Electric Power Systems Research 80 (2010) 465–472

Contents lists available at ScienceDirect

Electric Power Systems Research

journal homepage: www.elsevier.com/locate/epsr

Adaptive simulated annealing schedule to the unit commitment problem

Grzegorz Dudek*

Institute of Electrical Power Engineering, Czestochowa University of Technology, 42-200 Czestochowa, Al. Armii Krajowej 17, Poland

ARTICLE INFO

Article history:
Received 13 July 2007
Received in revised form
23 December 2008
Accepted 18 October 2009
Available online 24 November 2009

Keywords:
Unit commitment
Simulated annealing
Constrained combinatorial optimization

ABSTRACT

This paper presents an approach for solving the unit commitment problem based on a simulated annealing algorithm with an adaptive schedule. The control parameter, temperature, is adapted to the cost levels on which the algorithm operates during the annealing process. This shortens the time taken to find a good solution meeting all constraints and improves the convergence of the algorithm. The operators specific to this problem, mutation and transposition, are used as the transition operators. The method incorporates time-dependent start-up costs, demand and reserve constraints, minimum up and down time constraints and unit power generation limits. There are different definitions of the objective function for the feasible and infeasible solutions. Test results showed an improvement in effectiveness compared to results obtained from simulated annealing with a static schedule, genetic algorithm and other techniques.

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. The optimization of this important problem 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 [1,2], dynamic programming [3–6], Lagrangian relaxation [7,8], the branch-and-bound method [9], and the integer and mixed-integer programming [10] (a detailed literature synopsis is summarized in [11–13]). 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 [3,4], 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 [14–16], neural networks [6,15,17], fuzzy logic [15,16,18], genetic algorithms [19–22] and simulated annealing [18,23,24]. 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 recognise 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 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

* Tel.: +48 34 3250 896; fax: +48 34 3250 803.
E-mail address: dudek@el.pcz.czest.pl.

0378-7796/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.epsr.2009.10.019

optimization method, principally for combinatorial optimization 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 [25–27], the neural network [28], the tabu search [25], and the evolutionary programming [29].

This paper presents an adaptive SA algorithm to solve the UC problem. In the proposed approach the solution space is not homogeneous because there are three different definitions of the cost functions (for feasible and infeasible solutions) and thus the algorithm operates on the three cost levels. Adaptive SA schedules have parameters that are tuned during the annealing process to the specific properties of the solution space. The adaptive annealing and the specialized search operators accelerate the searching process and provide better results.

2. The mathematical model of unit commitment

The UC problem can be mathematically formulated as follows:
Objective function:

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

Constraints:

(a) Load balance

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

(b) Unit power generation limits

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

(c) Set of unit power generation limits

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

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

(d) Minimum up/down time

$$\forall i : t_{offi} \ge t_{downi} \quad (6)$$

$$\forall i : t_{oni} \ge t_{upi} \quad (7)$$

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

$$C_i(P_i) = a_iP_i^2 + b_iP_i + c_i \quad (8)$$

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:

$$SC_i(t_{offi}) = e_i\exp(-g_it_{offi}) + f_i\exp(-h_it_{offi}) \quad (9)$$

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 ($\tau \in \{1, 2, 3, \ldots\}$).

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

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

3. The proposed adaptive simulated annealing approach

The SA algorithm consists of exploring the solution space starting from a randomly selected solution and generating a new one by perturbing it. When a new candidate solution x' is generated, its cost is calculated and the new solution is either accepted or rejected according to an acceptance probability [30]:

$$p(x') = \min\left[1, \exp\left(\frac{-\Delta\Phi}{\psi}\right)\right] \quad (11)$$

where

$$\Delta\Phi = \Phi(x') - \Phi(x) \quad (12)$$

The effectiveness of the SA depends on:

- the definition of decision variables and their representation,
- the means by which the current solution x can be perturbed to generate the next one x', i.e. the transition operators,
- the procedure for infeasible solutions,
- the cooling schedule which specifies: an initial value of the control parameter called temperature ψ_0, a temperature update function $\psi_{k+1} = f(\psi_k)$ and the inner-loop and outer-loop criterions.

The inner-loop criterion is satisfied when a number of transitions proportional to the size of the problem have been attempted. The outer-loop criterion is satisfied when there is no significant improvement in the solution after a pre-specified number of iterations or when the maximum number of iterations is reached.

3.1. Decision variables and their representation

The decision variables are on/off unit states $\alpha_i(t)$. The solution string 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 $2^{N\cdot T}$.

