Parallel Processing Implementation of the Unit Commitment Problem with Full AC Power Flow Constraints

Carlos E. Murillo–Sánchez
Robert J. Thomas

Cornell University
School of Electrical Engineering
Ithaca, NY 14853

Abstract

In this paper, the authors describe a parallel implementation of the Lagrangian Relaxation Algorithm with variable duplication for the thermal unit commitment problem. The formulation was previously reported by the authors and allows inclusion of the full nonlinear AC network power flow model, which permits addressing voltage limits, as well as more realistic branch flow limits than is possible with a linear DC flow model. Thus, potential VAr production can be used as another criterion for commitment of otherwise expensive generators in strategic locations. The algorithm is highly parallelizable, and the authors have taken advantage of this in a version currently being developed for the Cornell Theory Center's Velocity AC3 NT cluster.

1 Introduction

As discussed by the authors when the algorithm used in this paper was presented for the first time in [26], the main reason for using Lagrangian relaxation for solving the unit commitment problem is that separation of a very specific kind is achieved. Lagrangian relaxation allows trading a direct solution approach exhibiting combinatoric complexity (such as dynamic programming or mixed integer programming) for an iterative process that is not guaranteed to find the global optimum, but which in practice finds solutions with very small duality gaps, and whose complexity for a given iteration is roughly proportional to the number of integer variables, not combinatoric.

The original Lagrangian relaxation method could only deal with linear constraints, so other authors never went beyond using linear approximations of the network constraints in their formulations. In [26], the authors employed a variable duplication technique that, together with the constraint structure of the problem, allowed them to reformulate the problem in a way that permits the inclusion of nonlinear network constraints. Furthermore, the resulting iterative process for the solution of the dual maximization problem is highly parallelizable, which is excellent because the algorithm does need a large amount of computational power: for every dual iteration, it needs to perform as many Optimal Power Flows (OPF's) as there are time slices in the planning horizon. Without resorting to a parallel implementation, it is difficult to present results for any larger systems. This paper is meant to relate the experiences of the authors obtained in the process of implementing the parallel algorithm. Section 1 presents the problem, the philosophy behind using Lagrangian relaxation and the reasons for considering a nonlinear AC power flow model. Section 2 presents the formulation and dual maximization algorithm. Section 3 presents some computational results obtained in a serial implementation. Section 4 describes a parallel implementation currently being developed, and Section 5 describes future work and conclusions so far.

1.1 The Unit Commitment Problem

The unit commitment problem is a mixed–integer mathematical program which can be formulated generally as:

$$\min_{P,Q,U} \{\ F(P,U) + K(U) \quad | \quad (P,U) \in \mathcal{D},\ (P,Q,U) \in \mathcal{S},\ (P,Q,U) \in \mathcal{C}\ \} \quad (1)$$

where (P, Q, U) are vectors of optimization variables grouped in like cathegories:

n_t:	Length of the planning horizon
n_g:	Number of generators to schedule
$p^{i,t}$:	Real power output for generator i at time t
$q^{i,t}$:	Reactive power output for generator i at time t
$u^{i,t}$:	On/off status (one or zero) for generator i at time t
P:	$(p^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$
Q:	$(q^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$
U:	$(u^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$
$F(P,U)$:	The total production cost
$K(U)$:	The sum of any startup costs
$\mathcal{D}$:	A set of *dynamic* generator-wise constraints
$\mathcal{S}$:	A set of *static* instantaneous constraints
$\mathcal{C}$:	A set of *nonseparable* constraints

The production cost F is assumed to be a convex (in fact quadratic) and separable over each generator and time period, so that we can write:

$$F(P,U) = \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} u^{i,t} F^i(p^{i,t}). \qquad (2)$$

The constraints in the problem are classified into three kinds: Cathegory $\mathcal{D}$ groups constraints that relate to a single generator, but may span several time periods (inter–temporal, or *dynamic*, constraints). Typical constraints in this cathegory are minimum up or down times and ramping constraints. Cathegory $\mathcal{S}$ groups constraints that span the complete system but involve only one time period at a time (also called system–wide, or *static* constraints). This cathegory includes such constraints as as load/demand matching, voltage limits, branch flow limits, reserve constraints and generator operating range limits. Finally, cathegory $\mathcal{C}$ groups constraints that involve more than one generator and more than one time period. A typical example is the infeasibility of turning on more than one unit at a time in a given location because of crew constraints. Typically, it is impossible to separate the generators bound by such constraints, resulting in increased complexity of the related dynamic program.

