Simulated Annealing Combined with Evolutionary Algorithm to Unit Commitment Problem

Grzegorz Dudek
Electrical Engineering Department
Czestochowa University of Technology, Poland

1 Introduction

Unit commitment (UC) is the process of determining the optimal set of generating units and their generation levels within a power system to satisfy the required demand and system operating constraints at any time. The scheduling period is from a day to a week. This problem is defined as a non-linear, mixed-integer combinatorial optimization problem. UC in the daily operation and planning of the power system may save the electric utilities millions of dollars per year in production costs.

The optimal solution to the problem can be obtained only by the complete enumeration method, which is, however, useless for realistic power systems because of the immense size of the solution space. Various optimization methods have been employed to approach the UC problem, such as the priority ordering methods (Lee, 1988; Senjyu *et al*., 2007), dynamic programming (Pang *et al*., 1981; Kumara & Palanisamy, 2007), Lagrangian relaxation (Merlin & Sandrin, 1983; Zhuang & Galiana, 1988), the branch-and-bound method (Cohen & Yoshimura, 1983), and the integer and mixed-integer programming (Garver, 1962) (a detailed literature synopsis is summarized in (Sheblé & Fahd, 1994; Sen & Kothari, 1998; Yamin, 2004)). Among these methods, the priority list is easy to implement and the simplest, but the quality of the solution is usually far from optimal due to the incomplete search of the solution space. Many classical methods such as branch-and-bound, dynamic and integer programming suffer from the "curse of dimensionality" because the problem size and the solution time increase rapidly with the number of generating units to be committed. To reduce the search space several approaches have been adopted. Most of them are based on the priority list technique (Pang *et al*., 1981), thus the solution obtained is suboptimal. The Lagrangian relaxation approach, compared with other methods, has higher computational efficiency and is more flexible for handling different types of constraint. However, because of the dual nature of the algorithm, its primary difficulty is associated with obtaining solution feasibility. Furthermore, the optimal value of the dual problem is not generally equal to that of the primal (original) problem.

Another class of methods applied to the UC problem are the artificial intelligence methods such as the expert systems (Padhy *et al*., 1997; Padhy, 2000; Ouyang & Shahidehpour, 2004), neural networks (Padhy *et al*., 1997; Wong *et al*., 2000; Kumara & Palanisamy, 2007), fuzzy logic (Padhy *et al*., 1997; Padhy, 2000; Saber *et al*., 2007), genetic algorithms (Dasgupta & McGregor, 1994; Kazarlis *et al*., 1996; Dudek, 2004; Dang & Li, 2007) and simulated annealing (Mantawy *et al*., 1998; Wong, 1998; Saber *et al*., 2007; Dudek, 2010). In the expert system approach, the knowledge of experienced power system operators and UC experts is combined to create an expert system rule base. However, a great deal of operator interaction is required in this approach, making it inconvenient and time-consuming. Neural networks based on a database holding typical load curves and corresponding UC schedules are trained to recognize the most economical UC schedule associated with the pattern of the current load curve. The fuzzy approach allows taking into account many uncertainties involved in the planning and operation of power systems. The key factors such as load demand and reserve margin are treated as fuzzy variables. A fuzzy decision system has been developed to select the units to be on or off based on these fuzzy variables. Genetic algorithms (GA) represent a class of stochastic, adaptive and parallel search techniques based on the mechanism of natural selection and genetics. They search from a population of individuals and use probabilistic transition rules. By adding problem-specific genetic operators and by the proper choice of variables and their representation, good near-optimal solutions to the UC problem can be obtained. Simulated annealing (SA) is a general-purpose stochastic optimization method, principally for combinatorial opti-

mization problems such as UC, which has been theoretically proved to converge with the optimum solution with probability 1. The main advantages of this method are that a complicated mathematical model of the problem under study is not needed, the starting point can be any given solution and the algorithm will attempt to improve the solution, the final solution does not strongly depend on the initial solution, and SA does not need large computer memory. One main drawback and limiting factor of this method is that it takes a great deal of CPU time to find the near-optimal solution. In order to improve the performance, SA is combined with other methods: the genetic algorithm (Mantawy *et al*., 1999; Yin & Wong, 2001; Cheng *et al*., 2002), the neural network (Nayak & Sharma, 2000), the tabu search (Mantawy *et al*., 1999), and the evolutionary programming (Rajan & Mohan, 2007).

This paper presents an optimization model for the unit commitment problem using a stochastic hybrid algorithm combining simulated annealing and evolutionary algorithm. Heuristic genetic operators: mutation, recombination and transposition, a method of the elimination of cost function constraints, and an annealing scheme are proposed. The cost function includes generation costs and start-up costs. The constraints are: the load demand constraint, units generation capability limits, units minimum up and down times and the spinning reserve. The efficiency of the proposed method is illustrated with an example described in Section 4. Results for comparative computations using simulated annealing, Monte Carlo method and the limit time characteristic method are reported in the same section.

2 The Mathematical Model of Unit Commitment

The UC problem can be mathematically formulated as follows (list of symbols is given in the Appendix):

Objective function:

$$F = \sum_{t=1}^{T}\sum_{i=1}^{N}\left(\alpha_i(t)C_i[P_i(t)] + \alpha_i(t)[1-\alpha_i(t-1)]SC_i(t_{offi})\right). \tag{1}$$

Constraints:

a) Load Balance

$$\forall t: \quad \sum_{i=1}^{N}\alpha_i(t)P_i(t) = D(t). \tag{2}$$

b) Unit Power Generation Limits

$$\forall i,t: \quad \alpha_i(t)P_{\min i} \le P_i(t) \le \alpha_i(t)P_{\max i}. \tag{3}$$

c) Set of Unit Power Generation Limits

$$\forall t: \quad \sum_{i=1}^{N}\alpha_i(t)P_{\min i} \le D(t), \tag{4}$$

$$\forall t: \quad \sum_{i=1}^{N}\alpha_i(t)P_{\max i} \ge D(t) + R(t). \tag{5}$$

d) Minimum Up/Down Time

$$\forall i : t_{offi} \geq t_{downi}, \tag{6}$$

$$\forall i : t_{oni} \geq t_{upi}, \tag{7}$$

where the variable production cost of unit i at time t, $C_i[P_i(t)]$ is conventionally approximated by the quadratic function:

$$C_i(P_i) = a_i P_i^2 + b_i P_i + c_i, \tag{8}$$

and the start-up cost of unit i, $SC_i(t_{offi})$ is expressed as a function of the number of hours the unit has been down:

$$SC_i(t_{offi}) = e_i \exp(-g_i t_{offi}) + f_i \exp(-h_i t_{offi}). \tag{9}$$

To take into account the costs connected with unit shut-down at time t, in the event that it remains in an off state to the end of time period T, it is assumed that:

- unit start-up costs are evenly distributed over the number of hours of unit down time,
- unit start-up occurs at time τ after the end of the optimization period T ($\tau \in \{1, 2, 3, ...\}$).

