An LU Decomposition Based Direct Integral Equation Solver of Linear Complexity and Higher-Order Accuracy for Large-Scale Interconnect Extraction

Wenwen Chai and Dan Jiao, *Senior Member, IEEE*

Abstract—**A fast LU factorization of linear complexity is developed to directly solve a dense system of linear equations for the capacitance extraction of any arbitrary shaped 3-D structure embedded in inhomogeneous materials. In addition, a higher-order scheme is developed to achieve any higher-order accuracy for the proposed fast solver without sacrificing its linear computational complexity. The proposed solver successfully factorizes dense matrices that involve more than one million unknowns in fast CPU run time and modest memory consumption. Comparisons with state-of-the-art integral-equation-based capacitance solvers have demonstrated its clear advantages. In addition to capacitance extraction, the proposed LU solver has been successfully applied to large-scale full-wave extraction.**

Index Terms—**Direct solver, fast solver, integral-equation-based methods, interconnect extraction, LU factorization.**

I. Introduction

In RECENT years, the semiconductor industry has turned to the exponential growth in transistor density to sustain the continued growth of computational performance; the new transistors added every 18–24 months are used in the form of additional cores. All these transistors are connected via interconnect lines. As a result, the interconnect extraction tools are required to solve larger and larger problems. In general, an extraction tool casts a physical problem into a matrix equation $\mathbf{A}x = b$ to solve, with $\mathbf{A}$ being either dense or sparse. To solve $\mathbf{A}$ of dimension N, generally speaking, the optimal computational complexity one can hope for is linear complexity $O(N)$. However, no linear complexity has been reported for direct matrix factorizations for general problems. State-of-the-art methods rely on iterative solvers to solve large-scale problems [1]–[6], the optimal complexity of which is $O(N_{\rm it}N_{\rm rhs}N)$, where $N_{\rm it}$ is the number of iterations, and $N_{\rm rhs}$ is the number of right hand sides. When the number of iterations or the number of right-hand sides is large, iterative solvers become inefficient. Fast direct integral-equation-based solvers have also been developed [7], [8] for interconnect extraction. In [8], we developed a fast inverse of linear complexity to solve a dense system of linear equations directly for capacitance extraction. In general, the number of right hands encountered in interconnect extraction is less than N, and hence an LU factorization based matrix solution is more efficient than an inverse based matrix solution. Therefore, in this work, we develop an LU factorization of linear complexity for integral-equation (IE)-based analysis of large-scale interconnects embedded in inhomogeneous materials. In addition, we develop higher-order schemes to achieve a higher-order accuracy for interconnect extraction. Little work has been reported on higher-order schemes for interconnect extraction. With the linear complexity and higher-order accuracy achieved, we pave the way for the precision extraction of very large-scale on-chip interconnects involving thousands and thousands of conductors.

In the proposed approach, the dense matrix resulting from an IE-based analysis of interconnects is represented as an $\mathcal{H}^2$ matrix [8]. The $\mathcal{H}^2$ matrix is a general mathematical framework that enables a compact representation and efficient numerical computation of the dense matrices [9]–[13]. It was shown that the storage requirements and matrix–vector products of $\mathcal{H}^2$ matrices are of complexity $O(N)$. However, the $O(N)$ complexity has yet not been established for LU factorization. In this work, after representing the dense system matrix by an $\mathcal{H}^2$ matrix, a higher-order scheme is developed to achieve any higher-order accuracy for the $\mathcal{H}^2$-based representation without sacrificing computational complexity. The dense matrix of $O(N^2)$ parameters is then compactly stored by a reduced set of parameters of $O(N)$. An LU factorization of linear complexity is then developed through orthogonal nested cluster bases, linear algebraic techniques, and efficient matrix multiplication. The proposed methods are applicable to arbitrary 3-D structures embedded in inhomogeneous materials. No structure regularity and problem specialty are employed, and hence the methods are general. In addition, they are applicable to any IE-based formulation.

II. Background

A. Integral Equation for Capacitance Extraction

The proposed LU-based direct IE solver is kernel independent, and hence suitable for any IE-based formulation. In the following, we use IE formulations for capacitance extraction as an example to illustrate the basic idea.

Consider multiple conductors embedded in an inhomogeneous material, the following integral equation can be derived [3]:

$$\mathbf{\Phi}(\vec{r}) = \int_{S_c} \sigma_c(\vec{r}')g(\vec{r},\vec{r}')dr' + \int_{S_d} \sigma_d(\vec{r}')g(\vec{r},\vec{r}')dr' \qquad (1)$$

Manuscript received December 31, 2009; revised April 13, 2010; accepted June 03, 2010. Date of publication July 26, 2010; date of current version January 07, 2011. This work was supported in part by a grant from Office of Naval Research under Award N00014-10-1-0482, in part by a grant from Intel Corporation, and in part by a grant from NSF under Award 0747578. This work was recommended for publication by Associate Editor J. Tan upon evaluation of the reviewers comments.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TADVP.2010.2053537

where σ_c is the surface charge density on conducting surfaces S_c, σ_d is the polarization charge density on dielectric interfaces S_d, Φ is the electric scalar potential, $\vec{r}$ is the observation point, $\vec{r}'$ is the source point, and g is the Green's function

$$g(\vec{r}, \vec{r}') = \frac{1}{4\pi\varepsilon|\vec{r} - \vec{r}'|} \tag{2}$$

where ε is the permittivity of the dielectric material. The following boundary condition needs to be satisfied at the dielectric interface that separates two regions:

$$\varepsilon_a \partial\Phi_a(\vec{r})/\partial n = \varepsilon_b \partial\Phi_b(\vec{r})/\partial n \tag{3}$$

where $\hat{n}$ is a unit vector normal to the dielectric interface at point $\vec{r}$, and ε_a and ε_b are the permittivity in the two regions.

