

Electromechanical Transients Simulation on a Multicomputer Via the VDHN–Maclaurin Method

Felipe Morales, Hugh Rudnick, and Aldo Cipriano

Abstract—This paper reports simulations of power systems electromechanical transients on a multicomputer, formulated as a nonlinear algebraic problem by using the time parallelization concept. The bi-factorized inversion, which is the most time consuming stage of the simulation, is solved by the “Very Dishonest Newton(VDHN)–Maclaurin” method, a fully parallel indirect method based on the decomposition of the nonupdated Jacobian matrix. This proposal is made to orient the search for the decomposition based on a sufficient condition for the convergence of the Maclaurin series, which is a desirable situation for the design of more robust algorithms for power system simulation. Such condition keeps a close relation with a physical coupling property exhibited by power systems, and the characteristics of the simulation method. Theoretical and numerical results show that a successful implementation of this method can be better reached when the Jacobian matrix is decomposed as a block diagonal matrix plus a matrix with off diagonal blocks elements, the latter representing weak couplings between the diagonal blocks. The ϵ Decomposition is used to satisfy the sufficient condition for convergence and the Longest Path Scheduling Method to prevent the uneven loading of processors, permitting to adapt the method in a efficient way on a coarse grain computer. The parallel simulation was written in C language and implemented on a Parsytec PowerXplorer multicomputer. Test using electromechanical models of the Chilean Central Interconnected system and the IEEE300 test system were made to evaluate the advantages and drawbacks of the parallel method.

I. INTRODUCTION

LARGE scale power system analysis is computationally intensive. In fact, the power industry and the academic community are requiring developments for high performance computing tools, such as parallel computers, efficient compilers, graphic interfaces and artificial intelligence-based algorithms to be able to obtain meaningful results on such systems [1]. Power system dynamic simulation is one of the problems needing a special treatment to reduce time and memory requirements [2]. It is required in the design, planning, operation and control stages of power systems.

Parallel processing is a promising possibility to obtain real time simulations and perform the real time dynamic security assessment on a more flexible and friendly environment. From a general point of view, the latter task is naturally parallelizable and it reaches high efficiency levels on coarse grain architectures without many adapting efforts [3]. On the contrary, the parallelization of a simulation as such is not trivial and it requires the design of parallel algorithms.

Spatial parallelization and time parallelization are two concepts often used to simulate power systems electromechanical transients on parallel computers. The former works by decoupling the power system in smaller subsystems based on connectivity properties between them, whereas the later, in a general sense, arises when multiple integration steps are simultaneously solved. Many parallel simulation methods based on spatial parallelization, time parallelization, or any mix of both have been proposed. Several of them have already been implemented on some particular architecture [1].

The specific way in which methods achieve parallelism has a strong relationship with the selected solver for the differential-algebraic initial value problem of the simulation. More common approaches for spatial and time parallelization work with two well known nonlinear algebraic problems that describe both the network and the machine discretized differential equations. Following the alternating implicit scheme, where the spatial parallelization is perfectly done around the noncoupled machine equations, Newton-like solvers as the VDHN method are commonly used. Main drawbacks are encountered in solving linear network equations. Mixed LU factorization and Conjugate Gradient methods [4], W matrix and approximate Jacobians from successive over relaxation, and Maclaurin methods are proposed to overcome this difficulty [5]. All of these methods permit to exploit the time parallelization concept by solving one integration step per processor. This kind of implementation in which Newton’s methods are relaxed along the integration steps leads to similar implementations of the Gauss–Jacobi–Newton or Gauss–Seidel–Newton algorithms reported in [6].

When the simultaneous implicit scheme is adopted it is possible to formulate an unique nonlinear algebraic problem where both the spatial and time parallelization concepts are exploited. As far as the solution based on the Newton–Raphson method is concerned, the linear algebraic problem that characterizes its iteration is doubtless the most time consuming stage of the simulation and therefore the main target for parallel processing. The Conjugate Gradient method, whose inner products and matrix-vector multiplications are very amenable for parallel and vector machines, was implemented in [4].

Two less conventional approaches are the waveform relaxation method [7], [8], whose implementation on parallel computers is reached by assigning differential and algebraic subproblems formulated according to the Gauss Jacobi method, and the shifted-Picard method [9]. The later exploits parallelism in matrix structures and matrix operations arise from the matrix exponential method, which was suggested to solve the linear differential-algebraic problem that updates state and algebraic trajectories of the nonlinear one.

Manuscript received April 6, 1999; revised February 6, 2001.

The authors are with the Department of Electrical Engineering, Universidad Católica de Chile, Casilla 306, Correo 22, Santiago, Chile.

Publisher Item Identifier S 0885-8950(01)06064-3.

Frequency domain techniques work like a time parallel method. They iterate through state and algebraic trajectories expanded in Fourier series, which being treated in the frequency domain, together with transfer functions of generating units and nonlinearities approximated by polynomials, lead to many vector and matrix operations efficiently solved on vector machines [10].

Methods based only on the spatial parallelization concept were mainly used in the first implementations of power system parallel simulations, but at the present those methods which are a mixture with the time parallelization concept are certainly more effective. Nevertheless, because of the simplicity of some approaches for spatial parallelization their implementations are still attractive, such as the approach suggested for the parallelization of university research level and production level programs reported in [11].

As mentioned above, a frequently used approach for power system parallel simulation is to decompose the transmission system into weakly coupled subsystems, or to group the generating units in an appropriate way. Such algorithms are attractive because they take advantage of some natural characteristics of power systems. Decompositions can be obtained by grouping generating units with similar time responses [12], [13], by means of the theoretic graph algorithm based on the ϵ Decomposition [14], the sparse eigenvalue-based approach for partitioning power networks [15] or the heuristic method based on the parallel simulated annealing [16], for instance. For many of those cases, the decomposition can be seen as a problem in which strongly coupled variables are grouped in subsystems with weak couplings between them. Nevertheless, when parallel processing is considered, subsystems must keep some type of constraint to prevent the uneven loading of processors.

An interesting nonlinear algebraic solver based on decomposition is the Newton–Maclaurin method. It has previously been used to circumvent the Newton–Raphson method sequentiality when the transmission system equations are solved for power system simulation. As suggested in [5], the Jacobian matrix is decomposed as the sum of a diagonal matrix plus an off diagonal matrix to formulate a new expression for the inverse Jacobian, which was simplified by means of a Maclaurin series evaluated up to its linear terms. The use of a decomposition to replace the Jacobian matrix for a block diagonal matrix is another approach for the factorized inversion process, which can be seen as based on a zero order Maclaurin series. Such a block diagonal matrix has been successfully used to solve the power flow problem [17].