Taking these assumptions into account, start-up costs in time period T for unit staying in down time until the end of time period T are calculated using the formula:

$$SC_i(T-t) = \frac{SC_i(T-t+\tau)}{T-t+\tau}(T-t). \tag{10}$$

3 The Proposed Hybrid Approach to Unit Commitment

The hybrid algorithm SA+EA is proposed combining features of the simulated annealing and evolutionary algorithm. SA+EA has a structure of SA, i.e. the inner-loop where the candidate solutions (chromosomes) are generated and the outer-loop where the temperature is updated and the chromosomes exchange information. There is many parallel annealing processes, i.e. the solution space is searched by a population of chromosomes (as in EA), which are modified independently by means of move operators: mutation and transposition. The Kirkpatrick cooling schedule specifying an initial value of the temperature and a temperature update function was applied (Kirkpatrick *et al.*, 1983). The chromosomes returned by the inner-loop exchange information among themselves in the process of recombination. The infeasible solutions are repaired or penalized. Initial population is generated randomly. The flowchart of SA+EA is shown in Fig. 1.

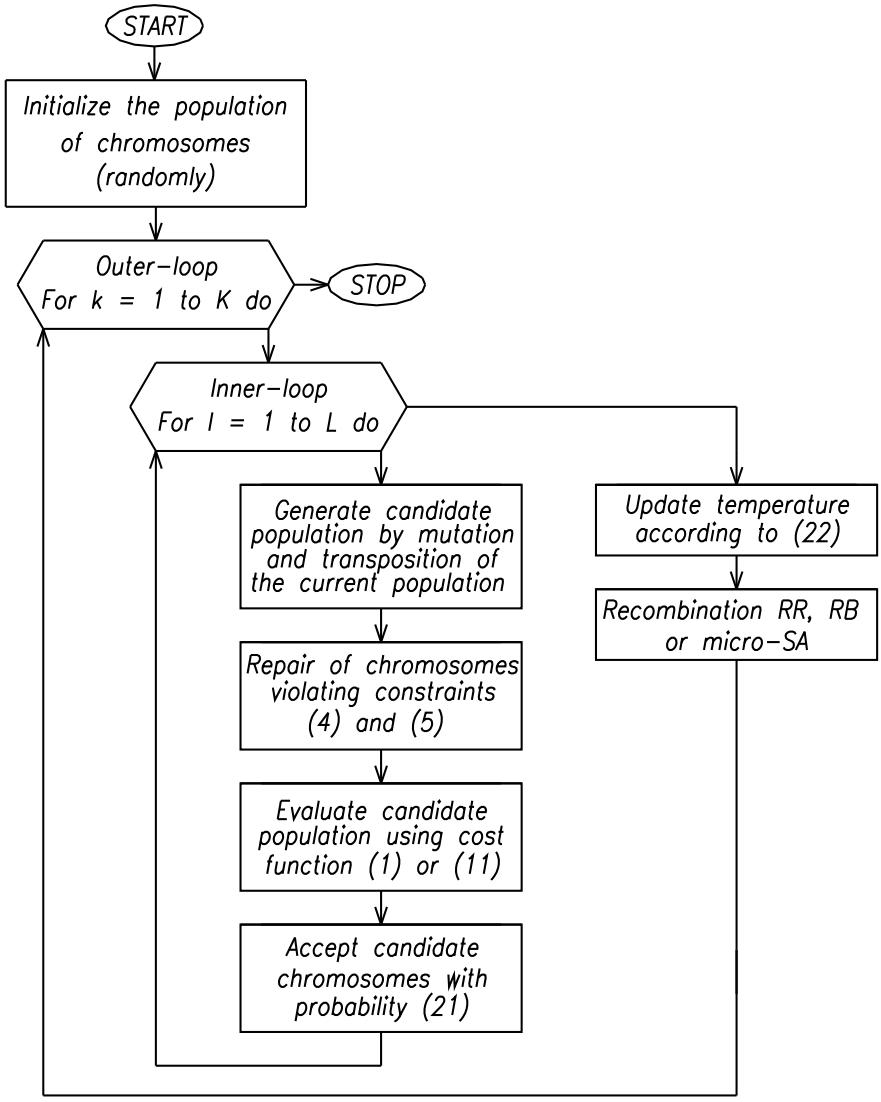


Figure 1: The flowchart of the proposed SA+EA algorithm.

3.1 Decision Variables and their Representation

The decision variables are on/off unit states $\alpha_i(t)$. The solution string (chromosome) is composed of bits representing all unit states at the following hours of the optimization period T: $x=(\alpha_1(1), \alpha_1(2),\ldots, \alpha_N(T))$. For the N units and T hours the string has $N\cdot T$ bits and the size of the solution space is $2^{N\cdot T}$.

Based on the solution x the generation levels of units $P_i(t)$ are determined using the economic dispatch procedure.

3.2 The Evaluation Procedure of Solutions

If solutions violate a set of unit power generation limit constraints, (4) or (5), the following repair algorithm is applied. Let $\Omega_1(t)$ be a set of units in on status at the t-th hour and let $\Omega_0(t)$ be a set of units in off status at the t-th hour (these sets are determined by the solution string). If constraint (4) is not met at the t-th hour, one unit i is chosen from the $\Omega_1(t)$ and its status is reset to off at the t-th hour (e.g. a bit in the solution string representing the status of unit i at the t-th hour is changed from 1 to 0). Similarly, if constraint (5) is not met, the status of one unit j from the $\Omega_0(t)$ is changed. Repair can be greedy – i is the most economical unit, e.g. one having the lowest incremental cost at full load, whereas j is the least economical unit, or random – units i and j are chosen at random from sets $\Omega_1(t)$ and $\Omega_0(t)$ respectively. The repair algorithm is activated for every hour t, until the moment constraints (4) and (5) are met.

For solutions which violate the minimum up/down time constraints (6) or (7), a penalty function is created (Dudek, 2003):

$$F' = M\left(1 + m\sum_{i=1}^{N}[g(i) + h(i)]\right), \tag{11}$$

where $g(i)$ is calculated as follows:

$$g(i) = \sum_{k=1}^{n_{downi}} \beta_i(k)[t_{downi} - t_{offi}(k)]. \tag{12}$$

$\beta_i(k)$ is expressed as follows:

$$\beta_i(k) = \begin{cases} 1 & \text{if } t_{offi}(k) < t_{downi}, \\ 0 & \text{if } t_{offi}(k) \geq t_{downi} \vee \tau_{oni}(k) > T. \end{cases} \tag{13}$$

$h(i)$ is given by:

$$h(i) = \sum_{k=1}^{n_{upi}} \gamma_i(k)[t_{upi} - t_{oni}(k)]. \tag{14}$$

$\gamma_i(k)$ is calculated as follows:

$$\gamma_i(k) = \begin{cases} 1 & \text{if } t_{oni}(k) < t_{upi}, \\ 0 & \text{if } t_{oni}(k) \geq t_{upi} \vee \tau_{offi}(k) > T. \end{cases} \tag{15}$$

The substitute cost function (11) is linearly dependent on the level of violation of constraints (6) and (7), and ensures a worse evaluation of individuals violating constraints (6) or (7) from feasible individuals. At the starting phase of the searching process the level of violation of the constraints (6) and (7) is minimized. At a certain point in the process chromosomes that are feasible according to these constraints start to appear and become the majority in the population due to their lower costs.