1.2 The motivation for employing an AC power flow model

Since the introduction of the Lagrangian relaxation technique for the solution of the unit commitment problem a little over 20 years ago [1, 3], researchers have improved the basic formulation by adding more ellaborate intertemporal constraints and costs [5, 6, 7], better dealing with primal–feasibility issues [9, 12, 17], incorporation of linear network constraints [11, 13, 15, 16, 18], ramp constraints [20] and several other issues. However, when it comes to expanding the original formulation of the problem, researchers have included new constraints either by lumping them into the separated dynamic programs for each generator (as in the case of adding ramp constraints by discretizing the generation range and imposing transition constraints in the dynamic program) or, if the constraints are linear, by relaxing them with appropriate multipliers. Such constraints do not hamper the separation structure of the dual functional if relaxed. It is, in fact, possible to add any linear constraint that one can think of to the Lagrangian, appropriately relaxed with a multipler, and expect to be able to preserve separation structure. Nonlinear constraints, on the other hand, are useless as candidates for relaxation because they will likely end up coupling subproblems in the dual functional. In particular, AC power flow constraints, being highly nonlinear and containing sines and cosines of angle differences and voltage cross products in each term, are not candidates for relaxation. Thus, researchers refrained from including constraints that depend on voltages or reactive flows, or, if they did something about them, they used linearizations. There are, however, genuine engineering considerations for including the full nonlinear model. These considerations stem from the fact that some very important system constraints can only be modeled accurately with the nonlinear AC flow network model. For example, branch flow limits are best expressed in terms of an MVA rating for transformers, or in terms of a maximum current for a transmission line. However, both actual MVA and current depend on its orthogonal active and reactive components. The DC flow model can predict (and even then, only for small angle deviations) the active component, and can thus be a poor model of what happens in the real system. Line limits inextricably tie together active and reactive dispatch restrictions. Another example is given by voltage limits; predicting them linearly in terms of reactive injections is not accurate enough. Finally, there are components in the network that the DC flow is not even capable of modeling, such as tapping transformers.

At this point, we asked ourselves: what good is solving the wrong (i.e., linearized) problem to within very small duality gaps, if the so–called primal–feasible solution that comes out of it will in fact require further processing and rescheduling in order to meet actual network constraints?

All of these considerations led us to consider an alternative formulation that could deal with nonlinear network constraints, as reported in [26], and repeated in the following section for clarity and for purposes of self-containment.

2 Unit commitment with AC OPF formulation

Our approach has its origins in the *variable duplication* technique credited to Guy Cohen in [13] by Batut and Renaud. This same technique was used later by Baldick [15] in his more general formulation of the unit commitment problem. The key idea is simple; it is based on the fact that the problem

$$\min \{f_1(x) + f_2(x) \mid s(x) \in \mathcal{S},\ \ d(x) \in \mathcal{D}\}$$

is equivalent to

$$\min \{f_1(x) + f_2(y) \mid s(y) \in \mathcal{S},\ \ d(x) \in \mathcal{D},\ \ y - x = 0\}$$

We start by defining two sets of variables, the *dynamic variables* and the *static* ones:

Dynamic:

$u^{i,t}$: Commitment status $\{0,1\}$ for generator i at time t

$d_p^{i,t}$: Real power output for generator i at time t

$d_q^{i,t}$: VAr output for generator i at time t

U: $(u^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$

D_p: $(d_p^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$

D_q: $(d_q^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$

D: (D_p, D_q)

Static:

$s_p^{i,t}$: Real power output for generator i at time t

$s_q^{i,t}$: VAr output for generator i at time t

S_p: $(s_p^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$

S_q: $(s_q^{i,t})$, $i = 1 \ldots n_g$, $t = 1 \ldots n_t$

S: (S_p, S_q)

Then the following optimization problem is defined

$$\min_{D,U,S} \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} [u^{i,t} F^i(d_p^{i,t}) + K^{i,t}(u^{i,\cdot})] \qquad (3)$$

subject to:
(1) $\mathcal{D}$–type constraints

$$u^{i,t} P^i_{min} \le u^{i,t} d_p^{i,t} \le u^{i,t} P^i_{max}, \qquad (4)$$

$$u^{i,t} Q^i_{min} \le u^{i,t} d_q^{i,t} \le u^{i,t} Q^i_{max}, \qquad (5)$$

$$U \text{ satisfies minimal up and down times,} \qquad (6)$$

(2) $\mathcal{S}$–type constraints

$$0 \le s_p^{i,t} \le P^i_{max}, \qquad (7)$$

$$Q^i_{min} \le s_q^{i,t} \le Q^i_{max}, \qquad (8)$$

$$(S_p, S_q) \begin{cases} \text{satisfies the network load flow} \\ \text{equations while respecting line} \\ \text{MVA limits and voltage limits} \end{cases} \qquad (9)$$

(3) and the following additional constraints

$$R^{l,t} - \sum_{i \in Z_l} u^{i,t} P^i_{max} \le 0,\ \ l = 1 \ldots n_z,\ \ t = 1 \ldots n_t \qquad (10)$$

$$s_p^{i,t} - u^{i,t} d_p^{i,t} = 0,\ \ i = 1 \ldots n_g,\ \ t = 1 \ldots n_t \qquad (11)$$

$$s_q^{i,t} - u^{i,t} d_q^{i,t} = 0,\ \ i = 1 \ldots n_g,\ \ t = 1 \ldots n_t \qquad (12)$$

where $R^{l,t}$ is the minimum combined capacity that is acceptable for the lth zone in the tth period and Z_l is the set of indices of generators in the lth zone.

