IEEE Transactions on Power Systems, Vol. 11, No. 3, August 1996

Conjugate Gradient Methods for Power System Dynamic Simulation on Parallel Computers

I.C. Decker* D.M. Falcão E. Kaszkurewicz
COPPE/Federal University of Rio de Janeiro
PO Box 68516, 21945-970 Rio de Janeiro RJ, Brazil

Abstract—**Parallel processing is a promising technology for the speedup of the dynamic simulations required in transient stability analysis. In this paper, three methods for dynamic simulation on parallel computers are described and compared. The methods are based on the concepts of spatial and/or time parallelization. In all of them, sets of linear algebraic equations are solved using different versions of Conjugate Gradient methods which have been successfully applied in other scientific and engineering applications. The algorithms presented in the paper were tested in a commercially available parallel computer using an actual large power system model. The results obtained in the tests showed a considerable reduction in computation time.**

Keywords—**Transient stability, vector processing.**

I. Introduction

Dynamic simulation for transient stability analysis is one of the most computer intensive power system applications. Practical planning studies may require several hours of computation in conventional machines. The introduction of transient stability considerations in real-time security assessment is up to date limited, among other problems, by difficulties in obtaining acceptable time delays in the computer response. Parallel processing is a promising technology for the speedup of dynamic simulation computations [1]. At present, parallel computers are commercially available in a price range acceptable for power system applications and offering enough computation power [2]. These machines can be used as high performance dedicated workstations or integrated with local area networks that are becoming available both in the off-line and on-line power system computer environments. However, to fully exploit the potential of this technology, new developments are required in the algorithms used for power system dynamic simulation.

In the last decades, an intensive research effort has generated several methods for dynamic simulation on parallel computers. These methods can be broadly classified in three categories: Spatial Parallelization [3]-[10], Waveform Relaxation [11, 12], and Spatial and Time Parallelization [13]-[15].

A basic numerical problem in all dynamic simulation methods is the solution of sets of linear algebraic equations. Direct methods, like LU factorization, have been dominating this application in conventional computers. If parallel computers are considered, however, the hegemony of direct methods is no more guaranteed. In several other engineering and scientific fields, parallel implementations of iterative methods have shown superior performance. Among the most successful iterative methods are the ones belonging to the Conjugate Gradient (CG) category [16, 17]. This class of methods is also suitable for vector processing. CG methods have been previously applied to power system problems with fairly good results [6, 7, 9, 18, 19].

This paper describes and compares three methods for the parallel solution of the dynamic simulation problem. The first two, including the one previously introduced in [6] and [7], are based on spatial parallelization concepts and the third one uses the idea of space and time parallelization. In all of them, the linear algebraic equations are solved using different versions of the CG methods. The methods were tested in a commercially available parallel computer using an actual large power system model. The results obtained in the tests showed considerable reduction in the computation time.

*On leave from the Federal University of Santa Catarina, Brazil

95 SM 506-6 PWRS A paper recommended and approved by the IEEE Power System Engineering Committee of the IEEE Power Engineering Society for presentation at the 1995 IEEE/PES Summer Meeting, July 23-27, 1995, Portland, OR. Manuscript submitted January 4, 1994; made available for printing June 5, 1995.

II. Problem Formulation

The electromechanical dynamics of a power system is described by a set of differential and algebraic equations which can be solved using different solution schemes [20]. The main differences between the schemes are in the numerical integration approach (implicit or explicit) and in the strategy to solve the differential and algebraic set of equations (simultaneous or alternating). Implicit integration methods, particularly the trapezoidal rule, have been mostly adopted for this application and is the one used in this work.

A. Alternating Implicit Scheme (AIS)

This solution scheme is better understood if the differential and algebraic equations are written as follows [20]:

$$\begin{aligned} \dot{x} &= Ax + Bu & (1)\\ I(E,V) &= YV & (2)\\ u &= h(E,V) & (3) \end{aligned}$$

where A is a square, real valued, block diagonal matrix in which each block is associated to one machine, B is a rectangular, real valued, blocked matrix in which each block is associated to one machine, u is a vector of interface variables, I is a vector of injected nodal currents, Y is a square, complex valued, nodal admittance matrix, V is a vector of complex nodal voltages representing the steady-state network behavior, E is a subvector of x required to calculate current injections in generation nodes, and h is a nonlinear vector function.

The alternating implicit scheme consists in transforming the differential equations in difference equations, by the application of an implicit integration method, and to solve these equations iteratively and alternately with the algebraic equations. A modified version of this scheme, called the Interlaced Alternating Implicit scheme [6, 7, 20], is obtained relaxing the convergence requirements on the network equation solution. The convergence test is performed in the state variables. This method usually presents better results than the conventional approach and is the one adopted in this work.

B. Simultaneous Implicit Scheme (SIS)

In this approach, the network algebraic equations and the algebrized differential equations

$$\begin{aligned} F(x,V^e) &= 0 & (4)\\ G(x,V^e) &= 0 & (5) \end{aligned}$$

where F is a vector function associated with the difference algebraic equations, G is a vector function associated with the algebraic equations, and V^e is a vector of nodal voltages expanded in its real and imaginary components, are lumped together in an enlarged set of equations which are, then, solved simultaneously. The Newton method is the usual choice of method to solve this set of equations.

III. Parallel Solution Methods

The difficulties for the parallelization of the dynamic simulation problem in the AIS are concentrated on the network solution. As can be seen in (1)-(3), the equations associated with the synchronous machines and their controllers are naturally decoupled and easily parallelizable. On the other hand, the network equations constitute a *tightly coupled* problem requiring ingenious decomposition schemes [21] and solution methods suitable for parallel applications [16, 17]. The SIS also requires the parallel solution of (4) and (5), in every integration step, with difficulties similar to the ones described for the AIS. Greater potential for parallelization exploitation can be achieved if several integration steps are considered simultaneously, as firstly suggested in [13].

Three different approaches for the parallel solution of the dynamic simulation problem were studied in the work described in this paper. The first two are based on the AIS scheme and the third one is based on the SIS. In all three, CG methods are used to solve totally or partially the associated systems of linear algebraic equations.