3.2. Transition operators

SA searches the solution space by exploring neighbourhoods of the current solution by means of the transition operators. Two types of transition operators are proposed. The first one is analogous to the mutation which was used in the genetic algorithm to the UC problem defined in [21]. In the case of this specialized search operator the probability of bit mutation is not the same for all bits but depends on the unit production cost, its start-up cost, and load demand. The probability of unit shut-down (probability of bit change from 1 to 0) is higher for units which have higher production costs and lower start-up cost. Moreover, the higher the load demands the higher the probability of unit start-up. The mutation procedure proceeds as follows. First, a bit from the solution x is randomly chosen. Then its value is changed with the probability calculated according to Eq. (23) in [21] when the bit value is equal

to 0 or with the probability calculated according to Eq. (30) in [21] when the bit value is equal to 1. When this bit $b(i, t)$, 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 probabilities of start-up (shut-down) for these moments are analysed. If the probability for $(t-1)$ is higher than for $(t+1)$, the value of bit $b(i, t-1)$ and succeeding bits $b(i, t-2)$, $b(i, t-3)$, ..., is changed, on the condition that they are of the same value as bit $b(i, t)$ before mutation. A bit with the opposite value finishes this process. Otherwise, if the probability for $(t+1)$ is higher than for $(t-1)$, the value of bit $b(i, t+1)$ and succeeding bits is changed. A change in succeeding bits of the same value means a change in the off or on state of units. This mechanism, suggested in [23], allows the avoidance of cases of multiple changes in the on or off state of units in period T and quickens the convergence of the algorithm.

The second transition operator is a transposition introduced in [31,21]. This operator exchanges fragments of the solution string x that encode all decision variables of two randomly chosen units. This transposition can considerably help the annealing process, particularly in the last phase, escaping the local minimums by changing the work states of pairs of units.

3.3. Economic dispatch and cost calculations. The procedure with infeasible solutions

Since the production cost is a quadratic function (convex and continuous), the economic dispatch problem is solved using a lambda-iteration method [32], based on the principle of equal incremental cost. For the following feasible solution strings created in the annealing process the lambda-iteration method is used to determine the generation levels $P_i(t)$. Then the unit production costs (8) and the value of the objective function (1) are calculated. This method guarantees that unit power generation limit constraints (3) are met.

For solutions which violate the minimum up/down time constraints (6) or (7), but do not violate a set of unit power generation limit constraints (4) and (5), a penalty function is created [31,21]:

$$F' = M\left\{1 + m\sum_{i=1}^{N}[g(i) + h(i)]\right\} \tag{13}$$

where M is a constant which is higher than the estimated maximal value if the objective function (1). M is expressed as below:

$$M = T\sum_{i=1}^{N} C_i(P_{\max i}) \tag{14}$$

The discreet functions defining the level of constraints (6) and (7) violation $g(i)$ and $h(i)$ are given by

$$g(i) = \sum_{j=1}^{n_{downi}} \{\beta_i(j)[t_{downi} - t_{offi}(j)]\} \tag{15}$$

$$h(i) = \sum_{j=1}^{n_{upi}} \{\gamma_i(j)[t_{upi} - t_{oni}(j)]\} \tag{16}$$

where

$$\beta_i(j) = \begin{cases} 1 & \text{if } t_{offi}(j) < t_{downi} \\ 0 & \text{if } t_{offi}(j) \geq t_{downi} \vee \tau_{oni}(j) > T \end{cases} \tag{17}$$

$$\gamma_i(j) = \begin{cases} 1 & \text{if } t_{oni}(j) < t_{upi} \\ 0 & \text{if } t_{oni}(j) \geq t_{upi} \vee \tau_{offi}(j) > T \end{cases} \tag{18}$$

If solutions violate the set of unit power generation limit constraints (4) or (5), a penalty function is formulated as follows [31]:

$$F'' = W\left[1 + w\sum_{t=1}^{T} f(t)\right] \tag{19}$$

where W is a constant bigger than the maximal value of the function (13), which takes place when the units start-up and then shut-down every hour. W is expressed as follows:

$$W = M\left\{1 + m\frac{T}{2}\sum_{i=1}^{N}[(t_{downi} - 1) + (t_{upi} - 1)]\right\} \tag{20}$$

$f(t)$ is given by

$$f(t) = \begin{cases} \sum\limits_{i \in \Omega(t)} P_{\min i} - D(t) & \text{if } \sum\limits_{i \in \Omega(t)} P_{\min i} > D(t) \\ D(t) + R(t) - \sum\limits_{i \in \Omega(t)} P_{\max i} & \text{if } D(t) + R(t) > \sum\limits_{i \in \Omega(t)} P_{\max i} \\ 0 \quad \text{otherwise} \end{cases} \tag{21}$$

or if all units have the same generation ranges as in the example described below:

$$f(t) = \begin{cases} n_{\min}(t) - n(t) & \text{if } n(t) < n_{\min}(t) \\ n(t) - n_{\max}(t) & \text{if } n(t) > n_{\max}(t) \\ 0 \quad \text{otherwise} \end{cases} \tag{22}$$

where

$$n_{\min}(t) = \min\left\{n : \sum_{i=1}^{n} P_{\max i} \geq D(t) + R(t)\right\} \tag{23}$$

$$n_{\max}(t) = \max\left\{n : \sum_{i=1}^{n} P_{\min i} \leq D(t)\right\} \tag{24}$$