We will assume that we can enforce the $\mathcal{D}$ constraints (4–6) on the $\mathcal{D}$ variables and the $\mathcal{S}$ constraints (7–9) on the $\mathcal{S}$ variables, so that we only relax the three last constraints (10–12), which leads to the following Lagrangian:

$$\begin{aligned} \mathcal{L}(U, D, \lambda, \beta) \ = \ & \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} [u^{i,t} F^i(d_p^{i,t}) + K^{i,t}(u^{i,\cdot})] \\ & + \sum_{t=1}^{n_t} \sum_{l=1}^{n_z} \beta^{l,t} (R^{l,t} - \sum_{i \in Z_l} u^{i,t} P^i_{max}) \\ & + \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \lambda_p^{i,t} (s_p^{i,t} - u^{i,t} d_p^{i,t}) \\ & + \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \lambda_q^{i,t} (s_q^{i,t} - u^{i,t} d_q^{i,t}) \qquad (13) \\ = \ & \sum_{i=1}^{n_g} \sum_{t=1}^{n_t} \left\{ u^{i,t} F^i(d_p^{i,t}) + K^{i,t}(u^{i,\cdot}) - \lambda_p^{i,t} u^{i,t} d_p^{i,t} \right. \end{aligned}$$

$$
\begin{aligned}
&\quad - \beta^{z(i),t} u^{i,t} P^i_{max} - \lambda^{i,t}_q u^{i,t} d^{i,t}_q \Big\} \\
&+ \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} (\lambda^{i,t}_p s^{i,t}_p + \lambda^{i,t}_q s^{i,t}_q) \\
&+ \sum_{t=1}^{n_t} \sum_{l=1}^{n_z} \beta^{l,t} R^{l,t} \qquad (14) \\
&= \mathcal{L}_1(U, D, \lambda, \beta) + \mathcal{L}_2(S, \lambda) + \mathcal{L}_3(\beta) \qquad (15)
\end{aligned}
$$

where $\lambda = (\lambda^{i,t}_p, \lambda^{i,t}_q)$ are multipliers on the relaxed equalities of the two kinds of variables, $\beta^{l,t}$ is the multiplier associated to the lth zone's reserve requirement at the tth period, and $z(i)$ returns the index of the zone to which generator i belongs.

The separation structure of the Lagrangian is obvious upon looking at equations (14) and (15). It makes it possible to write the dual objective as

$$
\begin{aligned}
q(\lambda, \beta) &= \min_{U,D,S} \{\mathcal{L}_1(U, D, \lambda, \beta) + \mathcal{L}_2(S, \lambda) + \mathcal{L}_3(\beta)\} \\
&= \min_{U,D} \mathcal{L}_1(U, D, \lambda, \beta) \\
&\quad + \min_{S} \mathcal{L}_2(S, \lambda) \\
&\quad + \mathcal{L}_3(\beta) \qquad (16)
\end{aligned}
$$

By looking again at (14) and (16), it can be seen that the first term can be computed by solving n_g dynamic programs again; the second term separates into n_t optimal power flow problems with all generators commited but with special cost curves $\lambda^{i,t}_p s^{i,t}_p + \lambda^{i,t}_q s^{i,t}_q$ for generator i at time t. Notice that $s^{i,t}_q$ also has a price. It is assumed that the solutions of the dynamic programs meet the $\mathcal{D}$ constraints and that the solutions of the optimal power flows meet the $\mathcal{S}$ constraints.

It would be tempting to apply a dual maximization procedure to the dual objective as stated, but there are some issues that prevent us from doing that without some modification of the Lagrangian. The first issue is that the cost of $d^{i,t}_q$ reflected in the dynamic programs, being linear, is not strongly convex; this can cause unwanted oscillations in the $d^{i,t}_q$ prescribed by the dynamic program (see [13]). Therefore we set out to fix this before addressing any other problems by augmenting the Lagrangian with quadratic functions of the equality constraints. This will introduce nonseparable terms, but using the Auxiliary Problem Principle described by G. Cohen in [4] and [8] we can linearize those terms about the previous iteration values, rendering them separable. Thus we write the new augmented Lagrangian as

$$
\begin{aligned}
&\mathcal{L}(U, D, S, \lambda, \beta) = \\
&\quad \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} [u^{i,t} F^i(d^{i,t}_p) + K^{i,t}(u^{i,\cdot})] \\
&\quad + \sum_{t=1}^{n_t} \sum_{l=1}^{n_z} \beta^{l,t} (R^{l,t} - \sum_{i \in Z_l} u^{i,t} P^i_{max}) \\
&\quad + \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \lambda^{i,t}_p (s^{i,t}_p - u^{i,t} d^{i,t}_p) \\
&\quad + \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \lambda^{i,t}_q (s^{i,t}_q - u^{i,t} d^{i,t}_q) \\
&\quad + \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \frac{c_p}{2} (s^{i,t}_p - u^{i,t} d^{i,t}_p)^2 \\
&\quad + \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \frac{c_q}{2} (s^{i,t}_q - u^{i,t} d^{i,t}_q)^2 \qquad (17)
\end{aligned}
$$