The parameter m in (11) is chosen so as to ensure the similar variation ranges of cost functions for feasible and infeasible solutions (1) and (11): $\max(F) - \min(F) \approx \max(F') - \min(F')$, where:

$$\max(F') = M\left(1 + m\frac{T}{2}\sum_{i=1}^{N}(t_{downi} + t_{upi} - 2)\right), \tag{16}$$

$$\min(F') \approx \max(F) \approx M, \tag{17}$$

$$\min(F) = \widehat{F}_{\min}. \tag{18}$$

Hence:

$$m = \frac{M - \widehat{F}_{\min}}{M\frac{T}{2}\sum_{i=1}^{N}(t_{downi} + t_{upi} - 2)}. \tag{19}$$

The constant M should be higher than the estimated maximal value of the objective function (1). Here M is expressed as below:

$$M = T\sum_{i=1}^{N} C_i(P_{\max i}). \tag{20}$$

Formula (20) corresponds to the situation when all units are loaded at full power at any hour of period T.

Formula (16) represents the substitute cost of the worst possible solution violating constraints (6) or (7), i.e. such solution in which units are alternately turned off and switched on in the successive hours.

To determine the generation levels of units P_i for each chromosome satisfying constraints (4) – (7) the economic dispatch (ED) procedure is performed. ED is a computational process where the total required generation is distributed among the generation units (the set of generating units is encoded in the chromosome) at the lowest possible cost. ED is performed for each hour t and determines the power output of each unit. Since the production cost (8) is a quadratic function (convex and continuous), the economic dispatch problem is solved using a method of Lagrange multipliers leading to the lambda-iteration procedure (Wood & Wollenberg, 1996), based on the principle of equal incremental cost. Lambda-iteration method is used for various generating unit schedules obtained by the SA+EA. Generation levels $P_i(t)$ determined in this procedure are used to calculate unit production costs (8) and the value of objective function (1). This method guarantees that unit power generation limit constraints (3) are met if the set of unit power generation limit constraints (4) and (5) are met.

3.3 The Annealing Schedule

The basic form of SA algorithm is the following:

1. Generate the initial solution x.
2. Calculate the cost of the initial solution x: $\Phi(x)$.
3. For $k = 1$ to K do
   - 3.1. For $l = 1$ to L do
     - 3.1.1. Generate the candidate solution x' from the neighborhood of the current solution x (move operation).
     - 3.1.2. Calculate the cost of the candidate solution x': $\Phi(x')$.
     - 3.1.3. Calculate the difference in costs of x and x': $\Delta = \Phi(x') - \Phi(x)$.

3.1.4. Accept the candidate solution ($x = x'$) with probability (for the minimization):

$$p_a(\Delta, T_k) = \min\left(1, \exp\left(\frac{-\Delta}{T_k}\right)\right). \tag{21}$$

3.2. Update the temperature T.

Parameters defining the cooling strategy (initial temperature T_1, the temperature update function, the number of the outer and inner-loop iterations K and L) are often chosen heuristically. The initial temperature is a parameter strictly dependent on the problem and according to the recommendations of the authors (Kirkpatrick *et al*., 1983) should be high enough to ensure acceptance of all candidate solutions. The acceptance probability of the candidate solution which is worse than the current solution x, at the end of the process of annealing approaches zero.

In the Kirkpatrick scheme the temperature changes in successive iterations k according to the geometric progression (Kirkpatrick *et al*., 1983):

$$T_{k+1} = r_T T_k, \qquad k = 1,2,...,(K-1), \tag{22}$$

where $r_T < 1$ is a temperature reduction factor. Assuming the acceptance probability of the worst candidate solution at the beginning of the annealing process: $p_a(\Delta_{\max}, T_1) = 0{,}99$, from (21) we get:

$$T_1 = \frac{\Delta_{\max}}{-\ln p_a(\Delta_{\max}, T_1)} \approx 100\Delta_{\max}, \tag{23}$$

where $\Delta_{\max}$ is the highest value of the difference in costs that occurs at the transition from the current solution to the candidate one.

It is assumed that in the final phase of the process, for $k > \kappa = 0.8K$, the acceptance probability of the candidate solution worse than the current one is less than 0.1, hence the temperature of the process advanced in 80% is:

$$T_\kappa = \frac{\Delta_{\min}}{-\ln p_a(\Delta_{\min}, T_\kappa)} \approx 0{,}43\Delta_{\min}, \tag{24}$$

where $\Delta_{\min}$ is a constant smaller than the smallest expected difference between cost of x and x'. Formula (22) can be written as:

$$T_{k+1} = r_T^k T_1, \qquad k = 1,2,...,(K-1). \tag{25}$$

Hence:

$$r_T^\kappa = \frac{T_\kappa}{T_1}. \tag{26}$$

Substituting (23) and (24) to (26) we get the formula for the temperature reduction factor:

$$r_T = \left(0{,}0043\frac{\Delta_{\min}}{\Delta_{\max}}\right)^{\frac{1}{\kappa-1}}. \tag{27}$$

The biggest change in cost takes place at the transition from the solution close to the optimal to the worst possible infeasible solution. Using (16) and (18) this value may be estimated as follows:

$$\Delta_{\max} = M\left(1 + m\frac{T}{2}\sum_{i=1}^{N}(t_{downi} + t_{upi} - 2)\right) - \hat{F}_{\min}. \tag{28}$$

3.4 Mutation

Using the classic mutation method, the probability of bit mutation (bits represent on/off states of units) does not depend on the unit production cost, its start-up cost, or load demand. Therefore this operator will turn off economical units at peak load as well as less economical units at minimum value of the load curve with the same probability. This leads the algorithm to "wander" and results in a much less effective search of the solution space. In the proposed method of mutation, probability of mutation is made dependent on the necessity of meeting the load demand of the number of units, cost of unit production and its start-up costs (Dudek, 2003). The probability of a bit change from 0 to 1, dependent on the number of units necessary to meet load demand at moment t is calculated in the formula:

$$p_{up1}(t) = q_1 + (1 - q_1)\frac{n_{\min}(t) + n_{\max}(t)}{2N} \qquad 1 \le t \le T, \tag{29}$$

where $q_1 \le p_{up1} \le 1$.

If through L_1 we denote a list of units sorted in decreasing order in terms of their upper generation limit $P_{\max}$, then the minimum number of units necessary to meet load demand is obtained by summing the $P_{\max}$ of succeeding units from list L_1 until the sum exceeds load demand and the spinning reserve:

$$n_{\min}(t) = \min\left\{n : \sum_{i=L_1(1)}^{L_1(n)} P_{\max i} \ge D(t) + R(t)\right\}, \tag{30}$$

where n is the auxiliary variable which denotes the number of units.

The maximum number of units necessary to meet load demand is obtained as follows: a list L_2 of units sorted in ascending order in terms of their lower generation limit $P_{\min}$ is introduced. The $P_{\min}$ of succeeding units from list L_2 is summed which gives the maximum number of units n, at which the sum does not exceed load demand:

$$n_{\max}(t) = \max\left\{n : \sum_{i=L_2(1)}^{L_2(n)} P_{\min i} \le D(t)\right\}. \tag{31}$$

Parameter q_1 in formula (29) has the function of limiting the range of probability p_{up1}.