For the sake of the proper operation of the annealing process the range widths of the functions (1), (13) and (19) should be similar: $\Delta_{\max} = \Delta F_{\max} \approx \Delta F'_{\max} \approx \Delta F''_{\max}$. The estimated maximal and minimal value of the objective function (1) is M and $F_{\min}$, respectively. The maximal and minimal value of the function (13) is W and M, respectively. The minimal value of the function (19) is W, and it takes its maximal value when all units are shut-down during the period T:

$$F''_{\max} = W\left\{1 + w\sum_{t=1}^{T}[D(t) + R(t)]\right\} \tag{25}$$

when $f(t)$ is defined by Eq. (21) or

$$F''_{\max} = W(1 + wTN) \tag{26}$$

when $f(t)$ is defined by Eq. (22).

The coefficient m in (13) can be calculated as follows:

$$\Delta F'_{\max} = W - M = Mm\frac{T}{2}\sum_{i=1}^{N}[(t_{downi} - 1) + (t_{upi} - 1)] \tag{27}$$

$$\Delta F_{\max} = \Delta F'_{\max} \rightarrow M - F_{\min} = Mm\frac{T}{2}\sum_{i=1}^{N}[(t_{downi} - 1) + (t_{upi} - 1)]$$

$$\rightarrow m = \frac{2(M - F_{\min})}{MT\sum_{i=1}^{N}[(t_{downi} - 1) + (t_{upi} - 1)]} \tag{28}$$

The coefficient w in (19), when $f(t)$ is given by (21), can be calculated as below:

$$\Delta F''_{\max} = F''_{\max} - W = Ww\sum_{t=1}^{T}[D(t)+R(t)] \tag{29}$$

$$\Delta F_{\max} = \Delta F''_{\max} \rightarrow M - F_{\min} = Ww\sum_{t=1}^{T}[D(t)+R(t)]$$

$$\rightarrow w = \frac{M - F_{\min}}{W\sum_{t=1}^{T}[D(t)+R(t)]} \tag{30}$$

or when $f(t)$ is given by (22):

$$\Delta F''_{\max} = F''_{\max} - W = WwTN \tag{31}$$

$$\Delta F_{\max} = \Delta F''_{\max} \rightarrow M - F_{\min} = WwTN \rightarrow w = \frac{M - F_{\min}}{WTN} \tag{32}$$

The penalty function definition (19) ensures that solutions violating constraints (4) or (5) are evaluated worse than solutions violating constraints (6) or (7). The penalty function (13) ensures a worse valuation of solutions violating constraints (6) or (7) from feasible solutions. Both penalty cost functions (13) and (19) are linearly dependent on the level of violation of constraints.

The annealing process runs as follows. At the starting phase there are no feasible solutions according to constraints (4)–(7) and the level of violation of these constraints is minimized. The solutions are evaluated by means of the penalty function (19)—SA operates on the first cost level. After solutions meeting constraints (4) and (5) have been found, they are evaluated by means of the function (13)—SA operates on the second cost level. At a certain point in the process solutions that are feasible according to all constraints start to appear and they are evaluated by means of the cost function (1)—SA works on the third cost level.

3.4. Adaptive annealing schedule

Heuristic annealing schedules have been successfully used in most applications. These heuristics can be classified into two broad categories: static and dynamic (adaptive) schedules [33]. In a static schedule parameters are fixed before the algorithm is started. A typical example of such a schedule is given by the geometric temperature update presented in the pioneering work [30]. Static schedules are easier to implement but have the disadvantage that their parameters have to be tuned to the particular applications. Adaptive schedules have parameters that are modified with information gathered during the operation of the algorithm. The proposed adaptive simulated annealing is based on a polynomial-time cooling schedule formulated by Aarts and Van Laarhoven [34] and analyzed in [35]. The initial value of temperature ψ_0 is obtained from the requirement that at this value of the temperature all generated transitions should be accepted. Let m_1 denote the number of proposed transitions for which $\Delta\Phi \leq 0$, and m_2 denote the number of proposed transitions for which $\Delta\Phi > 0$. Furthermore, let $\Delta\bar{\Phi}^+$ be the average difference in cost over the m_2 cost-increasing transitions. The acceptance ratio χ, defined as a ratio of a number of accepted transitions to a number of proposed transitions, can be approximated by the following expression [35]:

$$\chi \approx \frac{m_1 + m_2 \exp(-\Delta\bar{\Phi}^+/\Psi)}{m_1 + m_2} \tag{33}$$

from which we obtain:

$$\Psi = \frac{\Delta\bar{\Phi}^+}{\ln(m_2/m_2\chi - m_1(1-\chi))} \tag{34}$$

The initial value of temperature ψ_0 can be calculated from (34) assuming a high initial acceptance ratio, e.g. $\chi = 0.99$.