We discretize conductor–dielectric and dielectric–dielectric interfaces into $N = N_c + N_d$ small panels, with N_c panels residing on conductor–dielectric interfaces and N_d panels on dielectric–dielectric interfaces. By using a pulse function as the basis function, a method-of-moment solution of (1) and (3) results in a dense linear system

$$\mathbf{G}q = v \tag{4}$$

where $\mathbf{G} = [\begin{smallmatrix}\mathbf{P}_{cc} & \mathbf{P}_{cd} \\ \mathbf{E}_{dc} & \mathbf{E}_{dd}\end{smallmatrix}]$, $q = [\begin{smallmatrix}q_c \\ q_d\end{smallmatrix}]$, and $v = [\begin{smallmatrix}v_c \\ 0\end{smallmatrix}]$, in which 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

$$\begin{aligned}\mathbf{P}_{ij} &= \frac{1}{a_i}\frac{1}{a_j}\int_{S_i}\int_{S_j} g(\vec{r}, \vec{r}')drdr' \\ \mathbf{E}_{ij} &= (\varepsilon_a - \varepsilon_b)\frac{\partial}{\partial n_a}\frac{1}{a_i}\frac{1}{a_j}\int_{S_i}\int_{S_j} g(\vec{r}, \vec{r}')drdr'\end{aligned} \tag{5}$$

where a_i and a_j are the areas of panel S_i and S_j, respectively. The diagonal entries of $\mathbf{E}_{dd}$ are $\mathbf{E}_{ij} = (\varepsilon_a + \varepsilon_b)/(2a_i\varepsilon_0)$.

In a uniform dielectric, (4) is reduced to

$$\mathbf{P}_{cc}q_c = v_c. \tag{6}$$

A conventional way to solve (4) could be very expensive since the entries of $\mathbf{G}$ are all nonzero. In the following section, we introduce the $\mathcal{H}^2$ matrix, which enables a data sparse representation of $\mathbf{G}$.

B. *Definition of the $\mathcal{H}^2$ Matrix*

An $\mathcal{H}^2$ matrix is defined based on an important concept: admissibility condition. Denoting the full index set of all the panels used for the discretization of (4) by $\mathcal{I} := \{1, 2, \ldots, N\}$, for arbitrary two subsets t and s of $\mathcal{I}$, the strong admissibility condition is defined as [10]

$$(t, s)\text{are admissible} \\ := \begin{cases}\text{True, if } \max\{\text{diam}\,(Q_t), \text{diam}\,(Q_s)\} \le \eta\,\text{dist}(Q_t, Q_s) \\ \text{False, otherwise}\end{cases} \tag{7}$$

in which Q_t and Q_s are, respectively, the supports of the union of all the basis functions in t and s, $\text{diam}(\cdot)$ is the Euclidean diameter of a set, $\text{dist}(\cdot,\cdot)$ is the Euclidean distance of two sets, and η is a parameter that controls the admissibility condition. If two subsets t and s satisfy the strong admissibility condition (7), the corresponding matrix block formed by t and s can be replaced by a degenerate approximation, and hence becomes admissible. Otherwise, the matrix block is inadmissible. An $\mathcal{H}^2$ matrix is composed of admissible matrix blocks and inadmissible matrix blocks. The admissible block has a factorized form $\tilde{\mathbf{G}}^{t,s} := \mathbf{V}^t\mathbf{S}^{t,s}\mathbf{V}^{s^{\mathrm{T}}}$, where $\mathbf{V}^t \in \mathbb{C}^{m\times k_t}$, $\mathbf{S}^{t,s} \in \mathbb{C}^{k_t\times k_s}$, $\mathbf{V}^s \in \mathbb{C}^{n\times k_s}$ with k_t and k_s less than m and n. An $\mathcal{H}^2$ matrix also possesses a nested structure [9]–[13].

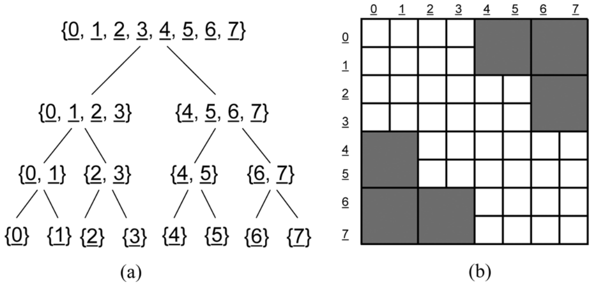

Fig. 1. (a) Cluster tree. (b) $\mathcal{H}^2$-matrix structure.

C. *Cluster Tree and Block Cluster Tree*

To obtain an $\mathcal{H}^2$ matrix representation, we need to use an efficient approach to partition a dense matrix into admissible blocks and inadmissible blocks. In [14] and [15], we have shown how to efficiently carry out the $\mathcal{H}^2$ matrix partition for a dense matrix resulting from an IE-based analysis of electromagnetic problems. The basic scheme is outlined as follows.

