

A New Quantum-Inspired Binary PSO for Thermal Unit Commitment Problems

Yun-Won Jeong, Jong-Bae Park, *Member, IEEE*, Se-Hwan Jang, and Kwang Y. Lee, *Fellow, IEEE*

Abstract—This paper proposes a new binary particle swarm optimization (BPSO) approach inspired from quantum computing, so-called quantum-inspired BPSO (QBPSO), for solving the unit commitment (UC) problems. Although BPSO-based approaches have been successfully applied to the combinatorial optimization problems of power systems, the BPSO algorithm has some drawbacks such as premature convergence when handling heavily constrained problems. The proposed QBPSO combines the conventional BPSO with the concept and principles of quantum computing such as a quantum bit and superposition of states. The QBPSO adopts a Q-bit individual for the probabilistic representation, which replaces the velocity update procedure in the particle swarm optimization. This paper also proposes an efficient rotation gate for updating Q-bit individuals to improve the searching capability of the quantum computing. To verify the performance of the proposed QBPSO, it is applied to the test systems of up to 100-units with 24-hour demand horizon.

Index Terms—Combinatorial optimization, unit commitment, binary particle swarm optimization, quantum computing, quantum evolutionary algorithm, constraint treatment technique.

I. INTRODUCTION

UNIT commitment (UC) problem involves scheduling the on/off states of generating units, which minimize the operating cost, start-up cost and shut-down cost for a given horizon under various operating constraints [1]. The optimal commitment scheduling can not only save millions of dollars for the power companies, but it can also maintain system reliability by keeping proper spinning reserves. In the UC problem, the decisions are the selection of the time for each unit to be on- and/or off-line (binary variables) as well as the unit's economic generation level (continuous variables). Thus, the UC problem can be formulated as a nonlinear mixed integer combinatorial optimization problem [2]-[16]. The number of combinations of 0-1 binary variables grows exponentially for a large-scale UC problem, which makes it difficult to solve a practical UC problem. Over the past decades, many salient methods have been developed for solving the UC problems. The exact solution to the problem can be obtained by complete enumeration, which cannot be applied to realistic power systems due to its computational

burdens [1]. The solution methods for UC problems can be divided into two classes: One is the numerical optimization techniques such as priority list methods [2], [3], dynamic programming [4], [5], Lagrangian relaxation methods [6], [7], branch-and-bound methods [8], and mixed-integer programming [9]. The other is the stochastic search methods such as genetic algorithms [10], [11], evolutionary programming [12], [13], simulated annealing [14], [15], quantum evolutionary algorithm [16], and particle swarm optimization [17].

Particle swarm optimization (PSO) suggested by Kennedy and Eberhart in 1995 is a population-based search algorithm and searches in parallel using a group of particles [18]. In the PSO, the particles are drawn stochastically toward the new position based on the present velocity of each particle, its own previous best performance, and the best previous performance of their neighbors [18], [19]. In 1997, Kennedy and Eberhart showed that the binary particle swarm optimization (BPSO) was able to successfully optimize the De Jong's suite of five test functions [20]. Quantum computing is a new paradigm which has been proposed as a consequence of applying quantum mechanics to computer science [21]-[24]. Research on merging evolutionary computation and quantum computing has been carried out since the late 1990s and can be classified into two fields; 1) new quantum algorithms using automatic programming techniques such as genetic programming [21], 2) quantum-inspired evolutionary computing for a digital computer as a branch of study on evolutionary computation that is characterized by certain principles of quantum mechanics such as uncertainty, superposition, and interference, etc. [22]-[24]. Quantum inspired computing was firstly introduced in [22]. Narayanan and Moore [23] proposed quantum-inspired genetic algorithms, where concepts and principles of quantum mechanics are used to inform and inspire more efficient evolutionary computing methods. Han and Kim [24] proposed a quantum-inspired evolutionary algorithm (QEA).