The temperature update rule is expressed by the following equation [35]:

$$\Psi_{k+1} = \frac{\Psi_k}{1 + \Psi_k \ln(1+\delta)/3\sigma_k} \tag{35}$$

Three cost levels on which SA operates complicate the choice of the initial temperature, its updating and the whole annealing process. There is a considerable cost difference $\Delta\Phi$ at the jump between the cost levels, i.e. when the new candidate solution is from another cost level than the current one. It results in a minor temperature reduction and the maintenance of a high probability of acceptance of solutions that are from the same cost level as the current solution. It leads to the lengthening of the annealing process and to "wandering". In order to prevent this situation the initial temperatures and the temperature update rules individually adapted to each cost level are proposed.

At each cost level the highest difference between the solutions from the same cost level is $\Delta_{\max}$. The initial temperature which ensures the acceptance of each new solution from the same level with probability of 0.99 is calculated from (11):

$$\Psi_0(l) = -\frac{\Delta_{\max}}{\ln(0.99)} \approx 100\Delta_{\max} \quad l = 1, 2, 3 \tag{36}$$

The temperature updating rule on each cost level is expressed by the equation:

$$\Psi_{k+1}(l) = \frac{\Psi_k(l)}{1 + \Psi_k(l)\ln(1+\delta)/[3\sigma_k(l)]} \quad l = 1, 2, 3 \tag{37}$$

When on the kth temperature level there are no solutions from the lth cost level (i.e. the current solution and the new solution generated from it both belonging to the lth cost level), the temperature for this cost level stays without change: $\Psi_{k+1}(l) = \Psi_k(l)$. The current temperature (i.e. the temperature needed for the acceptance probability calculation (11)) at the temperature level $(k+1)$ is the temperature of the cost level in which the kth temperature level finished. The annealing process starts on the first cost level at the initial temperature $\psi_0(1)$, which is updated on the successive temperature level according to (37). When the new solution meeting constraints (4) and (5) appear, which means the jump to the second cost level, the temperature $\psi(2)$ is also modified. If this jump is stable, i.e. the algorithm operates on the second cost level until the final iteration of the inner-loop, the temperature $\psi(2)$ becomes the current temperature used on the next temperature level for the acceptance probability calculation. Similarly, when the jump to the third cost level occurs (from the first or second level), the temperature $\psi(3)$ is updated and becomes the current temperature if the jump is stable. The proposed annealing process for the application example defined below is shown in Fig. 1. As a transition operator only standard uniform binary mutation (a bit to change in the solution string is chosen by random) is used for this figure.

A flowchart of the proposed adaptive SA algorithm is given in Fig. 2.

4. Application example

The SA algorithm described above for the UC problem was implemented in Matlab and has been applied to a practical power system with 12 units. The scheduling time horizon T is 24 h. These experiments were done on a personal computer with a Pentium III 800 MHz processor.

The unit and load data can be found in Tables 1 and 2, respectively. 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 $\tau = 7$ in (10) and $f(t)$ is calculated according to (22).

Table 1
Characteristics and initial state of units.

Unit	Initial status[a], h	a, \$/(MW^2 h)	b, \$/(MW h)	c, \$/h	e, \$	f, \$	g, h^{-1}	h, h^{-1}	P_{mini}, MW	P_{maxi}, MW	t_{downi}, h	t_{upi}, h
1	−24	0.004531	7.3968	643.24	−2889.45	5466.28	0.3680	−0.0112	180	350	5	5
2	−4	0.004683	7.5629	666.27	−2893.81	5474.51	0.3680	−0.0112	180	350	5	5
3	−4	0.004708	7.4767	672.77	−2888.84	5465.13	0.3680	−0.0112	180	350	5	5
4	On	0.004880	7.4742	686.58	−2882.77	5453.66	0.3680	−0.0112	180	350	5	5
5	On	0.004214	7.2995	601.53	−2863.94	5418.07	0.3680	−0.0112	180	350	5	5
6	On	0.004582	7.3102	641.99	−2843.13	5378.74	0.3680	−0.0112	180	350	5	5
7	On	0.004267	7.5494	609.07	−2876.16	5441.15	0.3680	−0.0112	180	350	5	5
8	On	0.003572	6.6577	531.63	−2903.29	5492.22	0.3680	−0.0112	180	350	5	5
9	On	0.004788	7.7184	678.40	−2892.73	5472.47	0.3680	−0.0112	180	350	5	5
10	On	0.003485	6.2115	503.60	−2928.65	5540.14	0.3680	−0.0112	180	350	5	5
11	On	0.003658	6.5492	528.19	−2894.88	5476.32	0.3680	−0.0112	180	350	5	5
12	On	0.003671	6.4137	527.81	−2915.53	5515.34	0.3680	−0.0112	180	350	5	5

[a] "On" indicates unit is in the on state, "−x" indicates unit is in the off state for x hours.

