Fast Capacitance Extraction Using Inexact Factorization

Shu Yan, Vivek Sarin, and Weiping Shi

Abstract — Most capacitance extraction algorithms based on Boundary Element Method (BEM) use iterative solvers, which is favorable for solving large systems. Different from the common practice, we present an approach that solves a small system for capacitance using the direct solver. Our study is based on the sparse formulation proposed in [8]. With proper ordering of the rows and columns, the sparse system can be approximated by its inexact factorization. Furthermore, with the proper ordering, the part of the solution vector, which contributes to capacitance, can be solved using the sub-matrix of the inexact factors. The dimension of the sub-matrix is $O(m)$, where m is the number of conductors. To our knowledge, this is the first BEM style method to solve capacitance extraction problem without using iterative solver. Experimental results show that the new algorithm is up to 100 times faster than FastCap [4] and is also much faster than the method in [8] (we call it PHiCap). The error of the new method with respect to FastCap is within 2%.

I. Introduction

Capacitance extraction is an important problem that has been extensively studied. The capacitance of an m-conductor geometry is summarized by an $m \times m$ capacitance matrix C. To determine the j-th column of the capacitance matrix, we compute the surface charges produced on each conductor by raising conductor j to unit potential while grounding the other conductors. Then C_{ij} is numerically equal to the charge on conductor i. This procedure is repeated m times to compute all the columns of **C**.

Many capacitance extraction algorithms are based on BEM. FastCap [4], HiCap [6], and other algorithms [1] are accelerated with Fast Multipole Method (FMM) [2]. The pFFT algorithm [5] and IES^3 algorithm [3] are accelerated using Fast Fourier Transform and singular value decomposition respectively. These methods split the conductor surfaces into small panels and formulate the problem using a linear system

$$\mathbf{P}\mathbf{q} = \mathbf{v}, \tag{1}$$

where $\mathbf{q} \in \mathbb{R}^n$ is the vector of unknown panel charges, $\mathbf{v} \in \mathbb{R}^n$ is the vector of known panel potentials, $\mathbf{P} \in \mathbb{R}^{n\times n}$ is the potential coefficient matrix and n is the number of panels. The linear system is dense and iterative methods are used for solving it.

In [8], we proposed a linear transformation which transforms the dense linear system to the equivalent sparse linear system, which is solved using preconditioned iterative methods. Experimental results show the incomplete LU or incomplete Cholesky factorizations are very efficient preconditioners. Inspired by this observation, instead of using the inexact factorizations as preconditioners for iterative solver, in this paper, we approximate the sparse linear system using its inexact factorization and solve the approximate system using direct method. The accuracy of the approximate system can be improved with proper row/column ordering. In addition, with proper row/column

This research was supported by the NSF grants -0098329, CCR-0113668, EIA-0223785, and ATP grant 512-0266-2001.

Shu Yan, Department of Electrical Engineering, Texas A&M University, College Station, TX 77843, Tel: 979-845-9578, Fax: 979-845-2630, shu@ee.tamu.edu

Vivek Sarin, Department of Computer Science, Texas A&M University, College Station, TX 77843, Tel: 979-458-2214, Fax: 979-847-9578, sarin@cs.tamu.edu

Weiping Shi, Department of Electrical Engineering, Texas A&M University, College Station, TX 77843, Tel: 979-458-0093, Fax: 979-845-2630, wshi@ee.tamu.edu

ordering, the capacitance matrix can be computed by solving a small system of size $O(m)$ obtained from the inexact factors.

II. PRELIMINARIES

The HiCap method [6] constructs a hierarchical data structure that contains the potential coefficient matrix $\mathbf{P}$, which is a dense matrix with $O(n)$ block entries. Fig. 1 shows an example of the hierarchical data structure. The panels are stored as nodes in the tree, and the block coefficient entries are stored as links between the nodes. Each tree represents the partition of a conductor surface. The root node represents the conductor surface. Each non-leaf node represents a panel that is further subdivided into two child panels. Each leaf node represents a panel that is not subdivided further. The union of all the leaf nodes covers the conductor surfaces completely. The rows and columns of $\mathbf{P}$, the entries of $\mathbf{q}$ and $\mathbf{v}$ correspond leaf nodes.