To construct a cluster tree, we start from the root cluster which is the full index set. We then find a disjoint partition of the index set and use this partition to create children clusters. We continue this procedure until the index number in each cluster is less than the *leafsize* which is a parameter to control the depth of the tree. This process generates a cluster tree shown in Fig. 1(a). Clusters with indices no more than *leafsize* are leaves. We denote a cluster tree constructed for the index set $\mathcal{I}$ by $T_\mathcal{I}$. A block cluster tree $T_{\mathcal{I}\times\mathcal{I}}$ is constructed from cluster trees $T_\mathcal{I}$ and $T_\mathcal{I}$ based on a given admissibility condition as the following. We test blocks level by level starting with the root clusters of $T_\mathcal{I}$ and $T_\mathcal{I}$, and descending in the tree. Given two clusters $t \in T_\mathcal{I}$ and $s \in T_\mathcal{I}$, we check the admissibility condition. If the two clusters are admissible, we stop at this level and do not check the admissibility of their children. If they are not admissible, we repeat the procedure for all combinations of the children of t and s. The resultant block cluster tree 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 a degenerate approximation is used and the unshaded ones are inadmissible blocks in which the original full matrix form is kept.

Based on the constructed cluster tree and block cluster tree, we introduce the following concepts and notations: 1) for each cluster $t \in T_\mathcal{I}$, the cardinality of the sets $\text{col}(t) := \{s \in T_\mathcal{I} :$

$(t,s) \in T_{\mathcal{I}\times\mathcal{I}}\}$ and $\text{row}(s) := \{t \in T_{\mathcal{I}} : \quad (t,s) \in T_{\mathcal{I}\times\mathcal{I}}\}$ is bounded by a constant C_{sp} ([10 p. 124]); 2) the *leafsize* is denoted by $n_{\min}$, and $\#t \leq n_{\min}$ if t is a leaf cluster; 3) the depth of a cluster tree is denoted by $\boldsymbol{L}$.

III. $\mathcal{H}^2$-Representation With Controlled Accuracy

In this section, we derive an $\mathcal{H}^2$-representation of $\mathbf{G}$ in (4) and analyze its accuracy. The error bound for an $\mathcal{H}^2$-representation of full-wave problems is also given.

A. $\mathcal{H}^2$-Representation

We choose two subsets t and s from the full index set of all the panels. If t and s satisfy the strong admissibility condition (7), the original kernel function $g(\vec{r},\vec{r}')$ in (2) can be replaced by a degenerate approximation

$$\tilde{g}^{t,s}(\vec{r},\vec{r}') = \sum_{v\in K^t}\sum_{\mu\in K^s} g\left(\xi_v^t,\xi_\mu^s\right) L_v^t(\vec{r}) L_\mu^s(\vec{r}') \tag{8}$$

where $K := \{v \in \mathrm{N}^d : v_i \leq p \quad \text{for all} \quad i \in \{1,\ldots,d\}\} = \{1,\ldots,p\}^d, \quad d = 1,2,3$, for 1-D, 2-D, and 3-D problems, respectively; p is the number of interpolation points; $(\xi_v^t)_{v\in K^t}$ 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 (8), (5) can be written as

$$\begin{aligned}\tilde{\mathbf{P}}_{ij}^{t,s} :&= \sum_{v\in K^t}\sum_{\mu\in K^s} 1/(a_i a_j)\\ &\quad\cdot g\left(\xi_v^t,\xi_\mu^s\right)\int_{S_i} L_v^t(\vec{r})dr \cdot \int_{S_j} L_\mu^s(\vec{r}')dr' \qquad (9)\\ \tilde{\mathbf{E}}_{ij}^{t,s} &= \sum_{v\in K^t}\sum_{\mu\in K^s} 1/(a_i a_j)\\ &\quad\cdot(\varepsilon_a-\varepsilon_b)\frac{\partial}{\partial n_a} g\left(\xi_v^t,\xi_\mu^s\right)\int_{S_i} L_v^t(\vec{r})dr\\ &\quad\cdot\int_{S_j} L_\mu^s(\vec{r}')dr' \qquad (10)\end{aligned}$$

from which the submatrix $\tilde{\mathbf{G}}^{t,s}$ in (4) can be written in a factorized form

$$\begin{aligned}\tilde{\mathbf{G}}^{t,s} &:= \mathbf{V}^t\mathbf{S}^{t,s}\mathbf{V}^{s^{\mathrm{T}}}, \quad \mathbf{V}^t \in \mathbb{R}^{\#t\times\#K^t},\\ \mathbf{S}^{t,s} &\in \mathbb{R}^{\#K^t\times\#K^s}, \quad \mathbf{V}^s \in \mathbb{R}^{\#s\times\#K^s},\end{aligned} \tag{11}$$

where

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

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

for $i \in t, j \in s, v \in K^t, \mu \in K^s$. In (11), "#" denotes the cardinality of a set.

The matrix form in (11) is an $\mathcal{H}^2$-matrix-based representation of $\mathbf{G}$ if the same space of polynomials are used across t and s. $\mathbf{V}^t(\mathbf{V}^s)$ are called cluster bases, and $\mathbf{S}$ is called coupling matrix.

B. Accuracy of an $\mathcal{H}^2$-Representation

1) Frequency-Independent Kernels: The kernel function in (2) is frequency-independent. Using the error analysis developed in [14], we can find that if the admissibility condition given in (7) is satisfied, the error of (8) is bounded by

$$\begin{aligned}&\left\|g(r,r') - \tilde{g}^{(t,s)}(r,r')\right\|_{\infty,Q_t\times Q_s}\\ &\leq \frac{4ed}{\pi}(\Lambda_p)^{2d} p\frac{1}{\text{dist}(Q_t,Q_s)}[1+\sqrt{2}\eta]\left[1+\frac{\sqrt{2}}{\eta}\right]^{-p}\end{aligned} \tag{12}$$

where Λ_p is a constant related to p and the interpolation scheme. Clearly, exponential convergence with respect to p can be obtained irrespective of the choice of η. Since $\mathbf{G}_{ij}^{(t,s)}$ is proportional to $1/\text{dist}(Q_t,Q_s)$, the relative error becomes a constant with respect to η. The smaller η is, the smaller the error. This also agrees with the fact that a smaller η leads to a stronger admissibility condition. However, too small an η could make the $\mathcal{H}^2$-based computation inefficient. Based on our numerical experiments, $1 \leq \eta \leq 2$ is, in general, a good choice.