The probability of a bit change from 0 to 1 is dependent on unit production costs as follows:

$$p_{up2}(i) = q_2 + (1 - q_2)\frac{u_i - u_{\min}}{u_{\max} - u_{\min}} \qquad 1 \le i \le N, \tag{32}$$

where:

$$u_i = \frac{\min\limits_{j=1,2,\ldots,N}\left\{C_j(P_{\max j})/P_{\max j}\right\}}{C_i(P_{\max i})/P_{\max i}}, \tag{33}$$

$$u_{\min} = \min_{i=1,2,\ldots,N}\{u_i\}, \tag{34}$$

$$u_{\max} = \max_{i=1,2,\ldots,N}\{u_i\}, \tag{35}$$

and $q_2 \le p_{up2} \le 1$.

The probability p_{up2} has the minimum value, equal to q_2 for units of the greatest production cost per unit at maximum load, and the maximum value, equal to 1, for units of the lowest production cost per unit at maximum load.

The joint probability of bit mutation, representing the state of unit i at moment t, from 0 to 1, is given by:

$$p_{up}(i,t) = \begin{cases} 1 & \text{if } p_{up1}(t) + p_{up2}(i) - \dfrac{1+q_2}{2} > 1, \\ 0 & \text{if } p_{up1}(t) + p_{up2}(i) - \dfrac{1+q_2}{2} < 0, \\ p_{up1}(t) + p_{up2}(i) - \dfrac{1+q_2}{2} & \text{otherwise,} \end{cases} \tag{36}$$

where

$$\max\left(0, q_1 - \frac{1-q_2}{2}\right) \le p_{up} \le 1 .$$

An analogous probability of bit change from 1 to 0, denoting a unit being turned off, is dependent on the number of units necessary to meet load demand according to the formula:

$$p_{down1}(t) = 1 - (1-r_1)\frac{n_{\min}(t) + n_{\max}(t)}{2N} \qquad 1 \le t \le T, \tag{37}$$

where $r_1 \le p_{down1} \le 1$.

The probability of bit change from 1 to 0, depending on the production costs of unit i are obtained using the formula:

$$p_{down2}(i) = 1 - (1-r_2)\frac{u_i - u_{\min}}{u_{\max} - u_{\min}} \qquad 1 \le i \le N, \tag{38}$$

where $r_2 \le p_{down2} \le 1$.

For units of the lowest production cost per unit at maximum load the probability p_{down2} assumes the maximum value, equal to 1, whereas for units of the highest production cost per unit at maximum load the probability p_{down2} assumes minimum value of r_2.

The dependence of the probability of bit mutation from 1 to 0 on unit i start-up cost is defined by the formula:

$$p_{down3}(i) = r_3 + (1-r_3)\frac{v_i - v_{\min}}{v_{\max} - v_{\min}} \qquad 1 \le i \le N, \tag{39}$$

where:

$$v_i = \frac{\min\limits_{j=1,2,\ldots,N}\{SC_j(t_{offx})\}}{SC_i(t_{offx})}, \tag{40}$$

$$v_{\min} = \min_{i=1,2,\ldots,N}\{v_i\}, \tag{41}$$

$$v_{\max} = \max_{i=1,2,\ldots,N}\{v_i\}, \tag{42}$$

and $r_3 \le p_{down3} \le 1$.

The probability p_{down3} assumes a minimum value of r_3 for units with the highest start-up costs after down time t_{offx}, and a maximum value of 1 for units with the lowest start-up costs.

The joint probability of bit mutation, representing the state of unit i at moment t, from 1 to 0, is calculated as follows:

$$p_{down}(i,t) = \begin{cases} 1 \ \text{ if } p_{down1}(t) + p_{down2}(i) + p_{down3}(i) - \dfrac{1+r_2}{2} - \dfrac{1+r_3}{2} > 1, \\ 0 \ \text{ if } p_{down1}(t) + p_{down2}(i) + p_{down3}(i) - \dfrac{1+r_2}{2} - \dfrac{1+r_3}{2} < 0, \\ p_{down1}(t) + p_{down2}(i) + p_{down3}(i) - \dfrac{1+r_2}{2} - \dfrac{1+r_3}{2} \ \text{ otherwise}, \end{cases} \tag{43}$$

where

$$\max\left(0, r_1 - \frac{1-r_2}{2} - \frac{1-r_3}{2}\right) \le p_{down} \le 1 .$$

The values of parameters q_1, q_2, r_1, r_2, r_3 and t_{offx} are chosen heuristically. For $q_1 = 1$ probability $p_{up1}(t)$ does not depend on the load demand and is equal to 1 for each unit i and each hour t. While $q_1 = 0$ the probability $p_{up1}(t)$ is the most diversified, dependent on the load demand within time period T. E.g. for $q_1 = 0$ and load data considered in the application example (Fig. 4) the p_{up1} changes in the range from 0,6667 (for the minimum load demand at time $t = 4$) to 0,9583 (for the peak load demand at time $t = 18$). The larger the value of q_1 ($0 \le q_1 \le 1$) the more the range of the $p_{up1}(t)$ narrows and nears 1, which means a reduction in the influence of the load demand value on the probability of unit start-up. The greatest "selective pressure" is acquired for $q_1 = 0$ and such a value is recommended.

The component $p_{up2}(i) - (1+ q_2)/2$ in formula (36) signifies the correction added to probability $p_{up1}(t)$ which allows the differentiation of the probability of unit start-up from unit production costs. This correction changes in the range from $-(1+ q_2)/2$ (for the unit of the highest production cost) to $(1+ q_2)/2$ (for the unit of the lowest production cost). If $q_2 = 1$ the probability of unit start-up $p_{up}(i, t)$ is not dependent on the production costs, whereas if $q_2 = 0$ this dependence is the greatest – the correction assumes values from the range (–0.5, 0.5). The value of $q_2 = 0.8$ gives the range of correction (–0.1, 0.1) and seems to be a reasonable compromise.

Formula (43) includes corrections differentiating the probability of the unit shut-down $p_{down}(i, t)$ from unit production costs: $p_{down2}(i) - (1+ r_2)/2$ and unit start-up costs: $p_{down3}(i) - (1+ r_3)/2$. The influence of the production costs on $p_{down}(i, t)$ is the greatest if $r_2 = 0$ (then the correction range is from –0.5 for the unit of the lowest production cost to 0.5 for the unit of the highest production cost). The influence of the start-up costs on $p_{down}(i, t)$ is the greatest if $r_3 = 0$ (then the correction range is from –0.5 for the unit of the

highest start-up cost after down time t_{offx} to 0.5 for the unit of the lowest start-up cost after down time t_{offx}). If $r_2 = r_3 = 0.9$ the range of each correction is (–0.05, 0.05) (jointly for both corrections (–0.1, 0.1)), which means an equal influence of production and start-up costs on the probability of unit shut-down.

The parameter t_{offx} means expected unit down time. If start-up costs change for each unit uniformly, i.e. v_i is constant for each unit, apart from down time t_{offx} (just like in the application) this parameter is not important. If the start-up curves (9) cross each other, it is safer not to take into account the start-up cost assuming $r_3 = 1$. In other cases, for different t_{offx} different probabilities $p_{down3}(i)$ are obtained but if the unit order in respect of start-up cost is constant for different t_{offx}, the unit order in respect of values of $p_{down3}(i)$ is constant as well.

