Enhancing the Efficiency of Reduction of Large RC networks By Pole Analysis via Congruence Transformations*

Zheng Hui, Zhang Wenjun, Tian Lilin, Yang Zhilian
Institute of Microelectronics, Tsinghua University
Beijing 100084, P R. China
Tel: 86-10-62784684 Fax: 86-l0-62771130
E-mail: zh@dns.ime.tsinghua.edu.cn

Abstract--- Among the RC reduction algorithms, the algorithm of PACT (Pole Analysis via Congruence Transformations) [41 has been proved to have several advantages. However, the original implementation of the algorithm destroys the sparsity of the internal capacitance matrix. Consequently, the LASO process [4], used in the computation of the dominant eigenvalues and eigenvectors, becomes very time-consuming. Therefore, the efficiency of the algorithm needs to be improved.

In this paper, a new method to implement the PACT algorithm is presented. In order to maintain the sparsity of the matrices, we use a special Lanczos algorithm to directly compute the eigenvalues and eigenvectors by solving a large sparse symmetric generalized eigenvalue problem. At the same time, this approach can avoid some matrix multiplication to speed up the reduction process. We have constructed a RC reduction tool with the new implementation method. The application of the tools to several RC networks has shown that this tool greatly outperforms the original implementation.

I. Introduction

With the current trends of feature size shrinking, signal speed enhancement and mixed-signal design, signal integrity has become an important factor determining the complexity of VLSI design. The signal integrity problem has two sources： propagation delay and crosstalk of interconnect and substrate coupling noise. To solve the problem, we should find efficient methods to analyze the behavior of interconnect and substrate. At present, interconnects and substrate are niainly modeled as RC networks which can be extracted from the layout. Since the parasitic RC networks are extremely large, it is very impractical to use the SPICE-like simulator (based on numerical integration) to calculate the waveform of the extracted circuit directly. Therefore, the RC networks should be reduced before simulation.

The algorithm AWE [l] (Asymptotic Waveform Evaluation) and PVL [2] (Padé Via Lanczos) have emerged as efficient methods for analysis and reduction of large linear networks. They are based on a technique called Padé approximation mathematically. However, they suffer from several fundamental shortcomings. First, it is difficult to predict the order of the reduced models. Secondly, the reduction procedure is carried out in frequency-domain. The reduced models can not been directly incorporated into a circuit simulator. Extra processing, such as convolution, is needed.

Very recently, a RC network reduction algorithm called PACT (Pole Analysis via Congruence Transformations) was presented in [4]. It has been proved to have several advantages. Instead of matching the moments explicitly in AWE, this algorithm handles the matrices directly while preserving the moments implicitly. The stability of the algorithm is guaranteed by preserving the passivity of the network. Moreover, The reduced model can be easily incorporated into SPICE-like simulator. However, in the implementation of the algorithm, the sparsity of the internal capacitance matrix is destroyed after the multiplication of matrices. Consequently, the LASO process[4], used in the computation of the dominant eigenvalues and eigenvectors, becomes very time-consuming. Therefore, the efficiency of the algorithm needs to be improved.

In this paper, we present a new method to implement the PACT algorithm. In order to maintain the sparsity of the matrices, we use a special Lanczos algorithm to directly compute the eigenvalues and eigenvectors by solving a large sparse symmetric generalized eigenproblem. We have constructed a new RC

* This work was supported by National Education Department under the contract 96-738-01-01-06.

reduction tool with the new implementation method. In section II, the original implementation of PACT is. briefly described and the memory requirement and computational complexity are analyzed. In section III, our new implementation of PACT and its advantages are presented. In section IV, We apply the different Implementations to several RC networks. The experimental results show that the reduction ratio of RC elements can be up to 10% and our implementation excels the original implementation in both memory requirement and computational complexity.

II. Original Implementation of PACT

The detailed description of PACT algorithm refers to [4]. Here, we list the critical steps, that are related to the efficiency of the original implementation, as follows.

PACT uses two congruence transformations during the reduction process. The first is congruence transformation based on Cholesky decomposition.

Since the matrix D is symmetric positive definite, it has the following Cholesky decomposition $D = LL^T$, where L is a lower triangular matrix.