This paper presents a new BPSO-based approach for solving the UC problems. To enhance the performance of the conventional BPSO, this paper proposes a quantum-inspired BPSO (QBPSO) which is based on the concept and principles of quantum computing such as a quantum bit and superposition of states. The proposed QBPSO introduces a Q-bit individual for the probabilistic representation of a particle, thereby replacing the velocity update procedure in the traditional PSO. In QBPSO, therefore, an inertia weight factor and two

Y.-W. Jeong, J.-B. Park, and S.-H. Jang are with the Department of Electrical Engineering, Konkuk University, Seoul 143-701, Korea, (e-mail: {ywjeong, jbaepark, shjang}@konkuk.ac.kr).

K. Y. Lee is with the Department of Electrical and Computer Engineering, Baylor University, Waco, TX 76798, USA. (e-mail: Kwang_Lee@baylor.edu).

acceleration coefficients can be removed and only one factor, rotation angle, is needed when modifying the position of particles. To improve the conventional rotation gate for Q-bit individual update, this paper proposes an efficient rotation gate. Therefore, the proposed QBPSO can obtain an efficient balance between exploration and exploitation with a smaller population size and shorter computation time. To demonstrate the performance of the QBPSO, test systems of up to 100-units along with 24-hour load demands are tested and their results are compared with those of the previous works.

II. FORMULATION OF UNIT COMMITMENT PROBLEM

A. Objective Function

The objective of the UC problem is to minimize the total operating cost, start-up cost and shut-down cost of all generating units during a time horizon, subject to a number of system and unit constraints [1].

1) *Fuel Cost Function*: For all committed generating units, the total fuel cost is minimized by economically dispatching the units. The fuel cost function of unit j at hour t can be expressed as a second-order polynomial as follows:

$$F_j(P_{j,t}) = a_j + b_j P_{j,t} + c_j P_{j,t}^2 \quad (1)$$

where $P_{j,t}$ is the power generation of unit j at hour t and a_j, b_j, c_j are the cost coefficients of unit j .

2) *Start-up Cost*: Start-up cost for restarting a de-committed generating unit, which is related to the temperature of the boiler, is included in the objective function. The start-up cost is associated with the number of hours during which the unit has been off. Start-up cost will be high, defined as the cold cost ($SU_{C,j}$), when down time duration exceeds the cold start hour ($T_{cold,j}$) in excess of the minimum down time; and will be low, defined as the hot cost ($SU_{H,j}$), when down time duration does not exceed the cold start hour in excess of the minimum down time. In general, the start-up cost is described as follows:

$$SU_{j,t} = \begin{cases} SU_{H,j} & \text{if } MDT_j \leq TOFF_{j,t} \leq MDT_j + T_{cold,j} \\ SU_{C,j} & \text{if } TOFF_{j,t} > MDT_j + T_{cold,j} \end{cases} \quad (2)$$

where $TOFF_{j,t}$ is the duration for which unit j is continuously off-line at hour t and MDT_j is the minimum down-time of the j th unit.

3) *Shut-down Cost*: Shut-down cost is usually modeled as a constant value for each unit per shutdown. In this paper, the shut-down costs have not been taken into consideration.

Consequently, the objective function of the UC problem is given by the minimization of the following cost function:

$$\min \sum_{t=1}^T \sum_{j=1}^N [F_j(P_{j,t})u_{j,t} + SU_{j,t}(1-u_{j,t-1})u_{j,t}] \quad (3)$$

where T is the number of scheduling period, N is the number of generating units, and $u_{j,t}$ is the on/off status of unit j at hour t (i.e., $u_{j,t} = 1$ when unit j is on-line, and $u_{j,t} = 0$ when unit j is off-line).

B. System and Unit Constraints

1) *Load Balance Constraints*: The sum of unit generation output at each hour must satisfy the system load demand requirement of the corresponding hour as follows:

$$\sum_{j=1}^N P_{j,t} u_{j,t} = PD_t \quad (4)$$

where PD_t is the total system demand at hour t .

2) *Generation Limit Constraints*: The power produced by each unit must be within its limits as indicated below:

$$u_{j,t} P_{j,\min} \leq P_{j,t} \leq u_{j,t} P_{j,\max} \quad (5)$$

where $P_{j,\min}$ and $P_{j,\max}$ are the minimum and maximum generation limits of unit j , respectively.