A. Spatial Parallelization

This approach is concerned with the parallel solution of the network equations in the AIS taking advantage of the geographical and topological properties of the power network. The use of CG methods for the solution of linearized versions of these equations, combined with a careful mapping of the problem structure onto the parallel computer architecture, presents great potential for speedup on parallel and vector computers. Two approaches are analyzed in this work:

1) Method 1: The network equations are ordered in such a way that Y appears in the Block Bordered Diagonal Form (BBDF) [3], as follows:

$$\begin{bmatrix} I_1(E_1,V_1)\\ I_2(E_2,V_2)\\ \vdots\\ I_q(E_q,V_q)\\ I_s(E_s,V_s) \end{bmatrix} = \begin{bmatrix} Y_1 & & & & \bar{Y}_1\\ & Y_2 & & & \bar{Y}_2\\ & & \ddots & & \vdots\\ & & & Y_q & \bar{Y}_q\\ \bar{Y}_1^t & \bar{Y}_2^t & \dots & \bar{Y}_q^t & Y_s \end{bmatrix} \begin{bmatrix} V_1\\ V_2\\ \vdots\\ V_q\\ V_s \end{bmatrix} \quad (6)$$

The BBDF of the network equations can be achieved by re-ordering the network nodes in $q+1$ sets in which the q first sets correspond to subnetworks connected to each other by the ***boundary buses*** in the $(q+1)^{th}$ set.

Equation (6) can be solved in a two-phase scheme by Block-Gaussian Elimination as follows:

Phase 1: Solve

$$\hat{Y}_s V_s = \hat{I}_s \quad (7)$$

where $\hat{Y}_s = Y_s - \sum_{i=1}^{q} \bar{Y}_i^t Y_i^{-1} \bar{Y}_i$ and $\hat{I}_s = I_s - \sum_{i=1}^{q} \bar{Y}_i^t Y_i^{-1} I_i$

Phase 2: For $i = 1, 2, \dots, q$, solve

$$Y_i V_i = I_i - \bar{Y}_i V_s \quad (8)$$

The above equations are then solved by a parallel scheme combining the CG and LU Factorization methods. The CG method is used to solve equation (7) while the LU factorization is used to solve the q independent equations sets defined in (8). This method was previously introduced in [6, 7], and has shown a fairly good performance on actual parallel implementations but presents the disadvantages of applying the CG method to a system of equations with relatively small dimensions and the need of an optimized BBDF.

2) Method 2: This method is an attempt to overcome the disadvantages presented by Method 1. The CG gradient method is supposed to present best performance, both on parallel and vector computers, for very large problems. Therefore, in this method the network equations are solved as a whole by a block-parallel version of the Preconditioned CG method (see the Appendix). The network matrix is decomposed in such a way that the blocks in the diagonal are weakly coupled to each other, i.e., in a Near Block Diagonal Form (NBDF) [3], as follows:

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_q \end{bmatrix}^k = \begin{bmatrix} Y_{11} & Y_{12} & \dots & Y_{1q} \\ Y_{21} & Y_{22} & \dots & Y_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{q1} & Y_{q2} & \dots & Y_{qq} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_q \end{bmatrix}^{k+1} \tag{9}$$

The NBDF is equivalent to a network decomposition in subnetworks weakly coupled [3, 21]. To optimize the algorithm performance, the NBDF should be obtained according to the following:

- The number of subnetworks is fixed and equal to the number of processors used in the simulation.
- The number of subnetwork connections is kept to a minimum in order to reduce communication requirements.
- The subnetworks have approximately the same dimensions and complexity.

Two approaches were used to obtain the NBDF's with the above characteristics: a trial and error method based on a careful analysis of the system one-line diagram and the semiautomatic methodology described in [21].

The matrix $M = -$ blockdiag$[\ Y_{11} \quad Y_{22} \quad \cdots Y_{qq}\ ]$, obtained from the NBDF neglecting the off-diagonal blocks, is used as a preconditioner.

3) AIS Parallel Algorithm: The general AIS parallel approach can be summarized in the following algorithm:

```
Initialization
For $t = 1,2,\dots,T$
    For $i = 1,2,\dots,q$
        Obtain $u_i^*(t)$ e $V_i^*(t)$, by extrapolation
        $x_i^0(t) = H_i[x_i^*(t), u_i^*(t)x_i(t-1), u_i(t-1)]$
    Endfor
    For $k = 0,1,2,\dots$
    For $i = 1,2,\dots,q$
        Obtain $E_i^k(t)$ e $I_i^k(t)$
        Solve $I^k(t) = YV^{k+1}(t)$
        $u_i^{k+1}(t) = h_i(E_i^k(t), V^{k+1}(t))$
        $x_i^{k+1}(t) = H_i[x_i^k(t), u_i^{k+1}(t), x_i(t-1), u_i(t-1)]$
        $\Delta x_i^{k+1}(t) = ||x_i^{k+1}(t) - x_i^k(t)||$
        If $\Delta x_i^k(t) < \epsilon$, any $i$, then quit
    Endfor
    Endfor
Endfor
```

In this parallel algorithm, each one of the q tasks are assigned to a processor. The network equation ($I^k(t) = YV^{k+1}(t)$), can be efficiently solved in parallel using either Method 1 or Method 2.

The use of preconditioning techniques is advised for a better performance of Method 1 and essential for an adequate performance of Method 2. Truncated Series preconditioning [16] is used for Method 1. For Method 2, the Incomplete LU Factorization [17] was chosen as the preconditioning approach after preliminary tests with several methods available in the literature.

