ERJ
Engineering Research Journal
Faculty of Engineering
Menoufiya University

UNIT COMMITMENT PROBLEM WITH RAMP RATE CONSTRAINT USING NEW HYBRIDIZED ADAPTIVE OPTIMIZATION METHOD

Ehab E. Elattar

Department of Electrical Engineering, Faculty of Engineering,
Menoufiya University, Shebin El-Kom, Egypt

ABSTRACT

Solving unit commitment (UC) problem is one of the most critical tasks in electric power system operations. Therefore, proposing an accurate method to solve this problem is of great interest. The original bacterial foraging (BF) optimization algorithm suffers from poor convergence characteristics for larger constrained optimization problems. In addition, the stopping criterion used in the original BF increases the computation burden of the original algorithm in many cases. To overcome these drawbacks, a hybridized adaptive bacterial foraging and genetic algorithm (HABFGA) approach is proposed in this paper to solve the unit commitment problem with ramp rate constraint. The HABFGA approach can be derived by combining adaptive stopping criterion, BF algorithm and genetic algorithm, so that the drawbacks of original BF algorithm can be treated before employing it to solve the complex UC problem. To illustrate the effectiveness of the HABFGA approach, several standard and real test systems with different numbers of generating units are used. The results of HABFGA approach are compared with the results obtained using other published methods employing same test systems. This comparison shows the effectiveness and the superiority of the proposed method over other published methods.

ان حل مشكلة تحديد مساهمة وحدات التوليد مع الأخذ فى الإعتبار قيود معدل المنحدر من الامور الهامة جدا فى تشغيل نظم القوى الكهربية، لذلك فإن إقتراح طريقه دقيقه لحل هذه المشكلة من الأمور ذات الإهتمام الكبير. طريقة البحث عن الطعام الجرثومى الأصلية تعانى من مشكلة التقارب البطيء. بالإضافة الى ان معيار التوقف المستخدم يزيد من زمن الحسابات للخوارزمية الأصلية. لحل هذه المشاكل فى هذا البحث تم إقتراح طريقة مثلى هجينه وذلك بدمج طريقة البحث عن الطعام الجرثومى التكيفيه مع الخوارزمية الجينية، وبالتالى يتم التغلب على عيوب طريقة البحث عن الطعام الجرثومى الأصلية قبل استخدامها فى حل مشكلة تحديد مساهمة وحدات التوليد المعقدة. تم تقييم أداء الطريقة المقترحة باستخدام أنظمة قياسية وأخرى حقيقية و مقارنة النتائج مع بعض الطرق المنشورة. أثبتت النتائج تفوق الطريقة المقترحة على الطرق الأخرى المستخدمة فى المقارنة.

Keywords: Unit commitment, ramp-rate constraint, adaptive bacterial foraging, hybridized adaptive bacterial foraging and genetic algorithm.

1. INTRODUCTION

It is very important to reduce the running costs of electric energy because of increasing the energy prices all over the world. The unit commitment (UC) problem can be defined as the problem of determining which generating units will be in service during a scheduling period with minimum operating cost. All committed units must satisfy the system demand and reserves subject to various constraints at minimum operating cost. UC is essential to the daily operation of modern power systems where the accurate UC schedule can save a lot of money by reducing the generation cost [1].

UC is a nonlinear mixed integer constrained optimization problem. The complexity of the UC problem increases when the dynamic constraints such as ramp rate constraint are considered. Ramp rate constraint is the maximum rate of change of the output in MW/min. The inclusion of this constraint requires the change of the range of output power for each generating unit at every time instant [2].

Various methods are proposed in the literature to solve the UC problem. These methods can be classified into three categories: single classical approaches, single non classical approaches (artificial intelligent based methods) and hybrid techniques. The former one includes many techniques such as: priority list (PL) [3], dynamic programming (DP) [4], Branch and Bound [5], integer/mixed integer programming [6] and Lagrangian relaxation (LR) [7]. The classical approaches are fast and simple. But, they are inadequacy in solving the nonlinear and non-convex search space of the UC [8].

The artificial intelligence (AI) techniques like tabu search (TS) [9], simulated annealing (SA) [10], particle swarm optimization (PSO) [11], genetic algorithms (GA) [12], shuffled frog leaping algorithm [13], artificial bee colony algorithm [14],

© Faculty of Engineering, Menoufiya University, Egypt

etc. attract the researchers, because of their ability to search not only for local optimal but also for global optimal. Although these methods impose no restriction on the problem formulation, these methods do not guarantee to find the global solution [1].

To solve the complex optimization problems such as the UC problem effectively, many hybrid methods are used in literature [1], [15]–[18]. The hybrid methods have some merits. They reduce the search space. In addition, they have better quality of solutions for small and large scale problems. Moreover, they give solution in an acceptable computation time. Finally, they can accommodate more constraints.

Since long time, GA is highly addressed to solve the UC problem [2], [19], [20]. GA is a search heuristic that mimics the process of natural selection. It generates solutions to the optimization problems using basic genetic operators such as selection, crossover and mutation. However, GA is successfully applied to several problems from different domains, it has a drawback. It gives a very fast initial convergence, followed by progressive slower improvements [21].