3) *Spinning Reserve Constraints*: Spinning reserve must be provided so as to minimize the probability of load interruption. The spinning reserve is considered to be a pre-specified amount or a given percentage of the forecasted peak demand. Spinning reserve can be specified in terms of excess megawatt capacity, which is expressed by

$$\sum_{j=1}^N P_{j,\max} u_{j,t} \geq PD_t + SR_t \quad (6)$$

where SR_t is the required spinning reserve at hour t .

4) *Generation Ramping Constraints*: Due to the mechanical characteristics and thermal stress limitations of a generating unit, the operating range of all on-line units is restricted by their corresponding ramp rate limits as follows:

$$RD_j \leq P_{j,t} - P_{j,(t-1)} \leq RU_j \quad (7)$$

where RD_j and RU_j are the ramp-down and ramp-up limits of unit j , respectively.

5) *Minimum Up-time/Down-time Constraints*: The unit cannot be turned on/off immediately once it is committed or de-committed. The minimum up/down time constraints indicate that a unit must be on/off during a certain number of hours before it becomes shut-down or start-up, respectively. These constraints are given by

$$u_{j,t} = \begin{cases} 1 & \text{if } 1 \leq TON_{j,t-1} < MUT_j \\ 0 & \text{if } 1 \leq TOFF_{j,t-1} < MDT_j \\ 0 \text{ or } 1 & \text{otherwise} \end{cases} \quad (8)$$

where $TON_{j,t}$ is the duration for which unit j is continuously on-line at hour t and MUT_j is the minimum up-time of unit j .

III. OVERVIEW OF PARTICLE SWARM OPTIMIZATION

Kennedy and Eberhart developed a PSO algorithm, which is a population-based parallel search algorithm using a group of particles [18]. It is based on the behavior of individuals of a swarm and its roots are in zoologist's modeling of the movement of individuals within a population. It has been noticed that members of a group seem to share information among them, a fact that leads to increased efficiency of the group [19]. A particle moves toward the optimum based on its present velocity, its previous experience, and the experience of its neighbors. In an n -dimensional search space, the position and velocity of the i th particle are represented as vectors

$X_i = \{x_{i1}, \dots, x_{in}\}$ and $V_i = \{v_{i1}, \dots, v_{in}\}$, where each element has the real values. Let $Pbest_i = \{x_{i1}^P, \dots, x_{in}^P\}$ and $Gbest = \{x_1^G, \dots, x_n^G\}$ be the best position of the i th particle and the group's best position so far, respectively. The velocity and position of each particle is updated as follows:

$$V_i^{k+1} = \omega \cdot V_i^k + c_1 \cdot m_1 \cdot (Pbest_i^k - X_i^k) + c_2 \cdot m_2 \cdot (Gbest^k - X_i^k) \quad (9)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \quad (10)$$

where V_i^k is the velocity of the i th particle at iteration k , ω is the inertia weight factor, c_1 and c_2 are the acceleration coefficients, m_1 and m_2 are random numbers between 0 and 1, and X_i^k is the position of the i th particle at iteration k . In the velocity updating process, the values of parameters such as ω , c_1 and c_2 should be determined in advance, which makes it cumbersome in solving large-scale optimization problems.

The BPSO, also introduced by Kennedy and Eberhart [20], enables the PSO to operate in binary spaces. The structure of the BPSO is effectively the same as that of the real-valued PSO. In BPSO, however, the position vector of a particle is a binary one. The velocity of the j th element in the i th particle is related to the possibility that the position of the particle takes a value of 1 or 0. It is implemented by defining an intermediate variable $S(v_{ij}^{k+1})$, called a sigmoid limiting transformation, as follows:

$$S(v_{ij}^{k+1}) = \frac{1}{1 + \exp(-v_{ij}^{k+1})} \quad (11)$$

The value of $S(v_{ij}^{k+1})$ can be interpreted as a probability threshold. If a random number m_{ij} , selected from a uniform distribution in $[0,1]$, is less than the value of $S(v_{ij}^{k+1})$ the position of the j th element in the i th particle at iteration $k+1$ (i.e., x_{ij}^{k+1}) is set to 1 and otherwise, set to 0. In the BPSO, therefore, (10) for modifying the position vector is replaced as follows:

$$x_{ij}^{k+1} = \begin{cases} 1 & \text{if } m_{ij} < S(v_{ij}^{k+1}) \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