B. Space and Time Parallelization

In this approach, the differential equations are algebrized for several integration steps, called integration windows, and solved together with the algebraic equations of this window by the Newton method, similarly to the conventional one-step approach. The resulting enlarged system of linear equations has the following structure:

$$\xi = -\Lambda.\Delta\delta \tag{10}$$

where

$$\xi = [\ \hat{F}_1 \quad \hat{G}_1 \quad \hat{F}_2 \quad \hat{G}_2 \quad \cdots \quad \hat{F}_p \quad \hat{G}_p\ ]^T$$

$$\Delta\delta = [\ \Delta\hat{x}_1 \quad \Delta\hat{V}_1^e \quad \Delta\hat{x}_2 \quad \Delta\hat{V}_2^e \quad \cdots \quad \Delta\hat{x}_p \quad \Delta\hat{V}_p^e\ ]^T$$

$$\Lambda = \begin{bmatrix} Q_1 & R_1 & & & & & & \\ S_1 & Y_1 & & & & & & \\ Q_{21} & R_{21} & Q_2 & R_2 & & & & \\ 0 & 0 & S_2 & Y_2 & & & & \\ & & \ddots & & \ddots & & & \\ & & & \ddots & & \ddots & & \\ & & & & Q_{pp-1} & R_{pp-1} & Q_p & R_p \\ & & & & 0 & 0 & S_p & Y_p \end{bmatrix}$$

where p is the number of integration steps taken simultaneously. Matrix Λ is asymmetric and very sparse. Assuming that each one of the m synchronous machines and its controllers is modeled by κ differential equations and that the network has n nodes, the dimension of (10) is $p.(\kappa m + 2n)$.

The initialization of the variables in each integration window is performed using a simplified model of the system and an approximated solution by Taylor Series Expansion [22]. This method can be implemented efficiently on parallel computers.

The calculation of the elements of the mismatch functions $\hat{F}_i$ and $\hat{G}_i$ $(i = 1,2,...,p)$, the elements of the Jacobian matrix, and the updating of the components of $\hat{x}_i$ and $\hat{V}_i^e$ $(i = 1,2,...,p)$ can be performed with perfect parallelism. Therefore, an obvious mapping of these operations onto the multiprocessor system is to allocate the equations corresponding to each integration step to a single processor. The decomposition of (10) should be done in the same way. The SIS parallel in time approach can be summarized in the following algorithm, where Δt is the integration step, $\delta = [\hat{x}, \hat{V}^e]^t$, and $\xi = [\hat{F}, \hat{G}]^t$:

SIS Parallel in Time Algorithm

```
Obtain $x^0$ e $V^{e0}$ em $t = 0^-$
For $\gamma = 1,2,\ldots,\Gamma$ $(\gamma = \Delta t.p)$
    Obtain $\delta_i^0 = [\hat{x}_i^0, \hat{V}_i^{e0}]^T$; $i = 1,2,\ldots,p$
    For $k = 0,1,2,\ldots$
    For $i = 1,2,\ldots,p$
        Calculate $\xi_i = [\hat{F}_i^t, \hat{G}_i^t]^T$
        If $||\xi_i||_2^2 < \epsilon$, any $i$, then quit
        Calculate $\Lambda = -\partial\xi/\partial\delta$
        Solve $\xi^k = \Lambda^k.\Delta\delta^{k+1}$ (by a CG type method)
        Calculate $\delta_i^{k+1} = \delta_i^k + \Delta\delta_i^{k+1}$
    Endfor
    Endfor
Endfor
```

In this parallel algorithm, each one of the p tasks are assigned to a processor. The elements of the Jacobian matrix Λ are calculated using p processors, and the equation $\xi^k = \Lambda^k.\Delta\delta^{k+1}$ can be also efficiently solved on parallel computers using CG type methods. The advantages of this decomposition approach are:

- Perfect processing load balance.
- Low communication requirements. Except for the convergence test, each processor needs to communicate only with processors assigned to integration steps before and after itself.
- Each processor may be a cluster of q other processors which could be used to solve each subset of equations in parallel in the so called space and time parallelization.

Two CG methods suitable for asymmetric sets of linear equations were experimented in the implementation of the parallel solution scheme introduced in this subsection. These implementations are referred to as:

- *Method 3a*: it uses the Bi-Conjugate Gradient method (Bi-CG)in which two sequences of mutually orthogonal residuals are generated by simple relations similar to the one used in the CG method (see the Appendix and [17]).
- *Method 3b* : it uses a recently introduced more robust and efficient variant of the Bi-CG method called the Bi-CGSTAB (see the Appendix and [23]).

Both methods were implemented using preconditioning. The preconditioning matrix used is the following block-diagonal matrix:

$$M = -\,\text{blockdiag}\left[\; Q_1 \quad Y_1 \quad Q_2 \quad Y_2 \quad \cdots \quad Q_p \quad Y_p \;\right]$$

IV. Computational Experiment Results

This section contains a summary of the results of the computational experiments conducted to assess the performance of the methods described in this paper. A comprehensive set of results can be found in [19].

A. Parallel Computer and Test System

The parallel machine used in the experiments is an Intel iPSC/860 with 8 processing nodes connected in a hypercubic topology. The test system is a planned configuration for the year 1987 of the interconnected electrical power system of the Brazil South-Southeastern region. Some areas of the system are represented by network equivalents. The configuration used in the test has 88 generators, 616 buses and 995 branches. The models used to represent the machines and their controllers are the same as the ones described in [7].

B. Numerical Results

Parallel and sequential versions of the algorithms described in this paper were programmed and tested in one or more nodes of the parallel computer. Also, a conventional dynamic simulation program, using the AIS approach and the network solution by LU Factorization, was used in the experiment. This last program was run in a mainframe computer (IBM 3090/170J) and in one node of the iPSC/860 for comparative purposes. The cpu times for these runs were 20.0 s and 62.3 s, respectively.

1) AIS Approach: Tests in an actual parallel machine were limited to the case of eight processors which was the maximum number of available processors in the used parallel computer. Simulated parallel solution of the problem in 2, 4, 8, 16, and 32 processors were performed in one node of the parallel machine to assess the influence of a larger number of processors. Table 1 presents the values of speedups ($S_{A1} = T_0/T_P$ and $S_{A2} = T_S/T_P$) and efficiencies (E_{A1} and E_{A2}) obtained in the tests with Methods 1 and 2. In these formulae, T_0, T_P, and T_S are the cpu time for the sequential implementation of a conventional program using LU Factorization, the parallel implementation of Methods 1 or 2, and the sequential implementations of these methods, respectively. The efficiency is calculated dividing the corresponding speedup by the number of processors used in the simulation.