Table 2
Load demand D (MW).

Hour	1	2	3	4	5	6	7	8	9	10	11	12
D	1950	1840	1844	1800	1817	1880	1952	2455	2672	2679	2618	2763
Hour	13	14	15	16	17	18	19	20	21	22	23	24
D	2835	2835	2508	2638	3217	3500	3325	3293	3146	2868	2415	2190

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

- maximum no. of inner-loop iterations: 100,
- maximum no. of outer-loop iterations: 1000,
- probability of solution mutation: 1,
- probability of solution transposition: 0.25,
- estimated value of the minimal cost F_{min}: \$645,000,
- parameters of the proposed mutation method were the same as in [21].

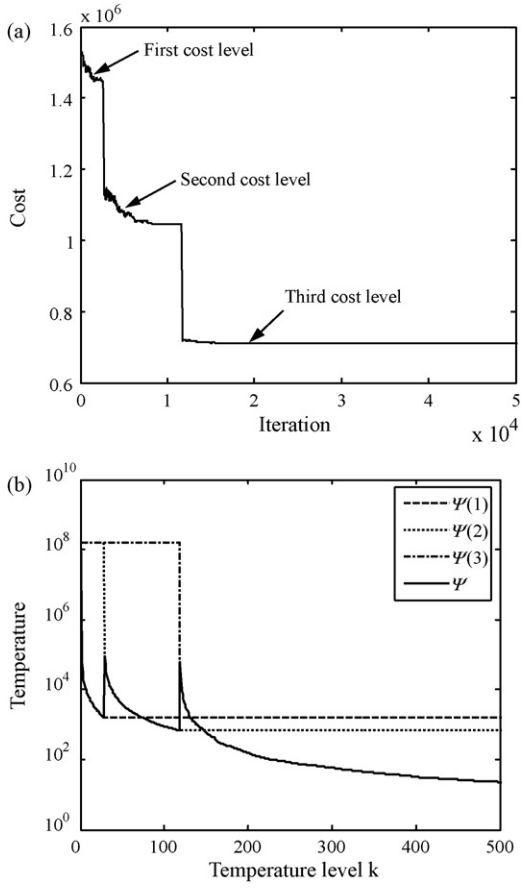

Fig. 1. The current solution costs (a) and temperatures (b) in the proposed adaptive annealing process with uniform binary mutation as a transition operator.

The following results were obtained:

- the minimum, maximum and average costs of the best solutions found by the algorithm in 10 runs were \$644,951, \$645,344, and \$645,116, respectively,
- the standard deviation of the costs of the best solutions found by the algorithm in 10 runs was \$180,
- the frequency of the best solution found by SA: 0.5,
- the average number of evaluations necessary to find the best solution: 57,637,
- the average computational time necessary to find the best solution: 2 h 25 min.[1]

In Fig. 3 the cost of current solutions in some variants of SA algorithm, implemented and tested by the author, is shown. It can be observed that SA with the adaptive annealing schedule converge much faster than with the static schedule and that the proposed operators allow the searching of the solution space more effectively because they provide the mechanisms to escape from a local minimum trap.

The same example was solved by genetic algorithm with the same binary representation of on/off unit status and specialized mutation and transposition operators [21]. Additionally the genetic algorithm has another one search operator—the one-point crossover. For solutions violating the constraints the repair algorithm combined with the penalty algorithm was used in genetic algorithm. The best solution found by genetic algorithm, shown in Table 5 in [21], was the same as that found by SA proposed in this article, but the frequency of finding the best solution was lower—0.2. But due to its parallel structure genetic algorithm was nearly twice faster then adaptive SA.

[1] Experiments were repeated on a PC with a Core 2 Duo 2.5 GHz processor in Matlab 7.7 environment and the execution time was reduced to 31 min. The professional implementation for industry needs using good compilers (e.g. C++, C#) which are much faster then Matlab.

