A Direct Integral-Equation Solver of Linear Complexity for Large-Scale 3D Capacitance and Impedance Extraction

Wenwen Chai, Dan Jiao, and Cheng-Kok Koh

School of Electrical and Computer Engineering, Purdue University, 465 Northwestern Avenue, West Lafayette, IN 47907, USA (phone: 765-494-5240; fax: 765-494-3371; e-mails: {wchai, djiao, chengkok}@purdue.edu)

ABSTRACT

State-of-the-art integral-equation-based solvers rely on techniques that can perform a matrix-vector multiplication in $O(N)$ complexity. In this work, a fast inverse of linear complexity was developed to solve a dense system of linear equations directly for the capacitance extraction of any arbitrary shaped 3D structure. The proposed direct solver has demonstrated clear advantages over state-of-the-art solvers such as FastCap and HiCap; with fast CPU time and modest memory consumption, and without sacrificing accuracy. It successfully inverts a dense matrix that involves more than one million unknowns associated with a large-scale on-chip 3D interconnect embedded in inhomogeneous materials. Moreover, we have successfully applied the proposed solver to full-wave extraction.

Categories and Subject Descriptors

B.7.2 **[Integrating Circuits]:** Design Aids - simulation, verification

General Terms

Algorithms

Keywords

Integral-equation-based methods, direct solver, capacitance extraction, full wave.

1. INTRODUCTION

Integral-equation-based (IE-based) methods have been methods of choice in extracting the capacitive parameters of 3D interconnects since they reduce the solution domain by one dimension, and they model an infinite domain without the need of introducing an absorbing boundary condition. Compared to their partial-differential-equation-based counterparts, however, IE-based methods generally lead to dense systems of linear equations. Using a naïve direct method to solve a dense system takes $O(N^3)$ operations and requires $O(N^2)$ space, with N being the matrix size. When an iterative solver is used, the memory requirement remains the same, and the time complexity is $O(N_{it}N^2)$, where N_{it} denotes the total number of iterations required to reach convergence. In state-of-the-art IE-based capacitance solvers, Fast Multipole Method (FMM) and hierarchical algorithms [1-3] were used to perform a matrix-vector multiplication in $O(N)$ complexity, thereby significantly reducing the complexity of iterative solvers. In the limited work reported on the direct IE solutions for capacitance extraction [4], no linear complexity has been achieved. Compared to iterative solvers, direct solvers have advantages when the number of iterations is large or the number of right hand sides is large. For example, if there exist N right hand sides, each solve of which costs $O(N)$ operations, the total cost is still $O(N^2)$, which is expensive.

The contribution of this paper is the development of a linear-complexity direct IE solver that is kernel independent, and hence suitable for solving both quasi-static and full-wave problems. To be specific, the inverse of a dense system matrix arising from a quasi-static or full-wave problem is obtained in linear CPU time and memory consumption without sacrificing accuracy. Our solution hinges on the observation that the matrices resulting from an IE-based method, although dense, can be thought of as *data-sparse*, i.e., they can be specified by few parameters. There exists a general mathematical framework, called the "Hierarchical ($\mathcal{H}$) Matrix" framework [5], which enables a highly compact representation and efficient numerical computation of dense matrices. Both Storage requirements and matrix-vector multiplications using $\mathcal{H}$-matrices are of complexity $O(N\log^{\alpha}N)$. $\mathcal{H}^2$-matrices, which are a specialized subclass of hierarchical matrices, were later introduced in [6]. It was shown that the storage requirements and matrix-vector products are of complexity $O(N)$ for $\mathcal{H}^2$-based representation of both quasi-static and electrodynamic problems [7-8]. The nested structure is the key difference between $\mathcal{H}$-matrices and $\mathcal{H}^2$-matrices, since it permits an efficient reuse of information across the entire hierarchy. Solvers based on $\mathcal{H}$- and $\mathcal{H}^2$-matrices are kernel independent, and are therefore suitable for any IE-based formulation.

Although the matrix-vector product involving an $\mathcal{H}^2$-matrix can be performed in $O(N)$ complexity, the complexity of $\mathcal{H}^2$-matrix-based inverse has not been clearly established in the literature. In this work, we developed a direct IE solver of linear complexity for solving large-scale quasi-static and electrodynamic problems. The remainder of this paper is organized as follows. In Section II, IE formulations for capacitance extraction in both uniform and non-uniform materials are presented. An $\mathcal{H}^2$-matrix-based representation of the dense system matrix is constructed, and its error bound derived. We show that exponential convergence with respect to the number of interpolation points can be achieved irrespective of the problem size. In Section III, we provide the details of the linear-complexity direct inverse, which includes the orthogonalization of cluster bases, a recursive inverse formula, and fast matrix-matrix multiplication in linear complexity. In

—This work was supported by NSF under award No. 0747578 and No. 0702567.

DAC'09, July 26-31, 2009, San Francisco, California, USA

Section IV, numerical results are given to demonstrate the accuracy and efficiency of the proposed IE solver for both capacitance and full-wave extraction. We conclude in Section V.

2. IE FORMULATION WITH $\mathcal{H}^2$ MATRIX

2.1 IE Formulation

An integral-equation-based analysis of a multi-conductor structure embedded in inhomogeneous materials results in the following linear system [3]

$$\underbrace{\begin{bmatrix}\mathbf{P}_{cc} & \mathbf{P}_{cd}\\ \mathbf{E}_{dc} & \mathbf{E}_{dd}\end{bmatrix}}_{\mathbf{G}}\underbrace{\begin{bmatrix}q_c\\ q_d\end{bmatrix}}_{q}=\underbrace{\begin{bmatrix}v_c\\ 0\end{bmatrix}}_{v} \tag{1}$$

where q_c and q_d are the charge vectors of the conductor panels and dielectric-dielectric interface panels, respectively, and v_c is the potential vector associated with the conductor panels. The entries of $\mathbf{P}$ and $\mathbf{E}$ are

$$\mathbf{P}_{ij}=\frac{1}{a_i}\frac{1}{a_j}\int_{S_i}\int_{S_j} g(\vec{r},\vec{r}')dr_i dr_j$$

$$\mathbf{E}_{ij}=(\varepsilon_a-\varepsilon_b)\frac{\partial}{\partial n_a}\frac{1}{a_i}\frac{1}{a_j}\int_{S_i}\int_{S_j} g(\vec{r},\vec{r}')dr_i dr_j \tag{2}$$