Let
$$V = \begin{bmatrix} I & 0 \\ -X & L^{-T} \end{bmatrix} \tag{1}$$

We apply the following congruence transformations to the matrix G and the matrix C :

$$G' = V^T G V = \begin{bmatrix} A - Q^T X & 0 \\ 0 & L^{-1} D L^{-T} \end{bmatrix} = \begin{bmatrix} A' & 0 \\ 0 & I \end{bmatrix} \tag{2}$$

$$C' = V^T C V = \begin{bmatrix} B - P^T X - X^T R & P^T L^{-T} \\ L^{-1} P & L^{-1} E L^{-T} \end{bmatrix} = \begin{bmatrix} B' & R'^T \\ R' & E' \end{bmatrix} \tag{3}$$

where $X = D^{-1}Q, \quad P = R - EX$.

Another congruence transformation is based on pole analysis. Since the matrix E' is still a symmetric matrix, it can be decomposed as $E' = U \Lambda U^T$, $\Lambda = diag(e_1, \ldots, e_n)$, where the diagonal elements of Λ are the eigenvalues of E'. U is an orthogonal matrix, whose column vectors are the relevant eigenvectors.

Let
$$Z = \begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix} \tag{4}$$

We apply the following congruence transformations to the matrix G' and the matrix C' :

$$G'' = Z^T G' Z = \begin{bmatrix} A' & 0 \\ 0 & I \end{bmatrix} \tag{5}$$

$$C'' = Z^T C' Z = \begin{bmatrix} B' & R''^T \\ R'' & \Lambda \end{bmatrix} \tag{6}$$

where $R'' = U^T R'$ (7)

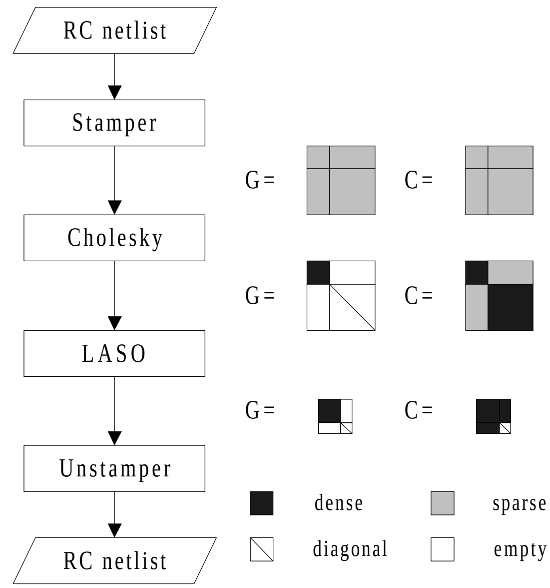

Fig.1. Flowchart of original implementation. The graphs on the right: the conductance and capacitance matrices after each steps.

The original implementation [4] is shown in Fig.1. First, the RC netlist is stamped into and stored as sparse matrices G and C. In the step of "Cholesky", internal conductance matrix D is factorized by a sparse Cholesky decomposition method. Then, calculation of G' and C' is performed according to (2) and (3). The matrix D^{-1}, the inversion of sparse matrix D, is generally a very dense matrix. Similarly, L^{-1} and L^{-T} are dense matrices. Consequently, E is transformed into a dense matrix E'. As we see below, this leads to the low efficiency of calculation of the eigenvalues and eigenvectors.

In the step of "LASO", two method of implementing LASO should be considered in terms of the scale of internal matrices. LASO (Lanczos Algorithm with Selective Orthogonalization)[5] uses a block Lanczos process. The time complexity of each iteration of LASO is determined by the multiplication of a fixed-sized vectors and the matrix whose eigenvalues to be solved. When n is small, we can compute E' and store it as a dense matrix. If we assume that the number of iteration is irrelevant to n , the total time complexity of LASO is O(n^2). We call this kind of implementation D-LASO, which means Direct LASO. When n is large, to save memory, we can not compute E' directly before the LASO process. During the LASO process, the matrix multiplication is accomplished by calculating a column of E' at a time so that only n elements need to be stored. However, we have to compute E' in each iteration. The time complexity of each iteration is O($n^3 n^{1\sim1.5}$). We call this kind of implementation MS-

LASO, which means Memory-Saving LASO.