IV. QUANTUM-INSPIRED BPSO ALGORITHM

A. Quantum Computing

The smallest unit of information stored in a quantum computer is called a quantum bit or *qubit* [24]. A qubit is analogous to a bit of storage in a traditional computer. A qubit may be in the "1" state, in the "0" state, or in any superposition of the two, while a bit in traditional computing can only hold a single state, either 0 or 1. To illustrate this, the traditional 0 and 1 values are written as $|0\rangle$ and $|1\rangle$, and the state of a qubit is represented as follows:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (13)$$

where α and β are complex numbers that specify the probability amplitudes of the corresponding states. Here, $|\alpha|^2$ and $|\beta|^2$ denote the probability that the qubit will be found in "0" state and "1" state, respectively. Normalization of the state to unity guarantees $|\alpha|^2 + |\beta|^2 = 1$. The state of a qubit can be changed by the operation with a quantum gate such as NOT gate, rotation gate, and Hadamard gate, etc [21].

In [24], Han and Kim suggested a novel QEA, inspired by the concept of quantum computing, in which a Q-bit representation is designed to represent a linear superposition of states (i.e., binary solutions). A *Q-bit* is defined as the smallest unit of information, which is defined with a pair of

numbers (α, β) as $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$, where $|\alpha|^2 + |\beta|^2 = 1$. A *Q-bit individual* as a string of n Q-bits is defined as

$$q = \begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_n \\ \beta_1 & \beta_2 & \dots & \beta_n \end{bmatrix} \quad (14)$$

where $|\alpha_j|^2 + |\beta_j|^2 = 1$, $j=1,2,\dots,n$. The Q-bit representation has the advantage of representing a linear superposition of states. All possible combinations of the decision variables can be derived from a single representation, while a system of n bits has 2^n possible single states in the classical computing. Evolutionary computing with Q-bit representation has a better characteristic of population diversity than other representations, since it can represent linear superposition of states probabilistically.

The following rotation gate is used as a variation operator, by which a Q-bit individual is updated.

$$U(\Delta\theta_j) = \begin{bmatrix} \cos(\Delta\theta_j) & -\sin(\Delta\theta_j) \\ \sin(\Delta\theta_j) & \cos(\Delta\theta_j) \end{bmatrix} \quad (15)$$

where $\Delta\theta_j$ is a rotation angle of the j th Q-bit toward either 0 or 1 state. The value of $\Delta\theta_j$ can be determined through a pre-defined lookup table [24].

B. BPSO with Quantum Computing

In the proposed QBPSO, the state of each element in a particle takes a value of 0 or 1 by the probability of $|\alpha|^2$ or $|\beta|^2$. In other words, the velocity update process (10) in the BPSO is replaced by the quantum computing. In the proposed QBPSO, therefore, an inertia weight factor (i.e., ω) and two acceleration coefficients (i.e., c_1 and c_2) can be removed, and only a rotation angle is added. The position vector of the i th particle (i.e., $X_i = \{x_{i1}, \dots, x_{in}\}$) is updated by probability of $|\beta_i|^2$ stored in the i th Q-bit individual (i.e., q_i). The j th element of the i th particle takes a value of 0 or 1 by the following:

$$x_{ij} = \begin{cases} 1 & \text{if } m_{ij} < |\beta_{ij}|^2 \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

for $i=1,2,\dots, NP$, $j=1,2,\dots,n$. Here, m_{ij} is the uniformly distributed random number between $[0,1]$ and NP is the population size.