Recently, bacterial foraging (BF) algorithm has been proposed and employed to solve many optimization problems [22]–[25]. The BF algorithm is based on social and cooperative behaviors found in nature. The three main stages of original BF algorithm are, chemotaxis, reproduction and elimination-dispersal [22], [23]. However, the original BF has been applied to solve several real world optimization problems, it has a drawback. The original BF suffers from poor convergence characteristics especially for non-convex and larger constrained problems. Hence, in order to use it to solve the complex and non-convex optimization problem, this drawback should be treated first [26].

Because of UC problem is a nonlinear mixed integer constrained optimization problem, the research work in this area is still a challenge to the scholars. Therefore, in this paper, a hybrid adaptive bacterial foraging and genetic algorithm (HABFGA) is proposed to solve the UC problem with ramp rate constrained. The HABFGA approach uses the same operators as GA and BF. In addition, an adaptive stopping criterion is used in the HABFGA approach so, the maximum number of iterations can be chosen based on the improvement of the objective function.

By using HABFGA method, the global optimization capability can be improved and the delay in reaching the global solution can be reduced. The proposed method is evaluated using different standard and real test systems and compared with some published methods employing the same data. The contributions of this paper are: to propose a hybridized adaptive optimization algorithm by combining GA, BF algorithm and adaptive stopping criterion, to apply the proposed method to solve the UC problem with ramp rate constraint and to improve the UC problem solution compared to other published methods.

The paper is organized as follows: Section 2 gives the mathematical formulation of the UC problem. Section 3 gives a brief overview of GA and BF algorithm. The HABFGA method is described in Section 4. Experimental results and comparisons with other methods are presented in Section 5. Finally, Section 6 concludes the work.

2. UNIT COMMITMENT PROBLEM FORMULATION

2.1 Objective Function

Unit commitment aims to minimize the total production cost over a time horizon while satisfying equality and inequality constraints. The total production cost equal to the production cost plus the startup costs of all generating units over the entire time horizon [27].

The objective function of the UC problem can be defined as following:

$$\min_{P_i^t,u_i^t}\left(\sum_{t=1}^{T}\sum_{i=1}^{N}u_i^t\left[F_i(P_i^t)+S_i^t(1-u_i^{t-1})\right]\right) \qquad (1)$$

where P_i^t is the output power of unit *i* at period *t*, u_i^t is the commitment state of unit *i* at period *t* (on = 1 and off = 0), $F_i(P_i^t)$ is the fuel cost of unit *i* at output power P_i^t, S_i^t is the startup price of unit *i* at period *t*, *N* is the number of generating unit and *T* is the total number of scheduling periods.

Fuel cost function $F_i(P_i^t)$ is always defined using the following second order equation:

$$F_i(P_i^t)=a_i+b_iP_i^t+c_i(P_i^t)^2 \qquad (2)$$

where a_i, b_i, c_i are the cost coefficients for the generating unit *i*.

The start-up cost (S_i^t) of unit *i* at time *t* is usually expressed by the following form:

$$S_i^t=\begin{cases}HS^i & \text{if } X_{off,i}^t\le T_i^{down}+CH^i\\ CS^i & \text{if } X_{off,i}^t>T_i^{down}+CH^i\end{cases} \qquad (3)$$

where HS^i, CS^i is the unit's hot/cold star-tup cost, CH^i is the cold start hour, $X_{off,i}^t$ is the number of hours the unit *i* has been off line and T_i^{down} is the minimum down time of unit *i*.

2.2 Constraints

1) Generating limits:

$$u_i^t P_i^{\min} \le P_i^t \le u_i^t P_i^{\max} \quad (4)$$

2) System power balance:

$$\sum_{i=1}^{N} u_i^t P_i^t = D^t \quad (5)$$

where D^t is the customers' demand at time t ($t = 1,..., T$).

3) Spinning reserve: It can be modelled as follows:

$$\sum_{i=1}^{N} u_i^t P_i^{\max} \ge D^t + R^t \quad (6)$$

where R^t is the spinning reserve requirements.

4) Minimum up time:

$$\left(X_{on,i}^{t-1} - T_i^{up}\right)\left(u_i^{t-1} - u_i^t\right) \ge 0 \quad (7)$$

$$X_{on,i}^t = \left(X_{on,i}^{t-1} + 1\right)u_i^t \quad (8)$$

where, $X_{on,i}^t$ is the number of hours the unit has been on line and T_i^{up} is the minimum up time.

5) Minimum down time:

$$\left(X_{off,i}^{t-1} - T_i^{down}\right)\left(u_i^t - u_i^{t-1}\right) \ge 0 \quad (9)$$

$$X_{off,i}^t = \left(X_{off,i}^{t-1} + 1\right)\left(1 - u_i^t\right) \quad (10)$$

where, $X_{off,i}^t$ is the number of hours the unit has been off line and T_i^{down} is the minimum down time.

6) Up/down ramp rate limits: The output power of generator i at time t may affect its output in the next time. So, the ramp-rate limitation can be mathematically stated for any unit as follows:

$$P_i^t - P_i^{t-1} \le UR_i \quad (11)$$

$$P_i^{t-1} - P_i^t \le DR_i \quad (12)$$

where UR_i and DR_i are the ramp-up and ramp-down limits of generator i (*MW/h*), respectively, $i = 1, ..,N$ and $t = 1, ..., T$.

