A UNIT DECOMMITMENT METHOD IN POWER SYSTEM SCHEDULING

by

Chung-Li Tseng, Shmuel S. Oren, Alva J. Svoboda,
and Raymond B. Johnson

Memorandum No. UCB/ERL/IGCT M96/31

23 May 1996

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A Unit Decommitment Method in Power System Scheduling

Chung-Li Tseng, Shmuel S. Oren
Department of Industrial Engineering
and Operations Research
University of California at Berkeley
Berkeley, CA 94720 USA

Alva J. Svoboda, Raymond B. Johnson
Pacific Gas & Electric Company
San Francisco, CA 94177
USA

Abstract–**This paper presents a unit decommitment method for power system scheduling. Given a feasible unit commitment, our algorithm determines an optimal strategy for decommitting overcommitted units based on dynamic programming. This method is being developed as a possible post-processing tool to improve the solution quality of the existing unit commitment algorithm used at PG&E. It can also be integrated into any other unit commitment method or used as a complete unit commitment algorithm in itself. The decommitment method can also be used as a tool to measure the solution quality of unit commitment algorithms. The proposed method maintains solution feasibility at all iterations. In this paper we prove that the number of iterations required by the method to terminate is bounded by the number of units. Numerical tests indicate that this decommitment method is computationally efficient and can improve scheduling significantly.**

Keywords: Power system scheduling, Unit commitment, Unit decommitment.

I Introduction

The unit commitment problem at a power utility like PG&E requires economically scheduling generating units over a planning horizon so as to meet forecast demand and system operating constraints. It has been an active research subject due to potential cost saving and the difficulty of the problem. The unit commitment problem is a mixed integer programming problem, and is considered to be among the class of the most intractable problems (NP-hard.) Many optimization methods have been proposed to solve the unit commitment problem (e.g. [4]). These methods include priority list methods [3], dynamic programming methods [10, 11, 14] and Lagrangian relaxation methods [2, 4, 5, 6], etc. Among them, Lagrangian relaxation methods are now among the most widely used approaches to solve unit commitment. At PG&E, the Hydro-Thermal Optimization (HTO) program was developed almost a decade ago based on the Lagrangian relaxation approach [5]. In recent work, the Lagrangian relaxation-based algorithm has been extended to schedule thermal units under ramp constraints ([12]).

The Lagrangian relaxation approach is a dual method. The basic idea is to relax the system demand and spinning reserve constraints using Lagrange multipliers. The resulting dual problem is then decomposed into unit subproblems, each of which can be easily solved. Despite efficiency in solving the dual problem due to its separability, the solution of the dual problem does not necessarily yield a feasible solution of the original problem. Therefore the Lagrangian relaxation methods for solving unit commitment have often been algorithms with two phases: a dual optimization phase and a feasibility phase (see [13, 15] for a discussion and interpretation.) A common phenomenon observed in the feasibility phase is overcommitment of generating units. For the purpose of improving cost savings, our attention has focused on how to improve a feasible unit commitment while maintaining feasibility, by developing decommitment algorithms.

In [8] a new unit commitment method was proposed. The method in [8] resembles the Lagrangian relaxation approach but only the system demand constraints are relaxed with multipliers. However the multipliers are not obtained and updated by the subgradient rule but from economic dispatch. This method starts with a unit commitment with all available units on-line at all hours in the planning horizon, and improves the commitment by gradually decommitting units.

In this paper, we propose a general decommitment algorithm for power system scheduling. Given any feasible unit commitment, the proposed algorithm determines an optimal strategy for decommitting overcommitted units. Without incorporating ramp constraints, the problem is formulated as an integer programming problem and is solved by dynamic programming. The method maintains solution feasibility at all iterations. We shall show in this paper that the number of iterations required by our method to terminate is bounded by the number of units. The computation involved is fast. Because problem feasibility is always satisfied, the method provides useful information at any iteration and can serve as a good and efficient post-processing method for any existing unit commitment algorithm. In this paper, we do not directly address the intractability of the unit commitment problem, but we will discuss applying the proposed decommitment method as a complete unit commitment algorithm.

This paper is organized as follows. In Section 2, the unit commitment problem is formulated. The unit decommitment algorithm and its convergence properties are presented in Section 3. In Section 4, approximate versions of the decommitment algorithm are discussed to improve algorithm performance. An attempt to apply the unit decommitment method presented in Section 3 as a complete unit commitment method is discussed in Section 5. We provide some numerical test results and the conclusions of this paper in Section 6 and Section 7.

II Problem formulation

In this paper the following standard notation will be used. Additional symbols will be introduced when necessary.

i : index for the number of units ($i = 1, \cdots, I$)

t : index for time ($t = 0, \cdots, T$)

u_{it} : zero-one decision variable indicating whether unit i is up or down in time period t

x_{it} : state variable indicating the length of time that unit i has been up or down in time period t

t_i^{on} (t_i^{off}) : the minimum number of periods unit i must remain on (off) after it has been turned on (off)

p_{it} : state variable indicating the amount of power unit i is generating in time period t

$p_i^{\min}$ ($p_i^{\max}$) : minimum (maximum) rated capacity of unit i

$C_i(p_{it})$: fuel cost for operating unit i at output level p_{it} in time period t

$S_i(x_{i,t-1}, u_{it}, u_{i,t-1})$: startup cost associated with turning on unit i at the beginning of time period t

D_t : forecast demand in time period t

R_t : spinning capacity requirement in time period t

The unit commitment problem is formulated as the following mixed-integer programming problem: (note that the underlined variables are vectors in this paper, e.g. $\underline{u} = (u_{11}, \cdots, u_{IT})$.)

$$\min_{\underline{u},\underline{x},\underline{p}} \sum_{t=1}^{T} \sum_{i=1}^{I} [C_i(p_{it})u_{it} + S_i(x_{i,t-1}, u_{it}, u_{i,t-1})] \tag{1}$$

subject to the demand constraints,

$$\sum_{i=1}^{I} p_{it}u_{it} = D_t, \quad t = 1, \cdots, T, \tag{2}$$

and the spinning capacity constraints,

$$\sum_{i=1}^{I} p_i^{\max} u_{it} \geq R_t, \quad t = 1, \cdots, T. \tag{3}$$

There are other unit constraints such as unit capacity constraints,

$$p_i^{\min} \leq p_{it} \leq p_i^{\max}, \; i = 1, \cdots, I; \; t = 1, \cdots, T, \tag{4}$$

the state transition equation for $i = 1, \cdots, I$,

$$x_{it} = \begin{cases} \max(x_{i,t-1}, 0) + 1, & \text{if } u_{it} = 1, \\ \min(x_{i,t-1}, 0) - 1, & \text{if } u_{it} = 0, \end{cases} \tag{5}$$

the minimum up/down time constraints for $i = 1, \cdots, I$,

$$u_{it} = \begin{cases} 1, & \text{if } 1 \leq x_{i,t-1} < t_i^{\text{on}}, \\ 0, & \text{if } -1 \geq x_{i,t-1} > -t_i^{\text{off}}, \\ 0 \text{ or } 1, & \text{otherwise}, \end{cases} \tag{6}$$

and the initial conditions on x_{it} at $t = 0$ for $\forall i$.