2) Frequency-Dependent Kernels: The error bound for an $\mathcal{H}^2$-based representation of frequency-dependent kernels is different from (12). For electrodynamic problems, the kernel function has the following form:

$$g(\overrightarrow{r},\overrightarrow{r}') = \frac{e^{-j\kappa|\overrightarrow{r}-\overrightarrow{r}'|}}{4\pi|\overrightarrow{r}-\overrightarrow{r}'|} \tag{13}$$

where κ is the wave number which is $\omega\sqrt{\mu\varepsilon}$ and ω is angular frequency. Based on the error analysis given in [14], we have

$$\begin{aligned}&\left\|g(r,r') - \tilde{g}^{(t,s)}(r,r')\right\|_{\infty,Q_t\times Q_s}\\ &\leq \frac{4ed}{\pi}(\Lambda_p)^{2d} p\frac{1}{\text{dist}(Q_t,Q_s)}\\ &\quad\times[1+\sqrt{2}\kappa\eta\,\text{dist}(Q_t,Q_s)+\sqrt{2}\eta]\\ &\quad\times\left[1+\frac{2}{\sqrt{2}\kappa\eta\,\text{dist}(Q_t,Q_s)+\sqrt{2}\eta}\right]^{-p}.\end{aligned} \tag{14}$$

Again, exponential convergence with respect to p can be obtained irrespective of the electric size $\kappa\eta\,\text{dist}(Q_t,Q_s)$ and η. In addition, given a required level of accuracy, when the electric size increases, the error of the $\mathcal{H}^2$-matrix approximation can be controlled to the same level either by decreasing η to maintain a constant $\kappa\eta\,\text{dist}(Q_t,Q_s)$, or by adaptively increasing the number of interpolation points p, or by the combination of both. It's obvious that (12) is a special case of (14). If κ is zero, (14) becomes (12).

IV. Proposed LU-Based Direct IE Solver of Linear Complexity and Higher-Order Accuracy

A. Higher-Order Scheme

As can be seen from (12), the larger p is, the smaller the error is. If p is chosen to be 1, (9) and (10) are the same as center-point-based integrations. In a very general sense, the work reported in [2] for capacitance extraction can be viewed as an $\mathcal{H}^2$-based matrix–vector multiplication having $p = 1$. Generally speaking, the accuracy offered by $p = 1$ is sufficient for static applications as can be seen from results reported in

[2]. However, if a higher-order accuracy is required, larger p is needed. In the following, we give three schemes that can efficiently achieve a higher-order accuracy without compromising the linear computational complexity. Users can employ these schemes to achieve a higher-order accuracy for capacitance extraction in both matrix–vector multiplication based iterative solvers developed in the past and LU-factorization based direct solvers proposed in this work.

1) Scheme 1: For general structures that have similar dimensions in x-, y-, and z-directions, we can use the same number of interpolation points p in each direction, the rank in one matrix block $\tilde{\mathbf{G}}^{t,s}$ shown in (11), k, is hence p^d, with $d = 1, 2, 3$, for 1-D, 2-D, and 3-D problems, respectively.

2) Scheme 2: For integrated circuits or other multilayered structures in which the dimension along one direction (stack-growth direction) is a constant, we can choose a fixed p along this direction, for example $p = 1$, and use larger p in the other two directions. By doing so, the rank in each $\mathcal{H}^2$ block is reduced to p^{d-1}. Compared to scheme 1, this scheme offers better efficiency while achieving the same accuracy. Scheme 2 can be generalized to $k = p_x \times p_y \times p_z$, i.e., having a different number of interpolation points along each direction.

3) Scheme 3: Both scheme 1 and scheme 2 are based on a constant rank, i.e., the same rank is used across all the admissible blocks when ascending an inverted tree. This is not the most efficient way because a larger rank is only needed at a lower level in an inverted tree. Therefore, we develop a rank function to adaptively determine the rank based on tree level. Compared to the constant-rank schemes, this scheme is computationally more efficient.

The rank function is defined as a polynomial order function which increases with the decreased tree level (assuming tree level is zero at the root). For an admissible block $b = (t, s)$, the rank function for each cluster t and s is defined as

$$p(b) = \hat{a} + \hat{b}(L - l(b)) \tag{15}$$

where

$$\begin{aligned} L &= L_{\min} = \min\{\text{level}(\tau) : \tau \in \mathcal{L}_{\mathcal{I}}\} \\ l(b) &= \text{level}(t) = \text{level}(s)) \\ p(b) &= \hat{a} \text{ if } L \leq l(b) \end{aligned} \tag{16}$$

and $\hat{a}, \quad \hat{b} \in \mathbb{N}_0$ are two constants that are used to control the accuracy. Given a cluster t in an $\mathcal{H}^2$ tree, the rank of $\mathbf{V} = (\mathbf{V}^t)_{t \in T_{\mathcal{I}}}$ in (11) becomes a variable $k_{\text{var}}(t)$, which can be determined from (16) as

$$k_{\text{var}}(t) = p^d(t) = [\hat{a} + \hat{b}(L - l(t))]^d. \tag{17}$$

Clearly, schemes 1 and 2 can be viewed as special cases of (17), with $\hat{b}$ chosen to be zero.