Figure 1 shows cpu and communication times required by Methods 1 and 2 in sequential execution (seq) and in actual parallel processing (par). Figure 2 presents a summary of the results obtained with Method 2, using the semiautomatic decomposition methodology and a trial and error method.

TABLE 1: SPEEDUPS AND EFFICIENCIES FOR THE AIS

Method	No. Proc.	T_P (s)	S_{A1}	E_{A1} %	S_{A2}	E_{A2} %
	2	38.7	1.61	80	1.95	98
1	4	26.7	2.33	58	2.85	71
	8	23.9	2.60	33	3.18	40
	2	37.6	1.66	83	1.90	95
2	4	26.0	2.40	60	2.82	71
	8	23.5	2.65	33	3.44	43

2) SIS Approach: Actual parallel implementation (par) and sequential processing (seq), including 1, 2, 4, and 8 integration steps, were performed in this case. Table 2 and Figure 3 present the speedups ($S_{S1} = T_S^p/T_P^p$ and $S_{S2} = T_S^1/T_P^p$), efficiencies (E_{S1} and E_{S2}), and cpu time requirements obtained in these tests, where T_S^p and T_P^p are the cpu time for the sequential and parallel implementations of Methods 3a or 3b, considering p integration steps simultaneously, and T_S^1 is the cpu time required by the sequential implementations of these methods considering only one integration step.

TABLE 2: SPEEDUPS AND EFFICIENCIES FOR THE SIS

Method	Proc./ Steps	T_S^p (s)	T_P^p (s)	S_{S1}	E_{S1} %	S_{S2}	E_{S2} %
	2	72.8	42.5	1.71	85	1.20	60
3a	4	105.0	33.6	3.12	78	1.52	38
	8	165.1	NC	—	–	—	–
	2	57.7	35.9	1.61	81	1.32	66
3b	4	79.7	28.1	2.83	71	1.68	42
	8	134.9	24.6	5.48	68	1.92	24

NC: no convergence

V. GENERAL REMARKS

This section contains some general conclusions regarding the computational experiment results and the experience in the implementation of the algorithms on an actual parallel computer.

1) Computation Time: Parallel processing has significantly reduced the computation time of the dynamic simulation. Moreover, cpu time requirements on the parallel machine are comparable to the one in a much more expensive mainframe computer (IBM 3090/170J).

2) Spatial Parallelization: Method 2 has consistently showed better performance than Method 1 although presenting higher communication requirements and more sensitivity to network decomposition. In both methods, a significant amount of the communication requirements is independent of the problem dimensions.

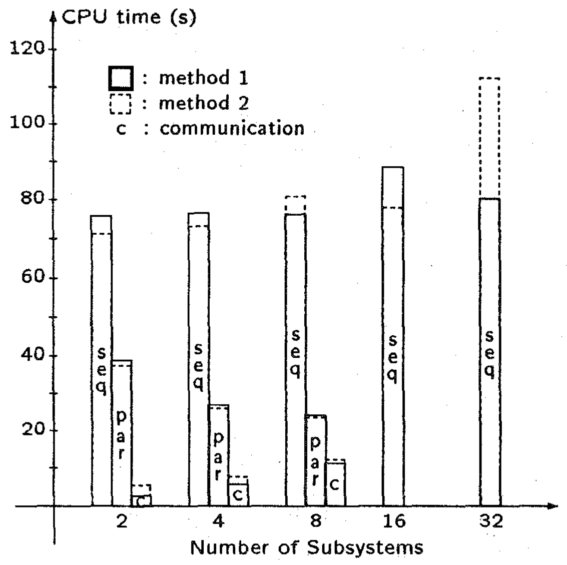

Fig. 1: CPU and Communication times for Methods 1 and 2

3) Space and Time Parallelization: This approach presented a considerable slowdown in the sequential implementation with the increase in the number of integration steps taken simultaneously (T_S^p in Table 2). There is evidence that this fact is directly connected to the used preconditioning technique and that more research in this area could improve the proposed method performance. The speedup is considerable in relation to the sequential version for the same number of integration steps (S_{S1}) which shows the effectiveness of the paralellization approach. Despite of the slowdown referred to above, there is still significant speed gain in the parallel version (speedup S_{S2}). The communication time remains practically constant with the increase in the number of processors. The Bi-CGSTAB method proved to be superior to the Bi-CG for this application.

4) CG Methods: The CG methods firmly establish themselves as alternatives for the solution of sets of linear equations originated from modeling electrical power networks. Except for a few cases in which the Bi-CG method fails to converge, the tested CG methods showed adequate robustness and accuracy. The best results with preconditioning in this work were obtained with the Truncated Series (Method 1) and the Incomplete LU Factorization (Methods 2, 3-a, and 3-b), in which the M matrix (see the Appendix) is made block-diagonal to allow the auxiliary system (see (16) in the Appendix) to be solved with perfect parallelism. Note that the decomposition represented in (9) is performed mainly to satisfy the preconditioning method requirements which has been shown to be more effective for weakly coupled subnetworks. In the SIS approach, the use of M as defined in section III.B is

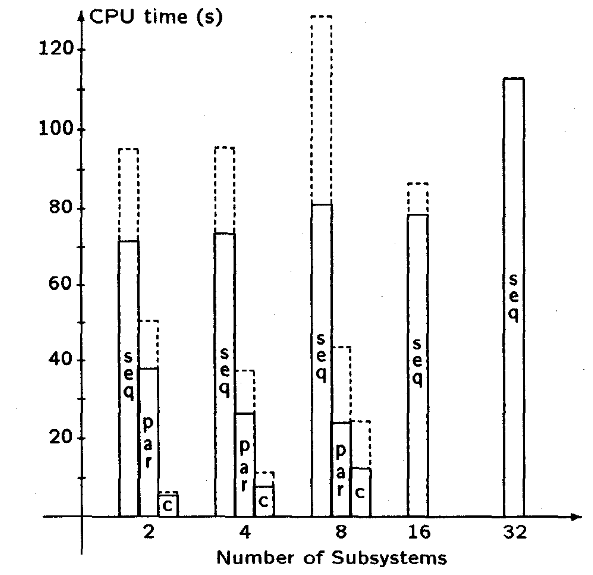

Fig. 2: CPU and Communication times for Method 2 with semiautomatic decomposition (—) and trial and error decomposition (...)