The generation capacity can be rewritten according to the ramp-rate limits as follows:

$$\max\left(P_i^{\min}, P_i^{t-1} - DR_i\right) \le P_i^t \quad (13)$$

$$\min\left(P_i^{\max}, P_i^{t-1} + UR_i\right) \ge P_i^t \quad (14)$$

3. GENETIC AND BACTERIAL FORAGING ALGORITHMS OVERVIEW

3.1 Genetic Algorithm

Genetic Algorithm (GA) which is invented by John Holland in the 1960's [21] is numerical optimization algorithm inspired by both natural selection and genetics. The first step of GA is generating a randomly population of individuals, which will be spread throughout the search space. There are no specific rules for the number of individuals in a population but it can be determined according to the nature of the problem. In general, the individuals can be encoded based on bits, labels, integers, real numbers, logic based rules and any other finite alphabet adequate to encode the solution space supplied with an en/decoder function.

Once a population is generated, a fitness function is used to evaluate the individuals and sort them according their fitness score. A fitness function is a function that assigns a quality measure to the individuals (chromosomes). Following this initial process the initial population of individuals is then processed by selection, crossover and mutation operators in order to drive the population towards the global optimum solution.

Selection is the next step of GA in which the better individuals based on their quality are chosen from a population to become parents of the next generation. Parent selection is responsible for pushing quality improvements. In the crossover step the useful segments of different parents are combined to form an offspring. The crossover step is followed by the mutation step. In the mutation step the offspring are randomly changed according to small rate called mutation rate. The mutation step is important to prevent loss of genetic diversity and premature convergence. After the application of these three operators, a new population will have been created and the number of generations is increased by one and the iterative cycle is repeated until a termination condition is reached [21], [28].

3.2 Bacterial Foraging Algorithm

Bacterial Foraging (BF) algorithm is accepted recently as a global optimization algorithm. The BF is based on social and cooperative behaviors found in nature [22]. The BF algorithm begins with creation of an initial population where its size equal to the number of bacteria. These bacteria try iteratively to reach a global optimum through four stages namely chemotaxis, swarming, reproduction and elimination dispersal [22].

The chemotaxis which is the first stage in the BF algorithm can be achieved through swimming and tumbling an E.coli cell via flagella. By changing between swimming and tumbling modes of motion, the bacterium spends its lifetime. A tumble can be

represented by a unit length $\phi(j)$ in a random direction. The step size in this random direction, $C(i, j)$, is depending on the dimensionality of the optimization problem and can be represented by the constant run-length unit.

Suppose that we want to minimize of the cost function $J(\theta)$, θ is the position of a bacterium, $\theta i(j, k, l)$ represents i^{th} bacterium at j^{th} chemotactic, k^{th} reproductive and l^{th} elimination dispersal step, $C(i)$ is the size of the step taken in the random direction specified by the tumble and $\phi(j)$ is the unit length random direction. Then the movement of the bacterium in the chemotaxis process can be represented by [22], [29]:

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\phi(j) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \quad (15)$$

where Δ indicates a vector in the random direction whose elements lie in [-1, 1].

Swarming is basically the cell to cell signaling stage. In this stage the E.coli cells congregate into groups and move as concentric patterns with high bacterial density. The following function presents the cell-to-cell signaling in E.coli swarm [29]:

$$J_{cc}(\theta, P(j,k,l)) = \sum_{i=1}^{S} J_{cc}^{i}(\theta,\theta^i) = \sum_{i=1}^{S}\left[-d_{attract}\exp\left(-w_{attract}\sum_{m=1}^{P}(\theta_m-\theta_m^i)^2\right)\right] + \sum_{i=1}^{S}\left[-h_{repellant}\exp\left(-w_{repellant}\sum_{m=1}^{P}(\theta_m-\theta_m^i)^2\right)\right] \quad (16)$$

where $J_{cc}(\theta, P(j, k, l))$ is a time varying function. This function is added to the actual cost function $(J(i, j, k, l))$ to present a time-varying fitness function, $d_{attract}$, $w_{attract}$, $h_{repellant}$ and $w_{repellant}$ are coefficients that represent the characteristics of the attractant and repellant signals released by the cell and θ_m^i is the m^{th} component of i^{th} bacterium position θ^i. $P(j, k, l)$ is the position of each bacterium of the population of the S bacteria and is defined as [26]:

$$P(j,k,l) = \{\theta^i(j,k,l),\ i = 1,2,...,S\} \quad (17)$$

where S is the size of the bacteria population.

In the reproduction stage, only the fitter bacteria which have lower value of the objective function, remains while the weaker bacteria eventually die. After that each of the fitter bacteria split into two bacteria at the same location thus keeps the swarm size constant. While in elimination and dispersal stage, some bacteria may be eliminated or disperse with a very small probability due to the sudden events. This avoids falling into premature convergence.