where a_i and a_j are the areas of panel S_i and S_j, respectively, $\hat{n}$ is a unit vector normal to the dielectric interface, and ε_a and ε_b are the permittivity in the two dielectric regions separated by the interface. The diagonal entries of $\mathbf{E}_{dd}$ are $e_{ij}=(\varepsilon_a+\varepsilon_b)/(2a_i\varepsilon_0)$. In a uniform dielectric, (1) is reduced to

$$\mathbf{P}_{cc}q_c=v_c. \tag{3}$$

2.2 Cluster Tree and Block Cluster Tree

In order to capture the nested hierarchical dependence present in $\mathbf{G}$ shown in (1), we explore the use of a cluster tree and a block cluster tree. Denoting the full index set of all the panels by $\mathcal{I}:=\{1, 2, \ldots, N\}$. A representative cluster tree $T_{\mathcal{I}}$ is shown in Fig. 1(a). Clusters with indices no more than *leafsize* are leaves. The set of leaves of $T_{\mathcal{I}}$ is denoted by $\mathcal{L}_{\mathcal{I}}$.

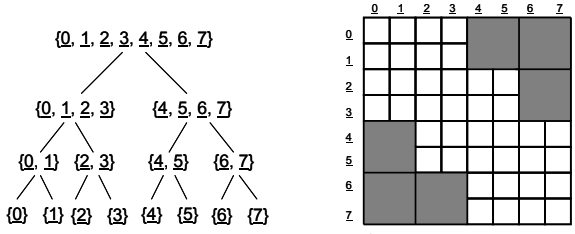

Fig. 1 (a) A cluster tree. (b) An $\mathcal{H}^2$-matrix structure.

Consider two subsets t and s of $\mathcal{I}$. We define a strong admissibility condition as follows [5]:

$$(t,s)\text{ are admissible}:=\begin{cases}\text{True} & \text{if } \max\{diam(\Omega_t),diam(\Omega_s)\}\le\\ & \eta dist(\Omega_t,\Omega_s)\\ \text{False} & \text{otherwise}\end{cases} \tag{4}$$

in which Ω_t and Ω_s are the supports of the union of all the panels in t and s respectively, and η is a parameter that controls the solution accuracy. Constructing an admissible block cluster tree from the cluster trees $T_{\mathcal{I}}$ and $T_{\mathcal{I}}$ itself (the testing and basis functions are the same in Galerkin-based IE solvers) and a given admissibility condition can be done recursively [5], in which the constructing procedure results in an admissible block cluster tree which can be mapped to a matrix structure shown in Fig. 1(b). Each leaf block cluster corresponds to a matrix block. The shaded matrix blocks are admissible blocks in which the $\mathcal{H}^2$-matrix representation is used; the un-shaded ones are inadmissible blocks in which a full matrix representation is employed.

2.3 $\mathcal{H}^2$-Matrix Representation and Its Error Bound

If two subsets t and s of $\mathcal{I}$ satisfy the strong admissibility condition (4), the original kernel function $g(\vec{r}_i,\vec{r}_j)$ in (2) can be replaced by a degenerate approximation

$$\tilde{g}^{t,s}(\vec{r},\vec{r}')=\sum_{v\in K^t}\sum_{\mu\in K^s} g(\xi_v^t,\xi_\mu^s)L_v^t(\vec{r})L_\mu^s(\vec{r}') \tag{5}$$

where $K:=\{v\in\mathbb{N}^d : v_i\le p \ \ \forall i\in\{1,\ldots,d\}\}=\{1,\ldots,p\}^d$, d = 1, 2, 3, for 1-, 2-, and 3-D problems, respectively; p is the number of interpolation points in one dimension; $(\xi_v^t)_{v\in K^t}$ and $(\xi_\mu^s)_{\mu\in K^s}$ are two families of interpolation points, respectively, in t and s; and $(L_v^t)_{v\in K^t}$ and $(L_\mu^s)_{v\in K^s}$ are the corresponding Lagrange polynomials. With (5), (2) are separated into two single integrals:

$$\begin{aligned}\tilde{\mathbf{P}}_{ij}^{t,s}&:=\sum_{v\in K^t}\sum_{\mu\in K^s}1/(a_ia_j)\cdot g(\xi_v^t,\xi_\mu^s)\int_{S_i}L_v^t(\vec{r})dr\cdot\int_{S_j}L_\mu^s(\vec{r}')dr'\\ \tilde{\mathbf{E}}_{ij}^{t,s}&=\sum_{v\in K^t}\sum_{\mu\in K^s}1/(a_ia_j)\cdot(\varepsilon_a-\varepsilon_b)\frac{\partial}{\partial n_a}g(\xi_v^t,\xi_\mu^s)\int_{S_i}L_v^t(\vec{r})dr\cdot\int_{S_j}L_\mu^s(\vec{r}')dr'\end{aligned} \tag{6}$$

Hence, the submatrix $\tilde{\mathbf{G}}^{t,s}$ can be written in a factorized form as:

$$\tilde{\mathbf{G}}^{t,s}:=\mathbf{V}^t\mathbf{S}^{t,s}\mathbf{V}^{s^T},\ \mathbf{V}^t\in\mathbb{R}^{t\times K^t},\ \mathbf{S}^{t,s}\in\mathbb{R}^{K^t\times K^s},\ \mathbf{V}^s\in\mathbb{R}^{s\times K^s} \tag{7}$$

$$\textit{where},\quad \mathbf{V}_{iv}^t=\int_{S_i}L_v^t(\vec{r})dr,\qquad \mathbf{V}_{j\mu}^s=\int_{S_j}L_\mu^s(\vec{r}')dr'$$

$$\mathbf{S}_{v\mu}^{t,s}=\begin{cases}g(\xi_v^t,\xi_\mu^s)/(a_ia_j) & (\ t\text{ contains conductor panels})\\ (\varepsilon_a-\varepsilon_b)/(a_ia_j)\dfrac{\partial g(\xi_v^t,\xi_\mu^s)}{\partial n_a} & (\ t\text{ contains dielectric panels})\end{cases}$$

for $i\in t$, $j\in s$, $v\in K^t$, and $\mu\in K^s$. The matrix $\mathbf{G}$ in (7) forms an $\mathcal{H}^2$-matrix representation if the same space of polynomials are used across t and s. After a detailed error analysis, we found that if the admissibility condition given in (4) is satisfied, the error of (5) is bounded by

$$\|g(r,r')-\tilde{g}^{(t,s)}(r,r')\|_\infty\le\frac{4ed}{\pi}(\Lambda_p)^{2d}p\frac{1}{dist(Q_t,Q_s)}[1+\sqrt{2}\eta][1+\frac{\sqrt{2}}{\eta}]^{-p} \tag{8}$$

where Λ_p is a constant related to p and the interpolation scheme, and $dist(Q_t,Q_s)$ is the Euclidean distance between cluster t and cluster s. Clearly, exponential convergence with respect to p can be obtained irrespective of the choice of η. The larger p is, the smaller the error is. In addition, the block entries represented by (7) can be kept to the same order of accuracy across tree levels.