The Auxiliary Problem Principle allows us to substitute the augmentation terms by the following at iteration k (see [13])

$$
\begin{aligned}
&\sum_{t=1}^{n_t} \sum_{i=1}^{n_g} c_p (\bar{s}^{i,t}_p - \bar{u}^{i,t}_p \bar{d}^{i,t}_p)(s^{i,t}_p - u^{i,t} d^{i,t}_p) \\
&+ \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \frac{b_p}{2} \left\{ (s^{i,t}_p - \bar{s}^{i,t}_p)^2 + (u^{i,t} d^{i,t}_p - \bar{u}^{i,t} \bar{d}^{i,t}_p)^2 \right\} \\
&+ \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} c_q (\bar{s}^{i,t}_q - \bar{u}^{i,t}_q \bar{d}^{i,t}_q)(s^{i,t}_q - u^{i,t} d^{i,t}_q) \\
&+ \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \frac{b_q}{2} \left\{ (s^{i,t}_q - \bar{s}^{i,t}_q)^2 + (u^{i,t} d^{i,t}_q - \bar{u}^{i,t} \bar{d}^{i,t}_q)^2 \right\} \qquad (18)
\end{aligned}
$$

where $\bar{u}^{i,t}$, $\bar{d}^{i,t}_p$, $\bar{d}^{i,t}_q$, $\bar{s}^{i,t}_p$ and $\bar{s}^{i,t}_q$ are the values obtained at the $(k-1)$th iteration. Since (18) is separable, we can collect terms of the augmented Lagrangian on a per-generator basis, so that at the kth iteration we are faced with

$$
\begin{aligned}
&\mathcal{L}(U, D, S, \lambda, \beta, \bar{U}, \bar{D}, \bar{S}) = \\
&\quad \sum_{i=1}^{n_g} \sum_{t=1}^{n_t} \Big\{ u^{i,t} F^i(d^{i,t}_p) + K^{i,t}(u^{i,\cdot}) \\
&\quad + \frac{b_p}{2} u^{i,t} (d^{i,t}_p)^2 + \frac{b_q}{2} u^{i,t} (d^{i,t}_q)^2 \\
&\quad + \left[-\lambda^{i,t}_p - c_p (\bar{s}^{i,t}_p - \bar{u}^{i,t} \bar{d}^{i,t}_p) - b_p \bar{u}^{i,t} \bar{d}^{i,t}_p \right] u^{i,t} d^{i,t}_p \\
&\quad + \left[-\lambda^{i,t}_q - c_q (\bar{s}^{i,t}_q - \bar{u}^{i,t} \bar{d}^{i,t}_q) - b_q \bar{u}^{i,t} \bar{d}^{i,t}_q \right] u^{i,t} d^{i,t}_q \\
&\quad + \left[-\beta^{z(i),t} P^i_{max} \right] u^{i,t} \Big\} \\
&+ \sum_{t=1}^{n_t} \sum_{i=1}^{n_g} \Big\{ \frac{b_p}{2} (s^{i,t}_p)^2 + \frac{b_q}{2} (s^{i,t}_q)^2 \\
&\quad + \left[\lambda^{i,t}_p + c_p (\bar{s}^{i,t}_p - \bar{u}^{i,t} \bar{d}^{i,t}_p) - b_p \bar{s}^{i,t}_p \right] s^{i,t}_p \\
&\quad + \left[\lambda^{i,t}_q + c_q (\bar{s}^{i,t}_q - \bar{u}^{i,t} \bar{d}^{i,t}_q) - b_q \bar{s}^{i,t}_q \right] s^{i,t}_q \Big\}
\end{aligned}
$$

$$+\sum_{t=1}^{n_t}\sum_{i=1}^{n_g}\left\{\frac{b_p}{2}\left[(\bar{s}_p^{i,t})^2+(\bar{u}^{i,t}\bar{d}_p^{i,t})^2\right]\right.$$
$$\left.+\frac{b_q}{2}\left[(\bar{s}_q^{i,t})^2+(\bar{u}^{i,t}\bar{d}_q^{i,t})^2\right]\right\}$$
$$+\sum_{t=1}^{n_t}\sum_{l=1}^{n_z}\beta^{l,t}R^{l,t} \quad (19)$$
$$=\mathcal{L}_1(U,D,\lambda,\beta,\bar{U},\bar{D},\bar{S})+\mathcal{L}_2(S,\lambda,\bar{U},\bar{D},\bar{S})$$
$$+\mathcal{L}_3(\beta) \quad (20)$$

Notice that (20) has the same separation structure of (16).