II.1 Model of cost function

The generating cost of a thermal unit includes fuel costs and the startup costs. In this paper, the fuel cost is modeled as a convex quadratic function of the power output(MWh) of the unit:

$$C_i(p_{it}) = a_{i0} + a_{i1}p_{it} + a_{i2}p_{it}^2,\ i = 1,\cdots,I, \qquad (7)$$

where each of the a_{ij} coefficients are taken to be nonnegative [1].

The startup costs vary with the temperature of the boiler and therefore depend on the length of time that the unit has been off. The longer a unit is off, the greater the cost will be to start up the unit. It is modeled as an exponential function. To further simplify the notation, we let $S_i(\underline{u},t) = S_i(x_{i,t-1}, u_{it}, u_{i,t-1})$.

II.2 Economic dispatch

Economic dispatch is a problem of allocating the system demand among all the on-line generating units at any time in the planning horizon. In this paper, the theory is developed with respect to a given commitment $\underline{\tilde{u}}$ satisfying (3), (5) and (6). All variables hatted with a *tilde* are related to this commitment. Define the index set of on-line units at time t with respect to this feasible commitment $J(t;\underline{\tilde{u}}) \equiv \{i|\tilde{u}_{it} = 1\}$. For simplicity, $\tilde{J}_t = J(t;\underline{\tilde{u}})$. The economic dispatch problem is to determine the generation levels of on-line units so as to minimize fuel costs subject to (2) and (4).

$$edp(\tilde{J}_t,t) \equiv \min_{p_i^{\min}\le p_{it}\le p_i^{\max}} \{\sum_{i\in\tilde{J}_t} C_i(p_{it})| \sum_{i\in\tilde{J}_t} p_{it} = D_t\}, \forall t, \qquad (8)$$

and

$$\tilde{p}_{it} = \arg \min_{p_i^{\min}\le p_{it}\le p_i^{\max}} \{\sum_{i\in\tilde{J}_t} C_i(p_{it})| \sum_{i\in\tilde{J}_t} p_{it} = D_t\}, \forall t, \qquad (9)$$

where $edp(\tilde{J}_t,t)$ is the economic dispatch value at hour t, and $\underline{\tilde{p}}$ is the economic dispatch solution. Note that the economic dispatch is a quadratic programming problem. Its necessary condition of optimality is generally stated as the operation of all generators at equal marginal (incremental) cost (e.g. [14].) More precisely, there exist $\tilde{\lambda}_t$, $t = 1,\cdots,T$, such that

$$\begin{array}{ll} C_i'(\tilde{p}_{it}) = \tilde{\lambda}_t, & \text{for } p_i^{\min} < \tilde{p}_{it} < p_i^{\max} \\ C_i'(\tilde{p}_{it}) \ge \tilde{\lambda}_t, & \text{for } \tilde{p}_{it} = p_i^{\min} \\ C_i'(\tilde{p}_{it}) \le \tilde{\lambda}_t, & \text{for } \tilde{p}_{it} = p_i^{\max}, \end{array} \qquad (10)$$

for $i = 1,\cdots,I$.

Note that to guarantee the existence of solution of $edp(\tilde{J}_t,t)$ in (8), the following minimum load condition is implicitly assumed.

$$\sum_{i\in\tilde{J}_t} p_i^{\min} \le D_t \le \sum_{i\in\tilde{J}_t} p_i^{\max},\ \forall t. \qquad (11)$$

III Unit decommitment method

Given a feasible schedule $(\underline{\tilde{u}};\underline{\tilde{p}})$ (satisfying (2) to (6) and (10)), we consider the problem of decommitting unit j in hour t. Before the decommitment, the total fuel cost in hour t is $edp(\tilde{J}_t,t)$. Decommitting unit j in hour t, its generated power $\tilde{p}_{jt}$ will be distributed to other on-line units in hour t in order to satisfy (2). The total fuel generating cost after decommitment is $edp(\tilde{J}_t\backslash\{j\},t)$. The increased fuel cost of all on-line units other than unit j due to its decommitment in hour t is denoted by $\Delta C_j(\underline{\tilde{u}},\underline{\tilde{p}},t)$. The exact value of $\Delta C_j(\underline{\tilde{u}},\underline{\tilde{p}},t)$ is

$$\Delta C_j(\underline{\tilde{u}},\underline{\tilde{p}},t) = edp(\tilde{J}_t\backslash\{j\},t) - (edp(\tilde{J}_t,t) - C_j(\tilde{p}_{jt})\tilde{u}_{jt}). \qquad (12)$$

Now we consider the following problem (P_j) to improve the commitment of unit j (with other units' commitments fixed.)

(P_j)

$$\min_{u_{jt}\in\{0,1\}} \sum_{t=1}^{T} [(C_j(\tilde{p}_{jt}) + S_j(\underline{u},t))u_{jt} + \Delta C_j(\underline{\tilde{u}},\underline{\tilde{p}},t)(1-u_{jt})]$$

subject to

$$u_{jt} = \begin{cases} 0 & \text{if } \tilde{u}_{jt} = 0, \\ 1 & \text{if } \tilde{u}_{jt} = 1 \text{ and } \sum_{i\in\tilde{J}_t\backslash\{j\}} p_i^{\max} < R_t, \end{cases} \qquad (13)$$

and the minimum up/down time constraints (5) and (6) for $i = j$ with initial conditions at $t = 0$. In the sequel, the solution of (P_j) will be called the *tentative* commitment of unit j.

In solving (P_j) with respect to $(\underline{\tilde{u}};\underline{\tilde{p}})$ it can be seen that unit j remains off in off-line hours; and may be turned off if its removal will not violate the system spinning capacity requirement. The objective function shows that in an on-line hour if unit j remains on, the generating cost in that hour is the original fuel cost; while if unit j is turned off, the generating cost in that hour will be $\Delta C_j(\underline{\tilde{u}},\underline{\tilde{p}},t)$. In both cases, the startup cost is imposed when unit j is started up.

Obviously, the tentative commitment obtained in (P_j) with respect to $(\underline{\tilde{u}};\underline{\tilde{p}})$ is no worse than $\{\tilde{u}_{jt}\}_{t=1}^{T}$, because

$\{\tilde{u}_{jt}\}_{t=1}^{T}$ itself is feasible to (P_j). Note that (P_j) is an integer programming problem and can be solved using either forward or backward dynamic programming. Figure 1 shows the state transition diagram of dynamic programming. Equation (13) can be regarded as a unit availability constraint, and is modeled in the transition cost in the state transition diagram.