There are no hard rules for setting up the above parameters of mutation method. In accordance with what was written above the advisable values of these parameters are $q_1 = r_1 = 0$, $q_2 = 0.8$, $r_2 = r_3 = 0.9$, $t_{offx} = 8$ if v_i is constant for each unit, apart from down time t_{offx} or $q_1 = r_1 = 0$, $q_2 = r_2 = 0.8$, $r_3 = 1$ in other cases. The probabilities p_{up} and p_{down} for the application example defined in Section 4 are shown in Fig. 2.

If the bit $b(i, t)$ chosen for mutation, which represents the state of unit i at moment t, changes its value from 0 to 1 (from 1 to 0) and the bits representing the state of unit i at neighbouring moments $(t–1)$ and $(t+1)$ have the same value as bit $b(i, t)$ before mutation, then the probability of a change in the state of unit i for these moments is analysed. If $p_{up}(i, t–1) > p_{up}(i, t+1)$ (or $p_{down}(i, t–1) > p_{down}(i, t+1)$ in the case of outage) then the value of bit $b(i, t–1)$ and succeeding bits $b(i, t–2)$, $b(i, t–3)$, ..., is changed, on condition that they are of the same value as bit $b(i, t)$ before mutation. A bit with the opposite value finishes this process. If $p_{up}(i, t–1) < p_{up}(i, t+1)$ (or $p_{down}(i, t–1) < p_{down}(i, t+1)$) then bits $b(i, t+1)$, $b(i, t+2)$ and so on, are changed analogously. A change in succeeding bits of the same value means a change in the off state or on state of units. This mechanism, suggested in (Mantawy *et al.*, 1998) as a solution to the problem of UC using simulated annealing, allows for the avoidance of cases of multiple changes in the on state or off state of units in period T and quickens the convergence of the algorithm.

3.4 Transposition

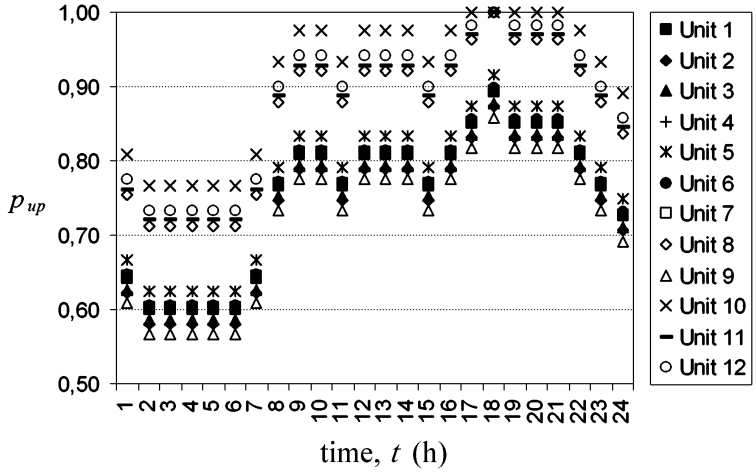
A transposition operation (introduced in (Dudek, 2003)) exchanges fragments of the chromosomes that encode the states (during period T) of two randomly chosen units. The expected number of the chromosome transpositions is n_t. This operation is shown in Fig. 3.

This transposition can considerably help the evolution process, particularly in the last phase, penetrating the local minimums by changing the work states of pairs of units.

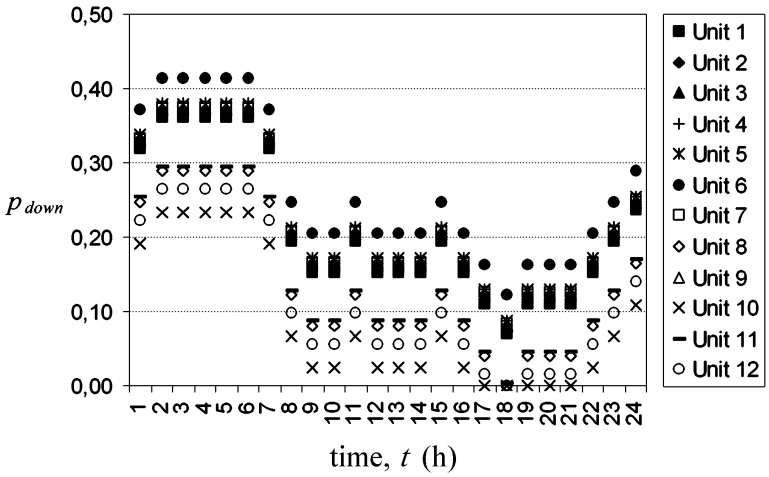
3.5 Recombinations and Micro-Simulated Annealing

3.5.1 Random Recombination

After the new temperature is determined, the chromosomes returned by the inner-loop undergo selection and recombination. A selection method is the binary tournament (Michalewicz, 1994). The recombination is analogous to the one-point crossover in EA. The chromosomes to crossover are chosen randomly. Crossover occurs with probability p_c. As a result of crossover new population of chromosomes is generated. This population is then processed in the inner-loop at a lower temperature value.



(a)



(b)

Figure 2: The probability of a bit change from 0 to 1 – p_{up}. **(a)** and from 1 to 0 – p_{down}. **(b)** for the problem defined in Section 4. Assumed: $q_1 = r_1 = 0$, $q_2 = 0.8$, $r_2 = r_3 = 0.9$, $t_{offx} = 8$.

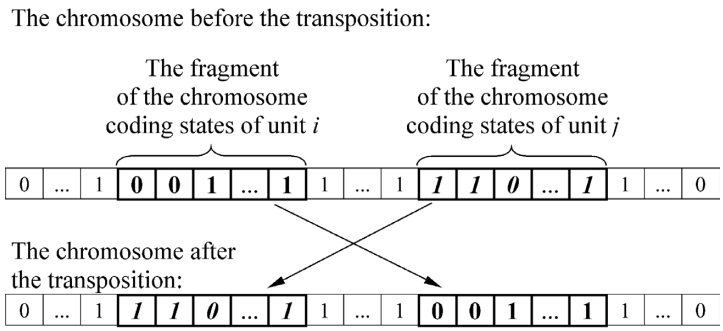


Figure 3: An illustration of the transposition.

3.5.2 Recombination with the Best Chromosome

In this approach, each chromosome of the population returned by the inner-loop recombine (one-point crossover) with the best chromosome found by the algorithm so far. Because the number of chromosomes in the population is constant, only one recombinant offspring is included to the population, replacing its parent. This strategy preserves genetic material of the best chromosome and searches the space around it.

3.5.3 Micro-Simulated Annealing

This approach is analogous to micro-genetic algorithm which refers to a small population GA with re-initialization. The idea was suggested by some theoretical results obtained by Goldberg (Goldberg, 1989), according to which a population size of 3 was sufficient to converge, regardless of the chromosomic length. The process suggested by Goldberg was to start with a small randomly generated population, then apply to it the genetic operators until reaching nominal convergence (e.g. when all the individuals have their genotypes either identical), and then to generate a new population by transferring the best individuals of the converged population to the new one. The remaining individuals would be randomly generated. This ensures delivery of new genetic material.