In this paper we use a block parallel VDHN–Maclaurin method for solving the nonlinear algebraic equations that result of the time parallelization-based simulation [18]. This fully parallel indirect method is used to approach the corrections obtained by means of the bi-factorized inversion process used in the VDHN method. We explore to decompose the nonupdated Jacobian matrix, J , as the sum of a block diagonal matrix, J_D , plus a matrix with off diagonal blocks elements, J_O , based on satisfying the following sufficient condition for the convergence of the Maclaurin series: $\|J_D^{-1}J_O\| < 1$. The use of such a decomposition, together with the evaluation of higher order terms in the Maclaurin series, improves

the performance of the simplified inversion process such that if sufficient terms are evaluated the VDHN–Maclaurin method converges in a similar way to the VDHN method. We have investigated the ϵ Decomposition [14] to satisfy the sufficient condition for convergence, and the Longest Path Scheduling Method to balance the size of the blocks obtained from the decomposition. The VDHN–Maclaurin method was implemented on a Parsytec PowerXplorer multicomputer [20] and tested using electromechanical models of the Chilean Central Interconnected system (SIC) and the IEEE300 test system.

II. ELECTROMECHANICAL TRANSIENTS SIMULATION

A. Electromechanical Power System Model

The electromechanical transients simulation is a differential-algebraic initial value problem associated to the power system model [21], [22]:

$$\begin{aligned}\dot{z} &= G(z, y) \\ 0 &= H(z, y)\end{aligned}\quad (1)$$

where

- $z \in \mathbb{R}^m$ state vector;
- $y \in \mathbb{R}^n$ algebraic variables vector;
- $G \in \mathbb{R}^m$ nonlinear vector function, with a quasilinear structure similar to two block diagonal matrices horizontally coupled.

It comprises the differential relations in the generating units, including the synchronous machine, excitation system, PSS, turbine and speed regulator models. $H \in \mathbb{R}^n$ is a nonlinear vector function that comprises the algebraic relations in the model, including the network admittance matrix, the synchronous machine stator relations and the Park’s transformation for voltage and current variables. When power systems equations are organized for simulation, it is useful to isolate algebraic relations from differential relations. Moreover, constant, nonconstant, linear and nonlinear terms can be isolated too. A comprehensive development of such a modeling can be found in [20].

B. Time Parallelization-Based Simulation

A standard procedure for digital dynamic simulation is obtained via trapezoidal integration. It is very attractive because its numerical stability characteristics are particularly suitable for solving problems associated with stiff differential equations [2], [18]. When this method is selected several schemes are able to formulate the simulation problem. In a broad context, the alternating or simultaneous solutions and the spatial or time parallelization approaches are the main alternatives. In this paper the simultaneous solution scheme and the time parallelization-based formulation proposed in [18] have been adopted. It is a nonconventional method whose main task requires to solve a very large set of sparse linear algebraic equations [4].

By using the trapezoidal rule the differential relations in (1) are discretized for consecutive integration steps and added to

the algebraic relations, resulting in a large nonlinear algebraic problem:

$$F(x) = 0, \quad (2)$$

for which, if equal integration steps, Δt , are considered on the total integration period, $]0, t_f = \hat{h}\Delta t]$,

$$F(x) = \begin{bmatrix} -z_1 + z_0 + \frac{\Delta t}{2} [G(z_1, y_1) + G(z_0, y_0)] \\ H(z_1, y_1) \\ -z_2 + z_1 + \frac{\Delta t}{2} [G(z_2, y_2) + G(z_1, y_1)] \\ H(z_2, y_2) \\ \vdots \\ -z_{\hat{h}} + z_{\hat{h}-1} + \frac{\Delta t}{2} [G(z_{\hat{h}}, y_{\hat{h}}) + G(z_{\hat{h}-1}, y_{\hat{h}-1})] \\ H(z_{\hat{h}}, y_{\hat{h}}) \end{bmatrix}. \quad (3)$$

In (2) $F \in \mathbb{R}^{\hat{h}(m+n)}$ is the algebraic vector function and $x \in \mathbb{R}^{\hat{h}(m+n)}$, defined as:

$$x^t = [z_1 \ y_1 \ z_2 \ y_2 \ \cdots \ z_{\hat{h}} \ y_{\hat{h}}], \quad (4)$$

is the unknown variables vector in the discrete time space. The parallelization of the problem defined in (2) is not trivial. Nevertheless, in general the solution to this problem is obtained via the Newton–Raphson method, characterized by the iteration:

$$x^{k+1} = x^k + \Delta x^k, \quad (5)$$

where x^k is the k th approximation to the solution of (2) and its correction, Δx^k , is obtained via:

$$-J_{|_{x^k}} \Delta x^k = -\nabla_x F(x)|_{x^k} \Delta x^k = F(x^k). \quad (6)$$

Then, a common approach for parallelization is to take advantage of some parallel technique to solve the linear problem defined in (6), in which $F(x^k)$ and $J_{|_{x^k}}$ are, respectively, the vector function F and the Jacobian matrix J , both evaluated in x^k . Defining the sub-matrices:

$$\begin{aligned} J_z^h G^{-(+)} &= -(+)I + \frac{\Delta t}{2} \nabla_{z_h} G(z_h, y_h) \\ J_y^h G &= \frac{\Delta t}{2} \nabla_{y_h} G(z_h, y_h) \\ J_z^h H &= \nabla_{z_h} H(z_h, y_h) \\ J_y^h H &= \nabla_{y_h} H(z_h, y_h) \end{aligned} \quad (7)$$

where I is the identity matrix, the following expression is obtained for the Jacobian matrix:

$$J = \begin{bmatrix} J_z^1 G^- & J_y^1 G & 0 & 0 & \cdots & 0 & 0 \\ J_z^1 H & J_y^1 H & 0 & 0 & \cdots & 0 & 0 \\ J_z^1 G^+ & J_y^1 G & J_z^2 G^- & J_y^2 G & \cdots & 0 & 0 \\ 0 & 0 & J_z^2 H & J_y^2 H & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & J_z^{\hat{h}} G^- & J_y^{\hat{h}} G \\ 0 & 0 & 0 & 0 & \cdots & J_z^{\hat{h}} H & J_y^{\hat{h}} H \end{bmatrix}. \quad (8)$$

The dimension of the problem described in (2), in a power system with N buses and M generating units, each one described for an equal number of states q , is $((q+6)M + 2N)\hat{h}$.

III. BLOCK PARALLEL NEWTON–MACLAURIN METHOD

Consider the problem described in (2) and its solution obtained via the Newton–Raphson method, whose initial estimate is iteratively updated by means of the corrections defined in (6).

