2520.0000	2520.0000	2519.74	2520.0000	2520.0000
TC ($/h)	24 398.23	24 970.91	24 275.71	24 266.44	24 261.05	24 177.79	24 170.7550	24 169.9790

Table 3: Relative frequencies of convergence for example 1 using GA-MU and the proposed IGAMU

Methods	Range of cost ($/h)						
	24 750–24 850	24 650–24 750	24 550–24 650	24 450–24 550	24 350–24 450	24 250–24 350	24 150–24 250
GA–MU	1	10	6	25	27	19	12
IGAMU	2	4	7	22	19	27	19

Table 4: **Summarised results of the comparison between GA–MU and the proposed IGAMU for example 1**

Methods	Mean time, s	Best time, s	Mean cost, \$/h	Maximum cost, \$/h	Minimum cost, \$/h
GA-MU	19.25	18.73	24 429.1202	24 759.3120	24 170.7550
IGAMU	7.54	7.14	24 385.4113	24 754.1450	24 169.9790

the lowest total cost (TC) of all methods tested, but also generates the exact total power (TP) for the system constraint (15), showing that the proposed algorithm is more effective than other methods for the practical PELD problem.

2. The performance of the proposed algorithm: Table 3 lists the relative frequencies of convergence between the proposed algorithm with the GA–MU for each cost range among 100 randomly initiated trials, to show the performance of the proposed IGAMU for this realistic example in a statistical manner. Table 4 itemises the tests of Table 3, and indicates that the proposed algorithm has both a better mean cost and the lower mean time than the GA–MU. Consequently, the proposed IGAMU is more effective and efficient than the GA–MU for solving this practical PELD problem.

4.2 Example 2

1. Simulation Results: To further demonstrate the effectiveness of the proposed algorithm, another practical system of units with POZs having non-convex cost functions was considered in this example, using the system data that are identical to those used in the work of Orero and Irving [5]. This system has 15 online units supplying a system demand of 2650 MW. Among these dispatching generators, units 2, 5 and 6 have three POZs, and unit 12 has two POZs, forming 192 decision subspaces for this realistic system. The implementation of this example can be represented as follows

$$\mathrm{L_a}(x, \nu, \upsilon) = f(x) + \alpha_1\left\{[h_1(x) + \nu_1]^2 - \nu_1^2\right\} + \sum_{k=1}^{4} \beta_k \left\{\langle g_k(x) + \upsilon_k \rangle_+^2 - \upsilon_k^2\right\} \qquad (16)$$

$$\text{objective}: \min_{x=P_1,P_2,\ldots,P_{15}} f(x) = \sum_{i=1}^{15} F_i(P_i) \qquad (17)$$

$$\text{subject to } h_1: \sum_{i=1}^{15} P_i - P_d = 0 \qquad (18)$$

$$\begin{aligned}
g_1 &: P_2^{150} \le P_2 \le P_{2,1}^{185}, \quad P_{2,1}^{225} \le P_2 \le P_{2,2}^{305}, \\
&\quad P_{2,2}^{335} \le P_2 \le P_{2,3}^{420} \quad \text{or} \quad P_{2,3}^{450} \le P_2 \le P_2^{455} \\
g_2 &: P_5^{105} \le P_5 \le P_{5,1}^{180}, \quad P_{5,1}^{200} \le P_5 \le P_{5,2}^{260}, \\
&\quad P_{5,2}^{335} \le P_5 \le P_{5,3}^{390} \quad \text{or} \quad P_{5,3}^{420} \le P_5 \le P_5^{470} \\
g_3 &: P_6^{135} \le P_6 \le P_{6,1}^{230}, \quad P_{6,1}^{255} \le P_6 \le P_{6,2}^{365}, \\
&\quad P_{6,2}^{395} \le P_6 \le P_{6,3}^{430} \quad \text{or} \quad P_{6,3}^{455} \le P_6 \le P_6^{460} \\
g_4 &: P_{12}^{20} \le P_{12} \le P_{12,1}^{30}, \quad P_{12,1}^{55} \le P_{12} \le P_{12,2}^{65}, \\
&\quad \text{or} \quad P_{12,2}^{75} \le P_{12} \le P_{12}^{80}
\end{aligned} \qquad (19)$$

This complex optimisation problem contains one objective function with 15 variable parameters, $(P_1, P_2, \ldots, P_{15})$,

Table 5: **Compared results of previous methods, GA-MU and the proposed IGAMU for example 2**