To enhance the rotation gate for updating Q-bit individuals, this paper suggests an efficient rotation gate as a variation operator. The conventional rotation gate requires a pre-specified lookup table to determine the rotation angle $\Delta\theta$ to obtain new (α, β) . However, the proposed simplified rotation gate determines the rotation angle without the lookup table information since it uses the current position, $Pbest$, and $Gbest$ of a swarm as in the following (17). Therefore, each particle can approach to the optimum solution through its own experience and its neighbor's experiences.

$$\Delta\theta_{ij} = \theta \times \left\{ \gamma_{1i} \times (x_{ij}^p - x_{ij}) + \gamma_{2i} \times (x_j^G - x_{ij}) \right\} \quad (17)$$

where θ is the magnitude of rotation angle. γ_{1i} and γ_{2i} can be obtained by comparing the fitness of current position of particle i with those of $Pbest_i$ and $Gbest$, respectively, as follows:

$$\gamma_{1i} = \begin{cases} 0 & \text{if } f(X_i) \leq f(Pbest_i) \\ 1 & \text{otherwise} \end{cases} \quad (18)$$

$$\gamma_{2i} = \begin{cases} 0 & \text{if } f(X_i) \leq f(Gbest) \\ 1 & \text{otherwise} \end{cases} \quad (19)$$

where $f(\cdot)$ is the value of the object cost function.

The procedure of the proposed QBPSO algorithm can be summarized as the following pseudo-code:

```

Begin
  Initialize Q-bit individual and position of a population.
  Set initial  $Pbest$  and  $Gbest$ .
  Do while
    For  $i=1$  to Population size
      Update Q-bit individual of the  $i$ th particle.
      Modify position of the  $i$ th particle.
      Update  $Pbest$  of the  $i$ th particle.
    Next  $i$ 
    Update  $Gbest$ .
  Until termination criterion is met
End

```

C. Implementation of QBPSO for UC Problems

Since the UC problems involve determining the on/off states of generating units, the decision variables are the on/off status of generating units. If the j th generator in the i th particle at hour t is ON, the $x_{ij,t}$ is set to be 1. Otherwise $x_{ij,t}$ is set to be 0. After determining the optimal commitment scheduling, the optimal power outputs of the units are determined through the conventional economic dispatch (ED) procedure. Since the fuel cost function of a generating unit is approximately represented as the quadratic function as (1), ED problem can

be easily solved by numerical techniques. In the subsequent sections, the detailed procedures of QBPSO for scheduling the on/off states of units are described.

1) *Creating Initial Q-bit Individual and Position of Particles*: In the initialization process, $\alpha_{ij,t}^0$ and $\beta_{ij,t}^0$ of all Q-bit individuals are set to be $1/\sqrt{2}$. It means that a Q-bit individual represents the linear superposition of all possible states with the same probability. The initial position of a set of particles is determined by the probability stored in the initialized Q-bit individuals. After generating a random number $m_{ij,t}$, an initial value of the j th element in the i th particle at hour t (i.e., $x_{ij,t}^0$) takes a value of 1 if $r_{ij,t}$ is less than $1/2$, otherwise it is set to be 0. The initial $Pbest$ of each particle is set as its initial position. And the initial $Gbest$ is determined as the position of the particle with the minimum cost.

2) *Q-bit Individual Update*: Q-bit individuals are updated by the rotation gate as the following (20). The updated Q-bit should satisfy the normalization condition, $|\alpha_{ij,t}^{k+1}|^2 + |\beta_{ij,t}^{k+1}|^2 = 1$.

$$\begin{bmatrix} \alpha_{ij,t}^{k+1} \\ \beta_{ij,t}^{k+1} \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta_{ij,t}^{k+1}) & -\sin(\Delta\theta_{ij,t}^{k+1}) \\ \sin(\Delta\theta_{ij,t}^{k+1}) & \cos(\Delta\theta_{ij,t}^{k+1}) \end{bmatrix} \begin{bmatrix} \alpha_{ij,t}^k \\ \beta_{ij,t}^k \end{bmatrix} \quad (20)$$

where t is the index of time ($t=1,2,\dots,T$). The rotation angle (i.e., $\Delta\theta_{ij,t}^{k+1}$) is determined as follows:

$$\Delta\theta_{ij,t}^{k+1} = \theta \times \left\{ \gamma_{1i}^k \times (x_{ij,t}^{P,k} - x_{ij,t}^k) + \gamma_{2i}^k \times (x_{j,t}^{G,k} - x_{ij,t}^k) \right\} \quad (21)$$