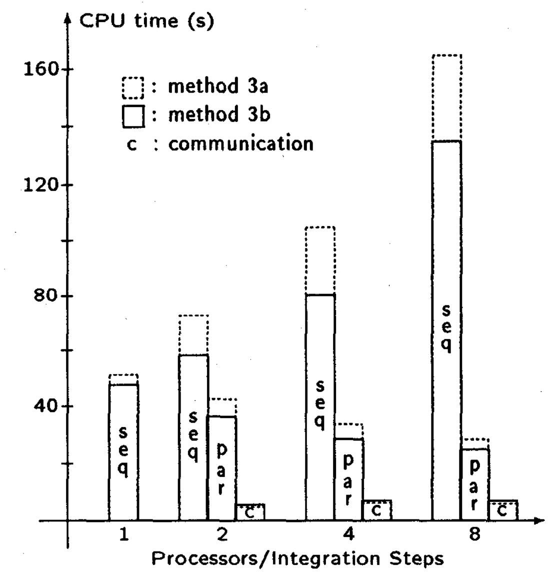

Fig. 3: CPU and Communication times for Methods 3a and 3b

equivalent to neglect the coupling between consecutive integration steps and the coupling between the synchronous machine and the electrical network. On the other hand, it may be observed that the effect of the preconditioning is only to reduce the residual in each iteration of the process which alters the rate of convergence but not the accuracy of the results. In this work, block-diagonal preconditioning matrices were used due to their advantage for parallelization. However, this field is still largely open to investigation and future research work may result in substantial improvements on the AIS and SIS parallel methods performance.

5) Efficiency and Speedup: The parallel computer used in the experiments has a relatively slow communication network and very fast node processors. This characteristic explains partially the relatively small speedup and efficiency achieved in some of the experiments. Other parallel machines already commercially available present more favorable communication/processing ratios. It is believed that in these machines, the proposed algorithms will be able to achieve higher levels of speedup and efficiency.

6) Decomposition Methodology: The system decomposition for the AIS obtained with the methodology described in [21], showed a consistently superior performance than the ones obtained by trial and error, apart from requiring less user experience and effort.

VI. Conclusions

This paper has described three methods for power system dynamic simulation on parallel computers. All methods use Conjugate Gradient techniques to solve sets of linear algebraic equations. Results of computational experiments in a commercially available parallel machine and an actual power system showed considerable reduction in the computation time. The Conjugate Gradient methods presented adequate robustness, accuracy, and computation speed establishing firmly as an alternative to direct methods in power system dynamic simulation.

References

[1] D.J. Tylavsky, A. Bose, et all., "Parallel Processing in Power Systems Computation", An IEEE Committee Report by a Task Force of the Computer and Analytical Methods Subcommittee of the Power Systems Engineering Committee, *IEEE Transactions on Power Systems*, vol. 7, no. 2, pp. 629-637, May 1992.

[2] "The Power of Parallelism", *IEEE Spectrum, Special Issue: Supercomputers*, vol. 29, no. 9, pp. 28-33, September 1992.

[3] W.L. Hatcher, F.M. Brasch, and J.E. Van Ness, "A Feasibility Study for the Solution of Transient Stability Problems by Multiprocessor Structures", *IEEE Transactions on Power Apparatus and Systems*, vol. 96, no. 6, pp. 1789-1797, Nov./Dec. 1977.

[4] F.M. Brasch, J.E. Van Ness, and S.C. Kang, "Simulation of a Multiprocessor Network for Power System Problems", *IEEE Transactions on Power Apparatus and Systems*, vol. 101, no. 2, pp. 295-301, 1982.

[5] J.S. Chai, N. Zhu, A. Bose, and D.J. Tylavsky, "Parallel Newton Type Methods for Power System Stability Analysis Using Local and Shared Memory Multiprocessors", *IEEE Transactions on Power Systems*, vol. 6, no. 4, pp. 1539-1545, November 1991.

[6] I.C. Decker, D.M. Falcão, and E. Kaszkurewicz, "An Efficient Parallel Method for Transient Stability Analysis", *Proceedings of the 10th Power Systems Computation Conference*, Graz, Austria , pp. 509-516, August 1990.

[7] I.C. Decker, D.M. Falcão, and E. Kaszkurewicz, "Parallel Implementation of a Power System Dynamic Simulation Methodology Using the Conjugate Gradient Method", *IEEE Transactions on Power Systems*, vol. 7, no. 1, pp. 458-465, February 1992.

[8] H. Taoka, I. Iyoda, H. Noguchi, N. Sato, and T. Nakazawa, "Real-Time Digital Simulator for Power System Analysis on a Hypercube Computer", *IEEE Transactions on Power Systems*, vol. 7, no. 1, pp. 1-10, February 1992.

[9] M.A. Pai, P.W. Sauer, and Y. Kulkarni, "Conjugate Gradient Approach to Parallel Processing in Dynamic Simulation of Power Systems", *Proceedings of the Automatic Control Conference*, pp. 1644-1647, 1992.

[10] J.S. Chai and A. Bose, "Bottlenecks in Parallel Algorithms for Power System Stability Analysis", *IEEE Transactions on Power Systems*, vol. 8, no. 1, pp. 9-15, February 1993.

[11] M.Ilić-Spong, M.L. Crow, and M.A. Pai, "Transient Stability Simulation by Waveform Relaxation Methods", *IEEE Transactions on Power Systems*, vol. 2, no. 4, pp. 943-952, November 1987.