Assume that $J_{|_{x^k}}$ results in a permuted matrix $\tilde{J}_{|_{\tilde{x}^k}}$ that is decomposed as:

$$\tilde{J}_{|_{\tilde{x}^k}} = E J_{|_{x^k}} E^t = \tilde{J}_{D|_{\tilde{x}^k}} + \tilde{J}_{O|_{\tilde{x}^k}} \quad (9)$$

where

- E permutation matrix;
- $\tilde{x} = Ex$ permuted unknown vector;
- $\tilde{J}_{D|_{\tilde{x}^k}}$ block diagonal matrix, with a number of blocks equal to the number of processors, and all nonzero element of the matrix $\tilde{J}_{O|_{\tilde{x}^k}}$ is outside the diagonal blocks defined in $\tilde{J}_{D|_{\tilde{x}^k}}$.

This decomposition suggests that the correction in (5) can be written as:

$$\Delta \tilde{x}^k = - \left[I + \tilde{J}_{D|_{\tilde{x}^k}}^{-1} \tilde{J}_{O|_{\tilde{x}^k}} \right]^{-1} \tilde{J}_{D|_{\tilde{x}^k}}^{-1} \tilde{F}(\tilde{x}^k) \quad (10)$$

where $\tilde{F}(\tilde{x}^k) = EF(x^k)$ is the set of permuted equations. Now, by assuming that the decomposed matrices satisfy the following property:

$$\left\| \tilde{J}_{D|_{\tilde{x}^k}}^{-1} \tilde{J}_{O|_{\tilde{x}^k}} \right\| < 1, \quad (11)$$

the first inverse in the correction of (10) can be simplified by a convergent Maclaurin series to obtain:

$$\Delta \tilde{x}^k = - \sum_{i=0}^{\eta} \Delta \tilde{x}_i^k, \quad (12)$$

where η is the number of terms to evaluate in the Maclaurin series and the corrections per term are:

$$\Delta \tilde{x}_i^k = \left[-\tilde{J}_{D|_{\tilde{x}^k}}^{-1} \tilde{J}_{O|_{\tilde{x}^k}} \right]^i \tilde{J}_{D|_{\tilde{x}^k}}^{-1} \tilde{F}(\tilde{x}^k), \quad i = 1, \dots, \eta. \quad (13)$$

Equation (12) may be programmed on a coarse grain multicomputer in a full parallel scheme. The inverse of $\tilde{J}_{D|_{\tilde{x}^k}}$ is

computed in parallel assigning one block per processor. Similarly, the multiplications of this inverse times the error matrix $J_{O|\tilde{x}^k}$ are naturally parallelizable by the blocks defined in the partition of $\tilde{J}_{|\tilde{x}^k}$. The addition of higher order terms in the Maclaurin series improves the approximation to the correction of Newton's method when $\eta \rightarrow \infty$.

In a nonlinear simulation, as in the case of power system electromechanical transients, the Jacobian matrix changes through the iterations of Newton's method. Then, it must be noted that a formal implementation of the Newton–Maclaurin method requires to obtain a decomposition with the property described in (11) for each $J_{|\tilde{x}^k}$ such that $J_{|\tilde{x}^k} \neq J_{|\tilde{x}^{k-1}}$. However, to compute one decomposition per iteration would cause much computational effort. Therefore, it seems reasonable to use the partition of the first Jacobian matrix for all the iterations, even though it could deteriorate the convergence of the algorithm. It should be pointed out that such situation is not an extreme simplification, since it is well known that one way of reducing computational efforts of Newton's method is to evaluate and factorize the Jacobian only in presence of topological changes or after a predetermined number of iterations is exceeded. This is the case of the VDHN method, in which $J_{|\tilde{x}^k} = J_{|\tilde{x}^0} \forall k$ excepting the cases previously mentioned. Experience with Newton-type algorithms has shown that the above method is the fastest sequential algorithm for power system simulation, and one the most used for comparative tests. Considering that the VDHN–Maclaurin method is a straightforward version of the Newton–Maclaurin method, and its sequential implementation corresponds to the VDHN method, this method was adopted here.

IV. ALGORITHMS FOR PRACTICAL DEVELOPMENT AND IMPLEMENTATION

A. ϵ Decomposition

In the VDHN–Maclaurin method it is required to decompose the Jacobian matrix taking in account the property defined in (11). There are many algorithms proposed to obtain decompositions. One way of finding a decomposition via trial and error consists in carrying out the operation $EJ_{|\tilde{x}^0}E^t$ and then saving the elements that remain inside the searched diagonal blocks. This decomposition does not assure that the elements outside the blocks have a small magnitude. Then, the number of permuted matrices under analysis to find an approximation that almost minimizes the norm in (11) can be very expensive in computational terms.

In terms of a graph, finding a permutation matrix E is equivalent to decompose the digraph \mathcal{D}_0 corresponding to $J_{|\tilde{x}^0}$ into Q_0 subgraphs \mathcal{D}_{0_q} , $q \in Q_0 = \{1, \dots, Q_0\}$, such that the edges interconnecting the subgraphs are the elements of $J_{|\tilde{x}^0}$ with magnitudes no larger than ϵ . Such a decomposition, $\tilde{\mathcal{Z}}_{\mathcal{D}_0}^\epsilon$ of \mathcal{D}_0 , is called an ϵ Decomposition [14].

The following properties arise from the ϵ Decomposition of $\tilde{J}_{|\tilde{x}^0}$ as $\tilde{J}_{D|\tilde{x}^0} + \tilde{J}_{O|\tilde{x}^0}$:

- Property 1. All elements into $\tilde{J}_{O|\tilde{x}^0}$ have a magnitude no larger than ϵ .

- Property 2. $\|\tilde{J}_{O|\tilde{x}^0}\|_\infty$ is bounded by:

$$\left\| \tilde{J}_{|\tilde{x}^0} - \tilde{J}_{D|\tilde{x}^0} \right\|_\infty = \left\| \tilde{J}_{O|\tilde{x}^0} \right\|_\infty \leq \epsilon \cdot \hat{\beta} \quad (14)$$

where $\hat{\beta} = \max\{\beta_i\}$ and β_i , $i = 1, \dots, \hat{h}(m+n)$, is the number of nonzero elements in the i th row of $\tilde{J}_{O|\tilde{x}^0}$.