Suppose that S is the number of bacteria, N_c is chemotactic loop limit, N_s is swim loop limit, N_{re} is the reproduction loop limit, S_r is number of bacteria for reproduction, N_{ed} is elimination-dispersal loop limit, C_i is step sizes (they depend on the dimensionality of the problem) and p_{ed} is probability of elimination dispersal. Based on the above steps and parameters, Passino [22] proposed the BF algorithm which is summarized in Fig. 1 [22]. More details of the BF optimization algorithm can be found in [22], [30]–[32]

Begin
  Initialize input parameters
  Create a random initial swarm of bacteria $\theta^s(i,j,k)$ $\forall i,\ i=1,...,S$
  Evaluate $f(\theta^s(i,j,k))$ $\forall i,\ i=1,...,S$
  For k = 1 to N_{ed} **Do**
    For j = 1 to N_{re} **Do**
      For i = 1 to N_c **Do**
        For m=1 to S **Do**
          Perform the chemotaxis step
          (Tumble-swim or tumble-tumble)
          For bacteria $\theta^s(i,j,k)$
        End For
      End For
      Perform the reproduction step by eliminating the S_r (half) worst bacteria and duplicating the other half
    End For
    Perform the elimination-dispersal step for all Bacteria $\theta^s(i,j,k)$ $\forall i,\ i=1,...,S$ with probability $0 \le P_{ed} \le 1$
  End For
End

Fig.1 BF algorithm [22]

4. HYBRIDIZED ADAPTIVE BACTERIAL FORAGING AND GENETIC ALGORITHM (HABFGA)

However, the conventional BF has been applied to solve several real-world optimization problems, it has some drawbacks. The problem with BF as indicated in [22] is that if the steps taken by bacteria are too large, the bacteria will tend to leap out of the search space by swimming without stopping. But, if the step size values are too small, convergence can be slow.

In addition, the premature convergence to local minima can occur, if the number of chemotactic steps is too short or if the number of reproduction levels is not sufficient. To overcome these drawbacks and to be able to apply BF to solve the complex and high dimensioned search space, there are some trials to hybridize BF with different other algorithms [23], [33].

However, hybridizing BF with different other algorithms as in [23], [33] improve the accuracy of the conventional BF algorithm, they still have high computation burden. In BF algorithm, the stopping criterion determined based on the maximum number of the chemotactic steps, the reproduction steps and the elimination/dispersal events which increases the computation burden of the BF algorithm in some cases. Therefore, Farahat et. al. [26] applied an adaptive stopping criterion by adjusting the maximum number of iterations depending on the improvement of the cost function. In this criterion, the chemotaxis operation stops either when the solution is found that satisfies minimum criteria or when the fixed number of chemotactic steps is reached.

In order to treat the drawbacks of conventional BF algorithm and the hybridize BF with different other algorithms [23], [33], hybridized adaptive bacterial foraging and genetic algorithm (HABFGA) is proposed in this paper. The proposed algorithm can be derived by combining BF algorithm, GA and adaptive stopping criterion. In the proposed algorithm, there are no cell to cell attraction and repellent effects while the new individuals are generated via mutating all the dimensions from the eliminated bacteria.

The steps of the HABFGA algorithm are as follows:

Step 1: Initialize the bacterial foraging parameters:

S: number of bacteria in the population.

p: dimension of the search space.

N_s: the length of a swim when it is on a gradient

N_c: number of chemotactic steps taken by bacteria in its lifetime.

N_{re}: number of reproduction steps.

N_{ed}: number of elimination-dispersal events.

P_{ed}: probability of elimination-dispersal.

$C(i)$: the size of the step taken in the random direction specified by the tumble.

$\phi(j)$: random vector which elements lie in [-1,1].

θ_i: the initial random location of each bacterium

Step 2: Starting of elimination-dispersal loop: ($l = l + 1$).

Step 3: Starting of reproduction loop: ($k = k + 1$).

Step 4: Starting of chemotaxis loop: ($j = j + 1$).

- Take a chemotactic step for each bacterium (i).
- Then the fitness function: $J(i, j, k, l)$ can be calculated. Let $J_{last} = J(i, j, k, l)$. so that the lower cost could be found.
- For $i = 1, 2, ...S$, the tumbling/ swimming decision can be taken as following:

  Tumble: a random vector $\Delta(i) \in R^p$ is generated with each element $\Delta_m(i), m = 1, 2, ...,$ p being a random number on [-1, 1].

  Move: Let

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$$

  Fixed step size in the direction of tumble for bacterium i is considered.
- Swimming loop: Let $z = 0$ (z is a counter for swim length).

  While $z < N_s$

  Let $z = z + 1$
- If $J(i, j + 1, k, l) < J_{last}$, then let $J_{last} = J(i, j + 1, k, l)$. Let

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\phi(j)$$

  Then the new fitness function $J(i, j + 1, k, l)$ can be computed.

  Else, let $z = N_s$ (the end of the while statement).
- Go to next bacterium ($i = i + 1$ if $i \neq S$).
- Apply GA operators (crossover and mutation).

  A modified simple crossover [21], [23] is used as following:

$$\begin{aligned} \tilde{y}_j^u &= \lambda \bar{y}_j^v + (1-\lambda)\bar{y}_j^u \\ \tilde{y}_j^v &= \lambda \bar{y}_j^u + (1-\lambda)\bar{y}_j^v \end{aligned} \quad (18)$$

  where $\bar{y}_j^u, \bar{y}_j^v$ refers to parent's generations and $\tilde{y}_j^u, \tilde{y}_j^v$ refers to offspring's generations and j is the chromosome of j^{th} step and λ is the multiplier.