Now that the separability issue has been resolved, it is easy to see that a dual maximization algorithm might proceed as follows:

Algorithm: *AC Augmented Lagrangian relaxation*

Step 0 $k \longleftarrow 0$

Step 1 Initialize $(\lambda_p^{i,t}, \lambda_q^{i,t})$ to the values of the multipliers on the power flow equality constraints at generator buses when running an OPF with all units commited. Initialize $(\bar{U}, \bar{D}, \bar{S})$ to zeros.

Step 2a Compute

$$(\hat{U},\hat{D}) \leftarrow \arg \min_{\text{feasible } U,D} \mathcal{L}_1(U,D,\lambda,\beta,\bar{U},\bar{D},\bar{S})$$

by solving n_g one-generator dynamic programs.

Step 2b Compute

$$\hat{S} \longleftarrow \arg \min_{\text{feasible } S} \mathcal{L}_2(S,\lambda,\bar{U},\bar{D},\bar{S})$$

by solving n_t OPF's in which all generators are committed, their generation range has been expanded to include $P_{min}^i = 0$ and the special cost $\mathcal{L}_2(S,\lambda,\bar{U},\bar{D},\bar{S})$ is used. Note: all tasks in steps 2a and 2b can be solved in parallel.

Step 3 If the commitment schedule $\hat{U}$ is not in a database of tested commitments, perform a cheap primal feasibility test. If the results are not encouraging, store the schedule in the database and label it "infeasible", then go to Step 6.

Step 4 Perform a more serious primal feasibility test by actually attempting to run n_t OPF's with the original P_{min} constraints. If all OPF's are successful, store the commitment in the database, together with the primal cost including startup costs, and the duality gap (the dual cost was available upon solving 2a and 2b). Else label the commitment as "infeasible", store it in the database, and go to Step 6.

Step 5 If the mismatch between the two sets of variables is small enough, stop.

Step 6 Update all multipliers using subgradient techniques, and

$$\bar{U} \longleftarrow \hat{U}$$
$$\bar{D} \longleftarrow \hat{D}$$
$$\bar{S} \longleftarrow \hat{S}$$
$$k \longleftarrow k+1$$

Step 7 Go to Step 2.

The proposed algorithm is very OPF–intensive: the major computational cost is that of computing n_t OPF's for every iteration in order to solve the static subproblems, plus extra OPF's in selected iterations when a given commitment is promising. Thus, every effort possible must be made to try to alleviate the burden of OPF computation. The first thing that can be done is to use as a starting point for the OPF the result of the previous iteration for the same time period. Most of the times, the only difference in the data for the OPF would be a small change in the costs (reflected by the change in λ from one iteration to another). This, in theory, should result in fewer iterations needed for the OPF.

Another drawback of the algorithm is that a different set of OPF computations must be performed to compute the value of the dual objective and to compute the value of the primal. Thus, before even trying to compute the value of the primal objective, one should make sure that such a costly computation is worth doing. Some of the cheap tests include verifying that the reserve constraint is met and that the mismatch between the S and the D variables is small. With respect to the latter, we have found that if $u^{1,t} = 1$, a smaller mismatch should be asked for as requisite to feasibility than if $u^{i,t} = 0$. More costly feasibility tests would involve power flow problems starting from appropriate initial values. Currently a constrained power flow is being performed at this stage.

3 Computational results

An implementation of the algorithm has been written in the $MATLAB^{TM}$ environment. The dynamic sub-

problems can accomodate minimal up or down times, warm start and cold startup costs. The static subproblems are solved using a version of *MINOS* [22] that has been incorporated into the *MATPOWER* package (see [23]). It includes box constraints on the generator's active and reactive output, piecewise-linear convex or polynomial cost functions for both P and Q, voltage constraints, line MVA limits and of course, the power flow equations. Additionally, any linear constraint on the optimization variables can be specified via a user-friendly mechanism. A preconditioner for *MINOS* that performs a constrained power flow is used if necessary. It implements a Levenberg–Marquardt–like minimization of the sum of squares of the power flow constraints, with penalty functions on some other box constraints and constrained variables in the case of voltage limits. Thus, each iteration involves solving a QP subproblem rather than a Newton step. The QP subproblem is solved using *BPMPD* (see [19]), an interior method QP solver.

The program was first tested on a modified IEEE 30-bus system [2] with 6 generators and a planning horizon of length 6. For comparison purposes, a version of the Lagrangian relaxation algorithm with DC Flow-based relaxed line limits was also written. It turns out that generator number 4, located at bus number 27, is needed for voltage support for many load levels even though it is most uneconomical to operate. The AC-based algorithm correctly identified this unit as a must-run for those time periods, even providing some price information on the MVArs that this unit produced by means of the corresponding $\lambda_q^{i,t}$. The number of iterations required was usually in the vicinity of one hundred. In contrast, the DC flow-based algorithm failed to commit unit 4 for any period, producing a commitment schedule that was infeasible in light of the AC power flow constraints.

The importance of proper selection of the (c_p, b_p, c_q, b_q) parameters was apparent from the beginning. We obtained good results with $c_p = 0.05$, $b_p = 4c_p$, $c_q = 0.08$ and $b_q = 4c_q$. However, other choices tended to produce somewhat smooth, damped oscillations in the values of some of the $(\lambda_p^{i,t}, \lambda_q^{i,t})$.