In our algorithm chromosomes returned by the inner-loop are replaced with the best chromosome found so far. This means that all processes at the next temperature value starts from the same chromosome and searches its neighborhood. This strategy does not apply recombination. New genetic material is introduced in the inner-loop through mutation and transposition operations. Subsequent restarts of the searching process occur at a lower temperature, which means the increasing the selective pressure.

3.6 Elitism

To protect the best solutions in SA+EA using recombination elitism is used: one of the processes in the inner-loop starts with the best chromosome found by the algorithm so far.

4 Application Example

The proposed hybrid algorithm for the UC problem described above was implemented in Matlab and has been applied to a practical power system with 12 units. The scheduling time horizon is 24 hours. Table 1

presents the parameters of unit cost characteristics and Fig. 4 presents the load data. The spinning reserve $R(t)$ for all t is equal to 5% of the maximum daily load demand, i.e. 175 MW. It is assumed: $\forall i$: $P_{\min i}$ = 180 MW, $P_{\max i}$ = 350 MW, $t_{upi} = t_{downi}$ = 5 h and τ = 7 in (10).

Unit	Initial status[a] h	*a* $/(MW²·h)	*b* $/(MW·h)	*c* $/h	*e* $	*f* $	*g* h^{-1}	*h* h^{-1}
1	–24	0.004531	7.3968	643.24	-2889.45	5466.28	0.3680	-0.0112
2	–4	0.004683	7.5629	666.27	-2893.81	5474.51	0.3680	-0.0112
3	–4	0.004708	7.4767	672.77	-2888.84	5465.13	0.3680	-0.0112
4	on	0.004880	7.4742	686.58	-2882.77	5453.66	0.3680	-0.0112
5	on	0.004214	7.2995	601.53	-2863.94	5418.07	0.3680	-0.0112
6	on	0.004582	7.3102	641.99	-2843.13	5378.74	0.3680	-0.0112
7	on	0.004267	7.5494	609.07	-2876.16	5441.15	0.3680	-0.0112
8	on	0.003572	6.6577	531.63	-2903.29	5492.22	0.3680	-0.0112
9	on	0.004788	7.7184	678.40	-2892.73	5472.47	0.3680	-0.0112
10	on	0.003485	6.2115	503.60	-2928.65	5540.14	0.3680	-0.0112
11	on	0.003658	6.5492	528.19	-2894.88	5476.32	0.3680	-0.0112
12	on	0.003671	6.4137	527.81	-2915.53	5515.34	0.3680	-0.0112

[a] "on" indicates unit is in the on-state, "–*x*" indicates unit is in the off-state for *x* hours.

Table 1: Parameters of the cost characteristics and initial state of units.

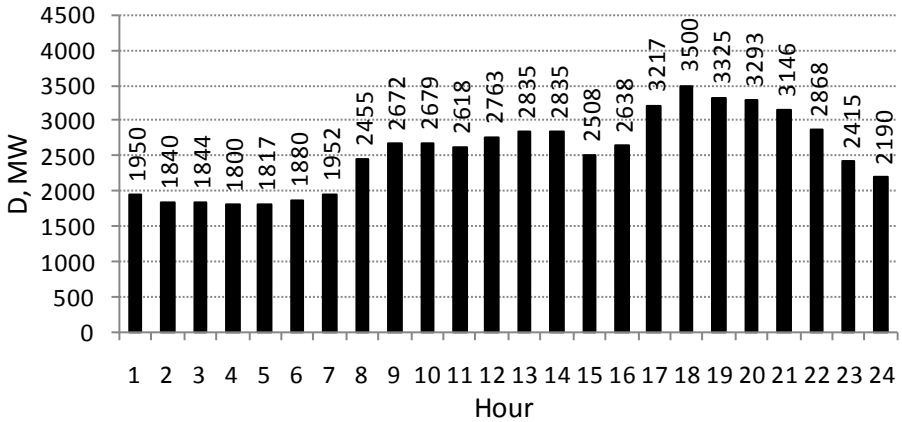


Figure 4: Load demand.

On the basis of preliminary experiments the following parameters of the algorithm were assumed:

- number of outer-loop iterations: K = 100,
- number of inner-loop iterations: L = 20,
- population size: 50,
- expected number of chromosome mutations: n_m = 1,

- expected number of chromosome transpositions: $n_t = 0.25$,
- probability of crossover in the random recombination: $p_c = 0.9$,
- parameters of the proposed mutation: $q_1 = r_1 = 0$, $q_2 = 0.8$, $r_2 = r_3 = 0.9$, $t_{offx} = 8$,
- estimated value of the minimal cost: $\widehat{F}_{\min}$ = \$645000,
- $\Delta_{\min}$ = \$2.5 (constant in (24)).

Experiments were carried out for three variants of the algorithm:

- using random recombination: SA+EA(RR)
- using recombination with the best chromosome: SA+EA(RB)
- using micro-simulated annealing: micro-SA+AE.

Each variant of the algorithm ran 10 times. All variants used the same initial populations (random). The results are shown in Table 2 where: $F_{\min}$, $F_{\max}$, F_{mean} are the minimum, maximum and average costs of the best solutions found by the algorithm in 10 runs, respectively, σ_F is the standard deviation of the costs of the best solutions P_{best}, f_{best} is the frequency of the best solution, n_{best} is the average number of evaluations necessary to find the best solution and t_{best} is the average computational time necessary to find the best solution (Pentium III 800 MHz). The best solution found by the algorithm is shown in Table 3 and Fig. 5.

Hour	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}
1	0.00	0.00	0.00	180.00	180.00	180.00	180.00	282.01	0.00	350.00	290.27	307.72
2	0.00	0.00	0.00	180.00	180.00	180.00	180.00	253.57	0.00	323.91	262.49	280.03
3	0.00	0.00	0.00	180.00	180.00	180.00	180.00	254.57	0.00	324.95	263.47	281.01
4	0.00	0.00	0.00	180.00	180.00	180.00	180.00	243.50	0.00	313.60	252.66	270.24
5	0.00	0.00	0.00	180.00	180.00	180.00	180.00	247.78	0.00	317.98	256.84	274.40
6	0.00	0.00	0.00	180.00	180.00	180.00	180.00	263.63	0.00	334.22	272.32	289.83
7	0.00	0.00	0.00	180.00	180.00	180.00	180.00	282.69	0.00	350.00	290.93	308.38
8	0.00	0.00	0.00	234.78	292.61	267.92	259.69	350.00	0.00	350.00	350.00	350.00
9	0.00	0.00	238.06	229.95	287.03	262.78	254.18	350.00	0.00	350.00	350.00	350.00
10	0.00	0.00	239.40	231.25	288.53	264.16	255.66	350.00	0.00	350.00	350.00	350.00
11	0.00	0.00	227.70	219.96	275.45	252.14	242.75	350.00	0.00	350.00	350.00	350.00
12	0.00	0.00	255.51	246.80	306.53	280.72	273.44	350.00	0.00	350.00	350.00	350.00
13	0.00	0.00	269.32	260.12	321.97	294.91	288.68	350.00	0.00	350.00	350.00	350.00
14	0.00	0.00	269.32	260.12	321.97	294.91	288.68	350.00	0.00	350.00	350.00	350.00
15	0.00	0.00	206.60	199.59	251.88	230.46	219.47	350.00	0.00	350.00	350.00	350.00
16	0.00	0.00	231.54	223.66	279.74	256.08	246.98	350.00	0.00	350.00	350.00	350.00
17	0.00	281.05	288.69	278.80	343.60	314.81	310.05	350.00	0.00	350.00	350.00	350.00
18	0.00	284.45	292.07	282.07	347.38	318.28	313.78	350.00	261.97	350.00	350.00	350.00
19	0.00	260.02	267.77	258.63	320.23	293.31	286.97	350.00	238.07	350.00	350.00	350.00
20	0.00	255.55	263.33	254.34	315.27	288.75	282.06	350.00	233.70	350.00	350.00	350.00
21	0.00	235.03	242.92	234.64	292.46	267.78	259.54	350.00	213.63	350.00	350.00	350.00
22	0.00	195.52	203.62	196.73	248.55	227.40	216.18	350.00	180.00	350.00	350.00	350.00
23	0.00	180.00	180.00	180.00	190.18	180.00	180.00	314.16	0.00	350.00	321.66	339.00
24	0.00	180.00	180.00	180.00	180.00	180.00	180.00	251.05	0.00	321.34	260.03	277.58