  The offspring produced in the crossover step is mutated with some probability to escape local minima [23]. Dynamic mutation [21] is used in the proposed approach as following:

$$y_j = \begin{cases} \tilde{y}_j + \Delta(k, y_j^{(U)} - \tilde{y}_j), & \tau = 0 \\ \tilde{y}_j - \Delta(y, \tilde{y} - y_j^{(L)}), & \tau = 1 \end{cases} \quad (19)$$

  where L and U are lower and upper domain bounds of the variable y_j , k is the generation number, the random constant τ becomes 0 or 1 and $\Delta(k, y_j)$ is given as:

$$\Delta(k, y_j) = y_j \times \eta \times \left(1 - \frac{k}{Z}\right)^b \quad (20)$$

  where, η is a random number from [0,1], Z is the maximum number of generations which is

defined by the user and b is a system parameter determining the degree of dependency on iteration number.

- After that the best (lower) cost obtained ($J_{best}(j)$) can be computed.
- The difference (D) in cost achieved in the current chemotactic step can be calculated:

  $D(j) = J_{best}(j) - J_{best}(j-1)$.
- If $j > N_c/v$ (e.g. $v = 2$).
- If $|D(j) - D(j-b)| < \varepsilon$, $b = 1, 2, \ldots b_m$ and $b_m < N_c/v$.

  $j = N_c$ (i.e. end chemotactic operations).

Step 5: If $j < N_c$, go to Step 4 (increase j by 1).

Step 6: Reproduction: Compute the health of the bacterium i as following:

$$J^i_{health} = \sum_{j=1}^{N_c+1} J(i, j, k, l) \qquad (21)$$

Sort the bacteria and chemotactic parameters $C(i)$ in ascending order where bacteria which have higher cost (higher cost means lower health) will die and the remaining bacteria reproduce.

Step 7: If $k < N_{re}$, go to Step 3 (increase k by 1), otherwise, go to Step 8.

Step 8: Elimination-Dispersal: The bacterium is eliminated and/or dispersed with probability P_{ed}. So, the number of bacteria in the population is kept constant.

Step 9: If $l < N_{ed}$, then go to step 2 (increase j by 1); otherwise end.

5. NUMERICAL RESULTS

Numerical results from different standard and real test systems are presented to show the effectiveness of the HABFGA algorithm. Because of the stochastic nature of HABFGA method, 30 independent runs with random initial solution for each run were conducted for each system and results (minimum, average, maximum) were calculated.

In the implementation of HABFGA method, many parameters should be selected. The selection of suitable values of these parameters is very important in improving the speed of convergence and solution's quality. The best values of these parameters for each system were selected from empirical tests by running the algorithm several times with different parameters combinations. The HABFGA method is implemented in Pentium 4 personal computer with 2.8 GHz clock frequency and 2 GB of random access memory using MATLAB R2012a.

5.1 Case 1: Using Standard Test Systems

To examine the performance of HABFGA method in comparison with some published methods, the HABFGA method was tested using different standard systems. These standard systems are 10, 20, 40, 60, 80 and 100 unit systems. The scheduling period is 24 h. The unit data for a 10-unit system and the load data for this system can be found in [34]. To get the data of the 20 unit problem, the data of 10 unit system were duplicated. Also, the load data of 10-unit system was multiplied by 2. The problem data were scaled appropriately for the problem with more units. These systems are considered in this paper because these systems are the most widely studied systems to solve UC problem in the literature [2].

In all systems, the spinning reserve is chosen to be constant and equal to 10% of the forecasted load demand at any time instant. In addition, when ramp rate constraint is considered both the ramp-up and ram-down limits of each generating unit are assumed to be 20% of the maximum limit of power output of that unit.

The HABFGA algorithm is applied to the 10, 20, 40, 69 80 and 100-unit systems 30 times with different initial solutions. Table 1 shows the best production costs, the worst production costs, the average production costs and the standard deviation, obtained over 30 independent runs for each power system. It can be observed that the standard deviation which demonstrates the small variation range of the total cost value obtained by the HABFGA method over 30 runs of a system is not so high, which illustrates the reliability of the method over different runs with different parameter settings. Also, one can notice that, by considering ramp rate a slight increase of the total cost of the system occurs.

The performance of HABFGA method without considering the ramp rate constraint is compared with genetic algorithms and tabu search (GA-TS) [35], Lagrangian relaxation and genetic algorithms (GA-LR) [36], differential evolution (DE) [37], ant colony (AC) [38], shuffled frog leaping algorithm (SFL) [39] and binary-real-coded genetic algorithm (BRGA) [2]. Table 2 shows the results of this comparison which shows that the HABFGA method yields enhanced results over other published methods.

In addition the performance of HABFGA method with considering the ramp rate constraint is compared with SA algorithm [10], improved priority list and enhanced particle swarm optimization (IPL-EPSO) [15] and BRGA [2]. This comparison is shown in Table 3. The results show the superiority of the HABFGA method over other published methods when considering the ramp rate constraint.

Table (1), The Solutions Obtained From 30 Independent Runs for Each Power System.

Number of units	Cost without the ramp rate constraint ($)				Cost with the ramp rate constraint ($)			
	Minimum	Average	Maximum	Standard deviation	Minimum	Average	Maximum	Standard deviation
10	560318	560458	560623	16	562022	5621968	562399	22
20	1121926	1122301	1122724	41	1124726	1125198	1125700	53
40	2241781	2242334	2243001	87	2248407	2249058	2249778	80
60	3360827	3361514	3362337	89	3370677	3371325	3372151	104
80	4481901	4482904	4484008	125	4494901	4496124	4497044	141
100	5601522	5603100	5604421	308	5617322	5619057	5620371	228

Table (2), The Solutions Obtained From 30 Independent Runs for Each Power System.