To highlight one of the new features found in the algorithm, we show the evolution of $(\lambda_p^{i,t}, \lambda_q^{i,t})$ vs. iteration number for a typical run in figure 1. The multipliers with the higher values are all P–type multipliers. Those with the smaller values correspond to the $\lambda_q^{i,t}$. Most of them settle to zero, indicating that Q is essentially free almost always. However, a few of them actually have high prices: these belong to generators and time periods where the OPF tries to use their MVArs in order to force feasibility or guided by

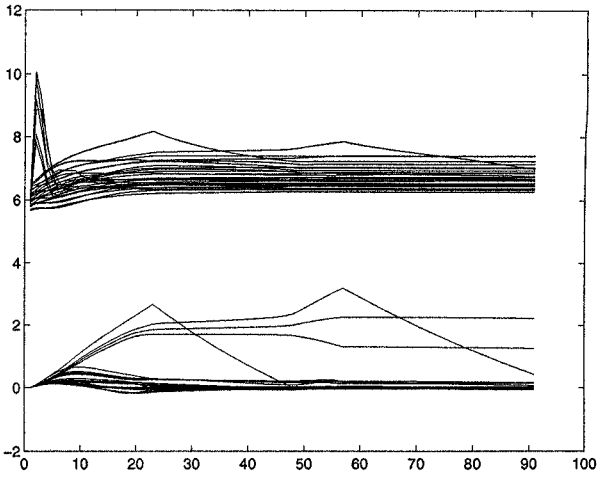

Figure 1: Evolution of multipliers in a typical run.

economic considerations, but the generators are not actually committed. In the course of the algorithm, these $\lambda_q^{i,t}$ may grow so large that they trigger the respective unit on. Once this happens, such multipliers tend to approach zero again, since Q is now plentiful. In figure 1 there are two clear examples of this behavior, corresponding to unit 4 being commited for certain time periods. As the multiplier approaches zero, the static copy $s_q^{i,t}$ will approach the dynamic $d_q^{i,t}$.

A slightly more ambitious test has been performed using the IEEE 118 bus system, with 54 generators and a time horizon of 24 slices, corresponding to two "weekdays" and one day of the weekend, each with 8 three–hour periods. The total variation of the load relative to the base case is -50% and $+40\%$. The behavior of the ℓ^1 norm of the active and reactive mismatches between the two sets of variables can be seen in Figure 2. The evolution of the multipliers in this case can be seen in Figure 3.

4 Parallel implementation

Profiling the algorithm indicates that up to 95% of the computation cost is OPF–related, and hence this section is the one that could benefit the most from parallelization. Although dynamic programs are also parallelizable, their granularity is much smaller and communications overhead is likely to reduce the efficiency of the parallelization. So the authors have opted for an initial version that parallelizes the OPF subproblems only.

The initial algorithm had been developed in

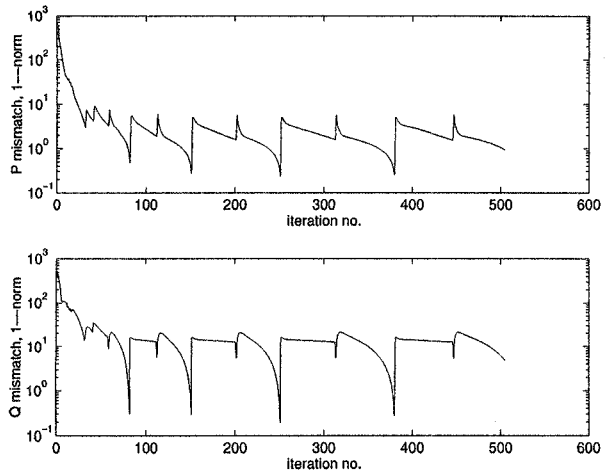

Figure 2: Evolution of power mismatch in the 118 bus system.

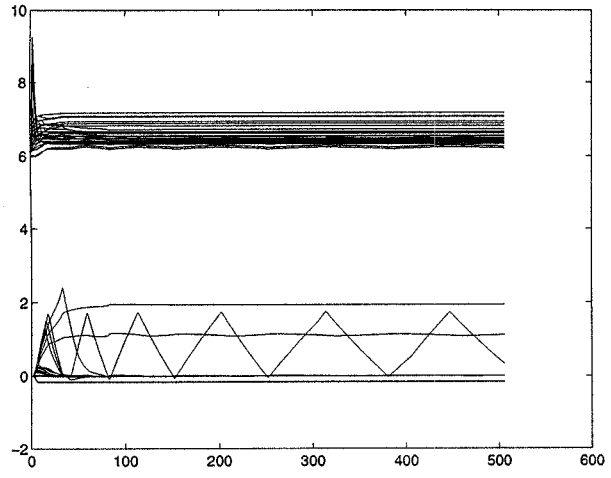

Figure 3: Evolution of multipliers in the 118 bus system.