For a fixed rank, it was shown that the storage requirement of $\mathcal{H}^2$ matrices is $O(N)$ [9]–[11]. By employing a variable-order rank, the $\mathcal{H}^2$-based $\mathbf{G}$ can still be stored in $O(N)$ parameters, the proof of which is given in [14]. By using (17) to determine rank k, we can achieve an $\mathcal{H}^2$-based representation of any higher-order accuracy without compromising the linear computational complexity of the proposed solver, which is proved in the next subsection.

B. $O(N)$ LU-Based Direct Matrix Solution

1) Algorithms: From (4), it can be seen that the computing task is to evaluate $\mathbf{G}^{-1}v$ instead of $\mathbf{G}^{-1}$, and hence an LU decomposition based direct matrix solution is, in general, more efficient than a direct inverse. In this section, we give a number of pseudo-codes to show a detailed procedure of performing a linear-complexity LU factorization, which are not reported anywhere else. In addition, we give a detailed complexity analysis.

We first orthogonalize the cluster bases $\mathbf{V} = (\mathbf{V}^t)_{t \in T_{\mathcal{I}}}$, and meanwhile preserving the nested properties using $\mathcal{H}^2$ arithmetics [10], [16]. This step is done in linear complexity. The new set of cluster bases are denoted by $(\tilde{\mathbf{V}}^t)_{t \in T_{\mathcal{I}}}$, which satisfy $\tilde{\mathbf{V}}^{t^{\mathrm{T}}} \tilde{\mathbf{V}}^t = \mathbf{I}$.

Next, we perform LU decomposition. The LU decomposition of the $\mathcal{H}^2$-based $\mathbf{G}$ can be defined as

$$\begin{aligned} \mathbf{G} &= \begin{bmatrix} \mathbf{G_{11}} & \mathbf{G_{12}} \\ \mathbf{G_{21}} & \mathbf{G_{22}} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{L}_{11} & 0 \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ 0 & \mathbf{U}_{22} \end{bmatrix} = \mathbf{L}_{\mathrm{H}^2} \mathbf{U}_{\mathrm{H}^2} \end{aligned} \tag{18}$$

```
LU Decomposition $\mathbf{G} = \mathbf{LU}$
Procedure $\boldsymbol{\mathcal{H}^2}-\mathbf{LU}(\mathbf{G})$ ($\mathbf{G}$ is the input matrix overwritten by $\mathbf{L}$ and $\mathbf{U}$)
if matrix $\mathbf{G}$ is a nonleaf matrix block
    $\boldsymbol{\mathcal{H}^2}-\mathbf{LU}(\mathbf{G}_{11})$
    Solve_LX$(\mathbf{L}_{11}, \mathbf{G}_{12}) \rightarrow \mathbf{U}_{12}$                    (19)
    Solve_XU$(\mathbf{G}_{21}, \mathbf{U}_{11}) \rightarrow \mathbf{L}_{21}$
    $-\mathbf{L}_{21} \times \mathbf{U}_{12} + \mathbf{G}_{22} \rightarrow \mathbf{G}_{22}$
    $\boldsymbol{\mathcal{H}^2}-\mathbf{LU}(\mathbf{G}_{22})$
else
    Full-LU$(\mathbf{G})$ (normal full matrix LU decomposition)
```

where $\mathbf{L}_{\mathrm{H}^2}$ and $\mathbf{U}_{\mathrm{H}^2}$ are both $\mathcal{H}^2$ matrices and stored in the same $\mathcal{H}^2$ format as $\mathbf{G}$. The LU factorization can be recursively realized by the pseudo-code shown in (19).

In (19), the function $\mathbf{Solve\_LX}(\mathbf{L}, \mathbf{G})$ is to solve a lower triangular system $\mathbf{LX} = \mathbf{G}$, where $\mathbf{L}$ and $\mathbf{G}$ are input matrices having $\mathcal{H}^2$ representations, and $\mathbf{G}$ is overwritten by solution $\mathbf{X}$. This function can be derived from