3) *Modification of Position of Particles*: The position vector of the i th particle at iteration k (i.e., X_i^k) is modified by the probability stored in the i th Q-bit individual as follows:

$$x_{ij,t}^k = \begin{cases} 1 & \text{if } m_{ij,t} < |\beta_{ij,t}^k|^2 \\ 0 & \text{otherwise} \end{cases} \quad (22)$$

4) *Update of Pbest and Gbest*: If X_i^{k+1} yields a smaller cost function value than $Pbest_i^k$, then $Pbest_i^{k+1}$ is set to X_i^{k+1} . Otherwise, $Pbest_i^k$ is retained:

$$Pbest_i^{k+1} = \begin{cases} X_i^{k+1} & \text{if } f(X_i^{k+1}) \leq f(Pbest_i^k) \\ Pbest_i^k & \text{otherwise} \end{cases} \quad (23)$$

Also, $Gbest^{k+1}$ is set as the best evaluated position among the $Pbest_i^{k+1}$.

5) *Stopping criteria*: The proposed QBPSO algorithm is terminated if the iteration reaches a pre-specified maximum iteration.

D. Constraint-Handling Techniques

Penalty functions are the most popular method in heuristic optimization techniques when treating the constraints, due to its simple concept and convenience for implementation. However, these methods have certain weaknesses in that the penalty functions tend to be ill-behaved near the boundary of the feasible region when the penalty parameters are sufficiently large [25]. As an alternative to the penalty functions, this paper applies the rule-based heuristic constraint-handling techniques

for the minimum up/down time and the spinning reserve constraints [16]. In evolutionary process for solving UC problem, random bits flipping of state variables is occurred, thereby the constraints may be frequently violated. Therefore, heuristic-based repair algorithms are applied to accelerate the solution quality and to avoid infeasible solutions.

V. NUMERICAL TESTS

The proposed QBPSO is applied to the test systems of up to 100-units with 24-hour demand horizon. For each test case, 50 independent trials are conducted to compare the solution quality and convergence characteristics. Numerical tests have been executed on a Pentium IV 2.0GHz computer. In implementing the proposed algorithms, some parameters must be determined in advance. In this paper the parameters were set through experiments as follows:

- Population size $NP = 20$;
- Maximum iteration count $iter_{max} = 1,000$;
- Rotation angles $\theta = 0.05 \pi$.

The proposed QBPSO is initially tested on the simple 10-unit base system with a 24-hour time horizon. The unit characteristics of the 10-unit system and the demand are given in [10]. Subsequently, the 40-, 80-, and 100-unit data are obtained by duplicating the base case, and the load demands are adjusted in proportion to the system size. In all cases, the spinning reserve requirements are assumed to be 10% of the hourly demand.

In Table I, the best, average, worst costs, and standard deviation for test systems obtained by the proposed QBPSO algorithm are summarized and compared with those of the conventional BPSO. Here, the BPSO employed the same constraint treatment techniques used in the QBPSO. The simulation results show that the proposed QBPSO provides much better solutions than the BPSO.

TABLE I
SIMULATION RESULTS OF EACH METHOD FOR TEST SYSTEMS

Units	Method	Best Cost (\$)	Average Cost (\$)	Worst Cost (\$)	Standard Deviation
10	BPSO	563,977	563,977	563,977	0.00
	QBPSO	563,977	563,977	563,977	0.00
40	BPSO	2,243,210	2,244,634	2,245,982	450.57
	QBPSO	2,242,967	2,244,557	2,245,509	545.66
80	BPSO	4,487,388	4,488,725	4,489,793	538.02
	QBPSO	4,484,351	4,486,334	4,487,304	715.10
100	BPSO	5,608,172	5,609,705	5,611,005	701.69
	QBPSO	5,603,795	5,605,885	5,607,511	889.31

The convergence characteristics of the BPSO and QBPSO for 40-unit system are illustrated in Fig. 1. We have observed that the QBPSO was improving the solution quality continuously while the BPSO experienced a premature convergence. In Table II, the best results of the proposed QBPSO are compared with those of Lagrange relaxation (LR) [10], genetic algorithm (GA) [10], evolutionary programming (EP) [12], simulated annealing (SA) [15], and improved

particle swarm optimization (IPSO) [17]. Table II reveals that the proposed QBPSO is obviously superior to the existing methods. In the 100-unit system, the QBPSO can save the operating costs of \$15,489 in a 24-hour compared to the SA [15] which is the best solution until now.