Method	Production cost in ($)					
	10-unit	20-unit	40-unit	60-unit	80-unit	100-unit
GA-TS [35]	561238	—	—	—	—	—
GA-LR [36]	564800	1122622	2242178	3371079	4501844	5613127
DE [37]	562921	1125546	2247570	3365125	4486010	—
AC [38]	563938	1123297	—	—	—	—
SFL [39]	564769	1123261	2246005	3368257	4503928	5624526
BRGA [2]	563938	1124290	2246165	3365431	4487766	5606811
HABFGA	560318	1121926	2241781	3360827	4481901	5601522

Table (3), Comparison of the Production Cost Produced by the HABFGA, With the Ramp Rate Constraint, With Those Produced by Some Other Published Techniques

System	Production Cost	Method			
		SA [10]	IPL-EPSO [15]	BRGA [2]	HABFGA
10-unit	Minimum ($)	–	–	565662	562022
	Average ($)	–	–	565858	5621968
	Maximum ($)	–	–	566073	562399
20-unit	Minimum ($)	–	1144278.52	1127110	1124726
	Average ($)	–	1144388.63	1127585	1125198
	Maximum ($)	–	1144578.68	1128103	1125700
40-unit	Minimum ($)	2255864	2288799.92	2252841	2248407
	Average ($)	2256971	2288833.43	2253592	2249058
	Maximum ($)	2258897	2288883.70	2254412	2249778
60-unit	Minimum ($)	–	–	3375332	3370677
	Average ($)	–	–	3376180	3371325
	Maximum ($)	–	–	3377106	3372151
80-unit	Minimum ($)	–	–	4500780	4494901
	Average ($)	–	–	4502043	4496124
	Maximum ($)	–	–	4503423	4497044
100-unit	Minimum ($)	–	–	5622688	5617322
	Average ($)	–	–	5624623	5619057
	Maximum ($)	–	–	5626737	5620371

The results show that the HABFGA method performs well despite the high dimensionality of the search space with or without considering the ramp rate constraint.

5.2 Case 2: Using real test systems

In this case two real test systems are used to show the effectiveness of the HABFGA method. The first system is from the practical Taiwan Power (Taipower) and consists of 38 generating units. The time horizon in this case is also 24 h. The details of this system can be found in [40]. In this system, start-up costs of units are constant. The maximum spinning reserve constraint is used as a constant and equal to 11% of the total power demand.

Table 4 shows the comparison of total production costs obtained from the HABFGA method and those obtained from constraint logic programming (CLP) [40], fuzzy optimization (FO) [41], matrix real coded genetic algorithm (MRCGA) [42], absolutely stochastic simulated annealing (ASSA) [43] and improved priority list and augmented Hopfield Lagrange neural network (IPLALH) [16] for two cases, with and without ramp rate constraints. The results show that the HABFGA method gives total production costs less than the total production costs obtained from other published methods in all cases.

Table (4), Comparison of the Total Production Cost by the HABFGA with those Produced by Some Other Published Techniques for 38 Unit System.

Problem	Method	Total Cost ($ x 10^6)
Neglecting ramp rate constraint	CLP [40]	208.1
	FO [41]	207.8
	MRCGA [42]	204.6
	ASSA [43]	196.70
	IPL-ALH [16]	196.31
	HABFGA	193.82
With ramp rate constraint	CLP [40]	213.8
	FO [41]	213.9
	MRCGA [42]	206.7
	ASSA [43]	198.84
	IPL-ALH [16]	197.05
	HABFGA	194.97

The second real system used in this paper which consists of 45 generating units is extracted from generation system mainland Spain. The details of this system can be found in [44]. As in the first system, the time horizon is 24 h and start-up costs of units are constant. While the maximum spinning reserve is set to be 10% of the power demand. Table 5 shows the comparison of total production costs obtained from the HABFGA method and those obtained from Parallel Repair Genetic Algorithm (PRGA) [44] and IPL-ALH [16]. From these results, one can notice the superiority of the HABFGA method over others.

Table (5), Comparison of the Total Production Cost by the HABFGA with those Produced by Some other Published Techniques for 45 Unit System.

Method	Total Cost ($)
PRGA [44]	1032415327.0
IPL-ALH [16]	1029104737.4
HABFGA	1024115931.0

From the above results (Tables 1- 5), the HABFGA method are compared with some published methods. The results of the published methods used in these comparisons have been directly quoted from their corresponding references. Observing the results obtained by the HABFGA method, the following remarks are made. By comparing of the total cost in these tables, the effectiveness and the superiority of the proposed method are clearly determined considering or without considering ramp rate constraint. Also, the HABFGA method obtains lower total production cost than other published methods using standard or real systems. In addition, the difference between minimum and maximum cost of the HABFGA method are small, which show the stability of the results obtained by the HABFGA method.

Based on the above results, the HABFGA method has high-speed convergence, but its computational burden (several minutes) is high compared with some published method. The real life UC problem is solved off line and solution time of several minutes is acceptable. This makes it possible to use the HABFGA method to solve the real life UC problem.