In relation to the Newton–Maclaurin method, it was mentioned that to avoid computational efforts $\mathcal{Z}_{\mathcal{D}_0}^\epsilon$ can also be applied to the following digraphs \mathcal{D}_k , $k > 0$, despite the relation among the value of ϵ and the edge magnitudes in \mathcal{D}_k . Furthermore, it is possible to assume that if the digraphs \mathcal{D}_k , $k > 0$, were decomposed by removing the same edges removed in \mathcal{D}_0 , it would lead to $\|\tilde{J}_{D|\tilde{x}^k}^{-1} \tilde{J}_{O|\tilde{x}^k}\| < 1 \forall k$, and then, on that assumption, the decomposition stage would only be required for the first iteration. In absence of hard nonlinearities, as saturations, and system topological changes, only the terms relating voltages and currents in d - q axis to state, $B_{I_{dq}}$ and $B_{V_{dq}}$, and the Park's transformation, T , change between iterations. Without network topological changes, the decomposition that results in grouping generating units by means of an ϵ Decomposition of the network leads to $\tilde{J}_{O|\tilde{x}^k}$ constant and the nonlinear terms remain in $\tilde{J}_{D|\tilde{x}^k} \forall k$. Then, $\mathcal{Z}_{\mathcal{D}_k} = \mathcal{Z}_{\mathcal{D}_0}^\epsilon$ if each element obtained from the nonlinear terms is larger than the same evaluated in the first iteration, that is:

$$\mathcal{Z}_{\mathcal{D}_k} = \mathcal{Z}_{\mathcal{D}_0}^\epsilon \quad \forall k > 0 \quad \left/ \quad \begin{array}{l} |b_{I_{dq}|k}| > |b_{I_{dq}|0}| \quad \forall b_{I_{dq}} \\ |b_{V_{dq}|k}| > |b_{V_{dq}|0}| \quad \forall b_{V_{dq}} \\ |t_k| > |t_0| \quad \forall t \end{array} \quad (15)$$

where $b_{I_{dq}}$, $b_{V_{dq}}$ and t denote the elements of $B_{I_{dq}}$, $B_{V_{dq}}$ and T , respectively.

These bounds have a special importance if it is considered that the convergence of the inversion stage is warranted while the decomposition satisfies the assumption in (11).

B. The Longest Path Scheduling Method

In the ϵ Decomposition process of a digraph it is possible to obtain digraphs with only one node. Similarly, in the ϵ Decomposition of the transmission system, subsystems with only one bus can be obtained. Such situations are not attractive for parallel processing. This result can be avoided by using a balancing stage complementing the decomposition.

In this paper we propose to use the Longest Path Scheduling Method [19] to automatically balance the size of the blocks obtained via the ϵ Decomposition. A previous stage is to obtain an ϵ Decomposition that consists of blocks whose size must be no larger than the balance index wanted, γ . For an efficient parallel application, the index:

$$\gamma = \frac{\hat{h}(m+n)}{p} \quad (16)$$

is used, where p is the number of available processors. The algorithm based on the Longest Path Scheduling Method considers the p biggest blocks as initial load. Then, the biggest free block is grouped with the smallest block that has already been assigned.

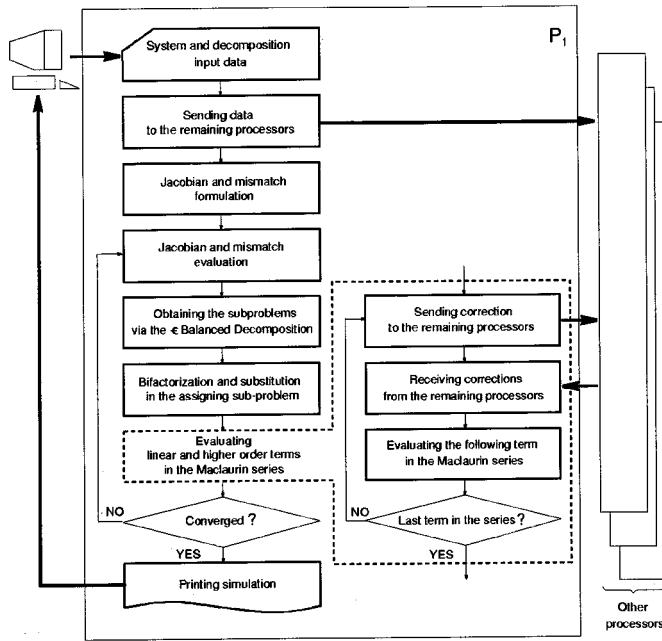


Fig. 1. Flowchart for the simulation of electromechanical transients on the Parsytec PowerXplorer multicomputer.

C. Implementation on a Multicomputer

The Parsytec PowerXplorer multicomputer consists of two PowerXplorer systems, each one with four PowerPC 601 RISC processors that work with 64 bits and 80 Mflops. The processing nodes have 32 MB of RAM, 1 MB of cache memory and an Inmos T805 transputer to perform the communication. Each PowerXplorer is connected, via an SCSI controller, to a Sun SPARCclassic 5 workstation that operates as server.

The algorithm was programmed in C language by using PARIX, an operating system based on UNIX with proper compilers and available for the multicomputer. Fig. 1 represents the simulation of power system electromechanical transients, as it solved by using the time parallelization concept and the VDHN–Maclaurin method. It includes a diagram that represents the operation of the first processor and its interaction with the remaining processors, and also the flowchart with the simulation stages.

Input and output data stages are carried out for the first processor, the other stages are made for all the processors. This specific processor reads power flow, dynamic and decomposition input data from the server and, after storing them in its local memory, it sends them to the remaining processors. Then, simultaneously, all the processors formulate the same problem. By using the ϵ Balanced Decomposition each processor obtains the vectors and matrices that define the assigned linear algebraic subproblem. The effective parallelization begins when the processors solve just the setting subproblem. Thus, each processor bi-factorizes only one of the p blocks in $\tilde{J}_{D|_{x^0}}$. The solution obtained via the substitution in the subproblem, which is equivalent to evaluate the correction associated with the zero order term of the Maclaurin series, $\Delta\tilde{x}^0 = -\Delta\tilde{x}_0^0$, where $\Delta\tilde{x}_0^0 = \tilde{J}_{D|_{x^0}}^{-1} \tilde{F}(x^0)$, is also carried out in a simultaneous and independent way. At that instance each processor keeps only one of the

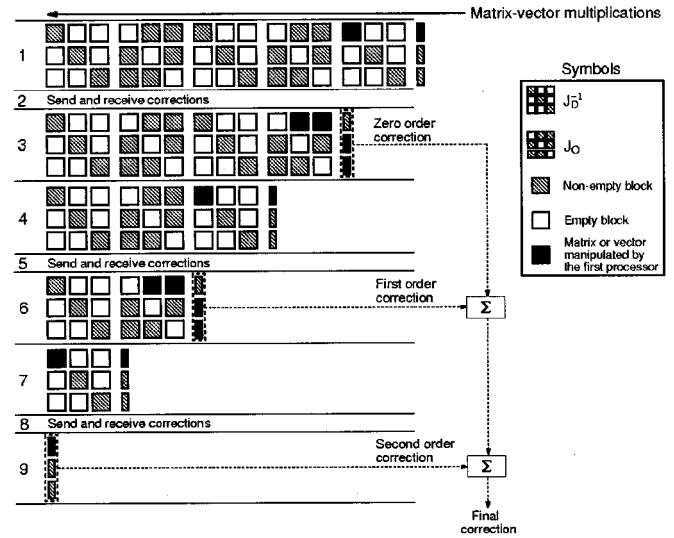


Fig. 2. First processor task assignment for the evaluation of a second order Newton–Maclaurin method on three processors.