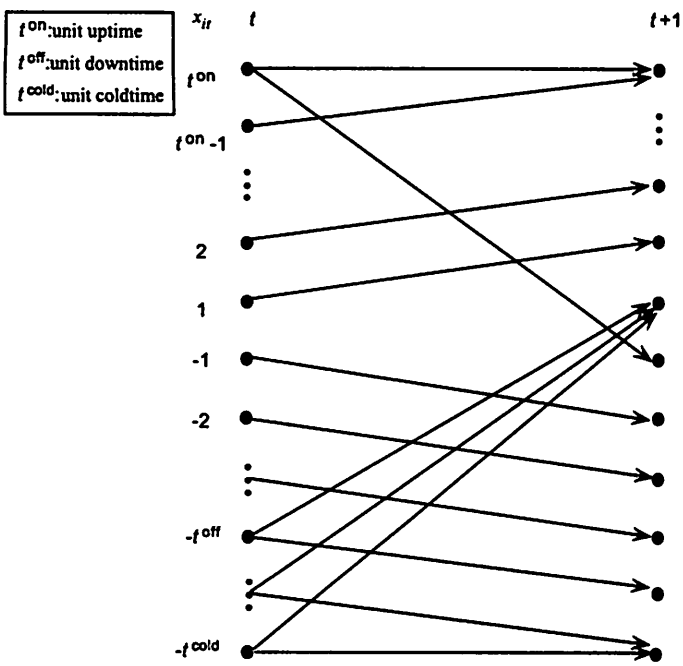

Figure 1: The state transition diagram

Based on a given unit commitment $(\tilde{\underline{u}};\tilde{\underline{p}})$, problem (P_j) improves the commitment of unit j with the commitments of all other units fixed. If we include the total generating cost of all the units other than j into the objective of (P_j), we have

$$\min_{u_{jt}\in\{0,1\}} \sum_{t=1}^{T}[(C_j(\tilde{p}_{jt})+S_j(\underline{u},t))u_{jt}+\Delta C_j(\tilde{\underline{u}},\tilde{\underline{p}},t)(1-u_{jt})]$$

$$+\sum_{i\neq j}\sum_{t=1}^{T}[C_i(\tilde{p}_{it})\tilde{u}_{it}+S_i(\tilde{\underline{u}},t)] \qquad (14)$$

$$=\min_{u_{jt}\in\{0,1\}} \sum_{t=1}^{T}[(C_j(\tilde{p}_{jt})+S_j(\underline{u},t))u_{jt}+\Delta C_j(\tilde{\underline{u}},\tilde{\underline{p}},t)(1-u_{jt})]$$

$$+\sum_{i\neq j}\sum_{t=1}^{T}[C_i(\tilde{p}_{it})\tilde{u}_{it}(u_{jt}+1-u_{jt})+S_i(\tilde{\underline{u}},t)] \quad (15)$$

$$=\min_{u_{jt}\in\{0,1\}} \sum_{t=1}^{T}[(edp(\tilde{J}_t,t)+S_j(\underline{u},t))u_{jt}$$

$$+\,edp(\tilde{J}_t\backslash\{j\},t)(1-u_{jt})]+\sum_{i\neq j}\sum_{t=1}^{T}S_i(\tilde{\underline{u}},t). \qquad (16)$$

(Note: if $j\notin\tilde{J}_t$, let $\tilde{J}_t\backslash\{j\}=\tilde{J}_t$.) Since the last term in (14) (also the last term in (16)) is a constant, the inclusion of this term to (P_j) will not affect the minimization solution of (P_j). It can be seen from (16) that solving (P_j) with respect to $(\tilde{\underline{u}};\tilde{\underline{p}})$ will determine the optimal strategy to improve $(\tilde{\underline{u}};\tilde{\underline{p}})$ by only decommitting unit j at some hours.

In the following algorithm, superscript k denotes the k-th iteration of the algorithm. Let $\tilde{\Lambda}_i^k$, $i=1,\cdots,I$ be the total generating cost (fuel cost and startup cost) of unit i of the feasible schedule $(\tilde{\underline{u}}^k;\tilde{\underline{p}}^k)$; and Λ_i^k, $i=1,\cdots,I$, the optimal objective value of (P_i^k) solved with respect to feasible solution $(\tilde{\underline{u}}^k;\tilde{\underline{p}}^k)$. We now state the decommitment algorithm.

Generic unit decommitment algorithm

Data: Feasible solution $(\tilde{\underline{u}}^0;\tilde{\underline{p}}^0)$ and the corresponding $\tilde{\Lambda}_i^0$, $i=1,\cdots,I$ are given.

Step 0: $k\leftarrow 0$.

Step 1: Solve (P_i^k) with respect to $(\tilde{\underline{u}}^k;\tilde{\underline{p}}^k)$ and obtain Λ_i^k for all $i=1,\cdots,I$.

Step 2: Select a unit m such that $(\tilde{\Lambda}_m^k-\Lambda_m^k)>0$. If there is no such a unit, stop; otherwise update the commitment of unit m in $\tilde{\underline{u}}^k$ by the commitment obtained in (P_m^k). The resultant unit commitment is assigned to be $\tilde{\underline{u}}^{k+1}$.

Step 3: Perform the economic dispatch on $\tilde{\underline{u}}^{k+1}$ to obtain $\tilde{\underline{p}}^{k+1}$ and evaluate $\tilde{\Lambda}_i^{k+1}$, the total generating cost of unit i, $i=1,\cdots,I$.

Step 4: $k\leftarrow k+1$, go to Step 1. ■

At each iteration, the tentative commitment problem (P_i) of each unit i is obtained, and the potential savings for running the tentative commitment $\tilde{\Lambda}_i-\Lambda_i$ is also calculated. The algorithm chooses the tentative commitment which can yield savings to replace the original commitment. The rule for selecting a unit to improve in Step 2, corresponding to choosing a descent direction as in continuous optimization theory, is not unique. For example,

- $m=\arg\max\{\tilde{\Lambda}_i^k-\Lambda_i^k|i=1,\cdots,I\}$ – the steepest descent direction (e.g. [9]).

- $m = \arg\max\{(\tilde{\Lambda}_i^k - \Lambda_i^k)/\Theta | i = 1, \cdots, I\}$ – the deflected gradient direction (e.g. [9]), where Θ can be, say, $p_i^{\max}$. The rule used in [8] belongs to this category.
- Assuming $C_i(p) = a_{i0} + a_{i1}p + a_{i2}p^2$ and $a_{12} \geq a_{22} \cdots \geq a_{I2}$, m is the smallest index i that has not been selected in previous iterations such that $\tilde{\Lambda}_i^k - \Lambda_i^k > 0$ – the coordinate descent method (e.g. [9]).

Theoretically, different unit selection rules in Step 2 may yield different convergence. However, our experience shows that the performance of the algorithm is insensitive, at least, to the above three selection rules. Issues about evaluation of $\Delta C_i(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$ will be discussed in Section IV.

III.1 Convergence analysis

In this section, we discuss the convergence properties of the decommitment algorithm. We will show that the algorithm will terminate within a finite number of iterations, and the number of iterations is bounded by the number of units.

Lemma 1 Given a feasible solution $(\tilde{\underline{u}}; \tilde{\underline{p}})$ and its associated $\underline{\lambda}$, for unit j in time t the following statements are true.

(i) $C_j(\tilde{p}_{jt}) \leq \lambda_t \tilde{p}_{jt}$, if $\lambda_t > C_j'(p_j^{\min})$; $C_j(\tilde{p}_{jt}) = C_j(p_j^{\min})$, if $\lambda_t \leq C_j'(p_j^{\min})$.

(ii) $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t) \geq \lambda_t \tilde{p}_{jt}$, for $\forall t$.