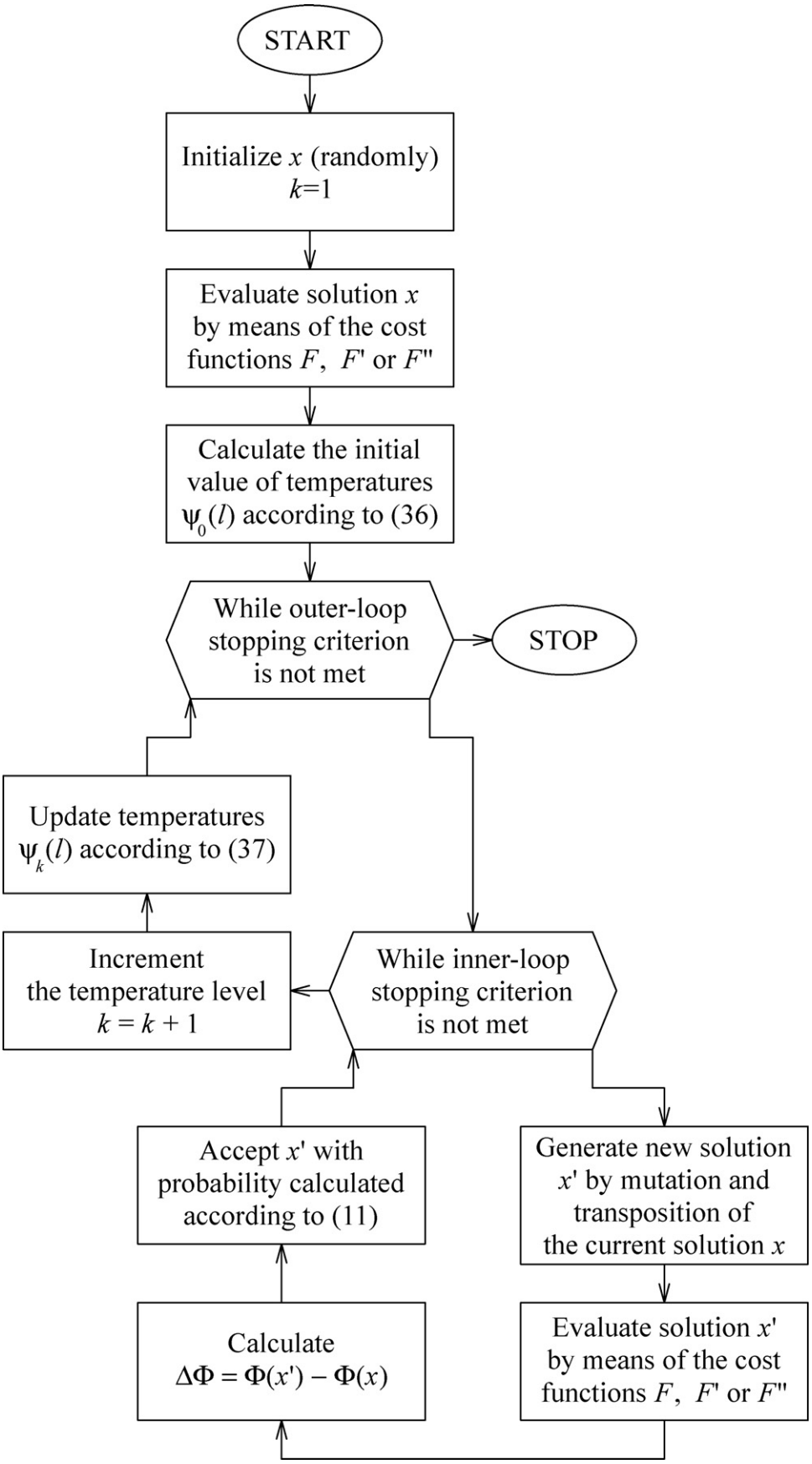

Fig. 2. The flowchart of the proposed adaptive SA algorithm.

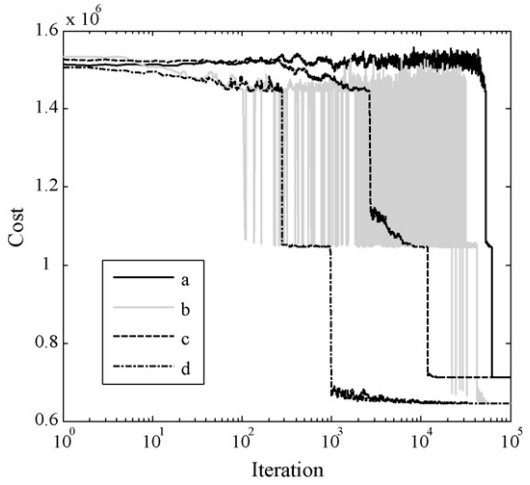

Fig. 3. The cost of current solutions in SA with static schedule (the geometric temperature update [30]) and standard binary uniform mutation (a), in SA with static schedule and proposed specialized mutation and transposition as the transition operators (b), in proposed adaptive SA with standard binary uniform mutation (c) and in proposed adaptive SA with specialized mutation and transposition (d).

The solution found by adaptive SA method is better than those reported in [21] obtained from:

- the simple SA ($702,379), in which representation of the variables is the same as for adaptive SA, but the solutions are generated by a change of one bit in the base solution (standard binary mutation);
- the Monte Carlo method (no acceptable solution), in which points in the solution space are sampled randomly, remembering the best solution;
- the heuristic method of limit time characteristics ($665,634) [36], which was used for many years in the Polish Electrical Power System. This method is based on the criterions of the profit from shut-down or start-up of units.