3. Direct Inverse of Linear Complexity

The algorithms in the proposed direct solver are outlined below:

Direct IE solver of linear complexity
1. Orthogonalize the cluster basis $\mathbf{V}^t$
2. Compute the inverse of $\mathcal{H}^2$–based $\mathbf{G}$
   - Recursive inversion
   - Linear-time matrix multiplication
3. Compute the capacitance matrix by $q=\mathbf{G}^{-1}v$

Before we provide the details, we introduce the following concepts and notation: 1) For each cluster $t \in T_{\mathcal{I}}$, the cardinality of the sets $col(t) := \{s \in T_{\mathcal{J}} : \ (t,s) \in T_{\mathcal{I}\times\mathcal{J}}\}$ and $row(s) := \{t \in T_{\mathcal{I}} : \ (t,s) \in T_{\mathcal{I}\times\mathcal{J}}\}$ is bounded by a constant C_{sp} [5]. 2) Each non-leaf cluster t has two child nodes. 3) The rank of $\mathbf{V} = (\mathbf{V}^t)_{t\in T_{\mathcal{I}}}$ is denoted by k. 4) The parameter *leafsize* is denoted by $n_{\min}$, and $\#t \le n_{\min}$ if $t \in \mathcal{L}_{\mathcal{I}}$.

3.1 Orthogonalization of the Nested Cluster Basis

Recall that when constructing the $\mathcal{H}^2$-matrix representation of system matrix $\mathbf{G}$, we use the same space of polynomials for all clusters. Consider a cluster t', which is a child of t, $L_v^t(\vec{r})$ in (5) can be written as $L_v^t(\vec{r}) = \sum_{v'\in K^{t'}} \mathbf{T}_{v'v}^{t'} L_{v'}^{t'}(\vec{r})$, with $\mathbf{T}_{v'v}^{t'} = L_v^t(\xi_{v'}^{t'})$. As a result,

$$\mathbf{V}_{iv}^t = \int_{S_i} L_v^t(\vec{r})dr = \sum_{v'\in K^{t'}} \mathbf{T}_{v'v}^{t'} \int_{S_i} L_{v'}^{t'}(\vec{r})dr = \sum_{v'\in K^{t'}} \mathbf{T}_{v'v}^{t'} \mathbf{V}_{iv'}^{t'} = (\mathbf{V}^{t'}\mathbf{T}^{t'})_{iv},$$

where $\mathbf{T}^{t'} \in \mathbb{R}^{K^{t'}\times K^t}$ is called transfer matrix for cluster t'. Hence, assuming that $\text{children}(t) = \{t_1, t_2\}$ with $t_1 \ne t_2$, we have

$$\mathbf{V}^t = \begin{pmatrix} \mathbf{V}^{t_1}\mathbf{T}^{t_1} \\ \mathbf{V}^{t_2}\mathbf{T}^{t_2} \end{pmatrix} = \begin{pmatrix} \mathbf{V}^{t_1} & \\ & \mathbf{V}^{t_2} \end{pmatrix} \begin{pmatrix} \mathbf{T}^{t_1} \\ \mathbf{T}^{t_2} \end{pmatrix} \tag{9}$$

This means that we only need to store the matrices $\mathbf{V}^t$ for leaf clusters t and use the transfer matrices $\mathbf{T}$ to represent all other clusters. This nested property of $\mathbf{V}^t$ as shown in (9) enables $O(N)$ storage of $\mathbf{G}$ and $O(N)$ matrix vector multiplication [6-8].

To make the inverse calculation efficient, we first orthogonalize $\mathbf{V}^t$ while still preserving the nested property of $\mathbf{V}^t$.

For leaf cluster bases, we can construct an orthogonal matrix $\mathbf{Z}^t$ such that

$$\tilde{\mathbf{V}}^t = \mathbf{V}^t\mathbf{Z}^t \quad \text{and} \quad \tilde{\mathbf{V}}^{t\,\mathrm{T}}\tilde{\mathbf{V}}^t = (\mathbf{V}^t\mathbf{Z}^t)^{\mathrm{T}}\mathbf{V}^t\mathbf{Z}^t = (\mathbf{Z}^t)^{\mathrm{T}}\mathbf{G}^t\mathbf{Z}^t = \mathbf{I}$$

with $\mathbf{G}^t = \mathbf{V}^{t\,\mathrm{T}}\mathbf{V}^t$. To find $\mathbf{Z}^t$, we first perform Schur decomposition $\mathbf{G}^t = \mathbf{P}\mathbf{D}\mathbf{P}^{\mathrm{T}}$, where $\mathbf{P}$ contains the eigenvectors of $\mathbf{G}^t$ and the diagonal matrix $\mathbf{D} = diag(\lambda_1, ..., \lambda_k)$ contains the corresponding eigenvalues $\lambda_1 \ge \lambda_2 \ge \ ... \ \ge \lambda_k \ge 0$. We fix a rank $k^t \in \{0, \ ... \ , k\}$ such that $\lambda_i > 0$ holds for all $i \in \{1, \ ... \ , k^t\}$. Define matrix $\tilde{\mathbf{D}} \in \mathbb{R}^{k\times k^t}$ by $\tilde{\mathbf{D}}_{ij} = \delta_{ij}/\sqrt{\lambda_i}$. If $\mathbf{Z}^t = \mathbf{P}\tilde{\mathbf{D}}$, we obtain

$$\tilde{\mathbf{V}}^{t\,\mathrm{T}}\tilde{\mathbf{V}}^t = (\mathbf{Z}^t)^{\mathrm{T}}\mathbf{G}^t\mathbf{Z}^t = \tilde{\mathbf{D}}^{\mathrm{T}}\mathbf{P}^{\mathrm{T}}\mathbf{P}\mathbf{D}\mathbf{P}^{\mathrm{T}}\mathbf{P}\tilde{\mathbf{D}} = \tilde{\mathbf{D}}^{\mathrm{T}}\mathbf{D}\tilde{\mathbf{D}}^{\mathrm{T}} = \mathbf{I}$$

Hence, $\mathbf{Z}^t = \mathbf{P}\tilde{\mathbf{D}}$ is the matrix that can orthogonalize leaf cluster bases $\mathbf{V}^t$. Based on the nested property of $\mathbf{V}^t$, the total complexity of obtaining $\mathbf{Z}^t$ using the above procedure is $O(N)$. The non-leaf clusters can be orthogonalized in a similar fashion with the nested property preserved.