	Methods $\lambda-\delta$	SGA	DCGA	ETQ	ESO	GA–MU	IGAMU
unit 1	455	451.4	406.1	450	455.00	454.9783	454.9794
unit 2	455	455	453.8	450	455.00	454.9881	454.9638
unit 3	130	130	130	130	130.00	129.9941	129.9848
unit 4	130	129.1	130	130	130.00	129.9946	129.9919
unit 5	295.3[a]	337.1	355	335	304.24	259.9989	259.9884
unit 6	460	429.5	456.8	455	460.00	459.9893	459.9892
unit 7	465	464.4	459.8	465	465.00	464.9859	464.9731
unit 8	60	60	60	60	60.00	60.0172	60.0355
unit 9	25	26.6	26.6	25	25.00	25.0110	25.0137
unit 10	20	27.1	21.6	20	20.00	20.0095	20.0383
unit 11	43.4	25.7	36.2	20	29.15	58.4360	70.0417
unit 12	56.3	59	59	55	59.24	76.5870	64.9759
unit 13	25	25	25	25	25.00	25.0040	25.0105
unit 14	15	15	15	15	17.28	15.0029	15.0077
unit 15	15	15	15	15	15.00	15.0032	15.0061
TP (MW)	2650.0	2649.9	2649.9	2650	2649.91	2650.0000	2650.0000
TC (\$/h)	32 503[b]	32 517	32 515	32 507.5	32 506.6	32 506.3740	32 506.3390

[a]Unit loading in a prohibited zone
[b]Infeasible result

Table 6: **Relative frequencies of convergence for example 2 using GA-MU and the proposed IGAMU**

Methods	Range of cost ($/h)						
	32 650–32 675	32 625–32 650	32 600–32 625	32 575–32 600	32 550–32 575	32 525–32 550	32 500–32 525
GA-MU	3	3	4	5	5	52	28
IGAMU	0	2	0	4	4	47	43

Table 7: **Summarised results of the comparison between GA-MU and the proposed IGAMU for example 2**

Methods	Mean time, s	Best time, s	Mean cost, $/h	Maximum cost, $/h	Minimum cost, $/h
GA-MU	20.72	20.62	32 54 3.2681	32 664.9540	32 506.3740
IGAMU	7.69	7.47	32 530.5600	32 640.5020	32 506.3390

one equality constraint, (h_1) and four inequality constraints, (g_1 to g_4), since four units have the POZs. Table 5 compares the proposed IGAMU with $\lambda-\delta$ [5], standard GA (SGA) [5], deterministic crowding GA (DCGA) [5], ETQ [9], ESO [3] and GA–MU with respect to the best dispatch solution obtained. Unit loadings in Table 5 show why $\lambda-\delta$ is not capable of solving this PELD problem. For instance, using $\lambda-\delta$ produces a solution that requires unit 5 to operate in one of the POZs. However, the proposed algorithm provides final optimal loadings that do not fall into any of the 'illegal' zones. Table 5 shows that the proposed IGAMU also has the lowest feasible cost of all methods tested. Hence, for PELD problems of different size and complexity, the proposed IGAMU proves to be the best algorithm among those surveyed.

2. The performance of the proposed algorithm: Table 6 lists the relative frequencies of convergence for the proposed algorithm and the GA–MU from 100 randomly initiated trials, and indicates that the proposed IGAMU has a high probability of obtaining the best solution quality, demonstrating its excellent performance. Table 7 presents the compared results. The proposed IGAMU performs better than the GA–MU in terms of mean cost as well as mean time, even if the proposed method uses a small population. Tables 6 and 7 clearly reveal that the proposed IGAMU is more effective and efficient than the GA–MU.

5 Discussions

Because the proposed IGAMU actively adopts a flexible MU strategy to probe for the new solution, while the fixed-penalty-based algorithm only uses a fixed (large) penalty function to test passively feasible and infeasible solutions. A benefit of the proposed approach is that the ALF can be scaled to avoid the ill-condition leading to difficulty in finding a solution.

Even if the objective function $f(x)$ has many variable parameters, and the ALF has many equality and inequality constraints, the proposed algorithm is straightforward to understand [from the NLP with general constraints (11) to the ALF (12)] and implement (the proposed algorithm applies the IGA to solve the optimal solution, and employs the MU to automatically adjust the penalties and multipliers for handling the system constraints).

Although the proposed algorithm combines the IGA and MU, it only adopts the IGA to solve the objective function in the inner loop, and does not concern the penalty parameters or multipliers. The MU can manage system constraints by automatically updating the penalty parameters and multipliers in the outer loop. Therefore the proposed algorithm is easier to implement than fixed-penalty-based optimisations.