Table 2: Simulation results.

Algorytm	F_{min}, \$	F_{max}, \$	F_{mean}, \$	σ_F, \$	f_{best}	n_{best}	t_{best}, h
SA+EA(RR)	645065	646450	645466	488	0.0	-	-
SA+EA(RB)	644951	645081	645026	50	0.2	57925	1.58
Micro-SA+EA	644951	645440	645091	168	0.3	62411	1.40

Table 3: The best solution P_{best}.

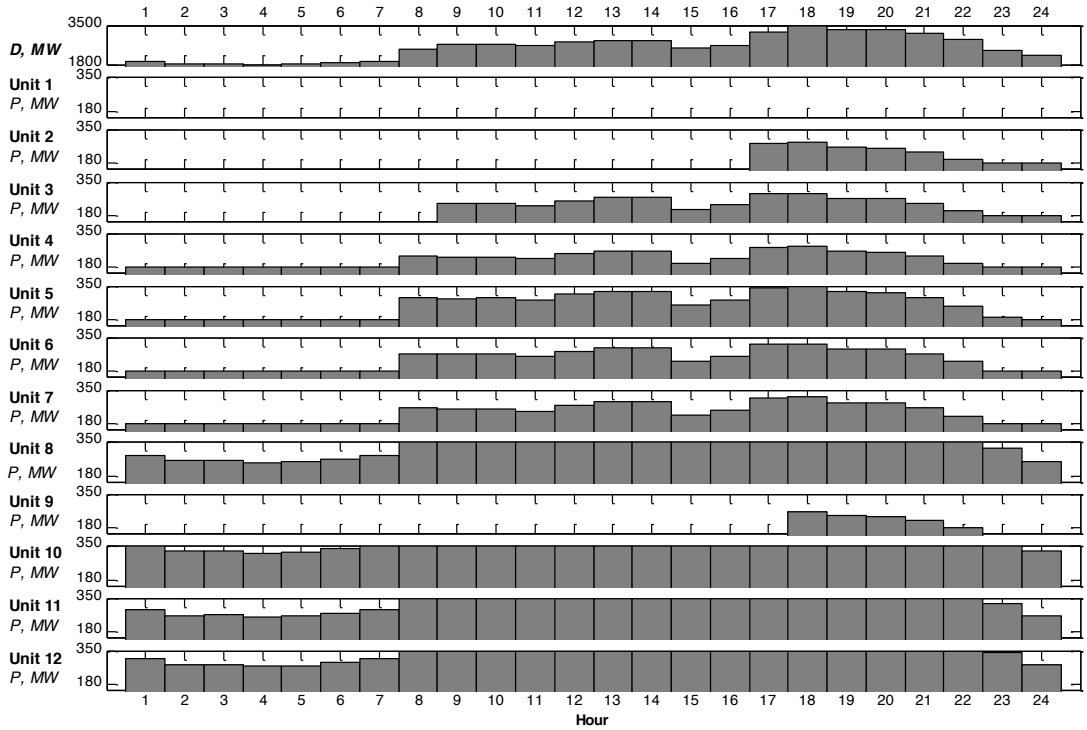


Figure 5: The best solution P_{best}.

The same example was solved by:

- genetic algorithm with the same binary representation of on/off unit status and specialized mutation and transposition operators, and one-point crossover (Dudek, 2004). For solutions violating the constraints the repair algorithm combined with the penalty algorithm was used.
- genetic algorithm with integer representation of unit start-up and shut-down times (Dudek, 2007). Penalty functions were applied to the infeasible solutions.
- adapted simulated annealing with the same representation and move operators as SA+EA (Dudek, 2010). Penalty algorithm was used for solutions violating the constraints. The temperature was adapted to the cost levels on which the algorithm operates during the annealing process.

The best solution found by these algorithms was the same as that found by SA+EA. The frequency of the best solution f_{best} was the highest for adaptive SA (0.5). The average number of evaluations necessary to find the best solution n_{best} and the average computational time t_{best} were the lowest for the GA with integer representation: n_{best} = 36550, t_{best} = 50 min. The computational time in SA+EA was lower than for adaptive SA and GA with the binary representation where t_{best} = 2 h 25 min and 2 h 15 min, respectively. The most stable solutions gave SA+AE(RB) (σ_F = \$50) and GA with integer representation (σ_F = \$48).

For comparison, calculations were done using the simple simulated annealing algorithm, Monte Carlo method and the heuristic method of limit time characteristics (Toroń, 1962), which was used for many years in the Polish Electrical Power System. In SA the variable representation is the same as for the proposed algorithm, but the candidate solutions are generated by a change of one randomly chosen bit in the current solution. In the Monte Carlo method points in the solution space are randomly chosen from the uniform distribution, remembering the best solution. The number of evaluations of the cost function in these algorithms has been set at 100000, similar to the proposed SA+EA algorithm, and the calculations for every algorithm are done ten times. The costs of the best solutions found by these algorithms are: SA: \$702379, the Monte Carlo method: no acceptable solution, the heuristic method: \$665634.

5 Conclusions

In this work we have proposed a novel method for unit commitment problem based on a hybrid model which combine simulated annealing and evolutionary algorithm. The model gives stable and acceptable solutions that are near optimal. The best result was achieved using recombination with the best solution and micro-simulated annealing. The effectiveness of the algorithm achieved by the introduction of genetic operators specific to the problem: mutation that makes the probability of bit change dependent on load demand, production costs and start-up costs of units, as well as the transposition searching through local minimums. The solutions obtained in simulation example are better by 3% from those obtained with the help of the heuristic method used in the Polish Electrical Power System.

The calculation time can be reduced by implementing the algorithm in a programming environment that is faster than Matlab, and doing the calculations in a parallel machine environment.

Acknowledgements

This work was supported by the Polish State Committee for Scientific Research under Grant 8T10B03921.