6. CONCLUSION

In this paper, an optimization method called HABFGA was employed to solve the UC problem considering the ramp rate constraint. The HABFGA approach can be derived by combining adaptive stopping criterion, BF algorithm and genetic algorithm, so that the drawbacks of original BF algorithm can be treated. The feasibility and efficiency of the proposed method have been demonstrated using the commonly used standard test systems and two real world test systems. The numerical results were compared with the recently reported approaches. The numerical results revealed that the UC solution obtained by the HABFGA method led to a smaller total generating cost than those obtained using other published methods, which

showed the ability of the HABFGA algorithm to find the global or near global solutions for the UC problem considering ramp rate constraint. These results show that the HABFGA method is a promising tool for solving UC problem in practical large power systems. Future work includes incorporating security issues and renewable energy sources as an extension to this work.

7. REFERENCES

[1] X. Yu and X. Zhang, "Unit commitment using Lagrangian relaxation and particle swarm optimization", Electrical Power and Energy Systems. vol. 61, pp.510–522, 2014.

[2] D. Datta, "Unit commitment problem with ramp rate constraint using a binary-real-coded genetic algorithm", Applied Soft Computing, vol. 13, pp. 3873–3883, 2013.

[3] T. Senjyu, K. Shimaukuro, K. Uezato and T. Funabashi, "A fast technique for unit commitment problem by extended priority list", IEEE Transactions on Power Systems, vol. 18, no. 3, pp. 882–888, 2003.

[4] Z. Ouyang and S. Shahidehpour, "An intelligent dynamic programming for unit commitment application", IEEE Transactions on Power apparatus and Systems, vol, 6, no. 3, pp. 1203–1209, 1991.

[5] A. Cohen and M. Yoshimura, "A Branch-and-Bound algorithm for unit commitment", IEEE Transactions on Power apparatus and Systems, vol. 12, no. 3, pp. 444–451, 1983.

[6] B. Venkatesh, T. Jamtsho and H. Gooi, "Unit commitment a fuzzy mixed integer Linear Programming solution", IET Generation, Transmission and Distribution, vol. 1, no. 5, pp. 836–846, 2007.

[7] D. Murtaza and S. Yamashiro, "Unit commitment scheduling by Lagrange relaxation method taking into account transmission losses", Electrical Engineering in Japan, vol. 152, no. 4, pp. 27–33, 2005.

[8] Z. Wu and T. Chow, "Binary neighbourhood field optimisation for unit commitment problems", IET Generation, Transmission and Distribution, vol. 7, no. 3, pp. 298–308, 2013.

[9] A. H. Mantawy, Y. L. Abdel-Magid and S. Z. Selim, "Unit commitment by tabu search", IEE Proceedings Generation Transmission and Distribution, vol. 145, no. 1, pp. 56–64 1998.

[10] D. Simopoulos and S. Kavatza, "Consideration of ramp rate constraints in unit commitment using simulated annealing", In: IEEE Power Tech. Conference, St. Petersburg, Russia, 2005. p. 1–7.

[11] A. Kherameh, M. Aien, M. Rashidinejad, M. Fotuhi-Firouzabad, "A particle swarm optimization approach for robust unit commitment with significant vehicle to grid penetration", In: Iranian Conference on Intelligent Systems (ICIS), Iran, 2014, pp. 1–6.

[12] M. Gupta and S. Haque, "Unit commitment using Micro Genetic Algorithm", In: International Conference on Engineering Nirma University, Ahmedabad, 2012. pp. 1–6.

[13] M. Barati and M. Farsangi, "Solving unit commitment problem by a binary shuffled frog leaping algorithm", IET Generation, Transmission and Distribution, vol. 8, no. 6, pp. 1050–1060, 2008.

[14] P. Singhal, R. Naresh and V. Sharma, "A modified binary artificial bee colony algorithm for ramp rate constrained unit commitment problem", International Transactions on Electrical Energy Systems, 2015, pp. in press.

[15] G. Weixun, "Ramp rate constrained unit commitment by improved priority list and enhanced particle swarm optimization", In: International Conference on Computational Intelligence and Software Engineering (CiSE). Wuhan, 2010. pp. 1–8.

[16] V. Dieu, W. Ongsakul, "Ramp rate constrained unit commitment by improved priority list and augmented Lagrange Hopfield network", Electric Power Systems Research. Vol. 78, pp. 291–301, 2008.

[17] S. Hosseini, H. Siahkali and Y. Ghalandaran, "Thermal unit commitment using hybrid binary particle swarm optimization and genetic algorithm", In: Asia-Pacific Power and Energy Engineering Conference (APPEEC). Shanghai, 2012. pp. 1–5.

[18] Sarjiya, A. Mulyawan, A. Setiawan and A. Sudiarso, "Thermal unit commitment solution using genetic algorithm combined with the principle of tabu search and priority list method", In: International Conference on Information Technology and Electrical Engineering (ICITEE). Yogyakarta, 2013, pp. 414–419.

[19] N. Amjady, A. Shirzadi, "Unit commitment using a new integer coded genetic algorithm", European Transactions on Electrical Power, vol. 19, no. 8, pp. 1161–1176, 2009.

[20] G. Dudek, "Genetic algorithm with integer representation of unit start-up and shut-down times for the unit commitment problem", European Transactions on Electrical Power, vol. 17, no. 5, pp. 500–511, 2007.