Proof. (i) is from the convexity of C_j and (10). Assume that solving $edp(\tilde{J}_t \backslash \{j\}, t)$ yields generation level $\tilde{p}_{it} + \Delta\tilde{p}_{it}$ for unit $i \in \tilde{J}_t \backslash \{j\}$ such that $\sum_{i \in \tilde{J}_t \backslash \{j\}} \Delta\tilde{p}_{it} = \tilde{p}_{jt}$. Since all the fuel cost functions are assumed to be smooth (7), we have

$$\begin{aligned} \Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t) &= \sum_{i \in \tilde{J}_t \backslash \{j\}} C_i(\tilde{p}_{it} + \Delta\tilde{p}_{it}) - C_i(\tilde{p}_{it}) \\ &\geq \sum_{i \in \tilde{J}_t \backslash \{j\}} C_i'(\tilde{p}_{it}) \Delta\tilde{p}_{it} \\ &\geq \tilde{\lambda}_t \sum_{i \in \tilde{J}_t \backslash \{j\}} \Delta\tilde{p}_{it} = \tilde{\lambda}_t \tilde{p}_{jt}. \end{aligned} \tag{17}$$

(Note that for those i such that $C_i'(\tilde{p}_{it}) \leq \lambda_t$, $\Delta\tilde{p}_{it} = 0$.) ■

Lemma 2 Given a feasible solution $(\tilde{\underline{u}}; \tilde{\underline{p}})$ and its associated $\underline{\lambda}$, for unit j in time t the following statements are true.

(i) If $\tilde{u}_{jt} = 0$, $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t) = 0$.

(ii) If $\tilde{u}_{jt} = 1$, $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t) > 0$.

(iii) If $\tilde{u}_{jt} = 0$ at the k'-th iteration, i.e., $\tilde{u}_{jt}^{k'} = 0$, then $\tilde{u}_{jt}^k = 0$ and $\Delta C_j(\tilde{\underline{u}}^k, \tilde{\underline{p}}^k, t) = 0$ for $\forall k > k'$.

(iv) Both $\{C_j(\tilde{p}_{jt}^k)\}$ and $\{\lambda_t^k\}$ are nondecreasing sequences in k, for all t.

(v) $\{\Delta C_j(\tilde{\underline{u}}^k, \tilde{\underline{p}}^k, t)\}$ is a nondecreasing sequences in k before unit j is decommitted in time t at some iteration.

Proof. Statements (i) to (iii) are obvious. Since the algorithm only involves unit decommitment and the load balance equation is satisfied at all iterations, (iv) is true. To prove statement (v), assume at iteration k, the generation levels of an on-line unit $i \neq j$ will increase Δp_{it} due to the decommitment of unit j. We have

$$\begin{aligned} \Delta C_j(\tilde{\underline{u}}^k, \tilde{\underline{p}}^k, t) = \min_{\Delta p_{it}} & \sum_{i \in \tilde{J}_t^k \backslash \{j\}} C_i(\tilde{p}_{it}^k + \Delta p_{it}) - C_i(\tilde{p}_{it}^k) \\ \text{s.t. } & \textstyle\sum_{i \in \tilde{J}_t^k \backslash \{j\}} \Delta p_{it} = \tilde{p}_{jt}^k \\ & 0 \leq \Delta p_{it} \leq p_i^{\max} - \tilde{p}_{it}^k, \ \forall i \in \tilde{J}_t^k \backslash \{j\} \\ = \min_{\Delta p_{it}} & \sum_{i \in \tilde{J}_t^k \backslash \{j\}} C_i'(\tilde{p}_{it}^k) \Delta p_{it} + 2a_{i2} \Delta p_{it}^2 \\ \text{s.t. } & \textstyle\sum_{i \in \tilde{J}_t^k \backslash \{j\}} \Delta p_{it} = \tilde{p}_{jt}^k \\ & 0 \leq \Delta p_{it} \leq p_i^{\max} - \tilde{p}_{it}^k, \ \forall i \in \tilde{J}_t^k \backslash \{j\}. \end{aligned}$$

Consider two iterations k' and k'' such that $k' < k''$ and $\Delta C_j(\tilde{\underline{u}}^{k'}, \tilde{\underline{p}}^{k'}, t)$, $\Delta C_j(\tilde{\underline{u}}^{k''}, \tilde{\underline{p}}^{k''}, t)$ are nonzero. Note that $\tilde{J}_t^{k''} \subseteq \tilde{J}_t^{k'}$, $\tilde{p}_{it}^{k'} \leq \tilde{p}_{it}^{k''}$ and $0 < C_i'(\tilde{p}_{it}^{k'}) \leq C_i'(\tilde{p}_{it}^{k''})$ for all $i \in \tilde{J}_t^{k''}$. Based on this information, it is straightforward to show that $\Delta C_j(\tilde{\underline{u}}^{k'}, \tilde{\underline{p}}^{k'}, t) \leq \Delta C_j(\tilde{\underline{u}}^{k''}, \tilde{\underline{p}}^{k''}, t)$ under these conditions. ■

Note that the results in Lemma 2 can be generalized to the case where the fuel cost functions for generating units are convex, not limited to quadratic functions. An intuitive interpretation of the final step in the proof is stated as follows: $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$ is the economic dispatch of $\tilde{p}_{jt}$ to other on-line units. Compare two distinct iterations k' and k'' ($k'' > k'$). In the later iteration k'', one needs to dispatch more power ($\tilde{p}_{jt}^{k''} \geq \tilde{p}_{it}^{k'}$) to with fewer units ($\tilde{J}_t^{k''} \subseteq \tilde{J}_t^{k'}$) while the dispatching fee gets higher as the iteration proceeds ($0 < C_i'(\tilde{p}_{it}^{k'}) \le C_i'(\tilde{p}_{it}^{k''})$). Therefore $\Delta C_j(\underline{\hat{u}}^{k'}, \underline{\tilde{p}}^{k'}, t) \le \Delta C_j(\underline{\hat{u}}^{k''}, \underline{\tilde{p}}^{k''}, t)$.

Theorem 3 A unit, once it has been selected in Step 2 in the generic unit decommitment algorithm at some iteration, will not be selected again in Step 2 at any future iteration. So the decommitment algorithm terminates within I iterations, where I is the number of the units.

Proof. Suppose unit j is selected at iteration k', and its tentative commitment of unit j is $\{\hat{u}_{jt}^{k'}\}$ $(=\{\tilde{u}_{jt}^{k'+1}\})$, so

$$\sum_{t=1}^{T}[(C_j(\tilde{p}_{jt}^{k'})+S_j(\underline{\hat{u}}^{k'},t))\hat{u}_{jt}^{k'}+\Delta C_j(\underline{\hat{u}}^{k'},\underline{\tilde{p}}^{k'},t)(1-\hat{u}_{jt}^{k'})] \le$$

$$\sum_{t=1}^{T}[(C_j(\tilde{p}_{jt}^{k'}) + S_j(\underline{u},t))u_{jt} + \Delta C_j(\underline{\hat{u}}^{k'},\underline{\tilde{p}}^{k'},t)(1-u_{jt})], \tag{18}$$

for any $\{u_{jt}\}$ satisfying (13).