3.2 Fast Inversion of Linear Complexity

(1) Recursive Inversion Equation: Casting the $\mathcal{H}^2$-matrix representation of $\mathbf{G}$ into the following form $\mathbf{G} = \begin{bmatrix} \mathbf{G}_{11} & \mathbf{G}_{12} \\ \mathbf{G}_{21} & \mathbf{G}_{22} \end{bmatrix}$. Its inverse can be recursively computed by using the equation

$$\mathbf{G}^{-1} = \begin{bmatrix} \mathbf{G}_{11}^{-1} \oplus \mathbf{G}_{11}^{-1} \otimes \mathbf{G}_{12} \otimes \mathbf{S}^{-1} \otimes \mathbf{G}_{21} \otimes \mathbf{G}_{11}^{-1} & -\mathbf{G}_{11}^{-1} \otimes \mathbf{G}_{12} \otimes \mathbf{S}^{-1} \\ -\mathbf{S}^{-1} \otimes \mathbf{G}_{21} \otimes \mathbf{G}_{11}^{-1} & \mathbf{S}^{-1} \end{bmatrix} \tag{10}$$

where $\mathbf{S} = \mathbf{G}_{22} \oplus (-\mathbf{G}_{21} \otimes \mathbf{G}_{11}^{-1} \otimes \mathbf{G}_{12})$, and $\oplus$, $\otimes$ are respectively addition and multiplication defined for the $\mathcal{H}^2$ matrix to be elaborated soon. The recursive inverse equation (10) can be realized by the pseudo-code shown below

Recursive inverse algorithm ($\mathbf{X}$ is used for temporary storage)
Procedure $\mathcal{H}^2$-inverse($\mathbf{G}$, $\mathbf{X}$) ($\mathbf{G}$ is input matrix, $\mathbf{X}$ is inverse)
If matrix $\mathbf{G}$ is a non-leaf matrix block
  $\mathcal{H}^2$-inverse ($\mathbf{G}_{11}$, $\mathbf{X}_{11}$)
  $\mathbf{G}_{21} \otimes \mathbf{X}_{11} \to \mathbf{X}_{21}$, $\mathbf{X}_{11} \otimes \mathbf{G}_{12} \to \mathbf{X}_{12}$, $-\mathbf{X}_{22} \oplus (\mathbf{X}_{21} \otimes \mathbf{G}_{12}) \to \mathbf{X}_{22}$ (11)
  $\mathcal{H}^2$-inverse ($\mathbf{X}_{22}$, $(\mathbf{G}^{-1})_{22}$)
  $-(\mathbf{G}^{-1})_{22} \otimes \mathbf{X}_{21} \to (\mathbf{G}^{-1})_{21}$, $-\mathbf{X}_{12} \otimes (\mathbf{G}^{-1})_{22} \to (\mathbf{G}^{-1})_{12}$, $\mathbf{X}_{11} \oplus (-\mathbf{G}_{12} \otimes \mathbf{X}_{21}) \to (\mathbf{G}^{-1})_{11}$
else
  Inverse ($\mathbf{G}$) (normal full matrix inverse)

From (11), it can be seen that the computation of inverse involves a full-matrix inverse at the leaf level and a number of matrix-matrix multiplications at other levels. Hence, efficient matrix-matrix multiplication is essential to an efficient inverse in linear time, which is elaborated in next section.

(2) Fast Matrix-Matrix Multiplication in Linear Time

The fast multiplication in (11) can be done recursively. Assuming $b_1 = (t,s) \in T_{\mathcal{I}\times\mathcal{I}}^G$, $b_2 = (s,r) \in T_{\mathcal{I}\times\mathcal{I}}^G$, and the multiplication target block is $b = (t,r) \in T_{\mathcal{I}\times\mathcal{I}}^G$. Matrix blocks $\mathbf{G}^{b1}$, $\mathbf{G}^{b2}$, and $\mathbf{G}^b$ can be admissible blocks, non-admissible blocks, or non-leaf blocks. The $\mathbf{G}^{b1} \otimes \mathbf{G}^{b2} \to \mathbf{G}^b$ encountered in (11) can be divided into the following cases, each of which has a constant complexity.

	$\mathbf{G}^{b1}$	$\mathbf{G}^{b2}$	$\mathbf{G}^b$	complexity
1	admissible	admissible	admissible	$O(k_1^3)$
2	admissible	admissible	non-leaf	$O(k_1^3)$
3	admissible	non-leaf	admissible	$O(k_1^3)$
4	admissible	non-leaf	non-leaf	$O(k_1^3)$
5	admissible	non-admissible	admissible	$O(k_1^3)$
6	non-admissible (full matrix)	non-admissible (full matrix)	non-admissible (full matrix)	$O(k_1^3)$
7	non-leaf	non-leaf	admissible	$O(k_1^3)$
8	non-leaf	non-leaf	non-leaf	$O(k_1^3)$

In the Table above, $k_1 = \max(k, n_{\min})$ is a constant that is independent of N for quasi-static applications.

Next, we will use only cases 1, 2 and 3 to explain how the fast multiplication is performed and omit other cases due to space constraint.