[12] M.L. Crow and M. Ilić, "The Parallel Implementation of the Waveform Relaxation Method for Transient Stability Simulations", *IEEE Transactions on Power Systems*, vol. 5, no. 3, pp. 922-932, August 1990.

[13] F. Alvarado, "Parallel Solution of Transient Problems by Trapezoidal Integration", *IEEE Transactions on Power Apparatus and Systems*, vol. 98, no. 3, pp. 1080-1090, May/June 1979.

[14] M. LaScala, A. Bose, D.J. Tylavsky, and J.S. Chai, "A Highly Parallel Method for Transient Stability Analysis", *IEEE Transactions on Power Systems*, vol. 5, no. 4, pp. 1439-1446, November 1990.

[15] M. LaScala, M. Brucoli, F. Torelli, and M. Trovato, "A Gauss-Jacobi-Block Newton Method for Parallel Transient Stability Analysis", *IEEE Transactions on Power Systems*, vol. 5, no. 4, pp. 1168-1177, November 1990.

[16] J. M. Ortega, *Introduction to Parallel and Vector Solution of Linear Systems*, Plenum Press, New York, 1988.

[17] J.J. Dongarra, I.S. Duff, D.C. Sorensen, and H.A. van der Vorst, *Solving Linear Systems on Vector and Shared Memory Computers*, SIAM, 1991.

[18] F.D. Galiana, H. Javidi, and S. McFee, "On the Application of a Pre-conditioned Conjugate Gradient Algorithm to Power Network Analysis", *Proceedings of the 18th Power Industry Computer Applications Conference*, Scottsdale, AZ, May 1993.

[19] I.C. Decker, *"Algorithms for Power Systems Dynamics Simulation with Parallel Computers"*, Ph.D. Thesis, Federal University of Rio de Janeiro, Brazil, (in Portuguese), September 1993.

[20] B. Stott, "Power System Dynamic Response Calculations", *Proceedings of the IEEE*, vol. 67, no. 2, pp. 219-241, February 1979.

[21] M.H.M. Vale, D.M. Falcão, and E. Kaszkurewicz, "Electrical Power Network Decomposition for Parallel Computations", *Proceedings of the IEEE Symposium on Circuits and Systems*, San Diego, CA, pp. 2761-2764, May 1992.

[22] M. Ribbens-Pavella, B. Lemal, and W. Pirard, "On-Line Operation of Lyapunov Criterion for Transient Stability Studies", *Proceedings of the IFAC Symposium*, Melbourne, Australia, pp. 292-296, February 1977.

[23] H.A. van der Vorst, "Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems", *SIAM J. Sci. Stat. Comput.*, vol. 13, no. 2, pp. 631-644, March 1992.

Appendix

This appendix introduces the fundamental concepts of the Conjugate Gradient Type Methods used in the algorithms proposed in this paper. More details can be found in [16, 17, 19, 23].

The Conjugate Gradient type methods derive from the family of methods for the solution of linear equations sets named Krylov subspace methods or projected gradient methods [16, 17]. The most popular of them is the Conjugate Gradient (CG) method, applicable only to sets of equations with symmetric coefficient matrices [16, 17]. In the case of asymmetric matrices, other methods have been used like the Generalized Minimal Residual (GMRES) [17], the Biconjugate Gradient (Bi-CG) [17], the Conjugate Gradient-Squared (CGS) [17] and, more recently, the Bi-CGSTAB [23].

The methods used in this work were the CG for the symmetric case and the Bi-CG and Bi-CGSTAB for the asymmetric case. These methods are briefly reviewed in the following.

The CG Method - Basic Concepts

The solution of a set of linear algebraic equations, like the one obtained from (2) for a particular value of the vector $I(E,V)$ (see AIS approach), can be achieved by the unconstrained minimization of the quadratic function

$$Q(V) = \frac{1}{2}V^tYV - I^tV \tag{11}$$

The minimum of $Q(V)$ coincides with the solution of the linear system $I = YV$.

The CG method is an unconstrained optimization technique in which the search directions are *conjugate*. The CG algorithm can be formulated in general terms as

$$V^{k+1} = V^k - \alpha^k p^k, \quad k = 0,1,2,\ldots \tag{12}$$

where

$p^k =$ conjugate gradient direction at the k^{th} step
$\alpha^k =$ optimum step-size in direction p^k

and the vectors p^k satisfy the following orthogonality condition

$$(p^i)^T A\, p^j = 0, \quad i \neq j \tag{13}$$

In the absence of rounding errors, it can be shown that (12) converges to the solution of the linear system in a number of iterations not greater than the number of equations [16]. In practical applications, depending on the conditioning of Y, the iterative process may converge in much less iterations. The usual termination rule is determined by a tolerance in the norm of the residual vector

$$r^k = I - YV^k \tag{14}$$

Preconditioning

The convergence rate of the CG method is affected by the condition of Y. Therefore, this rate can be improved if the condition number of Y is made closer to 1. This can be achieved by preconditioning Y by the congruence transformation [16, 17]

$$\hat{Y} = SYS^t \tag{15}$$

where S is chosen such that $cond(\hat{Y}) \leq cond(Y)$.

The introduction of the preconditioning of Y in the CG method algorithm [16] is performed by the modification of the residual vector in certain steps of the algorithm as follows:

$$M\tilde{r} = r \tag{16}$$

where $M = (S^tS)^{-1}$ and $\tilde{r}$ is the modified residual vector. Since (16) has to be solved at every iteration, it is important that S, or equivalently M, should be chosen in such way that (16) is easy to solve.