Now assume on the contrary that unit j is selected again in Step 2 (for the first time) at iteration $k'' > k'$, and assume that the tentative commitment is $\{\hat{u}_{jt}^{k''}\}(=\{\tilde{u}_{jt}^{k''+1}\})$, i.e.,

$$\sum_{t=1}^{T}[(C_j(\tilde{p}_{jt}^{k''})+S_j(\underline{\hat{u}}^{k'},t))\hat{u}_{jt}^{k'}+\Delta C_j(\underline{\hat{u}}^{k'},\underline{\tilde{p}}^{k''},t)(1-\hat{u}_{jt}^{k'})] >$$

$$\sum_{t=1}^{T}[(C_j(\tilde{p}_{jt}^{k''})+S_j(\underline{\hat{u}}^{k''},t))\hat{u}_{jt}^{k''}+\Delta C_j(\underline{\hat{u}}^{k''},\underline{\tilde{p}}^{k''},t)(1-\hat{u}_{jt}^{k''})]. \tag{19}$$

Let $\Gamma = \{t | \hat{u}_{jt}^{k'} \neq \hat{u}_{jt}^{k''}, t = 1, \cdots, T\}$, i.e., $\hat{u}_{jt}^{k'} = 1$ but $\hat{u}_{jt}^{k''} = 0$ for $\forall t \in \Gamma$. With $u_{jt} = \hat{u}_{jt}^{k''}$ substituted into (18), (18) is reduced to

$$\sum_{t\in\Gamma} C_j(\tilde{p}_{jt}^{k'}) + \Delta S_j \le \sum_{t\in\Gamma} \Delta C_j(\underline{\hat{u}}^{k'}, \underline{\tilde{p}}^{k'}, t), \tag{20}$$

where $\Delta S_j = \sum_{t=1}^{T}(S_j(\underline{\hat{u}}^{k'},t)\hat{u}_{jt}^{k'} - S_j(\underline{\hat{u}}^{k''},t)\hat{u}_{jt}^{k''})$. Similarly (19) is equivalent to

$$\sum_{t\in\Gamma} C_j(\tilde{p}_{jt}^{k''}) + \Delta S_j > \sum_{t\in\Gamma} \Delta C_j(\underline{\hat{u}}^{k''}, \underline{\tilde{p}}^{k''}, t). \tag{21}$$

From Lemmas 1 and 2, we know that at any t either $\Delta C_j(\underline{\hat{u}}^{k}, \underline{\tilde{p}}^{k}, t) \ge C_j(\tilde{p}_{jt}^{k})$, for $\forall k \ge k'$, or $\Delta C_j(\underline{\hat{u}}^{k}, \underline{\tilde{p}}^{k}, t) < C_j(\tilde{p}_{jt}^{k})$ at $k = k'$ but $\Delta C_j(\underline{\hat{u}}^{k}, \underline{\tilde{p}}^{k}, t)$ is getting closer to or even becomes greater than $C_j(\tilde{p}_{jt}^{k})$ as k increases. This implies that

$$\sum_{t\in\Gamma} C_j(\tilde{p}_{jt}^{k}) + \Delta S_j \le \sum_{t\in\Gamma} \Delta C_j(\underline{\hat{u}}^{k}, \underline{\tilde{p}}^{k}, t) \tag{22}$$

should continue to hold for all $k > k'$. But this contradicts (21). So unit j should not be selected again to improve after iteration k'. ■

To summarize Theorem 3, note that Lemma 2 (v) shows that it becomes less advantageous to decommit the same unit in the same hour as iteration proceeds. Therefore it could never occur that a unit facing a lower cost for decommitment is not decommitted in an hour at some iteration, but is decommitted in the same hour at a future iteration facing a higher cost. Also it is worthy to note that the properties stated in Lemmas 1 and 2, and Theorem 3 are independent of the unit selection rule in Step 2 in the algorithm.

Remark Based on the result in Theorem 3, those units which have been selected in Step 2 at some iteration can be exempt from consideration in Step 1 and Step 2 of future iterations. This can improve the computational speed of the algorithm.

IV Approximate methods

In this section, we discuss methods to approximate $\Delta C_j(\underline{\tilde{u}}, \underline{\tilde{p}}, t)$ in the objective function of (P_j). As previously mentioned, $\Delta C_j(\underline{\tilde{u}}, \underline{\tilde{p}}, t)$ is an estimate of the increased generating costs of all on-line units other than unit j due to its decommitment. The exact value of $\Delta C_j(\underline{\tilde{u}}, \underline{\tilde{p}}, t)$ is given in (12). To solve (12), equivalently an extra economic dispatch has to be solved, i.e. $edp(\tilde{J}_t \backslash \{j\}, t)$, for $\forall t$. Therefore at each iteration of the decommitment algorithm $I + 1$ economic dispatches are performed. By the Theorem, the algorithm will require performing no more than $I + I^2$ economic dispatches in total. Based on the remark in the end of the previous section, the upper bound on the number of economic dispatches required can be reduced to $I + I(I + 1)/2$. Even though the economic dispatch can be efficiently solved (e.g. [7, 14]), in a large-scale system it is desired to approximate $\Delta C_j(\underline{\tilde{u}}, \underline{\tilde{p}}, t)$ without performing an extra economic dispatch. Next we discuss two methods to approximate $\Delta C_j(\underline{\tilde{u}}, \underline{\tilde{p}}, t)$.

IV.1 Guaranteed descent method

If $\Delta C_j(\underline{\tilde{u}}, \underline{\tilde{p}}, t)$ is approximated, a sufficient condition to guarantee that solving (P_j) with respect to $(\underline{\tilde{u}}; \underline{\tilde{p}})$ can yield a commitment no worse than $\tilde{\underline{u}}$ is that $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$ satisfies

$$\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t) \geq edp(\tilde{J}_t \backslash \{j\}, t) - (edp(\tilde{J}_t, t) - C_j(\tilde{p}_{jt})\tilde{u}_{jt}). \tag{23}$$

That is, $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$ should be overestimated. Since the last two terms (in parentheses) in (23) are fixed and known when solving (P_j), the exact value of ΔC_j given in (12) can be regarded as the value of the solution of a minimization problem, i.e., the economic dispatch $edp(\tilde{J}_t \backslash \{j\}, t)$. To overestimate the solution value of a minimization problem, any feasible solution of the minimization problem will do. So any feasible dispatch among the unit index set $\tilde{J}_t \backslash \{j\}$ can be applied to satisfy (23). An easy way to create a feasible and reasonably good dispatch is to apply a priority list. All the units are assigned a priority order which is determined by the slope of the incremental cost curve $2a_{i2}$ of the unit. At time t, if unit $i = j$ is to be decommitted, its generation amount $\tilde{p}_{jt}$ will be distributed to other on-line units. All the on-line units other than unit j consume $\tilde{p}_{jt}$ in the order of the priority list subject to unit capacity constraint until $\tilde{p}_{jt}$ is totally consumed. The algorithm to overestimate the exact value of $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$ by the priority list is given below.