***Case 1*:** $\mathbf{G}^{b1} = \mathbf{V}^t \mathbf{S}_{b1} \mathbf{V}^{s^{\mathrm{T}}}$ and $\mathbf{G}^{b2} = \mathbf{V}^s \mathbf{S}_{b2} \mathbf{V}^{r^{\mathrm{T}}}$. Then,

$$\mathbf{G}^{b1} \otimes \mathbf{G}^{b2} = \mathbf{V}^t \mathbf{S}_{b1} \mathbf{V}^{s^{\mathrm{T}}} \otimes \mathbf{V}^s \mathbf{S}_{b2} \mathbf{V}^{r^{\mathrm{T}}} = \mathbf{V}^t \mathbf{S}_{b1} \mathbf{I} \mathbf{S}_{b2} \mathbf{V}^r = \mathbf{V}^t (\mathbf{S}_{b1} \mathbf{S}_{b2}) \mathbf{V}^r$$

$$\mathbf{G}_b^{new} = \mathbf{G}^b + \mathbf{G}^{b1} \otimes \mathbf{G}^{b2} = \mathbf{V}^t \mathbf{S}_b \mathbf{V}^r + \mathbf{V}^t (\mathbf{S}_{b1} \mathbf{S}_{b2}) \mathbf{V}^{r^{\mathrm{T}}} = \mathbf{V}^t (\mathbf{S}_b + \mathbf{S}_{b1} \mathbf{S}_{b2}) \mathbf{V}^{r^{\mathrm{T}}}$$

with $\mathbf{S}_b^{new} = \mathbf{S}_b + \mathbf{S}_{b1} \mathbf{S}_{b2}$. This process does not involve any approximation. Since the dimension of each of $\mathbf{S}_b, \mathbf{S}_{b1}, \mathbf{S}_{b2}$ is $k \times k$, the complexity of computing $\mathbf{S}_b^{new}$ is at most $O(k^3)$.

***Case 2*:** $\mathbf{G}^{b1} = \mathbf{V}^t \mathbf{S}_{b1} \mathbf{V}^{s^{\mathrm{T}}}$ and $\mathbf{G}^{b2} = \mathbf{V}^s \mathbf{S}_{b2} \mathbf{V}^{r^{\mathrm{T}}}$, while $\mathbf{G}^b$ is a non-leaf block. We first compute the multiplication as in Case (1) to get an admissible block as shown in step (a) in Fig. 2. We then split the resultant admissible block into four small admissible blocks as shown in step (b). However, the sub-blocks in $\mathbf{G}^b$ are not necessarily all admissible blocks. Based on the block structure in the target matrix, we may convert an admissible block to a full matrix block as shown in step (c). We add the resultant matrix upon $\mathbf{G}^b$.

Fig.2. A scheme to compute the product of two admissible blocks and format the product to be a non-leaf. (R—an admissible block, F—an inadmissible block.

Steps (b) and (c) are called split operation and conversion operation respectively, which are performed as follows:

Split operation: A split operation is in fact a transformation from parents to children, which does not involve any approximation. For one block $b = (t, s) \in T^{\mathrm{G}}_{\mathcal{I} \times \mathcal{I}}$, we perform

$$\mathbf{V}^t \mathbf{S}_b \mathbf{V}^{s^{\mathrm{T}}} = \begin{bmatrix} \mathbf{V}^{t1} \mathbf{T}^{t1} \\ \mathbf{V}^{t2} \mathbf{T}^{t2} \end{bmatrix} \mathbf{S}_b \begin{bmatrix} \mathbf{V}^{s1} \mathbf{T}^{s1} \\ \mathbf{V}^{s2} \mathbf{T}^{s2} \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \mathbf{V}^{t1} (\mathbf{T}^{t1} \mathbf{S}_b \mathbf{T}^{s1^{\mathrm{T}}}) \mathbf{V}^{s1^{\mathrm{T}}} & \mathbf{V}^{t1} (\mathbf{T}^{t1} \mathbf{S}_b \mathbf{T}^{s2^{\mathrm{T}}}) \mathbf{V}^{s2^{\mathrm{T}}} \\ \mathbf{V}^{t2} (\mathbf{T}^{t2} \mathbf{S}_b \mathbf{T}^{s1^{\mathrm{T}}}) \mathbf{V}^{s1^{\mathrm{T}}} & \mathbf{V}^{t2} (\mathbf{T}^{t2} \mathbf{S}_b \mathbf{T}^{s2^{\mathrm{T}}}) \mathbf{V}^{s2^{\mathrm{T}}} \end{bmatrix} \quad (12)$$

where $\mathrm{children}(t) = \{t_1, t_2\}$, and $\mathrm{children}(s) = \{s_1, s_2\}$. It can be seen that one admissible block is divided into four admissible blocks with $S_{b_{ij}} = T^{t_i} S_b T^{s_j^{\mathrm{T}}}$ in (12). Hence, one split operation costs $O(k^3)$.

Conversion operation: If we convert the admissible block to a full matrix block as shown in step (c), we just need to compute $\mathbf{V}^{t2} \mathbf{S}_{b22} \mathbf{V}^{r2^{\mathrm{T}}}$. Since the largest dimension of a full matrix block is $n_{\min} \times n_{\min}$, the cost is at most $O(n_{\min}{}^2 k)$. If we convert a full matrix block to an admissible block, the best approximation of $\mathbf{G}_{full}^{(t2,r2)}$ for the admissible block is $\mathbf{V}^{t2} \mathbf{V}^{t2^{\mathrm{T}}} (\mathbf{G}_{full}^{(t2,r2)}) \mathbf{V}^{r2} \mathbf{V}^{r2^{\mathrm{T}}} = \mathbf{V}^{t2} \mathbf{S}_{b22} \mathbf{V}^{r2^{\mathrm{T}}}$ with $\mathbf{S}_{b22} = \mathbf{V}^{t2^{\mathrm{T}}} (\mathbf{G}_{full}^{(t2,r2)}) \mathbf{V}^{r2}$. Hence, each conversion operation at most costs $O(n_{\min}{}^2 k)$. In summary, each of steps (a), (b), and (c) shown in Fig. 2 has complexity $O(k_1^3)$, and hence the total complexity is $O(k_1^3)$.

***Case 3*:** If $\mathbf{G}^{b1}$ is an admissible block, $\mathbf{G}^{b2}$ is a non-leaf block, and $\mathbf{G}^b$ is an admissible block, we first do split operations on $\mathbf{G}^{b1}$ and get a new block $\tilde{\mathbf{G}}^{b1}$ as shown in step (a) of Fig. 3. We then use simple recursive multiplication to compute $\tilde{\mathbf{G}}^{b1} \otimes \mathbf{G}^{b2}$ and obtain a non-leaf block with four admissible sub-blocks in step (b). In step (c), a collect operation is performed to get a single admissible block, which is depicted below. After obtaining the single admissible block, we directly add it to original matrix block $\mathbf{G}^b$ and get $\mathbf{G}_b^{new}$. The only approximation in this case is from the collect operation done in step (c), the accuracy of which is controllable.