p -corrections: the setting correction. To evaluate a new correction, $\Delta\tilde{x}^0 = \Delta\tilde{x}^0 - \Delta\tilde{x}_0^0$, improved with the following term of the series, $\Delta\tilde{x}_1^0 = -\tilde{J}_{D|_{x^0}}^{-1} \tilde{J}_{O|_{x^k}} \Delta\tilde{x}_0^0$, the processors require to know the corrections in the remaining processors at the term lately evaluated. Then, sequentially, each processor sends the setting correction to the other processors. This task is repeated to evaluate $\Delta\tilde{x}^0 = \Delta\tilde{x}^0 + \Delta\tilde{x}_2^0$, where $\Delta\tilde{x}_2^0 = -\tilde{J}_{D|_{x^0}}^{-1} \tilde{J}_{O|_{x^k}} \Delta\tilde{x}_1^0$, and so on until all terms are evaluated. With the $\Delta\tilde{x}^0$ correction already evaluated, each processor gets the same numerical solution x^1 , updates the error function $\tilde{F}(x^1)$ and checks for convergence; then, if it is required, updates the Jacobian $\tilde{J}_{|_{x^1}}$ and the previously setting blocks. At this stage the processors have renewed the setting subproblems and can begin a new iteration, similar to the previously described one, and repeat it until the convergence is reached. Fig. 2 represents the task assignment for the first processor when the terms associated to a second order Maclaurin series are evaluated on three processors.

V. SIMULATION STUDIES ON A MULTICOMPUTER

The following results were obtained from simulation studies of power systems electromechanical transients. They illustrate some characteristics of the parallel simulation, as it solved via the VDHN–Maclaurin method and implemented on the Parsytec PowerXplorer multicomputer. The objective of the selected tests is to evaluate the advantages and drawbacks in several stages of the time parallelization-based simulation, the parallelization strategy and methods used in its development. The speed-up:

$$S = \frac{T_1}{T_p} \quad (17)$$

was the selected performance index, where T_1 and T_p are, respectively, the computing times required to solve the nonlinear equations, no data preparing time, on 1 and p -CPUs.

Most of the results were obtained with a simplified electromechanical model used in planning studies of the Chilean Central Interconnected System (SIC). Because the SIC model has a relatively small size, the IEEE300 test system was considered in

TABLE I
SUMMARY OF THE TEST SYSTEMS

	SIC	IEEE300
Generating units (M)	14	69
Loads	24	198
Buses (N)	94	300
Transmission lines	66	304
Transformers	53	107
Shunt devices	13	29

TABLE II
GENERATING UNITS IN THE SIC MODEL

Unit	MVA	Power factor	kV (nominal)
Rapel	333.1	0.961	13.8
Sauzal	49.11	0.977	13.2
Sauzalito	7.07	0.990	13.8
Cipreses	103.1	0.969	13.2
Isla	68.1	0.969	13.8
Pehuenche	469.58	0.958	13.8
Colbún	465.91	0.944	13.8
Machicura	100.18	0.948	13.8
Toro	341.48	0.937	13.8
Antuco	314.56	0.954	13.8
Abanico	43.74	0.983	13.8
Concepción	7.60	0	13.8
Canutillar	130	0.969	13.8
Alfalfal	157.34	0.890	12.0

order to evaluate the method on a larger-scale problem. Table I summarizes the general characteristics for both systems.

The SIC generating units dynamically represented are listed in Table II. Generating units data for the IEEE300 test system were estimated from typical units [22] considering the nominal MVAs. For both systems each synchronous machine was described by a third order model representing the electro-mechanical and the rotor field circuit dynamics. All the models for generating units include a first order model representing the automatic voltage regulator. Loads are represented as constant admittances.

The opening of one 500 kV circuit between Colbún and Alto–Jahuel was simulated for the SIC. A line to ground three phase short circuit with a small reactance of 0.001 pu was simulated for the IEEE300 test system; fault location is at bus 169 and the clearing time is $t_{cl} = 0.06$ sec.

The tolerance for convergence is $|F(\tilde{x}^k)| < 0.001$. The Jacobian matrix was held constant through the iterations, and then its decomposition too. The integration step and the simulation time are specified in each case. As a manner to overcome the problems that present the use of a decomposition based on the natural Jacobian [23], before obtaining the decomposition this matrix was always scaled by its diagonal entries.

A. Evaluating the Time Parallelization

The first test evaluates the time parallelization-based simulation by solving, simultaneously, several number of consecutive integration steps \hat{h} . The integration step is $\Delta t = 0.02$ sec. and only one processor was used. Fig. 3 shows that the processing times required per simulation stage increase in a proportional form with the increment of the number of simultaneously solved

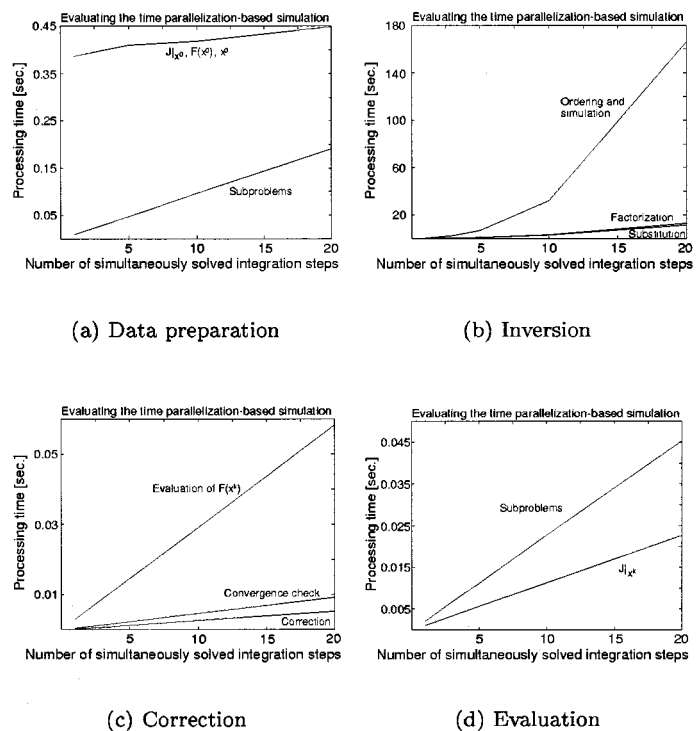


Fig. 3. Processing time per stage versus number of simultaneously solved integration steps.