Guaranteed descent method: by priority list

Step 0: $d \leftarrow \tilde{p}_{jt}$; $J \leftarrow \tilde{J}_t \backslash \{j\}$; $\Delta C \leftarrow 0$.

Step 1: Let $l = \arg\min_{i \in J} a_{i2}$. Obtain the following:

$$\hat{p}_{lt} = \min(\tilde{p}_{lt} + d, p_l^{\max}), \tag{24}$$

$$d \leftarrow d - (\hat{p}_{lt} - \tilde{p}_{lt}), \tag{25}$$

and

$$\Delta C \leftarrow \Delta C + C_l(\hat{p}_{lt}) - C_l(\tilde{p}_{lt})$$

$$= \Delta C + a_{l1}(\hat{p}_{lt} - \tilde{p}_{lt}) + a_{l2}(\hat{p}_{lt}^2 - \tilde{p}_{lt}^2). \tag{26}$$

Step 2: If $d = 0$, stop and ΔC is the estimation; otherwise $J \leftarrow J \backslash \{l\}$ and go to Step 1. ■

The priority list algorithm above is very efficient. It obtains a good approximation of $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$ based on our simulation experience.

IV.2 First order approximate method

Another method of approximating ΔC_j obtains a first order approximation. Assume that solving $edp(\tilde{J}_t \backslash \{j\}, t)$ yields generation level $\tilde{p}_{it} + \Delta\tilde{p}_{it}$ for unit $i \in \tilde{J}_t \backslash \{j\}$ such that $\sum_{i \in \tilde{J}_t \backslash \{j\}} \Delta\tilde{p}_{it} = \tilde{p}_{jt}$. Since all the fuel cost functions are assumed to be smooth (7), we have

$$\begin{aligned} \Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t) &= \sum_{i \in \tilde{J}_t \backslash \{j\}} C_i(\tilde{p}_{it} + \Delta\tilde{p}_{it}) - C_i(\tilde{p}_{it}) \\ &\approx \sum_{i \in \tilde{J}_t \backslash \{j\}} C_i'(\tilde{p}_{it}) \Delta\tilde{p}_{it} \\ &\approx \tilde{\lambda}_t \sum_{i \in \tilde{J}_t \backslash \{j\}} \Delta\tilde{p}_{it} = \tilde{\lambda}_t \tilde{p}_{jt}, \end{aligned} \tag{27}$$

where $\tilde{\lambda}_t$ is the marginal cost at hour t obtained in solving $edp(\tilde{J}_t)$. Note that in (27) we apply the optimality condition of economic dispatch, i.e. operating all units at the same marginal cost. However, this is only an approximation (cf. (10)) of the incremental costs of generating units when unit capacity constraints are present.

Also by Lemma 1 (ii) we know that the first order approximate method always underestimates the exact value of $\Delta C_j(\tilde{\underline{u}}, \tilde{\underline{p}}, t)$. Therefore the selected tentative commitment does not necessarily guarantee improvement. This sometimes causes the algorithm to terminate prematurely.

After substituting ΔC_j in (13) by $\tilde{\lambda}_t \tilde{p}_{jt}$ and rearranging terms, the objective of (P_j) is equal to the following.

$$\min_{u_{jt} \in \{0,1\}} \sum_{t=1}^{T} [C_j(\tilde{p}_{jt}) u_{jt} - \tilde{\lambda}_t \tilde{p}_{jt} u_{jt} + S_j(\underline{u}, t)] + \sum_{t=1}^{T} \tilde{\lambda}_t \tilde{p}_{jt}. \tag{28}$$

The last term in (28) is a constant and can be ignored in the objective. Since $\tilde{\lambda}_t$ is the marginal cost obtained from economic dispatch, we have the following proposition.

Proposition 4 The following problem $(\bar{P}_j)$ yields the same commitment solution as (P_j) with respect to $(\tilde{\underline{u}}; \tilde{\underline{p}})$.

$$(\bar{P}_j) \quad \min_{u_{jt}, p_{jt}} \sum_{t=1}^{T} [C_j(p_{jt}) u_{jt} - \tilde{\lambda}_t p_{jt} u_{jt} + S_j(\underline{u}, t)]$$

subject to (3), (4), (5), (6) and (13), where $\{\lambda_t\}$ are associated with $(\tilde{\underline{u}}; \tilde{\underline{p}})$ satisfying (10).

Proof. Since $\tilde{p}_{it}$ and $\tilde{\lambda}_t$ satisfy (10), when $u_{jt} = 1$, $\tilde{p}_{jt}$ solves $\min[C_j(p_{jt}) - \tilde{\lambda}_t p_{jt}]$ subject to the unit capacity constraint $p_j^{\min} \leq p_{jt} \leq p_j^{\max}$. ■

$(\bar{P}_j)$ resembles a unit subproblem in the Lagrangian relaxation approach for solving the unit commitment problem, but in which only the system load equations are relaxed, and the values of Lagrangian multipliers are taken from economic dispatch. This was also the basic idea behind the method in [8]. The method in [8] initially turns on as many units as possible. It solves problem $(\bar{P}_j)$ but only subject to (4), (5) and (6), and repeatedly updates the problem in the manner of the decommitment algorithm proposed in Section 3 of this paper. Since originally all units were turned on, any improvement in the commitment through $(\bar{P}_j)$ can only involve unit decommitment.

Finally, it is not difficult to show that the ΔC_j generated by both approximate methods presented in this section satisfies the properties stated in Lemma 1 (ii) and Lemma 2. Therefore a result similar to that of Theorem 3 holds even for the approximate methods. We state the final theorem without a proof.

Theorem 5 The generic unit decommitment algorithm, using either the guarantee descent method or the first order approximate method to approximate ΔC_j, terminates within I iterations, where I is the number of the units. ■

V Discussion

V.1 Optimal unit decommitment

The proposed unit decommitment algorithm at each iteration determines an optimal strategy to improve by a single unit at a time. Similarly, an optimal strategy to improve two units at a time can be devised. In that case, in each hour in the planning horizon, at most four combinations of the corresponding on-line units are considered for decommitment. If solved by dynamic programming, this requires extending the state space to include these combinations of on-line units. Intuitively, an *optimal unit decommitment* in an hour considers all possible combinations of all on-line units to be decommitted and determines the optimal strategy, which itself is a constrained unit commitment problem. Therefore, the optimal unit decommitment problem is a difficult combinatorial problem. In our design, the unit decommitment algorithm is a post-processing method to aid the two-phase (dual optimization phase and feasibility phase) Lagrangian relaxation method in improving the solution quality. Given a feasible solution obtained by the Lagrangian relaxation method, it is believed that the possibility of decommitting more than one unit at a time without affecting feasibility is relatively rare.

V.2 Solving unit commitment by unit decommitment

Applied to solve a complete unit commitment problem with all the available units turned on at all hours initially, our unit decommitment algorithm falls into the class of methods as proposed in [8]. When the system is thus overcommitted (in terms of surplus between the system capacity and system demand to measure the possible combinations of unit decommitment), it is not clear whether improving one unit at a time with other units fixed is a near-optimal strategy to improve the current unit commitment. Recently we have conducted extensive testing on the unit decommitment method proposed herein as a unit commitment algorithm, and compared it with the Lagrangian relaxation method. The test results suggest that this method could be an efficient way to obtain a reasonably good solution of the unit commitment problem. We shall present further results of research along this direction in a future paper.