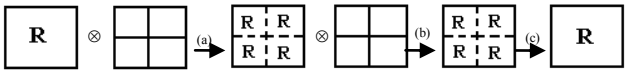

Fig. 3. A scheme to compute the product of an admissible block with a non-leaf block with the target block being an admissible block.

Collect operation: This process is a transformation from children to parents, which involves an approximation since we are not able to express the cluster bases of children in terms of the parent cluster bases. However, we can get the best approximation of the children blocks in the cluster bases corresponding to the parent using the orthogonal cluster basis.

We approximate the child matrix block b_{ij} by the parent block b. The best approximation in the cluster bases $\mathbf{V}^t$ and $\mathbf{V}^s$ is

$$\begin{aligned} &\mathbf{V}^t \mathbf{V}^{t^{\mathrm{T}}} \begin{bmatrix} \mathbf{V}^{t1} \mathbf{S}_{b11} \mathbf{V}^{s1^{\mathrm{T}}} & \mathbf{V}^{t1} \mathbf{S}_{b12} \mathbf{V}^{s2^{\mathrm{T}}} \\ \mathbf{V}^{t2} \mathbf{S}_{b_{21}} \mathbf{V}^{s1^{\mathrm{T}}} & \mathbf{V}^{t2} \mathbf{S}_{b22} \mathbf{V}^{s2^{\mathrm{T}}} \end{bmatrix} \mathbf{V}^s \mathbf{V}^{s^{\mathrm{T}}} \\ &= \mathbf{V}^t \Big(\begin{bmatrix} \mathbf{V}^{t1} \mathbf{T}^{t1} \\ \mathbf{V}^{t2} \mathbf{T}^{t2} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mathbf{V}^{t1} \mathbf{S}_{b11} \mathbf{V}^{s1^{\mathrm{T}}} & \mathbf{V}^{t1} \mathbf{S}_{b12} \mathbf{V}^{s2^{\mathrm{T}}} \\ \mathbf{V}^{t2} \mathbf{S}_{b_{21}} \mathbf{V}^{s1^{\mathrm{T}}} & \mathbf{V}^{t2} \mathbf{S}_{b22} \mathbf{V}^{s2^{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \mathbf{V}^{s1} \mathbf{T}^{s1} \\ \mathbf{V}^{s2} \mathbf{T}^{s2} \end{bmatrix} \Big) \mathbf{V}^{s^{\mathrm{T}}} \\ &= \mathbf{V}^t \Big(\sum_{i \in \mathrm{children}(t)} \sum_{j \in \mathrm{children}(s)} \mathbf{T}^{t_i^{\mathrm{T}}} \mathbf{S}_{b_{ij}} \mathbf{T}^{s_j} \Big) \mathbf{V}^{s^{\mathrm{T}}} \end{aligned} \quad (13)$$

It can be seen that four admissible blocks are collected to be one admissible block with $\mathbf{S}_b = \sum_{i \in \mathrm{children}(t)} \sum_{j \in \mathrm{children}(s)} \mathbf{T}^{t_i^{\mathrm{T}}} \mathbf{S}_{b_{ij}} \mathbf{T}^{s_j}$. Hence, one collect operation costs $O(k^3)$. The total complexity of steps (a)-(c) shown in Fig. 3 is $O(k_1^3)$.

3.3 Compute Capacitance Matrix

Since the inverse obtained from (11) is also an $\mathcal{H}^2$ matrix, and $\mathcal{H}^2$-matrix-vector multiplication has linear complexity [6-8], we can compute $q = \tilde{\mathbf{G}}^{-1} v$ in $O(N)$ time. By adding all the entries of q in each conductor, the capacitance matrix element can be obtained.

3.4 Complexity Analysis and Error Analysis

Complexity Analysis: The cost of orthgonalization of the cluster basis described in Section 3.1 is $O(N)$. The cost of direct inverse shown in (11) can be analyzed below

$$\begin{aligned} &Comp(G^{-1}) = \sum_{l=0}^{L} (\#blocks\ at\ level\ l) O(k_1^3) \le \sum_{l=0}^{L} 2^l C_{sp} O(k_1^3) \\ &= C_{sp} O(k_1^3) \#T_{\mathcal{I}} = C_{sp} k_1^3 O(N) \end{aligned} \quad . \quad (14)$$

in which L is the number of tree levels. The inverse procedure shown in (11) essentially traverses a block cluster tree from bottom to top. At each tree level, the matrix block at that level is formed by a matrix-matrix multiplication. Since each matrix-

matrix multiplication has an $O(k_1^3)$ complexity as shown in Section 3.3, and there are at most $2^l C_{sp}$ matrix blocks in level l, we obtain a linear cost for matrix inverse as shown in (14). Computing the capacitance matrix described in Section 3.3 also costs $O(N)$. Therefore, the total CPU cost of the proposed direct inverse is $O(N)$. It is worth mentioning that if the linear system is symmetric, we can compute only half of the entries in the inverse, further reducing the CPU cost.

Accuracy Analysis: In Section 3.1, orthogonal bases $\tilde{\mathbf{V}}^t$ are constructed. The best approximation of a general $\mathbf{V}^t$ in the space $\tilde{\mathbf{V}}^t$ is given by $\tilde{\mathbf{V}}^t(\tilde{\mathbf{V}}^t)^{\mathrm{T}}\mathbf{V}^t$. The error of this approximation is:

$$\| \mathbf{V}^t - \tilde{\mathbf{V}}^t(\tilde{\mathbf{V}}^t)^{\mathrm{T}}\mathbf{V}^t \|_2^2 = \lambda_{k^t+1} \quad , \tag{15}$$