$$\begin{aligned} \mathbf{LX} = \mathbf{G} &\rightarrow \begin{bmatrix} \mathbf{L}_{11} & 0 \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{G_{11}} & \mathbf{G_{12}} \\ \mathbf{G_{21}} & \mathbf{G_{22}} \end{bmatrix} \rightarrow \begin{cases} \mathbf{L}_{11}\mathbf{X}_{11} = \mathbf{G_{11}} \\ \mathbf{L}_{11}\mathbf{X}_{12} = \mathbf{G_{12}} \\ \mathbf{L}_{22}\mathbf{X}_{21} = \mathbf{G_{21}} - \mathbf{L}_{21}\mathbf{X_{11}} \\ \mathbf{L}_{22}\mathbf{X}_{22} = \mathbf{G_{22}} - \mathbf{L}_{21}\mathbf{X_{12}} \end{cases} \end{aligned} \tag{20}$$

```
Recursively Solve Triangular System $\mathbf{LX}=\mathbf{G}$
Procedure $\mathbf{Solve\_LX}(\mathbf{L},\mathbf{G})$
($\mathbf{L},\mathbf{G}$ are input matrices, $\mathbf{G}$ is overwritten by $\mathbf{X}$)
If matrix $\mathbf{L}$ is a nonleaf matrix block
    If $\mathbf{G}$ is a nonleaf matrix block
        $\mathbf{Solve\_LX}(\mathbf{L}_{11},\mathbf{G}_{11}),\quad \mathbf{Solve\_LX}(\mathbf{L}_{11},\mathbf{G}_{12})$
        $-\mathbf{L}_{21}\times\mathbf{G}_{11}+\mathbf{G}_{21}\rightarrow\mathbf{G}_{21},\quad \mathbf{Solve\_LX}(\mathbf{L}_{22},\mathbf{G}_{21})$
        $-\mathbf{L}_{21}\times\mathbf{G}_{12}+\mathbf{G}_{22}\rightarrow\mathbf{G}_{22},\quad \mathbf{Solve\_LX}(\mathbf{L}_{22},\mathbf{G}_{22})$
    else if $\mathbf{G}$ is an admissible block
        $\mathrm{Split}(\mathbf{G}),\mathbf{Solve\_LX}(\mathbf{L},\mathbf{G}),\mathrm{Collect}(\mathbf{G})$        (21)
    else
        $\mathbf{Solve\_LF}(\mathbf{L},\mathbf{G})$
else
    If $\mathbf{G}=\mathbf{VSW}^{\mathrm{T}}$ is an admissible block
        $\mathbf{VS}\rightarrow\mathbf{F},\quad \mathbf{Full\_LX}(\mathbf{L},\mathbf{F}),\quad \mathbf{V}^{\mathrm{T}}\mathbf{F}\rightarrow\mathbf{S}^{\mathrm{new}}$
    else
        $\mathbf{Full\_LX}(\mathbf{L},\mathbf{G})$ (Solve full matrix triangular system)
```

which can be recursively computed from the pseudo-code shown in (21).

In (21), we employ the split operation and collect operation in [17], which has a linear complexity.

```
Recursively Solve Triangular System $\mathbf{LX}=\mathbf{F}$
Procedure $\mathbf{Solve\_LF}\,(\mathbf{L},\mathbf{F})$
($\mathbf{L},\mathbf{F}$ are input matrices, $\mathbf{F}$ is overwritten by $\mathbf{X}$)
If matrix $\mathbf{L}$ is a nonleaf matrix block
    $\mathbf{Solve\_LF}(\mathbf{L}_{11},\mathbf{F}_{1})$
    $-\mathbf{L}_{21}\times\mathbf{F}_{1}+\mathbf{F}_{2}\rightarrow\mathbf{F}_{2}$        (22)
    $\mathbf{Solve\_LF}(\mathbf{L}_{22},\mathbf{F}_{2})$
else
$\mathbf{Full\_LX}(\mathbf{L},\mathbf{F})$
```

The function $\mathbf{Solve\_XU}(\mathbf{G},\mathbf{U})$ is to solve an upper triangular system, which can be derived in a similar fashion as (21). The function $\mathbf{Solve\_LF}(\mathbf{L},\mathbf{G})$ in (21) is to solve the triangular system $\mathbf{LX}=\mathbf{G}$, with $\mathbf{G}$ being a full matrix and $\mathbf{L}$ an $\mathcal{H}^2$-matrix. It can be realized recursively from the pseudo-code shown in (22).

In (21) and (22), $\mathbf{Full\_LX}(\mathbf{L},\mathbf{F})$ is to solve a normal full matrix triangular system, which can be realized straightforwardly.

2) Complexity Analysis: From (19), it can be seen that the LU factorization is built upon two functions, $\mathbf{Solve\_LX}(\mathbf{L},\mathbf{G})$ and $\mathbf{Solve\_XU}(\mathbf{G},\mathbf{U})$. The former is a block forward substitution, and the latter is a block backward substitution. Both have the same complexity as a matrix–matrix multiplication. Take $\mathbf{Solve\_LX}(\mathbf{L},\mathbf{G})$ as an example, as shown in (21), if $\mathbf{L}$ is a full matrix block in the leaf level, then $\mathbf{Solve\_LF}(\mathbf{L},\mathbf{F})$ in (22) is performed.

Proof 1: $\mathrm{Comp}(\mathrm{Solve}\mathbf{LX}=\mathbf{F})=\mathrm{Comp}(\mathbf{L}\times\mathbf{F})$

1) As shown in (22), in the leaf level $l=0,\mathbf{L}$ is a full matrix block, and hence a normal full-matrix triangular system is solved. It is clear that $\mathrm{Comp}(\mathrm{Solve}\mathbf{L}^{l}\mathbf{X}^{l}=\mathbf{F}^{l})=\mathrm{Comp}(\mathbf{L}^{l}\times\mathbf{F}^{l})$, where "Comp" denotes complexity.
2) In the level $l,\mathbf{LX}=\mathbf{F}$ is computed from $\mathbf{L}_{11}\mathbf{X}_{1}=\mathbf{F}_{1}$ and $\mathbf{L}_{22}\mathbf{X}_{2}=\mathbf{F}_{2}-\mathbf{L}_{21}\mathbf{X}_{1}$. Since we have proved $\mathrm{Comp}(\mathrm{Solve}\mathbf{L}^{l-1}\mathbf{X}^{l-1}=\mathbf{F}^{l-1})=\mathrm{Comp}(\mathbf{L}^{l-1}\times\mathbf{F}^{l-1})$ in the level $l-1$, it is straightforward to see that each step in computing $\mathbf{LX}=\mathbf{F}$ has the same cost as block multiplication. And hence $\mathrm{Comp}(\mathrm{Solve}\mathbf{L}^{l}\mathbf{X}^{l}=\mathbf{F}^{l})=\mathrm{Comp}(\mathbf{L}^{l}\times\mathbf{F}^{l})$ in level l.