VI Numerical results

The decommitment algorithms proposed in this paper have been implemented in FORTRAN on an HP 700 workstation. To compare the performance of the decommitment algorithm (DA) with its two approximate algorithms, the guaranteed descent method (DA-GD) and the first order method (DA-FO) introduced in Section 4, we randomly generate unit commitment instances and then apply these three algorithms to solve. All three methods use the steepest descent type of selection rule in Step 2 in the algorithm. Eight cases of systems with different numbers of units and lengths of planning horizon are tested. Cases are denoted by (no. of units × no. of hours in planning horizon): (10 × 24), (10 × 168), (20 × 24), (20 × 168), (30 × 24), (30 × 168), (40 × 24) and (40 × 168). Unit commitment instances of each case are randomly generated. Let $rand(x, y)$ denote a random number generator which generates numbers uniformly between numbers x and y. The detailed configuration of randomly generating the unit commitment instance is shown in Table 1.

Table 1: Configuration of unit commitment instances

t_i^{on}	the nearest integer of $rand(1,6)$
t_i^{off}	the nearest integer of $rand(1,6)$
$t_i^{cold\,\dagger}$	t_i^{off} + the nearest integer of $rand(5,25)$
x_{i0}	the nearest nonzero integer of $rand(1,20) - 10$
p_i^{min}	$rand(50,300)$
p_i^{max}	$p_i^{min} + rand(100,500)$
$C_i(p_{it})$	$a_{i0} = rand(175,800)$; $a_{i1} = rand(7,9)$; $a_{i2} = rand(0.001,0.005)$
$S_i(\underline{u},t)$	$S_i(\underline{u},t) = b_{i1}(1 - \exp(-x_{it}/\rho_i) + b_{i2})$, $x_{it} < 0$. $b_{i1} = rand(1200,2400)$; $b_{i2} = rand(2000,3500)$; $\rho_i = rand(2,5)$
D_t	$rand(\sum_i p_i^{min}\tilde{u}_{it}, 0.9\sum_i p_i^{max}\tilde{u}_{it})$
R_t	$1.07D_t$

† : t_i^{cold}: unit cold time.

The given initial feasible commitment $\tilde{\underline{u}}$ is generated uniformly among $\{0,1\}$ in succession, starting from the first hour, while the minimum uptime and downtime constraints are satisfied. Note that the demand D_t is generated after $\tilde{\underline{u}}$ to guarantee (11) is satisfied.

When testing each case, 100 instances are generated based on the configuration in Table 1. For each instance, the three algorithms are applied. Since each instance is different, only relative performances of the three algorithms is recorded, including the relative total generating costs (in Table 2) and the relative CPU time (in Table 3). However, we also provide the average cost and CPU time of the 100 instances of each case solved by DA in Table 2 and 3 respectively. Note that in the following tables, the two numbers inside the parentheses under an averaged number indicate the range of all the corresponding sample points obtained during testing of all instances.

Table 2 and Table 3 show that the DA-GD and the DA-FO methods are both good approximate methods for the DA. In terms of cost saving, DA-GD, on average, obtains better solutions than DA and DA-FO. The differences between the solutions obtained by these three methods are generally within 2% based on our testing. It is clear that both approximate methods, DA-GD and DA-FO, require much less computational time than the DA, and require lesser CPU time as the system gets larger. The DA-GD requires a sorting procedure to determine the priority list, and therefore always takes more time than the DA-FO. Based on the comparison and our experience, we recommend the DA-GD method because of its guaranteed descent property; on the other hand without this property DA-FO in very rare cases will terminate in the first few iterations due to a poorer estimate of $\Delta C_j(\tilde{\underline{u}}, \underline{\tilde{p}}, t)$ in (12) as discussed in Section 3.1.

Table 2: Average relative costs

Case	DA	DA-GD	DA-FO	Avg Cost of DA ($\$10^6$)
10×24	1	0.999 (0.978-1.011)	1.011 (0.995-1.018)	0.492
10×168	1	0.999 (0.978-1.011)	1.001 (0.999-1.011)	3.338
20×24	1	0.999 (0.978-1.011)	1.001 (0.995-1.018)	0.953
20×168	1	0.999 (0.978-1.011)	1.001 (0.995-1.018)	6.662
30×24	1	0.998 (0.978-1.011)	1.001 (0.995-1.018)	1.404
30×168	1	0.999 (0.978-1.011)	1.001 (0.995-1.018)	9.915
40×24	1	0.999 (0.978-1.011)	1.001 (0.995-1.018)	1.876
40×168	1	0.998 (0.978-1.011)	1.001 (0.995-1.018)	13.313

Table 3: Average relative CPU time

Case	DA	DA-GD	DA-FO	Avg CPU time of DA (sec.)
10×24	1	0.717 (0.500-0.727)	0.670 (0.416-0.882)	0.189
10×168	1	0.732 (0.995-1.018)	0.643 (0.978-1.011)	1.786
20×24	1	0.510 (0.416-0.540)	0.467 (0.349-0.637)	1.147
20×168	1	0.541 (0.426-0.513)	0.472 (0.344-0.585)	9.130
30×24	1	0.401 (0.319-0.370)	0.356 (0.223-0.444)	3.2783
30×168	1	0.419 (0.355-0.427)	0.359 (0.289-0.450)	31.305
40×24	1	0.340 (0.250-0.312)	0.298 (0.192-0.391)	5.858
40×168	1	0.348 (0.305-0.337)	0.298 (0.230-0.351)	56.001

Table 4 records the number of iterations required by these three methods on the testing cases. The result justifies the convergence analysis in Section 3.1.

As aforementioned, the unit decommitment method is

Table 4: Number of iterations

case	DA	DA-GD	DA-FO
10x24	6.01 (4-9)	5.88 (3-9)	5.66 (3-9)
10x168	8.83 (6-10)	8.87 (6-10)	7.81 (5-10)
20x24	10.91 (7-15)	10.71 (7-14)	10.46 (7-14)
20x168	16.14 (12-19)	15.83 (13-19)	14.65 (10-18)
30x24	16.03 (11-21)	15.66 (10-20)	15.32 (9-19)
30x168	23.34 (19-28)	22.91 (19-27)	21.54 (16-26)
40x24	20.49 (15-26)	20.04 (15-25)	19.88 (12-25)
40x168	30.33 (26-36)	29.48 (25-35)	28.51 (22-35)