Appendix A

List of main symbols:

$\alpha_i(t)$ – on/off status of the i-th unit at the t-th hour, $\alpha_i(t) \in \{0, 1\}$

a_i, b_i, c_i – production cost function parameters of unit i

$C_i(P_i(t))$ – variable production cost of unit i at time t (\$/h)

$D(t)$ – load demand at the t-th hour (MW)

e_i, f_i, g_i, h_i – start-up cost function parameters of unit i

F, F' – cost functions for the feasible solutions and the solutions violate constraints (6) or (7), respectively

$\hat{F}_{\min}$ – estimated value of the minimal cost (\$)

i – unit number, $i = 1, 2, ..., N$

N – total number of units

n_{downi} – the number of periods in which unit i is in continuous off state during the optimization period T

n_{upi} – the number of periods in which unit i is continuously in on state in the optimization period T

$P_i(t)$ – power generation of unit i at time t (MW)

$P_{\min i}$, $P_{\max i}$ – lower/upper generation limit of unit i (MW)

$R(t)$ – spinning reserve requirement at the t-th hour (MW)

$SC_i(t_{offi})$ – start-up cost of unit i after t_{offi} hour off state (\$)

t – hour of the optimisation period, $t = 1, 2, ..., T$

T – optimisation period (24 h)

t_{offi}, t_{oni} – time period during which unit i is continuously off /on (h)

t_{upi}, t_{downi} – minimum up/down time of unit i (h)

References

Cheng, C.-P., Liu, C.-W., Liu, C.-C. (2002). Unit commitment by annealing-genetic algorithm. Electrical Power Energy Syst., 24, 149–158.

Cohen, A.I., Yoshimura, M.A. (1983). A branch-and-bound algorithm for unit commitment. IEEE Trans. Power Apparatus Syst., 102, 444–451.

Dang, C., Li, M. (2007). A floating-point genetic algorithm for solving the unit commitment problem. European Journal of Operational Research, 181, 1370–1395.

Dasgupta, D., McGregor, D.R. (1994). Thermal unit commitment using genetic algorithms. IEE Proc.-Gener. Transm. Distrib., 141 (5), 459–465.

Dudek, G. (2003). Economic dispatch and unit commitment using evolutionary algorithms. Ph.D. Dissertation (in Polish), Dept. Elect. Eng., Technical University of Czestochowa, Poland.

Dudek, G. (2004). Unit commitment by genetic algorithm with specialized search operators. Electric Power System Research, 72, 299–308.

Dudek, G. (2007). Genetic algorithm with integer representation of unit start-up and shut-down times for the unit commitment problem. European Transactions on Electrical Power, 17, 500–511.

Dudek, G. (2010).Adaptive simulated annealing schedule to the unit commitment problem. Electric Power Systems Research, 80(4), 465–472.

Garver, L.L. (1962). Power generation scheduling by integer programming – development of theory, AIEE Trans., 81, 730–735.

Goldberg, D.E. (1989). Sizing populations for serial and parallel genetic algorithms. In 3rd International Conference on Genetic Algorithms (pp.70–79).

Kazarlis, S.A., Bakirtzis, A.G., Petridis, V. (1996). A genetic algorithm solution to the unit commitment problem. IEEE Trans. Power Syst., 11 (1), 83–92.

Kirkpatrick S., Gelatt C. D., Vecchi M. P. (1983). Optimization by Simulated Annealing. Science, 220(4598), 671–680.

Kumara, S.S., Palanisamy, V. (2007). A dynamic programming based fast computation Hopfield neural network for unit commitment and economic dispatch. Electric Power Systems Research, 77, 917–925.

Lee, F.N. (1988). Short-term unit commitment – a new method. IEEE Trans. Power Syst., 3 (2), 691–698.

Mantawy, A., Abdel-Magid, Y., Selim, S. (1998). A simulated annealing algorithm for unit commitment. IEEE Trans. Power Syst., 13 (1), 197–204.

Mantawy, A., Abdel-Magid, Y., Selim, S. (1999). Integrating genetic algorithms, tabu search, and simulated annealing for the unit commitment problem. IEEE Trans. Power Syst., 14 (3), 829–836.

Merlin, A., Sandrin, P. (1983). A new method for unit commitment at Electricite de France. IEEE Trans. Power Apparatus Syst., 102(5), 1218–1225.

Michalewicz Z. (1994). Genetic Algorithms + Data Structures = Evolution Programs. Springer–Verlag, New York.

Nayak, R., Sharma, J. (2000). A hybrid neural network and simulated annealing approach to the unit commitment problem. Comput. Ind. Eng., 30 (4), 851–870.

Ouyang, Z., Shahidehpour, S.M. (2004). Short term unit commitment expert system. Electric Power System Research, 18 (1), 1–13.

Padhy, N. (2000). Unit commitment using hybrid models: a comparative study for dynamic programming, expert system, fuzzy system and genetic algorithms. Electrical Power Energy Syst., 23, 827–836.

Padhy, N., Paranjothi, S., Ramachandran, V. (1997). A hybrid fuzzy neural network – expert system for a short term unit commitment problem. Microelectron. Reliab., 37 (5), 733–737.

Pang, C.K., Sheblé, G.B., Albuyeh, F. (1981). Evaluation of dynamic programming based methods and multiple area representation for thermal unit commitments. IEEE Trans. Power Apparatus Syst., 100 (3), 1212–1218.

Rajan, C.C.A., Mohan, M.R. (2007). An evolutionary programming based simulated annealing method for solving the unit commitment problem. Electrical Power Energy Syst, 29, 540–550.

Saber, A.Y., Senjyu, T., Yona, A., Urasaki, N., Funabashi, T. (2007). Fuzzy unit commitment solution – A novel twofold simulated annealing approach. Electric Power Systems Research, 77, 1699–1712.

Sen, S., Kothari, D. (1998). Optimal thermal generating unit commitment: a review. Electrical Power Energy Syst., 20 (7), 443–451.

Senjyu, T., Miyagi, T., Yousuf, S.A., Urasaki, N. Funabashi, T. (2007). A technique for unit commitment with energy storage system. Electrical Power and Energy Systems, 29, 91–98.

Sheblé, G. B., Fahd, G. N. (1994). Unit commitment literature synopsis. IEEE Trans. Power Syst., 9 (1), 128–136.

Toroń, M. (1962). A method of selecting the optimal set of generating units in the power systems. Ph.D. Dissertation (in Polish), Dept. Elect. Eng., Technical University of Wroclaw, Poland.

Wong, M., Chung, T., Wong, Y. (2000). An evolving neural network approach in unit commitment solution. Microprocess. Microsyst., 24, 251–262.

Wong, S. (1998). An enhanced simulated annealing approach to unit commitment. Electrical Power Energy Syst., 20 (5), 359–368.

Wood, A., Wollenberg, B. (1996). Power Generation, Operation, and Control. John Wiley & Sons, New York.

Yamin, H.Y. (2004). Review on methods of generation scheduling in electric power systems. Electric Power System Research, 69, 227–248.

Yin, S., Wong, W. (2001). Hybrid simulated annealing/genetic algorithm approach to short-term hydro-thermal scheduling with multiple thermal plants. Electrical Power Energy Syst., 23, 565–575.

Zhuang, F., Galiana, F.D. (1988). Towards a more rigorous and practical unit commitment by Lagrangian relaxation. IEEE Trans. Power Syst., 3 (2), 763–773.