A test system was used to demonstrate further the effectiveness and efficiency of the proposed algorithm. The test system consisted of a realistic 40-unit Taiwan power system (which is a large-scale and mixed-generating system), considering valve-point loadings from Sinha *et al.* [2]. Similar to the work of Sinha *et al.* [2] and in order to compare results, the tests were performed without considering the loss. The proposed IGAMU was compared with two EPs (FEP: a fast EP, and IFEP: an improved FEP) [2], MPSO (a modified PSO) [16], Taguchi method (TM) [17], ESO [3] and the GA–MU in terms of the best dispatch solution for a load demand of 10 500 MW. This test utilised the same setting factors of the GA–MU and the proposed algorithm as examples 1 and 2. Table 8 lists the compared results. The proposed IGAMU has the lowest cost of all methods tested, demonstrating that the proposed algorithm is more effective than other methods for the realistic PELD problem. Table 9 shows the best solutions of the MPSO, ESO, GA–MU and IGAMU, and indicates that the proposed IGAMU has a lower cost and simulation time than GA–MU. Moreover, considering the non-smooth cost model with valve-point effects, the proposed algorithm not only had the best economic dispatch of all methods, but also completely satisfied the system constraints. Therefore analytical results clearly reveal that the proposed IGAMU can solve the PELD problem efficiently and effectively.

Table 8: **Compared results of previous methods, GA-MU and the proposed IGAMU for the discussion case**

	Methods						
	FEP	IFEP	MPSO	TM	ESO	GA–MU	IGAMU
Minimum cost ($)	122 679.71	122 624.35	122 252.265	122 477.78	122 122.16	122 000.2837	121 819.2521

Table 9: Compared results of the MPSO, ESO, GA-MU and the proposed IGAMU for the discussion case

	Methods			
	MPSO	ESO	GA–MU	IGAMU
unit 1	114.000	114.00	113.8630	111.9206
unit 2	114.000	114.00	74.1045	110.8223
unit 3	120.000	120.00	97.4762	97.4081
unit 4	182.222	190.00	179.9570	179.7336
unit 5	97.000	97.00	96.9560	88.2042
unit 6	140.000	140.00	139.9994	139.9976
unit 7	300.000	300.00	260.0023	259.7454
unit 8	299.021	288.88	299.9990	286.1419
unit 9	300.000	285.91	286.1980	285.6566
unit 10	130.000	203.02	204.7970	130.0000
unit 11	94.000	94.36	168.8147	168.7987
unit 12	94.000	96.45	168.8005	94.0006
unit 13	125.000	211.51	125.0028	304.4441
unit 14	304.485	301.46	304.5224	394.2794
unit 15	394.607	394.80	304.5173	394.2788
unit 16	305.323	300.15	484.0395	394.2808
unit 17	490.272	490.90	489.2827	489.2802
unit 18	500.000	489.13	489.3896	489.2832
unit 19	511.404	512.03	511.2863	511.2821
unit 20	512.174	512.08	511.2823	511.2809
unit 21	550.000	521.92	523.4731	523.3001
unit 22	523.655	532.09	525.1403	523.2806
unit 23	534.661	527.14	523.3914	523.2821
unit 24	550.000	524.54	523.4357	523.2796
unit 25	525.057	522.88	523.3986	523.2822
unit 26	549.155	522.42	523.3028	523.2791
unit 27	10.000	10.71	10.0004	10.0001
unit 28	10.000	10.71	10.0006	10.0002
unit 29	10.000	13.96	10.0114	10.0001
unit 30	97.000	85.35	90.7687	87.9679
unit 31	190.000	189.92	189.9989	160.2761
unit 32	190.000	190.00	189.9987	162.2347
unit 33	190.000	189.16	189.9956	165.9863
unit 34	200.000	197.28	170.0588	165.8653
unit 35	200.000	166.31	199.9990	199.9996
unit 36	200.000	199.36	165.4893	164.8574
unit 37	110.000	109.69	89.9315	89.8252
unit 38	110.000	108.57	109.9995	91.4569
unit 39	110.000	108.90	109.9998	89.7082
unit 40	512.964	514.02	511.3154	511.2792
TP (MW)	10 500.000	10 500.61	10 500.0000	10 500.0000
TC ($/h)	122 252.265	122 122.16	122 000.2837	121 819.2521
CPU time, s	—	—	61.42	27.03

6 Conclusion

This paper presented the IGAMU to solve practical PELD problems of different size and complexity having non-convex cost curves where conventional mathematical methods are inapplicable. The IGA enhances the ability of the proposed method to efficiently search and actively explore the solution, and the MU enables the proposed approach to manage system constraints efficiently. The proposed algorithm integrates the IGA and the MU such that it has the following merits: straightforward concept; easy implementation; better effectiveness than previous methods; better effectiveness and efficiency than the GA–MU; automatic adjustment of the randomly assigned penalty to an appropriate value and the requirement for only a small population in realistic PELD problems. Comparative results demonstrate that the proposed algorithm has these merits mentioned earlier in real-world PELD operations.