where λ_{k^t+1} is the $(k^t+1)th$ eigenvalue of $\mathbf{V}^{t\mathrm{T}}\mathbf{V}^t$, in which k^t is the rank of cluster basis $\tilde{\mathbf{V}}^t$. Clearly, if k^t is chosen to be the same as the rank of $\mathbf{V}^t$, the error of (15) is zero. Therefore $\tilde{\mathbf{V}}^t\tilde{\mathbf{V}}^{t\mathrm{T}}\mathbf{G}\tilde{\mathbf{V}}^s\tilde{\mathbf{V}}^{s\mathrm{T}}$ is the best approximation of a matrix block $\mathbf{G}^{(t,s)}$ in the bases $\tilde{\mathbf{V}}^t$ and $\tilde{\mathbf{V}}^s$. In Section 3.2, the inverse is performed by using formatted multiplication. For example, when computing $\mathbf{G}_{21}\otimes\mathbf{X}_{11}\rightarrow\mathbf{X}_{21}$ in (11), the block structure of $\mathbf{X}_{21}$ is assumed to be the same as that of $\mathbf{G}_{21}$. The goal of a formatted multiplication is to represent the $\mathcal{H}^2$ tree of $\mathbf{G}^{-1}$ by the same $\mathcal{H}^2$ tree used to represent $\mathbf{G}$. Certainly, one can assume a different tree to represent the tree of $\mathbf{G}^{-1}$, or perform unformatted multiplication, i.e. without specifying the target matrix. However, using the $\mathcal{H}^2$ tree of $\mathbf{G}$ to represent that of $\mathbf{G}^{-1}$ is an ideal choice based on physical understanding. Here, the capacitance matrix $\mathbf{G}^{-1}$ is a sparse matrix. In the $\mathcal{H}^2$ tree constructed for $\mathbf{G}$ and hence $\mathbf{G}^{-1}$, the blocks formed by clusters that satisfy admissibility condition (4) (i.e., they are far away) are represented by low rank matrices. In real $\mathbf{G}^{-1}$, these blocks can even be considered as zero. Hence, using the $\mathcal{H}^2$ tree of $\mathbf{G}$ to represent that of $\mathbf{G}^{-1}$ is indeed an ideal choice. The same argument holds true for full-wave cases. This has been verified by our numerical experiments. Therefore, the inverse performed here can be considered as an exact inverse if one neglects the round off error incurred in numerical computation.

4. Numerical Results

The first example is a $m\times m$ crossing bus structure embedded in free space or dielectric materials as shown in Fig. 4 [3]. Two

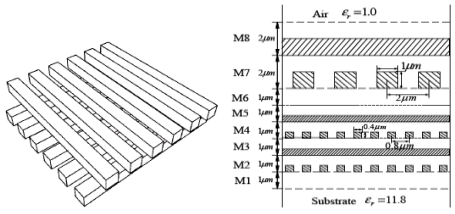

Fig. 4. (a) A bus structure. (b) An on-chip interconnect.

methods are compared: FastCap 2.0 and the proposed direct IE solver. The m in this bus structure varies from 4 to 16. The dimension of each bus is scaled to $1\times1\times(2m+1)$ m^3. The distance between buses in the same layer is 1 m, and the distance between the two bus layers is 1 m. For this bus structure, we simulated both free-space case and non-uniform dielectric case. For the case involving non-uniform dielectrics, the dielectric surrounding the upper layer conductors has relative permittivity of 3.9, and the lower layer conductors are in the dielectric having relative permittivity 7.5. Each bus is also scaled to $1\times1\times(2m+1)$ m^3. The distance between buses in the same layer is 1 m, and the distance between the two bus layers is 2 m. (Note that capacitances are scalable with respect to the length unit). In the proposed solver, the parameters used to construct the cluster tree and block cluster tree are *leafsize*=10 and η =1.6. The number of interpolation points p is determined by a function $p=a+b(L-l)$, with a=2, b=1, and L being the maximum number of tree level, and l tree level. In Fig. 5(a), we plot the error of $\mathcal{H}^2$-matrix representation of system matrix $\mathbf{G}$ (so called as original matrix error), and the error of extracted capacitances with respect to the number of unknowns. The former is measured by $\| \mathbf{G}-\tilde{\mathbf{G}} \|_F / \| \mathbf{G} \|_F$, where $\tilde{\mathbf{G}}$ is shown in (7), and $\|\cdot\|_F$ is the Frobenius norm; the latter is measured by $\| C-C' \|_F / \| C \|_F$, where C is the capacitance matrix obtained from FastCap 2.0, and C' is that generated by the proposed solver. As can be seen clearly from Fig. 5(b), very good accuracy of the proposed direct solver can be observed in both $\tilde{\mathbf{G}}$ and capacitance matrix C'. In addition, the error of $\tilde{\mathbf{G}}$ reduces with the number of unknowns because of increased p and hence increased accuracy as can be seen from (8). In addition, we are able to keep the accuracy of the capacitance matrix to the same order in the entire range.

In Fig. 6, we plot the total CPU time and memory consumption of the proposed direct inverse for the $m\times m$ bus structure in free space. In Fig. 7, we plot the same for the $m\times m$ bus structure embedded in multiple dielectrics. The performance of FastCap 2.0 is also plotted for comparison, the convergence tolerance of which is set to 1%. Compared with FastCap 2.0, the proposed direct solver is 9–25 times faster and reduces memory usage by 85–95%. Dell 1950 Server was used for all simulations in this paper.

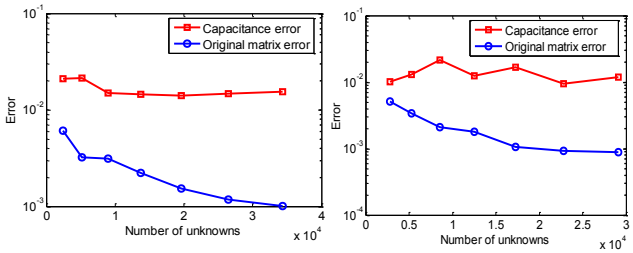

Fig. 5 Original matrix error and capacitance error with respect to N. (a) Uniform dielectric. (b) Non-uniform dielectric.

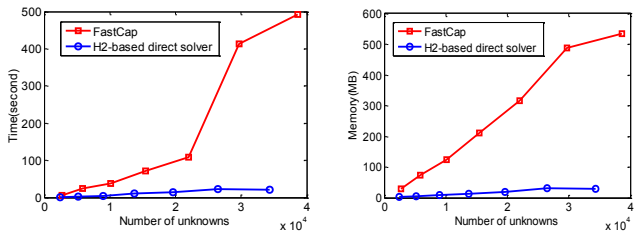

Fig. 6. Comparison of time and memory complexity in simulating the bus structure in free space.