As shown in Table II, D-LASO is highly memory consuming while MS-LASO is extremely time-consuming, which makes LASO implementation unsuitable for RC network with internal nodes greater than 2000. In next section, we propose a new implementation, which is efficient in both memory requirement and time complexity.

III. New Implementation of PACT

For decreasing memory and time, the key point is to preserve matrices D and E as sparse matrices. Fortunately, we find a method to calculating the eigenvalues and eigenvectors without transforming D and E. That is to solve the generalized symmetric eigenvalue problem as follows:

$$\det[E-\lambda D]=0 \tag{8}$$

Given two $n\times n$ matrices D and E, we can define the generalized eigenvalue problem as follows. Determine scalar λ and nonzero vectors x such that

$$Ex=\lambda Dx \tag{9}$$

Here D is positive definite matrix. The Cholesky factorization $D=LL^T$ is available. It is easy to deduce that this problem is equivalent to the real symmetric problem:

$$L^{-1}EL^{-T}y=\lambda y \text{ where } y=L^T x \tag{10}$$

The Lanczos recursion for the generalized eigenvalue problem is given by following equations:

$$\beta_{i+1}Dv_{i+1}=Ev_i-\alpha_i Dv_i-\beta_i Dv_{i-1} \tag{11}$$

$$\alpha_i=v_i^T(Ev_i-Dv_{i-1}) \tag{12}$$

$$|\beta_{i+1}|=\left\|L^{-1}(Ev_i-\alpha_i v_i-\beta_i v_{i-1})\right\| \tag{13}$$

During this Lanczos process, the sparsity of matrices D and E is unchanged. Therefore, we save. the memory that is needed to contain the dense matrix. Moreover, the time complexity of each iteration only is O($n^{1\sim1.5}$), as determined by the multiplication of sparse matrices and vectors.

If we get the eigenvectors V of (10), the eigenvectors of E' is determined by

$$U=L^T V \tag{14}$$

The (7) is then changed into

$$R''=U^T R'=V^T LL^{-1}P=V^T P \tag{15}$$

Thus, the matrix computation in (7) is reduced.

Using the above new implementation, we have constructed a RC network reduction CAD tool called RCRED in C language. The flowchart is shown in Fig.2. First, we use a software package called LANZ[6] to solve the generalized eigenvalue problem. We can get the desired eigenvalues and eigenvectors. In the process of "Cholesky", we first compute the Cholesky factorization of D. Then, matrices A', B' and R'' are calculated according to (2), (3) and (15). The computational complexity is the same as the original implementation.

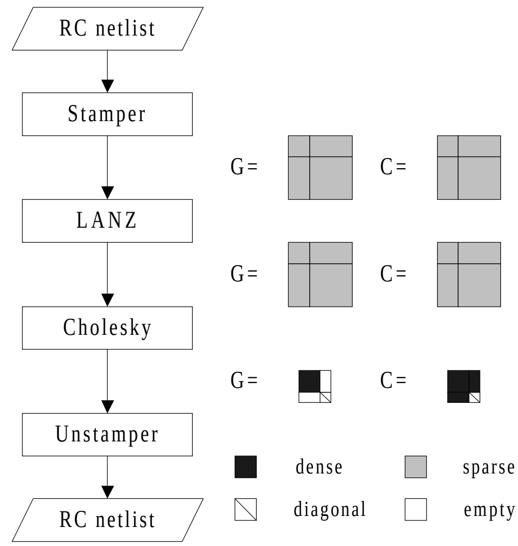

Fig.2. Flowchart of new implementation. The graphs on the right: the conductance and capacitance matrices after each steps.