PCGM Algorithm

A basic version of the Preconditioning Conjugate Gradient Method (PCGM) algorithm is given below:

```
Choose $V^0$
Compute $r^0 = I - YV^0$
Solve $M\tilde{r}^0 = r^0$
Set $p^0 = \tilde{r}^0$
For $k = 0, 1, \ldots$
    $\alpha^k = (\tilde{r}^k, r^k)/(p^k, Y.p^k)$
    $V^{k+1} = V^k - \alpha^k p^k$
    $r^{k+1} = r^k + \alpha^k Y.p^k$
    If $||r^{k+1}||_2^2 \leq \epsilon$, Then, Stop.
    Else,  Solve :
            $M\tilde{r}^{k+1} = r^{k+1}$
            $\beta^k = (\tilde{r}^{k+1}, r^{k+1})/(\tilde{r}^k, r^k)$
            $p^{k+1} = \tilde{r}^{k+1} + \beta^k p^k$
    Endif
Endfor
```

The main computational load of the above algorithm is concentrated in the calculation of inner products and matrix-vector multiplications.

The Preconditioning Bi-CG Method

For asymmetric coefficient matrices, in general, it is not possible to obtain orthogonality between vectors in the p direction and the residual vector r using a simple recurrence relation like in the symmetric case[17]. A proposed alternative [17] is to generate two sequences of residual vectors r and $\bar{r}$ and directions p and $\bar{p}$ satisfying the mutual orthogonality condition

$$\left.\begin{array}{rcl} (r^i, \bar{r}^j) & = & 0 \\ (\bar{p}^i, \Lambda.p^j) & = & 0 \end{array}\right. \quad \text{for } i \neq j \tag{17}$$

where Λ is the coefficient matrix.

The pre-conditioned Bi-CG algorithm, applied to the Jacobian equation defined in (10), is given below (M is the pre-conditioning matrix):

```
Choose $\Delta\delta^0$
Calculate $r^0 = \xi - \Lambda\Delta\delta^0$
Solve $M\tilde{r}^0 = r^0$
Set $p^0 = \bar{p}^0 = \bar{r}^0 = \tilde{r}^0$
For $k = 0, 1, \ldots$
    $\alpha^k = (\tilde{r}^k, \bar{r}^k)/(\bar{p}^k, M^{-1}\Lambda.p^k)$
    $\Delta\delta^{k+1} = \Delta\delta^k + \alpha^k p^k$
    $\tilde{r}^{k+1} = \tilde{r}^k - \alpha^k M^{-1}\Lambda.p^k$
    $\bar{r}^{k+1} = \bar{r}^k - \alpha^k \Lambda^t (M^{-1})^t.\bar{p}^k$
    If $||\tilde{r}^{k+1}||_2^2 \leq \epsilon$
            Calculate $r^{k+1} = M\tilde{r}^{k+1}$
            If $||r^{k+1}||_2^2 \leq \epsilon$ , Then Stop
            Endif
    Else,   Calculate :
            $\beta^k = (\bar{r}^{k+1}, \tilde{r}^{k+1})/(\bar{r}^k, \tilde{r}^k)$
            $p^{k+1} = \tilde{r}^{k+1} + \beta^k p^k$
            $\bar{p}^{k+1} = \bar{r}^{k+1} + \beta^k \bar{p}^k$
    Endif
Endfor
```

The main disadvantage of the Bi-CG method is the need of two matrix-vector multiplications for each iteration ($\Lambda.p^k$ and $\Lambda^T.\bar{p}^k$, assuming $M = I$).

The Preconditioning Bi-CGSTAB Method

The Bi-CGSTAB method was recently introduced in [23] and is a variant of the Bi-CG method for asymmetric systems. This method presents better characteristics of robustness and computational efficiency and do not require to operate with the transpose of the coefficient matrix (Λ^T in this case). The Bi-CGSTAB algorithm applied to equation (10), using M as preconditioning matrix, is given below:

```
Choose $\Delta\delta^0$
Calculate $r^0 = \xi - \Lambda\Delta\delta^0$
Set $\bar{r}^0 = r^0$ e $p^0 = r^0$
For $k = 0, 1, 2, \ldots$
    $M\hat{p} = p^k$
    $\alpha^k = (\bar{r}^0, r^k)/(\bar{r}^0, \Lambda.\hat{p})$
    $q = r^k - \alpha^k \Lambda.\hat{p}$
    $M\hat{q} = q$
    $\omega^k = [\Lambda.\hat{q}]^t.[q]/[\Lambda.\hat{q}]^t.[\Lambda.\hat{q}]$
    $\Delta\delta^{k+1} = \Delta\delta^k + \alpha^k \hat{p} + \omega^k \hat{q}$
    $r^{k+1} = q - \omega^k \Lambda\hat{q}$
    If $||r^{k+1}||_2^2 \leq \epsilon$ , Then Stop
    Else,   Calculate :
            $\beta^k = (\bar{r}^0, r^{k+1})/(\bar{r}^0, r^k) .\alpha^k/\omega^k$
            $p^{k+1} = r^{k+1} + \beta^k [p^k - \omega^k \Lambda.\hat{p}]$
    Endif
Endfor
```

Note that as in the CG method, most of the computational effort in the Bi-CG and Bi-CGSTAB methods is concentrated in the computation of inner products and matrix-vector multiplications. These vector and matrices can be trivially decomposed and, consequently, these operations can be performed efficiently on parallel and vector computers.

ACKNOWLEDGMENTS

This research project was supported by the Brazilian Government agencies FINEP and CNPq (RHAE). The first author wishes to thank the Federal University of Santa Catarina (Brazil) for the financial support and leave of absence. The authors thank M.H.M. Vale for her assistance in obtaining network decompositions to test the methods based on the AIS.

BIOGRAPHIES

Ildemar C. Decker received his B.Sc. from the Catholic University of Pelotas, RS., Brazil. He obtained his M.Sc. (1984) and D.Sc. (1993) degrees in Electrical Engineering from the Federal Universities of Santa Catarina and Rio de Janeiro, Brazil, respectively. From 1980 to 1985 he worked in the Federal University of Santa Maria,RS. Since 1985, he has been Associate Professor of the Federal University of Santa Catarina, in the Electrical Engineering Department. His general research interest is in the area of computer methods for power systems analysis and control and high performance scientific computing.