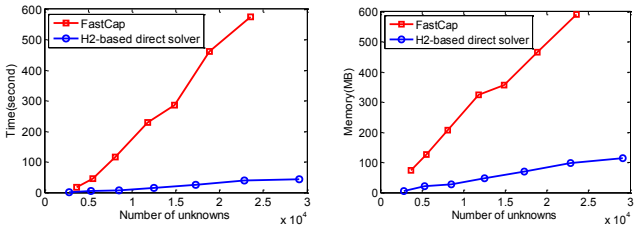

Fig. 7 Comparison of time and memory complexity in simulating the bus structure embedded in multiple dielectrics.

To test the performance of the proposed solver in simulating very large examples, we simulated a structure shown in Fig. 4(b) [3]. The relative permittivity is 3.9 in M1, 2.5 from M2 to M6, and 7.0 from M7 to M8. The discretization of this 48-conductor structure results in 25,556 unknowns. To test the large-scale modeling capability of the proposed solver, the 48-conductor structure shown in Fig. 4(b) is duplicated horizontally, resulting in 72, 96, 120, 144, 192, 240, 288, and 336 conductors, which lead to more than 1 million unknowns. The simulation parameters are chosen as *leafsize*=10, η =1, and p=1. The error of the $\mathcal{H}^2$-matrix representation of system matrix $\mathbf{G}$ in the maximal admissible block is shown in Fig. 8(a), with the number of

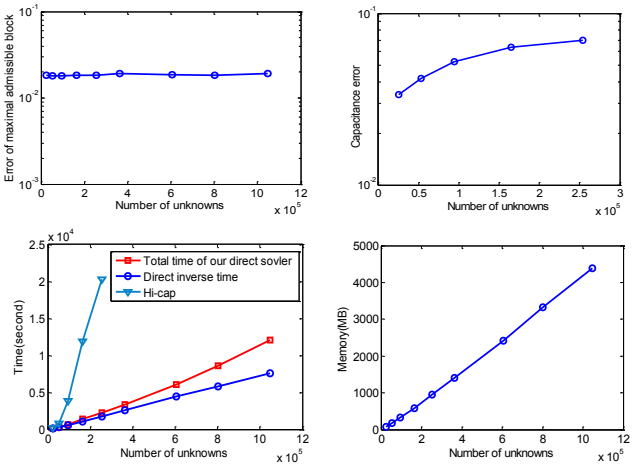

Fig. 8. (a) Error of maximal admissible block. (b) Error of Capacitance. (c) Time Complexity. (d) Memory Complexity.

unknowns varying from 25,556 to 1,047,236. Good accuracy is observed in the entire range. In Fig. 8(b), we show the capacitance error with respect to N. Again, good accuracy is observed. Since we need to use the capacitance C generated from an existing solver such as FastCap to assess the accuracy of the capacitance C" extracted by the proposed solver based on $\| C - C' \|_F / \| C \|_F$, and C is not available within feasible computational resources when the number of unknowns is too large, the error in Fig. 8(b) was only plotted up to 253792 unknowns. In Fig. 8(c), we plot the inverse time and the total CPU time of the proposed direct solver with respect to N. Clearly a linear complexity can be observed. For comparison, the solution time of the HiCap [2] is also plotted. The advantage of the proposed direct solver is clearly demonstrated even though HiCap only calculated the results for m right hand sides with m being the number of conductors, whereas the proposed solver obtained the entire inverse, i.e., the results for N right hand sides. In Fig. 8(d), we plot the memory complexity of the proposed solver, which again demonstrates a linear complexity.

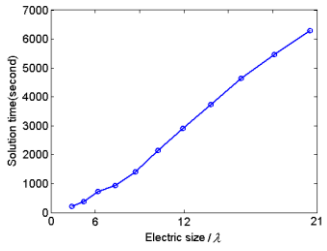

Fig. 9. Simulation of a 4λ–20λ plate.

The proposed $\mathcal{H}^2$-matrix-based method is kernel independent, and hence is equally applicable to electrodynamic problems. Using the proposed solver, we simulated a square plate having electric size from 4 wavelengths to 20 wavelengths. The simulation parameters were chosen as η =1, *leafsize*=20, and p=10. In Fig. 9, the CPU time was plotted with respect to the number of unknowns. Again, linear complexity is observed. In addition, the error $\| \mathbf{G} - \tilde{\mathbf{G}} \|_F / \| \mathbf{G} \|_F$ across all the electric sizes is smaller than 3.5e-3; and the error of the inverse matrix, which is $\| \mathbf{I} - \tilde{\mathbf{G}}^{-1} \mathbf{G} \|$, is smaller than 3.3%. For full-wave cases, the rank k used in the $\mathcal{H}^2$-based representation needs to be determined adaptively [8] in order to keep a constant order of accuracy across electric sizes without compromising the computational complexity.

5. CONCLUSIONS

A linear-complexity direct inverse was developed for fast integral-equation-based analysis of quasi-static and electro-dynamic systems. Numerical results demonstrated its superior performance.

6. REFERENCES

[1] K. Nabors and J. White, "FastCap: A multipole accelerated 3-d capacitance extraction program," *IEEE Trans. on CAD*, pp.1447–1459, 1991.

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

[3] S. Yan, V. Saren, and W. Shi, "Sparse Transformations and Preconditioners for Hierarchical 3-D Capacitance Extraction with Multiple Dielectrics," DAC 2004, pp. 788-793.

[4] D. Gope, I. Chowdhury, and V. Jandhyala, "DiMES: Multilevel fast direct solver based on multipole expansions for parasitic extraction of massively coupled 3D microelectronic structures," DAC 2005, pp. 159-162.

[5] S. Borm, L. Grasedyck, and W. Hackbusch, "Hierarchical matrices," Lecture note 21 of the Max Planck Institute for Mathematics, 2003.

[6] S. Borm. "$\mathcal{H}^2$-matrix approximation of integral operators by interpolation," Applied Numerical Mathematics, 43: 129-143, 2002.

[7] W. Chai and D. Jiao, "An $\mathcal{H}^2$-Matrix-Based Integral-Equation Solver of Linear-Complexity for Large-Scale Full-Wave Modeling of 3D Circuits," *IEEE 17th Conference on Electrical Performance of Electronic Packaging* (EPEP), pp. 283-286, Oct. 2008.

[8] W. Chai and D. Jiao, "An $\mathcal{H}^2$-Matrix-Based Integral-Equation Solver of Reduced Complexity and Controlled Accuracy for Solving Electrodynamic Problems," accepted for publication, *IEEE Trans. Antennas Propagat.*, 2009.