MATLAB, and fortunately, we can continue to use the same environment for testing the parallel version due to the existence of *MultiMATLAB*, currently being developed at Cornell University's Theory Center by John Zollweg; see [25] and earlier work in [24]. *MultiMATLAB* is implemented by means of having several copies of *MATLAB* each running in a different node and communicating via a subset of the Message Passing Interface (MPI) library. The calls to MPI are implemented as MEX files. A master processor, where the main algorithm and dynamic programming subproblems run, switches to a master/worker parallelization strategy whenever it needs to solve a set of OPF's. The workers' only task is to receive OPF input data, run the OPF solver and report back to the master the results of the computation.

So far, we have programmed two scheduling strategies to perform the parallel task. The simpler of the two is a straightforward round-robin strategy in which the master cycles through the workers, receiving the results of any prior assigned work by means of a blocking receive (MPI's *Recv* function), which ties the master until data arrives. When data does arrive, the master then sends the worker data for the next OPF, and turns its attention to the next worker. While having the master wait for the worker to be ready is not optimal, the only kind of message passing being done employs MPI's *Recv* and *Send* calls, whose implementation is more stable in *MultiMATLAB*.

A second, more sophisticated implementation makes use of MPI's *Irecv* non–blocking receive function. Right after sending data to a worker, the master processor posts a non–blocking receive, setting aside a data reception area for the incoming message from the worker. The master can then turn its attention to other workers and assign further work or read from them. From time to time, the master executes a call to MPI's *Testany* function, which informs the master of any pending *Irecv*'s being completed. If so, the master processes the data from the corresponding worker and assigns further work if needed. This strategy promises the most efficient use of the workers. In limited experiments with up to 7 workers, the master is actually free most of the time, which means that it is very efficiently keeping the workers occupied. The efficacy of this algorithm can be further enhanced by performing the OPF's in order of decreasing expected complexity, so that the end game is less likely to become a situation in which the master spends its time waiting for only a small subset of the workers occupied in solving long, difficult OPF's.

At the time this paper is being written, several pending issues in the implementation of the calls to the MPI

library have not been resolved, and as a result we do not have the results of a complete run available yet. However, it is only a matter of weeks before there is a stable *MultiMATLAB* implementation to complete our tests.

5 Future work

There are three main areas where efforts are under way. The first of them relates to the joint work with John Zollweg to make *MultiMATLAB* a more reliable on the NT cluster. The second general area involves improvement of the OPF's robustness and handling of very large systems. While *MINOS* has been found to do a good job of finding optima given a good starting point, the early stages of the algorithm, when prices are moving rapidly from iteration to iteration, result in OPF problems where the optimal solution lies far away from the initial point. We have found *MINOS* not to behave as well in these cases, especially for larger systems (i.e., 3000 buses), even with the constrained power flow preconditioner. Work is being done on making the preconditioner even smarter, so that it can predict binding constraints at the optimum.

The last area imvolves enhancing the software by adding ramp constraints in the formulation. There are two basic approaches poposed in the literature; the first one involves discretizing the generation range for the dynamic copy of the active power sources, and disallowing transitions that violate ramping restrictions in the dynamic programs. While relatively straight forward, the dynamic programs do become more complicated. The second approach is to relax the linear ramping constraints and add them to the Lagrangian. There is concern about the speed of convergence of the corresponding multipliers, however. It may be possible to have special updating strategies for these multipliers.

6 Acknowledgements

We wish to thank Ray Zimmerman and Deqiang Gan for providing us with their package *MATPOWER* [23] for the initial tests of the algorithm. We would also like to thank Csaba Mészáros, whose QP program [19] *BPMPD* we use in several of our programs, and finally, John Zollweg, principal developer of *MultiMATLAB*, with whom we have spent countless hours testing and debugging.

References

[1] Muckstadt, J.A. & Wilson, R.C., "An Application of Mixed-Integer Programming Duality to Scheduling Thermal Generating Systems", *IEEE Transactions on Power Apparatus and Systems*, Vol. PAS-87, No. 12, December 1968, pp. 1968–1978.

[2] Alsac, O. & Stott, B., "Optimal Load Flow with Steady-State Security", *IEEE Transactions on Power Apparatus and Systems*, Vol. PAS 93, No. 3, 1974, pp. 745–751.

[3] Muckstadt, J.A. & Koenig, S.A., "An Application of Lagrange Relaxation to Scheduling in Power–Generation Systems", *Operations Research*, Vol. 25, No. 3, May–June 1977, pp. 387–403.

[4] Cohen, G., "Auxiliary Problem Principle and Decomposition of Optimization Problems", *Journal of Optimization Theory and Applications*, Vol. 32, No. 3, November 1980, pp. 277–305.

[5] Lauer, G.S., Bertsekas, D.P., Sandell, N.R. & Posbergh, T.A., "Solution of Large–Scale Optimal Unit Commitment Problems", *IEEE Transactions on Power Apparatus and Systems*, Vol. PAS-101, No. 1, January 1982, pp. 79–85.