IV. Examples and Comparisons

All examples presented in this section are executed on a Sun SPARC-20 workstation. First, we apply RCRED to a large RC tree with 25 external ports and 4687 internal nodes, which is extracted from a clock distribution network. In Table I, which shows the reduction and simulation statistics, we can find that the reduction ratio of internal nodes is less than 0.l% and the reduction ratio of elements is less than 10%. Consequently, HSPICE simulation of the reduced network is more than ten times faster than that of the original RC network. Fig.3 illustrates the comparison of HSPICE simulations of reduced and non-reduced RC networks. It is clear that RCRED greatly reduces the time and memory consumption of HSPICE simulation and simultaneously guarantees the accuracy of the behavior of the RC network.

Table I

Simulation circuits	Total nodes	R's	C's	HSPICE simulation	
				Time(s)	Mem.(Mb)
Not reduced	4712	4711	4712	109.7	4.1
Reduced	29	192	419	7.7	0.5

We then apply the three kinds of implementation mentioned above to three RC network with different numbers of external ports and internal nodes. The results of time consumption and memory requirement are shown in Table II. MS-LASO can not handle RC network with more 1000 internal nodes for its unacceptable long execution time. D-LASO consumes so much memory that it can only deal with RC network with no more than 2000 internal nodes. Our implementation, LANZ, excels the other two implementations in both memory requirement and time complexity. RC network with nodes greater than 10,000 can be processed by RCRED in a few minutes and with reasonable memory.

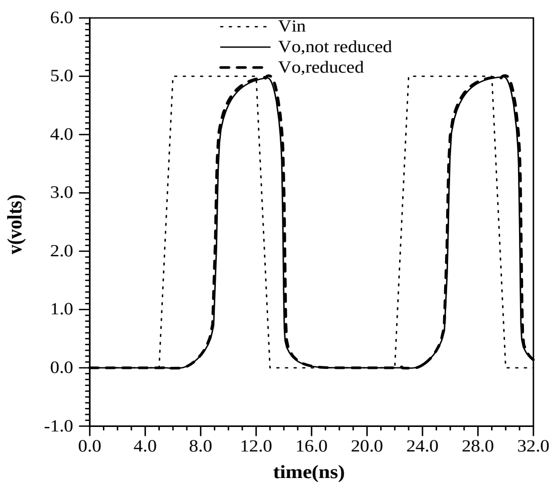

Fig.3 Comparison of HSPICE simulations

Table II

Examples		Exter: 13 Inter: 547	Exter: 47 Inter: 3019	Exter: 547 Inter: 12282
MS-LASO	Time(s)	3343.9	---	---
	Mem.(Mb)	0.4	---	---
D-LASO	Time(s)	7.2	1012.8	---
	Mem.(Mb)	4.0	158.3	---
LANZ	Time(s)	0.6	15.6	423.3
	Mem.(Mb)	0.7	13.2	50.4

V. Conclusion

A new implementation of PACT has been presented which utilize a special Lanczos process for generalized symmetric eigenproblem. This implementation preserves the sparsity of the matrices.. Consequently, memory requirement and time complexity are greatly reduced. Several examples have shown that it greatly outperform the original implementations with much higher time and memory efficiency.

References

[1] Pillage, Lawrence T., Rohrer, Ronald A., “Asymptotic waveform evaluation for timing analysis," IEEE Trans. on CAD, v 9, n 4, Apr 1990, p 352-366

[2] Feldmann, Peter, Freund, Roland W, “Efficient linear circuit analysis by Padé approximation via the Lanczos process," IEEE Trans. on CAD, v 14, n 5, May 1995, p 639-649

[3] Kevin J. Kerns, Ivan I. Wemple and Andrew T. Yang, “Stable and Efficient Reduction of Substrate Model Networks Using Congruence Transforms,” ICCAD 95, p 207-214

[4] Kevin J.Kerns and Andrew T. Yang, “Stable and efficient Reduction of Large, Multiport RC Networks by Pole Analysis via Congruence Transformations", IEEE/ACM DAC'96, pp. 280-285

[5] B.N. Parlett and D. S. Scott, "The Lanczos algorithm with selective orthogonalization," Mathematics of Computation, v 33, pp. 217-238,Jan. 1979

[6] Jones, M. T., and Patrick, M. L., “The Use of Lanczos`s Method to Solve the Large Generalized Symmetric Definite Eigenvalue Problem," Technical Report 89-67, Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA, l989.