[21] Z. Michalewicz, "Genetic algorithms + data structures = Evolving programs", Springer-Verlag, Berlin Heidelberg, 1996.

[22] K. Passino, "Biomimicry of bacterial foraging for distributed optimization and control", IEEE Control Systems Magazine, vol. 22, no. 3, pp. 52–67, 2002.

[23] O. Verma, R. Garg and V. Bisht, "Optimal time-table generation by hybridized bacterial foraging and genetic algorithms", In: 2012 International Conference on Communication Systems and Network Technologies, 2012, pp. 919–923.

[24] O. Abedinia, N. Amjady, A. Ghasemi and Z. Hejrati, "Solution of economic load dispatch problem via hybrid particle swarm optimization with time-varying acceleration coefficients and bacteria foraging algorithm techniques", International Transactions on Electrical Energy Systems, vol. 23, no. 8, pp. 1504–1522, 2013.

[25] R. Pandi, A. Biswas, S. Dasgupta and K. Panigrahi, "A hybrid bacterial foraging and differential evolution algorithm for congestion management", European Transactions on Electrical Power, vol. 20, no. 7, pp. 862–871, 2010.

[26] I. Farhat, M. El-Hawary, "Dynamic adaptive bacterial foraging algorithm for optimum economic dispatch with valve-point effects and wind power", IET Generation, Transmission and Distribution, vol. 4, no. 9, pp. 989–999, 2010.

[27] X. Yuan, B. Jia, S. Zhang, H. Tian and Y. Hou, "A new approach for unit commitment problem via binary gravitational search algorithm", Applied Soft Computing, vol. 22, pp. 249–260, 2014.

[28] D. Coley, "An Introduction to Genetic Algorithms for Scientists and Engineers", World Scientific Publishing Co. Pte. Ltd., 2001.

[29] S. Dasgupta, S. Das, A. Abraham and A. Biswas, "Adaptive computational chemotaxis in bacterial foraging optimization: An analysis", IEEE Transactions on Evolutionary Computation, vol. 13, no. 4, pp. 919–941, 2009.

[30] S. Das, A. Biswas, S. Dasgupta and A. Abraham, "Bacterial foraging optimization algorithm: theoretical foundations, analysis and applications", Studies in Computational Intelligence, Springer, vol. 203, pp. 23–55, 2009.

[31] B. Panigrahi and V. Pandi, "Bacterial foraging optimisation: Nelder Mead hybrid algorithm for economic load dispatch", IET Gen., Transm. and Distrib., vol. 2, no. 4, pp. 556–565, 2008.

[32] E. M. Montes and B. H. Ocana, "Bacterial foraging for engineering design problems: Preliminary results", In: Fourth Mexican Conference on Evolutionary Computation (COMCEV2008). Guanajuato, Mexico; 2008, pp. 33–38.

[33] D. Kim, A. Abraham and J. Cho, "A hybrid genetic algorithm and bacterial foraging approach for global optimization", Information Sciences, vol. 177, pp. 3918–3937, 2007.

[34] S. A. Kazarlis, A. G. Bakirtzis and V. Petridis, "A genetic algorithm solution to the unit commitment problem", IEEE Transaction on Power Systems, vol. 11, pp. 11:83–92, 1996.

[35] M. Sudhakaran and P. Raj, "Integrating genetic algorithms and tabu search for unit commitment problem", International Journal of Engineering, Science and Technology, vol. 2, no. 1, pp. 57–69, 2010.

[36] C. Cheng, C. Liu and C. Liu, "Unit commitment by Lagrangian relaxation and genetic algorithms", IEEE Transactions on Power Systems, vol. 15, no. 2, pp. 707–714, 2000.

[37] C. Chang, "An improved differential evolution scheme for the solution of large scale unit commitment problems", Informatica, vol. 21, no. 2, pp. 175–190, 2010.

[38] K. Vaisakh and L. Srinivas, "Evolving ant colony optimization based unit commitment", Applied Soft Computing, vol. 11, no. 2, pp. 2863–2870, 2011.

[39] J. Ebrahimi, S. Hosseinian and G. Gharehpetian, "Unit commitment problem solution using shuffled frog leaping algorithm", IEEE Transactions on Power Systems, vol. 26, no. 2, pp. 573–581, 2011.

[40] K. Huang, H. Yang and C. Huang, "A new thermal unit commitment approach using constraint logic programming", IEEE Transactions on Power Systems, vol. 13, no. 3, pp. 936–945, 1998.

[41] M. El-Saadawi, M. Tantawi and E. Tawfik, "A fuzzy optimization-based approach to large scale thermal unit commitment", Electric Power Syst. Res., vol. 72, pp. 245–252, 2004.

[42] L. Sun, Y. Zhang and C. Jiang, "A matrix real-coded genetic algorithm to the unit commitment problem", Electric Power Syst. Res., vol. 76, pp. 716–728, 2006.

[43] A. Saber, T. Senjyu, T. Miyagi, N. Urasaki and T. Funabashi, "Fuzzy unit commitment scheduling using absolutely stochastic simulated annealing", IEEE Transactions on Power Systems, vol. 21, pp. 955–964, 2006.

[44] J. Arroyo and A. Conejo, "A parallel repair genetic algorithm to solve the unit commitment problem", IEEE Transactions on Power Systems, vol. 17, no. 4, pp. 1216–1224, 2002.