Based on the hierarchical data structure, [8] constructs the transformation $\mathbf{W}$, which convert the dense linear system (1) to the equivalent sparse system

$$\tilde{\mathbf{P}}\tilde{\mathbf{q}} = \tilde{\mathbf{v}}, \tag{2}$$

where $\tilde{\mathbf{q}} = \mathbf{W}\mathbf{q}$, $\tilde{\mathbf{v}} = \mathbf{W}^{-\mathbf{T}}\mathbf{v}$ and $\mathbf{P} = \mathbf{W}^{\mathbf{T}}\tilde{\mathbf{P}}\mathbf{W}$. The entries of $\tilde{\mathbf{q}}$ and $\tilde{\mathbf{v}}$ correspond to the root nodes and the right child nodes. Matrix $\tilde{\mathbf{P}}$ is sparse with the number of nonzeros comparable to the number of block entries in $\mathbf{P}$. In [8], the sparse system is solved using iterative methods, with preconditioners constructed from incomplete LU or incomplete Cholesky factorizations.

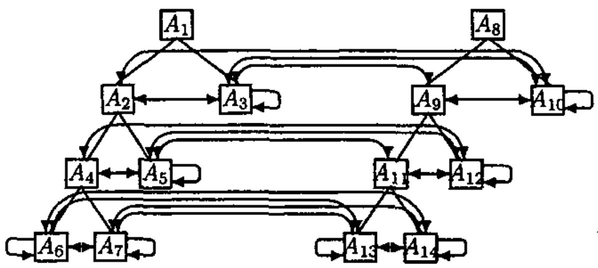

Fig. 1. The hierarchical data structure and potential coefficients.

Since the rows of $\mathbf{W}$ are mutually orthogonal, $\mathbf{W}\mathbf{W}^{\mathbf{T}}$ is a diagonal matrix. The sum of entries in each row of $\mathbf{W}$ are zero for all nodes except the root. As a result, the vector

$$\tilde{\mathbf{v}} = \mathbf{W}^{-\mathbf{T}}\mathbf{v} = (\mathbf{W}\mathbf{W}^{\mathbf{T}})^{-1}\mathbf{W}\mathbf{v}$$

has only nonzero entries in rows corresponding to root nodes of the conductor surfaces at unit potential. The rows of $\mathbf{W}$ corresponding to root nodes have identical nonzero entries which depend on the height. Thus, a root node entry in $\tilde{\mathbf{q}}$ is the sum of all leaf panel charges in that tree, scaled by a factor which is decided by the height. Capacitance is computed from those root entries.

III. NEW ALGORITHM

Let $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$ be the inexact lower and upper triangular factors of $\tilde{\mathbf{P}}$ obtained from incomplete LU or Cholesky factorization. From [8], with no more than 4 iterations, the norm residual converges to 1% using $\hat{\mathbf{L}}\hat{\mathbf{U}}$ as preconditioner, which means that $\hat{\mathbf{L}}\hat{\mathbf{U}}$ is a very good approximation of $\tilde{\mathbf{P}}$. The basic idea of the new algorithm is to solve

$$\hat{\mathbf{L}}\hat{\mathbf{U}}\hat{\mathbf{q}} = \tilde{\mathbf{v}} \tag{3}$$

for $\hat{\mathbf{q}}$, using forward and backward substitution. Here $\hat{\mathbf{q}}$ is an approximation of $\tilde{\mathbf{q}}$. Furthermore, with proper row/column ordering, only the small part of Equation (3), $\hat{\mathbf{L}}_{22}\hat{\mathbf{U}}_{22}\hat{\mathbf{q}}_2 = \tilde{\mathbf{v}}_2$, is necessary for solving capacitance. We explain this in detail later. The new algorithm is outlined below.

The New Algorithm

1) Construct hierarchical data structure of $\mathbf{P}$.
2) Transform the dense system $\mathbf{P}\mathbf{q} = \mathbf{v}$ to equivalent sparse system $\tilde{\mathbf{P}}\tilde{\mathbf{q}} = \tilde{\mathbf{v}}$, with perper row/column ordering.
3) Compute inexact factorization $\hat{\mathbf{L}}\hat{\mathbf{U}}$ for $\tilde{\mathbf{P}}$.
4) Solve $\hat{\mathbf{L}}_{22}\hat{\mathbf{U}}_{22}\hat{\mathbf{q}}_2 = \tilde{\mathbf{v}}_2$ by forward and backward substitution.
5) Compute capacitance.