TABLE III
NORMS TO EVALUATE THE CONVERGENCE OF THE MACLAURIN SERIES

CPUs.	$\ J\ _\infty$	$\ J_D\ _\infty$	$\ J_D^{-1}\ _\infty$	$\ J_0\ _\infty$	$\ J_D^{-1}J_0\ _\infty$
1	3043	-	-	-	-
2	-	3043	184	28	33
8	-	3043	184	131	35

integration steps. The exceptions are the bi-factorized inversion stage, which increases almost exponentially, and the data preparation, including the multimachine model, Jacobian matrix and error function buildings, but no their evaluations with the actual solution x^k , which increases insignificantly. It can also be noted that the bi-factorized inversion stage is the most time consuming.

The results show that the time parallelization is not convenient, at least with the sparse bi-factorized solution used here, because the processing time increases more than proportionally with the number of simultaneously solved integration steps. This problem is partially explained by the excessive fill-in at the end of the bi-factorization stage.

B. Evaluating Algorithms for Practical Development

1) ϵ Decomposition: The following test evaluates the ϵ Decomposition as a method to obtain a decomposition that approaches the sufficient condition for the convergence. In this case, the integration step is $\Delta t = 0.02$ sec. and the window size (simultaneous integration time) is $t_f = 0.04$ sec. Table III includes the infinity norm ($\|\cdot\|_\infty$) of the Jacobian matrix and its decomposed matrices. Partitions for 2 and 8 processors were considered. It can be viewed that the sufficient condition for the

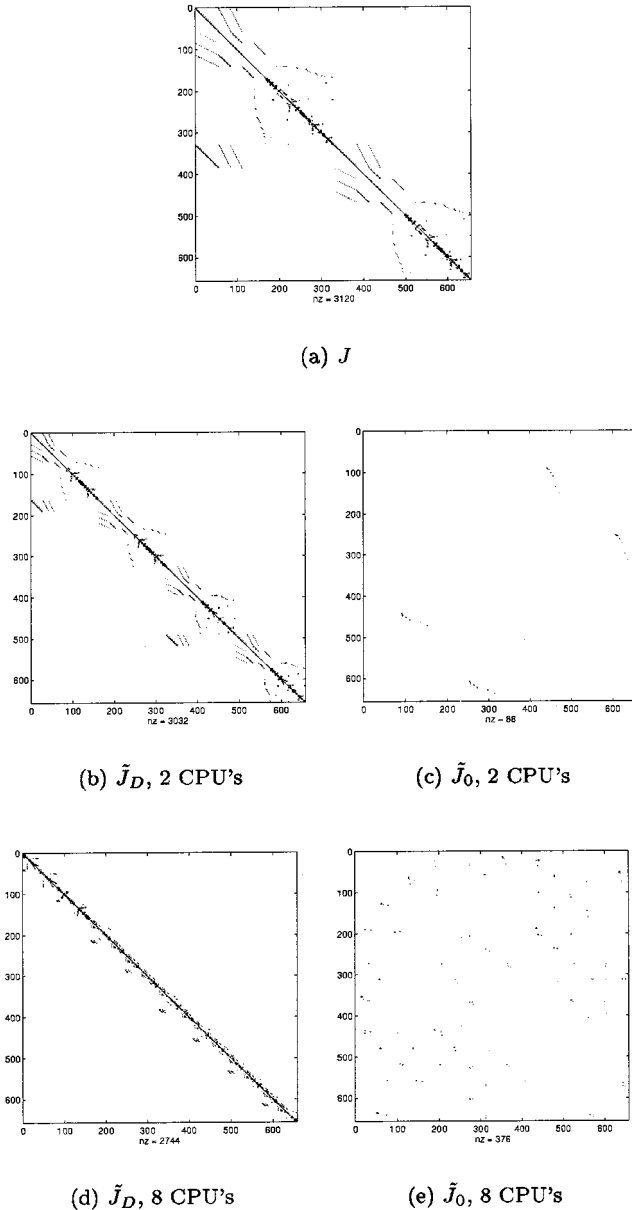


Fig. 4. ϵ balanced decompositions for parallel processing.

convergence in the inversion stage is not reached. Nevertheless, these results permit to get an insight into the performance of the ϵ decomposition. On the other hand, it must be noted that they do not imply that the Maclaurin series diverges, because the norm is only an upper bound on the properties of the decomposed matrices. Fig. 4(a) shows the Jacobian required to obtain the simulation. Fig. 4(b) and (c) show, respectively, the decomposition of the permuted Jacobian in a block diagonal matrix and a matrix with off diagonal blocks elements, as it obtained for processing on 2 processors. Fig. 4(d) and (e) show the same decomposed matrices, but considering a decomposition for 8 processors. It can be seen that the blocks preserve some symmetry, it because the ϵ decomposition distributed the nodes in such a way that the blocks keep the variables per generating unit and per trajectory (the variables defining a trajectory are held in the same cluster), that is, the blocks were obtained by grouping generating units.

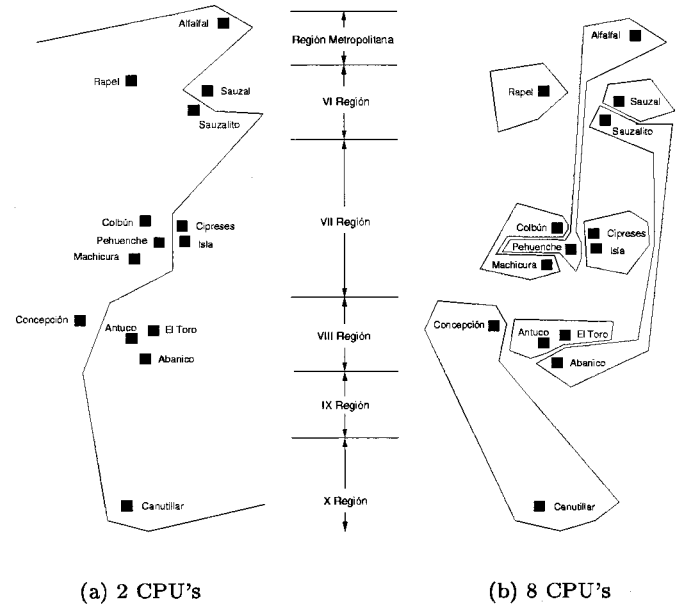


Fig. 5. Clustering of the SIC generating units via the ϵ balanced decomposition.

TABLE IV
RESULTS TO EVALUATE THE BALANCE AS IT OBTAINED VIA THE LONGEST PATH SCHEDULING METHOD

Processor	1	2	3	4
Size	3280	3240	3280	3320
$\delta\gamma$ (%)	0	-1.22	0	+1.22
Ordering-simulation (sec.)	45.104	45.952	39.133	37.122
Bi-factorization (sec.)	3.437	3.549	3.758	3.217
Inversion (sec.)	48.541	49.501	42.891	40.339
$\delta\bar{t}_I$ (%)	7.11	9.23	-5.36	-10.99

Fig. 5 shows the approximate geographical location of the SIC generating units, which is used for the graphical representation of the decompositions, indicating the clusters of generating units obtained via decompositions in 2 and 8 blocks. It is interesting to note that the clustering is similar, in the sense of grouping generating units, to the coherency recognition methods [12].