As a direct consequence of (1) and (2), $\mathrm{Comp}(\mathrm{Solve}\mathbf{LX}=\mathbf{F})=\mathrm{Comp}(\mathbf{L}\times\mathbf{F})$.

Proof 2: $\mathrm{Comp}(\mathrm{Solve}\mathbf{L}^{l}\mathbf{X}^{l}=\mathbf{G}^{l})=\mathrm{Comp}(\mathbf{L}^{l}\times\mathbf{G}^{l})$

With the complexity of $\mathbf{Solve\_LF}(\mathbf{L},\mathbf{F})$ derived, the complexity of $\mathbf{Solve\_LX}(\mathbf{L},\mathbf{G})$ can be analyzed based on (21) as follows.

1) It can be seen that if $\mathbf{L}$ is a full matrix block in the leaf level, $\mathbf{Solve\_LF}(\mathbf{L},\mathbf{G})$ is performed, and hence $\mathrm{Comp}(\mathrm{Solve}\mathbf{LX}=\mathbf{G})=\mathrm{Comp}(\mathbf{L}\times\mathbf{G})$.
2) In the nonleaf level, three cases are encountered:
    i) If $\mathbf{G}$ is a full matrix, $\mathbf{Solve\_LF}(\mathbf{L},\mathbf{G})$ is performed, then $\mathrm{Comp}(\mathrm{Solve}\mathbf{LX}=\mathbf{G})=\mathrm{Comp}(\mathbf{L}\times\mathbf{G})$.
    ii) If $\mathbf{G}$ is an admissible block, the process shown in (21) is exactly the same as an $\mathcal{H}^2$-based matrix multiplication $\mathrm{nonleaf}\times\mathrm{admissible}\rightarrow\mathrm{admissible}$.
    iii) If $\mathbf{G}$ is a nonleaf block, $\mathbf{LX}=\mathbf{G}$ has the form defined in (20). We have proved that in the level $l-1,\mathrm{Comp}(\mathrm{Solve}\mathbf{L}^{l-1}\mathbf{X}^{l-1}=\mathbf{G}^{l-1})=\mathrm{Comp}(\mathbf{L}^{l-1}\times\mathbf{G}^{l-1})$, which means the complexity of each step in (20) is the same as that of matrix block multiplication, and therefore $\mathrm{Comp}(\mathrm{Solve}\mathbf{L}^{l}\mathbf{X}^{l}=\mathbf{G}^{l})=\mathrm{Comp}(\mathbf{L}^{l}\times\mathbf{G}^{l})$ can be obtained in level l.

From the aforementioned analysis, it can be seen that the computation of the recursive LU factorization is essentially a full-matrix LU factorization at the leaf level, a full-matrix solution of a lower triangular system at the leaf level, a full-matrix solution of an upper triangular system at the leaf level, and a number of matrix–matrix multiplications at the other levels, and hence it has a linear complexity since $\mathcal{H}^2$-based matrix–matrix multiplication has a linear complexity. To explain, consider

$\mathbf{G}^{b1} = \mathbf{V}^t\mathbf{S}_{b1}\mathbf{V}^{s^{\mathrm{T}}}$ and $\mathbf{G}^{b2} = \mathbf{V}^s\mathbf{S}_{b2}\mathbf{V}^{r^{\mathrm{T}}}$, then due to the orthogonality of cluster bases $(\mathbf{V}^t)_{t\in T_{\mathcal{I}}}$, we have

$$\begin{aligned}\mathbf{G}^{b1} \times \mathbf{G}^{b2} &= \mathbf{V}^t\mathbf{S}_{b1}\mathbf{V}^{s^{\mathrm{T}}} \times \mathbf{V}^s\mathbf{S}_{b2}\mathbf{V}^{r^{\mathrm{T}}}\\ &= \mathbf{V}^t\mathbf{S}_{b1}\mathbf{I}\mathbf{S}_{b2}\mathbf{V}^r = \mathbf{V}^t(\mathbf{S}_{b1}\mathbf{S}_{b2})\mathbf{V}^r\\ &= \mathbf{V}^t\mathbf{S}_b^{\mathrm{new}}\mathbf{V}^r. \end{aligned}\tag{23}$$

Since the dimension of each of $\mathbf{S}_{b1}$ and $\mathbf{S}_{b2}$ is $k \times k$, the complexity of computing $\mathbf{S}_b^{\mathrm{new}}$ is at most $O(k^3)$. The recursive matrix–matrix multiplication is essentially a procedure that 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 with $k_1 = \max(k, n_{\min})$, and there are at most $2^l C_{sp}$ matrix blocks in level l and each block is computed at most C_{sp} times by multiplication, we obtain a linear cost for matrix multiplication.