[6] Bertsekas, D.P., Lauer, G.S., Sandell, N.R. & Posbergh, T.A., "Optimal Short-Term Scheduling of Large-Scale Power Systems", *IEEE Transactions on Automatic Control*, Vol. AC-28, No. 1, January 1983, pp. 1–11.

[7] Merlin, A. & Sandrin, P., "A New Method for Unit Commitment at Electricité de France", *IEEE Transactions on Power Apparatus and Systems*, Vol. PAS-102, No. 5, May 1983, pp. 1218–1225.

[8] Cohen, G. & Zhu, D.L., "Decomposition Coordination Methods in Large Scale Optimization Problems: The Nondifferentiable Case and the Use of Augmented Lagrangians", in *Advances in Large Scale Systems, Vol. 1; J.B. Cruz, Ed.*, JAI Press Inc, 1984, pp. 203–266.

[9] Zhuang, F. & Galiana, F.D., "Towards a More Rigorous and Practical Unit Commitment by Lagrangian Relaxation", *IEEE Transactions on Power Systems*, Vol. 3, No. 2, May 1988, pp. 763–773.

[10] Alsac, O., Bright, J., Prais, M. & Stott, B., "Further Development in LP–Based Optimal Power Flow", *IEEE Transactions on Power Systems*, Vol. 5, No. 3, August 1990, pp. 697–711.

[11] Ružić, S. & Rajaković, N., "A New Approach for Solving Extended Unit Commitment Problem", *IEEE Transactions on Power Systems*, Vol. 6, No. 1, February 1991, pp. 269–277.

[12] Guan, X., Luh, P.B. & Yan, H., "An Optimization–Based method for Unit Commitment", *Electrical Power & Energy Systems*, Vol. 14, No. 1, February 1992.

[13] Batut, J. & Renaud, A., "Daily Generation Scheduling Optimization with Transmission Constraints: A New Class of Algorithms", IEEE Transactions on Power Systems, Vol. 7, No. 3, August 1992, pp. 982–989.

[14] Sheble, G.B. & Fahd, G.N., "Unit Commitment Literature Synopsis", *IEEE Transactions on Power Systems*, Vol. 9, No. 1, February 1994, pp. 128–135.

[15] Baldick, R., "The Generalized Unit Commitment Problem", *IEEE Transactions on Power Systems*, Vol. 10, No. 1, February 1995, pp. 465–475.

[16] Wang, S.J., Shahidehpour, S.M., Kirschen, D.S., Mokhtari, S. & Irisarri, G.D., "Short-Term Generation Scheduling with Transmission and Environmental Constraints Using an Augmented Lagragian Relaxation", *IEEE Transactions on Power Systems*, Vol. 10, No. 3, August 1995, pp. 1294–1301.

[17] Shaw, J.J., "A Direct Method for Security–Constrained Unit Commitment", *IEEE Transactions on Power Systems*, Vol. 10, No. 3, August 1995, pp. 1329–1339.

[18] Abdul-Rahman, K.H., Shahidehpour, S.M., Aganagic, M. & Mokhtari, S., "A Practical Resource Scheduling with OPF Constraints", *IEEE Transactions on Power Systems*, Vol. 11, No. 1, February 1996, pp. 254–259.

[19] Mészáros, Cs., " The efficient implementation of interior point methods for linear programming and their applications", Ph.D. Thesis, Eötvös Loránd University of Sciences, 1996.

[20] Svoboda, A.J., Tseng, C.L., Li, C.A. & Johnson, R.B., "Short-Term Resource Scheduling with Ramp Constraints", *IEEE Transactions on Power Systems*, Vol. 12, No. 1, February 1997, pp. 77–83.

[21] Li, C.A., Johnson, R.A. & Svoboda, A.J., "A New Unit Commitment Method", *IEEE Transactions on Power Systems*, Vol. 12, No. 1, February 1997, pp. 113–119.

[22] Murtagh, B. A. & Saunders, M.A., "MINOS 5.5 User's Guide", Stanford University Systems Optimization Laboratory Technical Report SOL 83-20R.

[23] Zimmerman, R. & Gan, D., "*MATPOWER:* A Matlab Power System Simulation Package", *http://www.pserc.cornell.edu/matpower/* .

[24] Trefethen, A.E., Menon, V.S., Chang, C.C., Czajkowski, G.J, Myers, C. & Trefethen, L.N., "MultiMatlab: MATLAB on Multiple Processors", Cornell University Computer Science Technical Report 96TR239.

[25] Zollweg, J, "MultiMATLAB for NT Clusters", Cornell Theory Center Software Documentation, *http://www.tc.cornell.edu/UserDoc/Intel/Software/multimatlab/.*

[26] Murillo–Sánchez, C.E. & Thomas, R.J., "Thermal Unit Commitment Including Optimal AC Power Flow Constraints", *Proceedings of the 31st. HICSS Conference, Kona, Hawaii, Jan. 6–9 1998.*