Djalma M. Falcão (M'75) obtained his B.Sc. (1971) and M.Sc. (1974) from the Federal Universities of Paraná and Rio de Janeiro, Brazil, respectively, and the Ph.D. (1981) from UMIST, England. He is currently a professor of Electrical Engineering at the Federal University of Rio de Janeiro. From 1991 to 1993 he was in sabbatical leave at UC Berkeley. His general research interest is in the area of computer methods for power system simulation and control.

Eugenius Kaszkurewicz received his B.Sc. from the Instituto Tecnologico de Aeronautica(ITA) in 1970. He obtained his M.Sc. and D.Sc. degrees in Systems and Control at the Federal University of Rio de Janeiro in 1972 and 1981, where he joined the Electrical Engineering Department in 1974. From 1985 to 1987 he was a visiting Faculty at the EECS Dept. of Santa Clara University, California. His research interests are in Control and Parallel Computation in Large Scale Systems.

Discussion

H. Dağ and F. L. Alvarado (University of Wisconsin-Madison) The authors are to be commended for an interesting paper describing new parallel linear equation solvers. The increasing dimension and necessary amount of detail to mode power system will require faster (possibly parallel) solvers. Traditional Gaussian elimination based direct solvers are not well suited to parallel processing. Computation time increases faster than linearly with system dimension for direct solvers [D1]. As the dimension increases, direct methods become increasingly impractical [D2]. Iterative methods can be used as alternative solution techniques [D3].

The discussers would like to have authors' comments on the following issues:

- The conjugate gradient (CG) method is guaranteed to converge for symmetric positive definite (spd) set of linear equations with full arithmetic precision. The authors use CG method in Method 1 for equation (7) and in Method 2 for equation (9), both of which are symmetric. Are both of the equations also positive definite in order for CG to be successfully applied?

- When the CG method is used with a preconditioner, the preconditioner matrix is also required to be positive definite. The authors use ILU as preconditioner. However, ILU preconditioners are not guaranteed to be positive definite, even for spd matrices other than M-matrices and H-matrices. Do the authors use a special technique to guarantee the positive definiteness of the ILU? Have the authors had any nonconverged cases when they use ILU as a preconditioner?

- The paper offers iterative methods as alternatives to direct methods for the solution of linear equations to increase parallelism, but it uses ILU as preconditioner, which is inherently sequential. In other words, the use of ILU limits the parallelism associated with the CG method [D4]. Could the author comment on the parallel implementation of CG and its performance with respect to system size?

- In Method 3, the authors use Bi-CG and Bi-CGSTAB to solve the set of linear equations. The Bi-CGSTAB is an improvement over Bi-CG, and the latter can always be preferred to the former. However, both methods can break-down. Division by zero may occur and Bi-CG can exhibit irregular convergence behavior [D5]. Would it not be advantageous to use a better established asymmetric linear equation solver such as the generalized minimum residual method (GMRES) [D5] instead, or a method such as quasi-minimum residual (QMR) [D5] that overcomes the pitfalls of the above methods?

REFERENCES

[D1] F. L. Alvarado, *Computational Complexity in Power Systems*, IEEE Transactions on Power Apparatus and Systems, Vol.95, No.4, pp. 1028–1037,1976.

[D2] H. Dağ and F. L. Alvarado, *Direct Methods Versus GMRES and PCG For Power Flow Problems*, in the proceedings of NAPS,pp.274-278, october 5–9, 1993, Washington, DC.

[D3] F. L. Alvarado, H. Dağ and M. ten Bruggencate, *Block-Bordered Diagonalization and parallel iterative solvers*, in proceedings of Colorado Mountain Conference on Iterative Methods, April 5–9, 1994, Breckenridge,CO.

[D4] F. L. Alvarado and H. Dağ, *Incomplete Partitioned Inverse Preconditioners*, Submitted to Parallel Computing, June 1994.

[D5] R. Freund and N. Nachtigal, *QMR: A quasi-minimal residual method for non-hermitian linear systems*, Numerische Mathematik, Vol. 60, pp. 315–339, 1991.

[D5] Y. Saad and M. Schultz, *GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems*, SIAM Journal on Scientific and Statistical Computing, Vol.7, pp. 856-869, 1986.

I.C. Decker, D.M. Falcão, and E. Kaszkurewicz. We thank Mr. H. Dağ and Prof. F.L. Alvarado for the interest in our paper and the relevant questions raised on their discussion. We certainly agree with the discussants in their claim that iterative methods have an important role in the solution of large sets of equations arising from power system modeling and this is particularly true in a parallel computer environment. In fact, we have been defending this point since the publication in 1990 of our first results regarding application of conjugate gradient methods to power system dynamic simulation (Reference [6] of the paper).

The coefficient matrices defined in equations (7) and (9) are not guaranteed to be always positive definite. However, this positive definiteness is only a sufficient condition for the convergence of the CG method. For the test system reported in the paper, the matrices present a few non-positive eigenvalues. Despite this fact, in all the simulations, the CG method presented no convergence problem.

The preconditioner that was used is different from the usual ILU preconditioner which is inherently sequential, as pointed out by the discussants. The preconditioning effect is obtained through the solution of equation (16) where M is an approximation of the original system coefficient matrix. To obtain a perfect parallelism in the solution of this auxiliary system, in the case of the equations defined in (9), matrix M is formed by the diagonal blocks of Y (Y_{ii}, $i = 1, 2, \ldots, q$). This is equivalent to decompose the electrical network in weakly coupled subnetworks and neglect the weak couplings between these subnetworks (see section III, Method 2). Thus, the *LU* factors of *M* are also block-diagonal and, therefore, the solution of equation (16) can be performed by solving *q* perfectly decoupled sets of equations. This approach has proved effective in terms of algorithm convergence and parallel implementation. Moreover, its performance improves with the system dimension.

The Bi-CG and Bi-CGSTAB methods where chosen due to their implementation simplicity when compared to other methods like the QMR and GMRES. Also, in reference [23] of the paper, it is pointed out that the Bi-CGSTAB presents convergence characteristics superior to the CG-S (Conjugate Gradient-Squared) methods. However, the authors believe that the methods referred to by the discussants deserve further investigation regarding their application to power systems.

Manuscript received October 30, 1995.