The number of evaluations of the cost function in these algorithms has been set at 100,000, similar to the proposed SA algorithm, and the calculations for every algorithm are done 10 times.

5. Conclusions

The proposed adaptive SA for the UC problem gives stable and acceptable solutions that are near-optimal. The difference between the cost of the best and worst solution found in 10 runs of the algorithm in the example was 0.061% ($393). The effectiveness of the algorithm was achieved by two ways. First, by adaptation of SA temperature to the specificity of the solution space which is different for feasible and infeasible solutions. Second, by using transition operators specific to the UC problem: mutation with a probability of bit change depending on load demand, production and start-up costs of the generating units and transposition searching through local minimums.

The limiting factor of SA is the calculation time longer than in genetic algorithm because of the non-parallel algorithm structure. It can be reduced by implementing the algorithm in a programming environment that is faster than Matlab or by combing SA with genetic algorithm [37].

Acknowledgement

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

Appendix A. List of symbols

$\alpha_i(t)$	on/off status of the ith unit at the tth hour, $\alpha_i(t) \in \{0, 1\}$
χ	acceptance ratio defined as a ratio of a number of accepted transitions to a number of proposed transitions
δ	distance parameter, small δ-value leads to small decrements in the temperature; assumed $\delta = 0, 3$
$\Delta_{\max}$	range widths of the cost functions F, F' and F''
$\Delta\bar{\Phi}^+$	average difference in cost over the m_2 cost-increasing transitions
Φ	cost function F, F' or F''
$\sigma_k(l)$	standard deviation of differences between costs of the current and new solutions generated on the kth temperature level both belonging to the same lth cost level
$\tau_{offi}(j)$, $\tau_{oni}(j)$	shut-down/start-up hour of unit i after the jth on/off state period
$\Omega(t)$	set of units in on state at time t
Ψ	control parameter in the simulated annealing called temperature
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 tth hour (MW)
e_i, f_i, g_i, h_i	start-up cost function parameters of unit i
F, F', F''	cost functions for the feasible solutions, the solutions violate constraints (6) or (7) but do not violate constraints (4) and (5) and the solutions violate constraints (4) or (5), respectively
$F_{\min}$	estimated minimal value of the objective function (1)
k	temperature level index
l	cost level index, $l = 1, 2, 3$
m_1, m_2	number of proposed transitions for which $\Delta\Phi \leq 0$ or $\Delta\Phi > 0$, respectively
N	total number of units
$n(t)$	number of units that are in on state at time t
n_{downi}, n_{upi}	number of periods in which unit i is in continuous off/on state during the optimization period T
$n_{\min}(t)$, $n_{\max}(t)$	minimum and maximum number of units necessary to meet load demand at moment 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 tth hour (MW)
$SC_i(t_{offi})$	start-up cost of unit i after t_{offi} hour off state ($)
T	number of hours in the study period
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
x, x'	current and new solution, respectively.

References

[1] F.N. Lee, Short-term unit commitment—a new method, IEEE Trans. Power Syst. 3 (2) (1988) 691–698.
[2] T. Senjyu, T. Miyagi, S.A. Yousuf, N. Urasaki, T. Funabashi, A technique for unit commitment with energy storage system, Electr. Power Energy Syst. 29 (2007) 91–98.
[3] C.K. Pang, H.C. Chen, Optimal short-term thermal unit commitment, IEEE Trans. Power Apparat. Syst. PAS-95 4 (1976) 1336–1342.
[4] C.K. Pang, G.B. Sheblé, F. Albuyeh, Evaluation of dynamic programming based methods and multiple area representation for thermal unit commitments, IEEE Trans. Power Apparat. Syst. PAS-100 3 (1981) 1212–1218.
[5] A. Philpott, R. Schultz, Unit commitment in electricity pool markets, Math. Program., Ser. B 108 (2006) 313–337.
[6] S.S. Kumara, V. Palanisamy, A dynamic programming based fast computation Hopfield neural network for unit commitment and economic dispatch, Electr. Power Syst. Res. 77 (2007) 917–925.
[7] A. Merlin, P. Sandrin, A new method for unit commitment at Electricite de France, IEEE Trans. Power Apparat. Syst. 102 (5) (1983) 1218–1225.
[8] F. Zhuang, F.D. Galiana, Towards a more rigorous and practical unit commitment by Lagrangian relaxation, IEEE Trans. Power Syst. 3 (2) (1988) 763–773.
[9] A.I. Cohen, M.A. Yoshimura, A branch-and-bound algorithm for unit commitment, IEEE Trans. Power Apparat. Syst. 102 (1983) 444–451.