7 References

1 Lin, C.E., and Viviani, G.L.: 'Hierarchical economic dispatch for piecewise quadratic cost functions', *IEEE Trans. Power Appar. Syst.*, 1984 **103**, (6), pp. 1170–1175

2 Sinha, N., Chakrabarti, R., and Chattopadhyay, P.K.: 'Evolutionary programming techniques for economic load dispatch', *IEEE Evol. Comput.*, 2003, **7**, (1), pp. 83–94

3 Pereira-Neto, A., Unsihuay, C., and Saavedra, O.R.: 'Efficient evolutionary strategy optimisation procedure to solve the nonconvex economic dispatch problem with generator constraints', *IEE Proc. Gener. Transm. Distrib.*, 2005, **152**, (5), pp. 653–666

4 Wong, K.P., and Wong, Y.W.: 'Genetic and genetic/simulated-annealing approaches to economic dispatch', *IEE Proc. Gener. Transm. Distrib.*, 1994, **141**, (5), pp. 507–513

5 Orero, S.O., and Irving, M.R.: 'Economic dispatch of generators with prohibited operating zones: a genetic algorithm approach', *IEE Proc. Gener. Transm. Distrib.*, 1996, **143**, (6), pp. 529–534

6 Das, D.B., and Patvardhan, C.: 'Solution of economic load dispatch using real coded hybrid stochastic search', *Electr. Power Syst.*, 1999, **21**, pp. 165–170

7 Attaviriyanupap, P., Kita, H., Tanaka, E., and Hasegawa, J.: 'A hybrid EP and SQP for dynamic economic dispatch with nonsmooth fuel cost function', *IEEE Trans. Power Syst.*, 2002, **17**, (2), pp. 411–416

8 Victoire, T.A.A., and Jeyakumar, A.E.: 'Hybrid PSO-SQP for economic dispatch with valve-point effect', *Electr. Power Syst. Res.*, 2004, **71**, pp. 51–59

9 Lin, W.M., Cheng, F.S., and Tsay, M.T.: 'Nonconvex economic dispatch by integrated artificial intelligence', *IEEE Trans. Power Syst.*, 2001, **16**, (2), pp. 307–311

10 Yamamoto, K., and Inoue, O.: 'New evolutionary direction operator for genetic algorithms', *American Institute of Aeronautics and Astronautics (AIAA) J. Technical Notes*, 1995, **33**, (10), pp. 1990–1993

11 Chiou, J.P., and Wang, F.S.: 'A hybrid method of differential evolution with application to optimal control problems of a bioprocess system'. Conf. Rec. 1998 IEEE Int. Conf. on Evolutionary Computation (ICEC), pp. 627–632

12 Chiang, C.L., and Su, C.T.: 'Tracking control of induction motor using fuzzy phase plane controller with improved genetic algorithm', *Electr. Power Syst. Res.*, 2005, **73**, pp. 239–247

13 Chiang, C.L., Su, C.T., and Wang, F.S.: 'Augmented Lagrangian method for evolutionary optimization of mixed-integer nonlinear constrained problems', *Intern. Math. J.*, 2002, **2**, (2), pp. 119–154

14 Michalewicz, Z., and Schoenauer, M.: 'Evolutionary algorithms for constrained parameter optimization problems', *Evol. Comput.*, 1996, **4**, (1), pp. 1–32

15 Powell, M.J.D.: 'Algorithms for nonlinear constraints that use Lagrangian functions', *Math. Program.*, 1978, **14**, pp. 224–248

16 Park, J.B., Lee, K.S., Shin, J.R., and Lee, K.Y.: 'A particle swarm optimization for economic dispatch with nonsmooth cost functions,', *IEEE Trans. Power Syst.*, 2005, **20**, (1), pp. 34–42

17 Liu, D., and Cai, Y.: 'Taguchi method for solving the economic dispatch problem with nonsmooth cost functions,', *IEEE Trans. Power Syst.*, 2005, **20**, (4), pp. 2006–2014