In Step 1, we use HiCap algorithm to construct the dense linear system. In Step 2, we make the sparse transformation using PHiCap algorithm. When forming the $\tilde{\mathbf{P}}$ matrix, in order to improve the accuracy of the $\hat{\mathbf{L}}\hat{\mathbf{U}}$ approximation of the following step, we order the rows/columns carefully. Compared with the exact factorization, the inexact factors $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$ maintain the sparse pattern of $\tilde{\mathbf{P}}$ by dropping the nonzero entries, called fill-ins, in positions where $\tilde{\mathbf{P}}$ has zero entries. Since incomplete factorization updates the rows one by one from top to bottom, we can order the rows with less nonzeros to the top part and the rows with more nonzeros to the bottom to reduce the number of dropped fill-ins. Considering the hierarchical data structure in Fig. 1, after the sparse transformation, higher nodes have more links than lower nodes. We order the nodes according to the heights, with lower nodes before higher nodes. All root nodes are ordered last. As a result, the accuracy of the inexact factorization in Step 3 is improved.

In addition, with the proper ordering, the cost needed to solve system (3) can be reduced. Let

$$\hat{\mathbf{q}} = \begin{bmatrix} \hat{\mathbf{q}}_1 \\ \hat{\mathbf{q}}_2 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{v}} = \begin{bmatrix} \tilde{\mathbf{v}}_1 \\ \tilde{\mathbf{v}}_2 \end{bmatrix},$$

where vectors $\hat{\mathbf{q}}_1$ and $\tilde{\mathbf{v}}_1$ correspond to right child nodes and $\hat{\mathbf{q}}_2$ and $\tilde{\mathbf{v}}_2$ correspond to root nodes. According to the discussion in previous section, $\tilde{\mathbf{v}}_1 = \mathbf{0}$. Furthermore, each entry of $\hat{\mathbf{q}}_2$ is the sum of all the leaf panel charges in that tree, scaled by a factor that depends on the height of the root node. Thus, only $\hat{\mathbf{q}}_2$ is necessary for computing capacitance.

Following the same ordering, we represent $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$ as the follows.

$$\hat{\mathbf{L}} = \begin{bmatrix} \hat{\mathbf{L}}_{11} & 0 \\ \hat{\mathbf{L}}_{21} & \hat{\mathbf{L}}_{22} \end{bmatrix} \quad \text{and} \quad \hat{\mathbf{U}} = \begin{bmatrix} \hat{\mathbf{U}}_{11} & \hat{\mathbf{U}}_{12} \\ 0 & \hat{\mathbf{U}}_{22} \end{bmatrix}.$$

Thus, equation (3) can be written as

$$\begin{bmatrix} \hat{\mathbf{L}}_{11} & 0 \\ \hat{\mathbf{L}}_{21} & \hat{\mathbf{L}}_{22} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{U}}_{11} & \hat{\mathbf{U}}_{12} \\ 0 & \hat{\mathbf{U}}_{22} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{q}}_1 \\ \hat{\mathbf{q}}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{\mathbf{v}}_2 \end{bmatrix},$$

which results in

$$\hat{\mathbf{L}}_{11}\left(\hat{\mathbf{U}}_{11}\hat{\mathbf{q}}_1 + \hat{\mathbf{U}}_{12}\hat{\mathbf{q}}_2\right) = \mathbf{0} \tag{4}$$

$$\hat{\mathbf{L}}_{21}\left(\hat{\mathbf{U}}_{11}\hat{\mathbf{q}}_1 + \hat{\mathbf{U}}_{12}\hat{\mathbf{q}}_2\right) + \hat{\mathbf{L}}_{22}\hat{\mathbf{U}}_{22}\hat{\mathbf{q}}_2 = \tilde{\mathbf{v}}_2. \tag{5}$$

From Equation (4), since $\hat{\mathbf{L}}_{11}$ is nonsingular, we have

$$\hat{\mathbf{U}}_{11}\hat{\mathbf{q}}_1 + \hat{\mathbf{U}}_{12}\hat{\mathbf{q}}_2 = \mathbf{0}. \tag{6}$$

Substitute (6) in Equation (5), the problem is reduced to

$$\hat{\mathbf{L}}_{22}\hat{\mathbf{U}}_{22}\hat{\mathbf{q}}_2 = \tilde{\mathbf{v}}_2., \tag{7}$$

We only need to solve system (7) for $\hat{\mathbf{q}}_2$ to compute capacitance. That is Step 4 and Step 5. The dimension of system (7) equals the number of conductor surfaces, which is $O(m)$. Since the system dimension is small, it can be solved easily using forward and backward substitution.