2) *Load Balance Via the Longest Path Scheduling Method:* The performance of the Longest Path Scheduling Method was evaluated considering the bi-factorized inversion computing times per processor. In this case the integration step is $\Delta t = 0.015$ sec., the window size is 0.6 sec. and the decomposition considers the use of 4 processors. Table IV includes the size of the blocks in \tilde{J}_D matrix, the percentage deviation around the optimal balance index, $\delta\gamma$, and the times spent in the ordering-simulation stage and the bi-factorization stage, respectively. The addition of both these times, equal to the inversion time less the substitution time, and its percentage deviation around the average, $\delta\bar{t}_I$, are also included.

Table IV presents the results of the load balance among the processors when a sparsity based method is used for the inversion process. In these methods the number of branches and the fill-in can significantly affect the load balance. Nevertheless, a balance that takes in account this situation is very difficult. Remembering that the method used for balancing the blocks is only

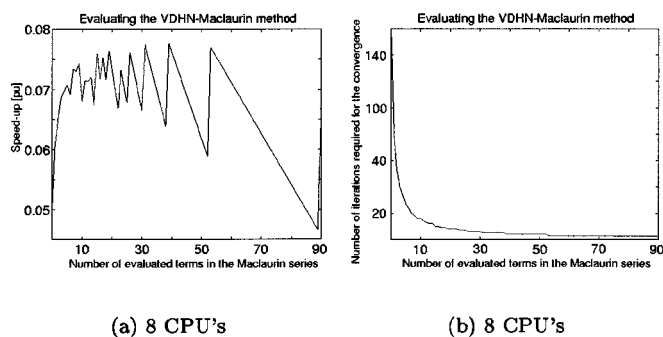


Fig. 6. Speed-up and number of iterations required for convergence as a function of the evaluated terms in the Maclaurin series.

based on the equilibrium of the blocks size in \tilde{J}_D , it can be noted that this method is very simple and effective.

C. Evaluating the VDHN Maclaurin Method in Step-by-Step Simulations

Considering that the step-by-step simulation has better performance than the time parallelization-based simulation, the following tests evaluate the VDHN–Maclaurin method by simulating only one integration step.

1) *Evaluation of Higher Order Terms of the Maclaurin Series*: Both the number of processors to be used and the number of terms to be evaluated in the Maclaurin series can be freely chosen in the algorithm. The following test considers this issue. Fig. 6 shows both speed-up reached on 8 CPUs and number of iterations required for convergence as a function of the evaluated terms in the Maclaurin series. The simulation parameters are $\Delta t = 0.02$ sec. and $t_f = 0.02$ sec. From Fig. 6(b) is viewed that the VDHN–Maclaurin method converges, and when 90 terms of the Maclaurin series are evaluated the number of iterations required for convergence is the same number than the VDHN method requires. Fig. 6(a) shows that the smaller slow down was obtained when 39 higher order terms were evaluated. Nevertheless, it is slightly better in relation to the evaluation of low order terms. The latter situation is preferable because the speed-up is very sensitive to the higher order terms, where the saw type effect is more notorious. This effect is produced because the tolerance for the simulation convergence is a fixed parameter, then some terms improve the solution but do not permit to reach the convergence without extra iterations. Similar results, that is, slow-down, saw effect, and convergence in all the cases were obtained using 2, 4 and 6 processors, but they are not shown here.

2) *Speed-Up*: Fig. 7 shows the speed-up obtained via parallel simulations of the SIC and the IEEE test system as a function of the number of processors used. The speed-up for the SIC corresponds to the best processing time with respect to the evaluated terms in the Maclaurin series. The parallel method does not reduce the serial processing time. Furthermore, the slow down is more significant when the number of processors is increased. In this respect, it is probable that the communication times can be larger than the processing times, which may be partially explained by the size of the model representing the SIC.

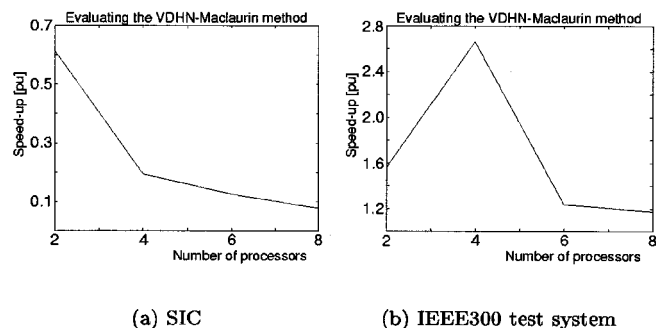


Fig. 7. Speed-up in step-by-step simulations.

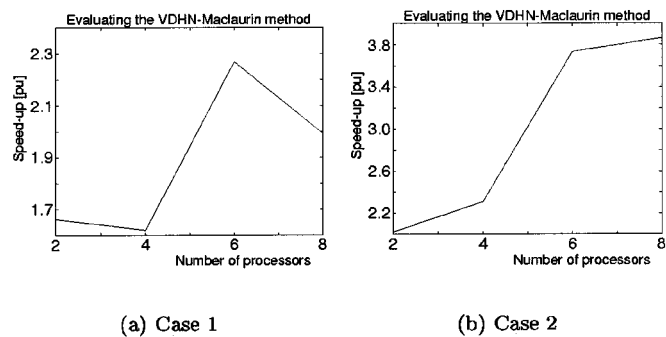


Fig. 8. Speed-up for time parallelization-based simulations of the SIC.

Simulation studies in the IEEE300 test system were obtained at the post-fault period. The simulation was began at $t_0 = 0.06$ sec. and finished at $t_f = 0.08$ sec. To avoid the saw effect, all the following simulations were obtained evaluating only 2 terms of the Maclaurin series.

Fig. 7(b) shows a satisfactory speed-up when 2 and 4 processors were used. The poor speed-up on 6 and 8 processors is mainly due to the relatively slow communication network and fast processing CPUs. However, the results indicate that the proposed algorithm presents advantages when it works with large scale systems.

D. Evaluating the VDHN Maclaurin Method in Time Parallelization-Based Simulations

In previous test a slow-down for the SIC and a moderate speed-up for the IEEE300 test system were obtained. It is obvious to think that an important factor in the performance of the parallelization strategy is the size of the formulated problems. To prove this supposition, the parallelization strategy was evaluated in larger size problems. In absence of large scale power system data, which lead to large nonlinear algebraic problems when step-by-step simulations are considered, the researchers studied nonlinear algebraic problems obtained via the time parallelization-based simulation.