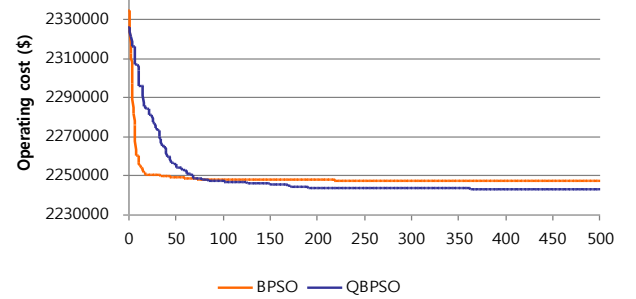


Fig. 1. Convergence characteristics of the BPSO and QBPSO for the 40-unit system.

TABLE II
COMPARISON OF BEST RESULTS OF EACH METHOD

Method	10-unit	40-unit	80-unit	100-unit
LR [10]	565,825	2,258,503	4,526,022	5,657,277
GA [10]	565,825	2,251,911	4,504,933	5,627,437
EP [12]	564,551	2,249,093	4,498,479	5,623,885
SA [15]	565,828	2,250,063	4,498,076	5,617,876
IPSO [17]	563,954	2,248,163	4,495,032	5,619,284
BPSO	563,977	2,243,210	4,487,388	5,608,172
QBPSO	563,977	2,242,967	4,484,351	5,603,795

For the 10-unit and 100-unit systems, the commitment schedules during the planning horizon obtained by the proposed QBPSO are described in Table III and Table IV, respectively.

TABLE III
UNIT SCHEDULING AND CORRESPONDING COSTS FOR THE 10-UNIT SYSTEM

Hr	Generation Output										Total Power	Fuel Cost	Startup Cost
	G1	G2	G3	G4	G5	G6	G7	G8	G9	G10			
1	455	245	0	0	0	0	0	0	0	0	700	13,683	0
2	455	295	0	0	0	0	0	0	0	0	750	14,554	0
3	455	370	0	0	25	0	0	0	0	0	850	16,809	900
4	455	455	0	0	40	0	0	0	0	0	950	18,598	0
5	455	390	0	130	25	0	0	0	0	0	1,000	20,020	560
6	455	360	130	130	25	0	0	0	0	0	1,100	22,387	1,100
7	455	410	130	130	25	0	0	0	0	0	1,150	23,262	0
8	455	455	130	130	30	0	0	0	0	0	1,200	24,150	0
9	455	455	130	130	85	20	25	0	0	0	1,300	27,251	860
10	455	455	130	130	162	33	25	10	0	0	1,400	30,058	60
11	455	455	130	130	162	73	25	10	10	0	1,450	31,916	60
12	455	455	130	130	162	80	25	43	10	10	1,500	33,890	60
13	455	455	130	130	162	33	25	10	0	0	1,400	30,058	0
14	455	455	130	130	85	20	25	0	0	0	1,300	27,251	0
15	455	455	130	130	30	0	0	0	0	0	1,200	24,150	0
16	455	310	130	130	25	0	0	0	0	0	1,050	21,514	0
17	455	260	130	130	25	0	0	0	0	0	1,000	20,642	0
18	455	360	130	130	25	0	0	0	0	0	1,100	22,387	0
19	455	455	130	130	30	0	0	0	0	0	1,200	24,150	0
20	455	455	130	130	162	33	25	10	0	0	1,400	30,058	490
21	455	455	130	130	85	20	25	0	0	0	1,300	27,251	0
22	455	455	0	0	145	20	25	0	0	0	1,100	22,736	0
23	455	420	0	0	25	0	0	0	0	0	900	17,685	0
24	455	345	0	0	0	0	0	0	0	0	800	15,427	0