[10] L.L. Garver, Power generation scheduling by integer programming—development of theory, AIEE Trans. 81 (1962) 730–735.
[11] G.B. Sheblé, G.N. Fahd, Unit commitment literature synopsis, IEEE Trans. Power Syst. 9 (1) (1994) 128–136.
[12] S. Sen, D. Kothari, Optimal thermal generating unit commitment: a review, Electr. Power Energy Syst. 20 (7) (1998) 443–451.
[13] H.Y. Yamin, Review on methods of generation scheduling in electric power systems, Electr. Power Syst. Res. 69 (2004) 227–248.
[14] Z. Ouyang, S.M. Shahidehpour, Short-term unit commitment expert system, Electr. Power Syst. Res. 18 (1) (2004) 1–13.
[15] N. Padhy, S. Paranjothi, V. Ramachandran, A hybrid fuzzy neural network—expert system for a short-term unit commitment problem, Microelectron. Reliab. 37 (5) (1997) 733–737.
[16] N. Padhy, Unit commitment using hybrid models: a comparative study for dynamic programming, expert system, fuzzy system and genetic algorithms, Electr. Power Energy Syst. 23 (2000) 827–836.
[17] M Wong, T. Chung, Y. Wong, An evolving neural network approach in unit commitment solution, Microprocess. Microsyst. 24 (2000) 251–262.
[18] A.Y. Saber, T. Senjyu, A. Yona, N. Urasaki, T. Funabashi, Fuzzy unit commitment solution—a novel twofold simulated annealing approach, Electr. Power Syst. Res. 77 (2007) 1699–1712.
[19] D. Dasgupta, D.R. McGregor, Thermal unit commitment using genetic algorithms, IEE Proc.-Gen. Transm. Distrib. 141 (5) (1994) 459–465.
[20] S.A. Kazarlis, A.G. Bakirtzis, V. Petridis, A genetic algorithm solution to the unit commitment problem, IEEE Trans. Power Syst. 11 (1) (1996) 83–92.
[21] G. Dudek, Unit commitment by genetic algorithm with specialized search operators, Electr. Power Syst. Res. 72 (2004) 299–308.
[22] C. Dang, M. Li, A floating-point genetic algorithm for solving the unit commitment problem, Eur. J. Oper. Res. 181 (2007) 1370–1395.
[23] A. Mantawy, Y. Abdel-Magid, S. Selim, A simulated annealing algorithm for unit commitment, IEEE Trans. Power Syst. 13 (1) (1998) 197–204.
[24] S. Wong, An enhanced simulated annealing approach to unit commitment, Electr. Power Energy Syst. 20 (5) (1998) 359–368.
[25] A. Mantawy, Y. Abdel-Magid, S. Selim, Integrating genetic algorithms, tabu search, and simulated annealing for the unit commitment problem, IEEE Trans. Power Syst. 14 (3) (1999) 829–836.
[26] S. Yin, W. Wong, Hybrid simulated annealing/genetic algorithm approach to short-term hydro-thermal scheduling with multiple thermal plants, Electr. Power Energy Syst. 23 (2001) 565–575.
[27] C.-P. Cheng, C.-W. Liu, C.-C. Liu, Unit commitment by annealing-genetic algorithm, Electr. Power Energy Syst. 24 (2002) 149–158.
[28] R. Nayak, J. Sharma, A hybrid neural network and simulated annealing approach to the unit commitment problem, Comput. Ind. Eng. 30 (4) (2000) 851–870.
[29] C.C.A. Rajan, M.R. Mohan, An evolutionary programming based simulated annealing method for solving the unit commitment problem, Electr. Power Energy Syst. 29 (2007) 540–550.
[30] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.
[31] G. Dudek, Economic dispatch and unit commitment using evolutionary algorithms, Ph.D. Dissertation (in Polish), Dept. Elect. Eng., Czestochowa University of Technology, Poland (2003).
[32] A. Wood, B. Wollenberg, Power Generation, Operation and Control, John Wiley & Sons, New York, 1996.
[33] F. Romeo, A. Sangiovanni-Vincentelli, A theoretical framework for simulated annealing, Algorithmica 6 (3) (1991) 302–345.
[34] E. Aarts and P. Van Laarhoven, A new polynomial time cooling schedule. Proc. IEEE Conf. Computer-Aided Design (1985) 206-208.
[35] E. Aarts, J. Korst, Simulated Annealing and Boltzmann Machines, John Wiley & Sons, Chichester, 1989.
[36] M. Toroń, 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 (1962).
[37] G. Dudek, Unit commitment with a hybrid algorithm using simulated annealing and genetic algorithm, Arch. Energ. XXXV (1) (2006) 3–28 (in Polish).

Grzegorz Dudek (born in 1969) received his M.Sc. degree in 1994 and Ph.D. degree in 2003 in electrical engineering from the Czestochowa University of Technology, Poland. He is currently a Lecturer at the Department of Electrical Engineering, Czestochowa University of Technology. His research interests are in using artificial intelligence and pattern recognition in the areas of generation planning and scheduling, unit commitment and load forecasting.