Fig. 8 shows the speed-up as a function of the number of processors. Two simulation cases were considered, whose data are provided in Table V. The speed-up in Case 2 is more satisfactory than Case 1, mainly because the size of the problem is two times bigger. It is noted that the speed-up in case 2 is close to 4

TABLE V
SUMMARY OF THE TIME PARALLELIZATION-BASED SIMULATION STUDIES

Case	$\Delta t(\text{sec.})$	$t_f(\text{sec.})$	\tilde{h}	$\tilde{h}(m+n)$
1	0.02	0.4	20	6560
2	0.01	0.4	40	13120

when 8 processors are used, and it does not present a considerable saturation. Larger problems obtained from the time parallelization-based simulation were not solved, because the available RAM was exhausted by the fill-in. This fact confirms that the bi-factorized inversion should not be a good solver when very large scale systems are simulated, unless a more efficient ordering and simulation algorithm is considered.

VI. CONCLUSION

In this paper the time parallelization concept was used to formulate the simulation of power system electromechanical transients as a nonlinear algebraic problem. Its parallel solution was successfully obtained via the VDHN–Maclaurin method. A Theoretical background about the convergence of the inversion process was provided, which is a desirable situation for the design of more robust algorithms. The ϵ Decomposition was used to satisfy the sufficient condition for convergence of the inversion process, and the Longest Path Scheduling Method to prevent the uneven loading of processors. The implementation of the VDHN–Maclaurin method was made on a Parsytec PowerXplorer multicomputer.

It was pointed out that the sparse bi-factorized inversion had better performance in step-by-step simulations than time parallelization-based simulations. A moderate speed-up of 2.65 using 4 CPUs was reported for step-by-step simulations of the IEEE300 test system, whilst a speed-up close to 4 using 8 CPUs was reported for larger size problems arising from the time parallelization-based simulation of the SIC. This suggests that more experience with simulations of large scale systems, in addition to the use of new technologies incorporating powerful hardware, can improve the block parallel VDHN–Maclaurin method.

REFERENCES

- [1] D. Falcão, "High performance computing in power system applications," in *Proceedings of the 2nd International Meeting on Vector and Parallel Processing (VECPAR'96)*, Porto, Portugal, Sept. 25–27, 1996.
- [2] B. Stott, "Power system dynamic response calculations," *Proceedings of the IEEE*, vol. 67, no. 2, pp. 219–241, Feb. 1979.
- [3] V. Vittal, G. M. Prabhu, and S. L. Lim, "A parallel computer implementation of power system transient stability assessment using the transient energy function method," *IEEE Trans. Power Systems*, vol. 6, no. 1, pp. 167–173, Feb. 1991.
- [4] I. C. Decker, D. Falcão, and E. Kaszkurewicz, "Conjugate gradient methods for power system dynamic simulation on parallel computers," *IEEE Trans. Power Systems*, vol. 11, pp. 1218–1227, Apr. 1996.
- [5] J. S. Chaff and A. Bose, "Bottlenecks in parallel algorithms for power system stability analysis," *IEEE Trans. Power Systems*, vol. 8, no. 1, pp. 9–15, Feb. 1993.
- [6] G. P. Granelli, M. Montagna, M. La Scala, and F. Torelli, "Relaxation-Newton methods for transient stability analysis on a vector/parallel computer," *IEEE Trans. Power Systems*, vol. 9, no. 2, pp. 637–643, May 1994.
- [7] M. Crow and M. Ilić, "The parallel implementation of the waveform relaxation method for transient stability simulations," *IEEE Trans. Power Systems*, vol. 5, no. 3, pp. 922–932, Aug. 1990.
- [8] L. Hou and A. Bose, "Implementation of the waveform relaxation algorithm on a shared memory computer for the transient stability problem," *IEEE Trans. Power Systems*, vol. 12, no. 3, pp. 1053–1060, Aug. 1997.
- [9] M. La Scala, G. Sblendorio, and R. Sbrizzai, "Parallel-in-time implementation of transient stability simulations on a transputer network," *IEEE Trans. Power Systems*, vol. 9, no. 2, pp. 1117–1125, May 1994.
- [10] P. E. Crouch, E. Brady, and D. J. Tylavsky, "Frequency domain transient stability simulation of power systems: Implementation by supercomputer," *IEEE Trans. Power Systems*, vol. 6, no. 1, pp. 51–58, Feb. 1991.
- [11] J. Q. Wu, A. Bose, J. A. Huang, A. Valette, and F. Lafrance, "Parallel implementation of power system transient stability analysis," *IEEE Trans. Power Systems*, vol. 12, no. 3, pp. 1126–1233, Aug. 1995.
- [12] H. Rudnick, R. Patiño, and A. Brameller, "Power system dynamic equivalents: Coherency recognition via the rate of change of kinetic energy," *Proceeding of the IEE*, pp. 325–333, Sept. 1981.
- [13] J. Chow, *Time Scale Modeling of Dynamic Networks with Applications to Power Systems*. New York: Springer Verlag, 1982.
- [14] M. E. Sezer and D. D. Šiljak, "Nested ϵ -decompositions and clustering of complex systems," *Automatica*, vol. 22, no. 3, pp. 321–331, Aug. 1986.
- [15] N. Muller and V. H. Quintana, "A sparse eigenvalue-based approach for partitioning power network," *IEEE Trans. Power Systems*, vol. 7, no. 2, pp. 520–527, May 1992.
- [16] H. Mori and K. Takeda, "Parallel simulated annealing for power system decomposition," *IEEE Trans. Power Systems*, vol. 9, no. 2, pp. 789–795, May 1999.
- [17] M. Amano and D. D. Šiljak, "An improved block-parallel Newton method via epsilon decompositions for load-flow calculation," *IEEE Trans. Power Systems*, vol. 11, no. 3, pp. 1519–1527, Aug. 1996.
- [18] F. L. Alvarado, "Parallel solution of transient problems by trapezoidal integration," *IEEE Trans. Power Apparatus and Systems*, vol. PAS-98, no. 3, pp. 1080–1089, May–June 1979.
- [19] M. T. Kaufman, "An almost-optimal algorithm for the assembly line scheduling problem," *IEEE Trans. Computers*, vol. c-23, pp. 1169–1174, May 1974.
- [20] F. Morales, "Simulación de Transitorios Electromecánicos en Sistemas Eléctricos de Potencia Mediante Procesamiento Paralelo," (in Spanish), Research Report, Pontificia Universidad Católica de Chile, 1999.
- [21] P. Kundur, *Power System Stability and Control*. New York: McGraw-Hill, 1994.
- [22] P. M. Anderson and A. A. Fouad, *Power System Control and Stability*. New York: Institute of Electrical and Electronics Engineers, Inc., 1994.
- [23] N. Gačić, A. I. Začević, and D. D. Šiljak, "Coherency recognition using epsilon decomposition," *IEEE Trans. Power Systems*, vol. 13, no. 2, pp. 314–319, May 1998.