TABLE I

EXPERIMENTAL RESULTS OF BUS CROSSING EXAMPLES IN UNIFORM DIELECTRIC. TIME IS CPU SECONDS. ITERATION IS AVERAGE FOR SOLVING ONE CONDUCTOR. MEMORY IS MB. ERROR IS WITH RESPECT TO FASTCAP(ORDER=2).

Bus crossing in uniform dielectric

	Bus4x4			Bus6x6			Bus8x8		
	FastCap (order=2)	PHiCap	New Algorithm	FastCap (order=2)	PHiCap	New Algorithm	FastCap (order=2)	PHiCap	New Algorithm
Time	18.6	0.4	0.3	113.9	1.5	1.1	206	3.3	2.8
Iteration	8	3	—	14.4	3.2	—	12	3.4	—
Memory	25.7	2.4	2.1	62.5	7.3	6.4	112	12.8	11.4
Error	—	2.1%	1.1%	—	2.3%	1.7%	—	3.0%	1.8%
Panel	2736	1088	1088	5832	3168	3168	10080	4224	4224

Bus crossing in multilayer dielectrics

	Bus4x4			Bus6x6			Bus8x8		
	FastCap (order=2)	PHiCap	New Algorithm	FastCap (order=2)	PHiCap	New Algorithm	FastCap (order=2)	PHiCap	New Algorithm
Time	63	2.0	1.5	162	5.7	3.4	324	14.2	6.9
Iteration	13	2	—	17.1	3	—	18	3	—
Memory	68	6.4	5.6	92	14.0	12.4	133	26.3	23.5
Error	—	0.7%	0.7%	—	1.3%	1.4%	—	1.4%	1.5%
Panel	3456	2120	2120	5448	4120	4120	7968	6784	6784

IV. EXPERIMENTAL RESULTS

We compare the new method with FastCap [4] and PHiCap [8]. Table I reports the experimental results. The bus crossing examples in uniform dielectric are standard benchmarks from [4]. The examples in multilayer dielectrics are from [8]. The algorithms are executed on a Sun UltraSPARC Enterprise 4000. The relative error in the capacitance matrix $\mathbf{C}'$, which is computed by the algorithms, is defined as $\|\mathbf{C}-\mathbf{C}'\|_\mathbf{F}/\|\mathbf{C}\|_\mathbf{F}$, where $\|\cdot\|_F$ denotes the Frobenius norm. The new algorithm uses less memory compared with PHiCap, because $\hat{\mathbf{L}}$ and $\hat{\mathbf{U}}$ overwrite $\tilde{\mathbf{P}}$ and the memory for iterative solver in PHiCap are not needed in the new algorithm.

V. CONCLUSIONS

The algorithm propsed in this paper is the first capacitance extraction algorithm based on BEM, which solves the linear system using direct method, instead of iterative solver. The new method is up to 100 times faster than FastCap and is also much faster than PHiCap.

REFERENCES

[1] M. W. Beattie and L. T. Pileggi, "Electromagnetic parasitic extraction via a multipole method with hierarchical refinement," in *Proceedings of the 1999 IEEE/ACM international conference on Computer-aided design.* IEEE Press, 1999, pp. 437–444.

[2] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems.* Cambridge, MA: MIT Press, 1988.

[3] S. Kapur and D. E. Long, "IES^3: a fast integral equation solver for efficient 3-dimensional extraction," in *Proceedings of the 1997 IEEE/ACM international conference on Computer-aided design.* IEEE Computer Society, 1997, pp. 448–455.

[4] K. Nabors and J. White, "FastCap: A multipole accelerated 3-d capacitance extraction program," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1447–1459, 1991.

[5] J. R. Phillips and J. White, "A precorrected FFT method for capacitance extraction of complicated 3-d structures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1059–1072, 1997.

[6] W. Shi, J. Liu, N. Kakani, and T. Yu, "A fast hierarchical algorithm for 3-d capacitance extraction," *IEEE Trans. on Computer Aided Design*, pp. 330–336, 2002.

[7] S. Yan, V. Sarin, and W. Shi, "Sparse transformations and preconditioners for hierarchical 3-d capacitance extraction with multiple dielectrics," in *Proceedings of the 41st annual conference on Design automation.* ACM Press, 2004, pp. 788–793.