Table 5: Comparison of methods for unit commitment

No. of iter.	Total cost ($)		Duality Gap (%)	CPU time (sec)
20	Dual Obj.	9252032		
	Feas. Ph.	9300814	0.53	17.56
	DA-GD	9288790	0.39	8.40
25	Dual Obj.	9254664		
	Feas. Ph.	9295743	0.44	18.36
	DA-GD	9287256	0.35	5.2
30	Dual Obj.	9255323		
	Feas. Ph.	9295024	0.43	22.11
	DA-GD	9287479	0.35	8.84
35	Dual Obj.	9255988		
	Feas. Ph.	9292481	0.39	23.57
	DA-GD	9288322	0.35	7.90
40	Dual Obj.	9256526		
	Feas. Ph.	9296280	0.43	23.27
	DA-GD	9280256	0.26	6.54
45	Dual Obj.	9257763		
	Feas. Ph.	9320547	0.68	25.78
	DA-GD	9281954	0.26	7.24
50	Dual Obj.	9257659		
	Feas. Ph.	9292240	0.37	26.46
	DA-GD	9280092	0.24	7.42
	DA	9395589	–	40.56
	DA-GD	9382742	–	21.28
	DA-FO	9433649	–	20.65

proposed as a post processing module for unit commitment algorithms. We integrate the three methods into a Lagrangian relaxation (LR) algorithm, and examine how the unit decommitment algorithm can improve the solution of the LR approach. A 30-unit-168-hour unit commitment instance is solved using the LR algorithm with different stopping criteria. For purpose of illustration, the stopping criterion adopted here is the number of iterations. The test result is given in Table 5. In the test, we operate the LR dual optimization and terminate it at a given number of iterations. The dual objective value is recorded in Table 5 in the first row in each section. A feasibility phase follows to obtain a feasible solution, whose corresponding total generating cost is recorded in the second row of each section in Table 5. Based on this feasible solution, we apply the DA-GD algorithm to improve the feasible solution and obtain an improved generating cost recorded in the third row. The duality gaps corresponding to before and after applying the DA-GD algorithm are also given in Table 5. In the column under CPU time, in each section, the upper number is the CPU time required by a complete LR algorithm. The lower number only indicates the CPU time required by the DA-GD.

In Table 5, one can see that the dual optimum should be approached between the 45th and the 50th iteration. Because of the highly nonlinear behavior of the feasibility phase, it is not clear when to terminate the dual optimization followed by feasibility phase to yield the best result. Of course the duality theory suggests fully solving the dual optimization. Compare the two cases: terminating the dual optimization at the 40th and the 45th iteration, the latter with higher dual objective value yields a worse solution after the feasibility phase. The badness of this solution implies overcommitment. After applying DA-GD, the duality gap is reduced from 0.68% to 0.26%. In Table 5, it can also be seen that the unit decommitment algorithm generally lessens the nonlinearity of the feasibility phase, and make the solution less sensitive to the number of iterations in the dual optimization as the stopping criterion. Also note that with unit decommitment, it may be a good strategy to moderately overestimate generating units in the feasibility phase. In terms of the CPU time required for obtaining a feasible solution, Table 5 suggests a possibly advantageous strategy of applying the (cheap) unit decommitment algorithm to substitute the (expensive) dual optimization. Also, by determining how much saving could be achieved by unit decommittment, our method can be used as a tool to measure the solution quality obtained by a unit commitment algorithm.

In the bottom section of Table 5, the three algorithms are applied as a complete unit commitment algorithm, with initially all the units turned on to the most extent without violating the minimum uptime and downtime constraints. As discussed in Section 5, the approximate method (DA-GD) efficiently obtains a reasonably good solution compared with regular unit commitment algorithms like LR.

VII Conclusions

In this paper we present a unit decommitment method for power system scheduling. Given a feasible unit commitment, our algorithm determines an optimal strategy for decommitting overcommitted units based on dynamic programming. The method was developed as a post processing tool to aid the existing unit commitment algorithm in improving solution quality. Two approximate methods are proposed. In our numerical test, we show that the two approximate methods are much faster than the proposed decommitment algorithm, and the difference between the solutions obtained by the proposed decommitment algorithm and the two approximate algorithms are within 2% on average. We also have integrated one approximate algorithm into a Lagrangian relaxation algorithm. The numerical results show that with unit decommitment, solution quality can be generally improved efficiently. Unit decommitment also lessens the dependency on heuristics like stopping criteria in algorithm operation. The method of unit decommitment presented in this paper has been extended to a more complicated case with ramp constraints. We shall present this extension in a future paper.

Acknowledgement

This work was partly supported by the National Science Foundation under Grant IRI-9120074 and by PG&E's R&D Department. Their support is greatly appreciated.

References

[1] J. C. Ammons and L. F. McGinnis, "Production costing in electric utilities", *Management Science*, vol. 29, pp. 307-316, 1983.

[2] D. P. Bertsekas, G. S. Lauer, N. R. Sandell, Jr., and T. A. Posbergh, "Optimal short-term scheduling of large-scale power systems," *IEEE Tran. Automatic Control*, vol. AC-28, no. 1, pp. 1-11, 1983.

[3] W. G. Chandler, P. L. Dandeno, A. F. Gilmn and L. K. Kirchmayer, "Short-range operation of a combined thermal and hydroelectric power system", *AIEE, Transactions*, paper no. 53-326, 1953.

[4] A. Cohen and V. Sherkat, "Optimization-based methods for operations scheduling," *Proceedings of IEEE*, vol. 75, no. 12, pp. 1574-1591, 1987.

[5] L. A. F. M. Ferreira, T. Andersson, C. F. Imparato, T. E. Miller, C. K. Pang, A. Svoboda, and A. F. Vojdani, "Short-term resource scheduling in multi-area hydrothermal power systems," *Electrical Power & Energy Systems*, vol. 11, no. 3, pp. 200-212, 1989.

[6] X. Guan, P. B. Luh, and H. Yan, "An optimization-based method for unit commitment," *Electrical Power & Energy Systems*, vol. 14, no. 1, pp. 9-17, February 1992.

[7] J. Guddat, Q. Römisch, and R. Schultz, "Some applications of mathematical programming techniques in optimal power dispatch", *Computing*, vol. 49, no. 3, pp. 193-200, 1992.

[8] C. A. Li, R. B. Johnson, and A. J. Svoboda, "A new unit commitment method", presented at the IEEE/PES 1995 Winter meeting.

[9] D. G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley, 1984.

[10] C. K. Pang and H. C. Chen, "Optimal short-term thermal unit commitment", *IEEE Transactions of Power Apparatus and Systems*, vol. PAS-95, no. 4, pp. 1336-1342, July/August 1976.

[11] C. K. Pang, G. B. Sheble and F. Albuyeh, "Evaluation of dynamic programming based methods and multiple area representation for thermal unit commitments", *IEEE Tran. Power Apparatus and Systems*, vol. PAS-100, no. 3, pp. 1212-1218, March, 1981.

[12] A. J. Svoboda, C. L. Tseng, C. A. Li, and R. B. Johnson, 'Short term resource scheduling with ramp constraints', presented at the 1995 IEEE/PES Winter meeting.

[13] C. L. Tseng, S. S. Oren, A. J. Svoboda, and R. B. Johnson, 'Price-based adaptive spinning reserve requirements in power system scheduling', Report UCB/ERL/IGCT M96/32, UC Berkeley, May 1996.

[14] A. J. Wood and B. F. Wollenberg, *Power Generation, Operation and Control*, John Wiley & Sons, New York, 1984.

[15] F. Zhuang and F. D. Galiana, "Towards a more rigorous and practical unit commitment by Lagrangian relaxation," *IEEE Trans Power Systems*, vol. PWRS-3, no. 2, pp. 763-770, 1988.