With the higher-order scheme developed in Section IV-A, the complexity of computing $\mathbf{S}_b^{\mathrm{new}}$ in (23) becomes $O(k_{\mathrm{var}}^3)$. We can still prove that the complexity of matrix–matrix multiplication is linear as the following:

$$\begin{aligned}&\mathrm{Comp(Multiplication)}\\ &\quad= \sum_{l=0}^{L}(\#\text{blocks at level } l)O\left(C_{sp}\,k_{\mathrm{var}}^3\right)\\ &\quad\le \sum_{l=0}^{L} 2^l C_{sp}^2 O\left(k_{\mathrm{var}}^3\right)\\ &\quad\le C_{sp}^2 O\left(\sum_{l=0}^{L}[\hat{a}+\hat{b}(L-l)]^{3-D}\cdot 2^l\right)\\ &\quad\le C_{sp}^2 O((\hat{a}+\hat{b})^{3-D})\sum_{l=0}^{L}(1+L-l)^{3-D}\cdot 2^l\\ &\quad\le C_{sp}^2 O((\hat{a}+\hat{b})^{3-D})\cdot 2^L\cdot\sum_{l=0}^{L}(l+1)^{3-D}\cdot 2^{-l}\\ &\quad\le C_{sp}^2 O\left(\left(\frac{2d(\hat{a}+\hat{b})}{\ln 1.5}\right)^{3-D}\right)\cdot N \end{aligned}\tag{24}$$

3) Compute Capacitance Matrix: After $\mathbf{G}$ matrix is factorized as $\mathbf{G} = \mathbf{LU}$, matrix equation (4) can be solved in two steps: 1) Solve the lower triangular system $\mathbf{L}x = v$; 2) Solve the upper triangular system $\mathbf{U}q = x$. These two can be performed in linear complexity following a procedure similar to (21). By adding all the entries of q for each conductor, the corresponding capacitance matrix element can be obtained.

C. Discussions

The accuracy and efficiency of the proposed method are controllable. They are controlled by four simulation parameters: η, leafsize $n_{\min}$, $\hat{a}$ and $\hat{b}$. The $\hat{a}$ and $\hat{b}$ are used to determine p. From (12), the smaller η is and the larger p is, the better the accuracy is. For static problems, $1 \le \eta \le 2$ is generally enough for achieving a good accuracy; for high frequency problems, $0.2 \le \eta \le 1$ can be chosen. Leafsize, $n_{\min}$, and $\hat{a}$ can be determined together. Give a required level of accuracy across a range of frequencies, $\hat{a}$ can be determined by achieving the accuracy requirement at the low end of the frequency range, and $n_{\min}$ can be chosen based on $n_{\min} \ge 0.5\hat{a}^d$. This can help make $\mathcal{H}^2$-approximation most efficient in both memory and CPU time. The parameter $\hat{b}$ can be chosen between 0 and $\hat{a}$.

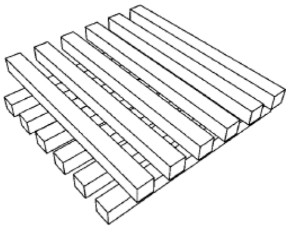

Fig. 2. Bus structure.

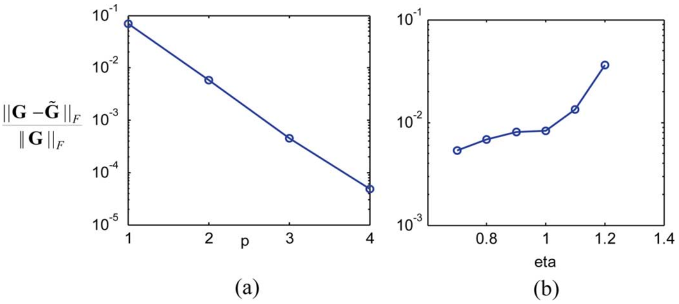

Fig. 3. (a) The effect of p on the $\mathcal{H}^2$-based representation ($\eta = 1$). (b) The effect of η on the $\mathcal{H}^2$-based representation ($p = 2 \times 2 \times 1$).

V. Numerical Results

A. Capacitance Extraction

We first tested the accuracy of the $\mathcal{H}^2$-based representation with respect to accuracy control parameters η and p. A bus structure shown in Fig. 2 involving nonuniform dielectrics was simulated [3]. The dielectric material surrounding the upper-layer conductors has relative permittivity of 3.9, and the lower-layer conductors are in the dielectric medium having relative permittivity 7.5. Each bus is scaled to $1 \times 1 \times (2m+1)$ (unit length)^3. The distance between buses in the same layer is 1 unit length, and the distance between the two bus layers is 2 unit lengths. In this example, a 10×10 bus was simulated, and the resultant number of unknowns was 11008.

In Fig. 3(a), we plotted the error of $\mathcal{H}^2$-based representation with respect to p with η fixed. An exponential convergence can be observed. In Fig. 3(b), we plotted the error of the $\mathcal{H}^2$-based representation with respect to η with p fixed. Clearly, the smaller η is, the better the accuracy. Fig. 3 agrees very well with the error bound we derived in (12).

We then tested the performance of the proposed LU-based direct IE solver. The first example considered was the structure shown in Fig. 2 but in a uniform material. The m in the $m \times m$ bus structure was from 4 to 30. The simulation parameters were chosen as *leafsize* $= 10, \quad \eta = 1.2, \quad p = 2\times 2\times 1$. In Fig. 4(a) and (b), we plotted the CPU time and memory consumption of the proposed LU-based direct solver in comparison with that cost by FastCap2.0, the convergence tolerance of which was set to be 1%. A clear speedup of the proposed solver is observed. In addition, both the time and memory complexity of the proposed solver are shown to be linear. In Fig. 4(c), we assessed the error

TABLE I
SOLUTION ERROR (FROBENIUS NORM) FOR THE UNIFORM DIELECTRIC CASE

Bus Size	4×4	6×6	8×8	10×10	12×12
Error	3.42E-02	4.73E-02	3.62E-02	6.52E-02	4.48E-02
Bus Size	14×14	16×16	18×18	20×20	22×22
Error	4.79E-02	3.17E-02	4.14E-02	1.29E-01	1.30E-01
Bus Size	24×24	26×26	28×28	30×30	
Error	5.66E-02	4.89E-02	4.90E-02	4.40E-02	

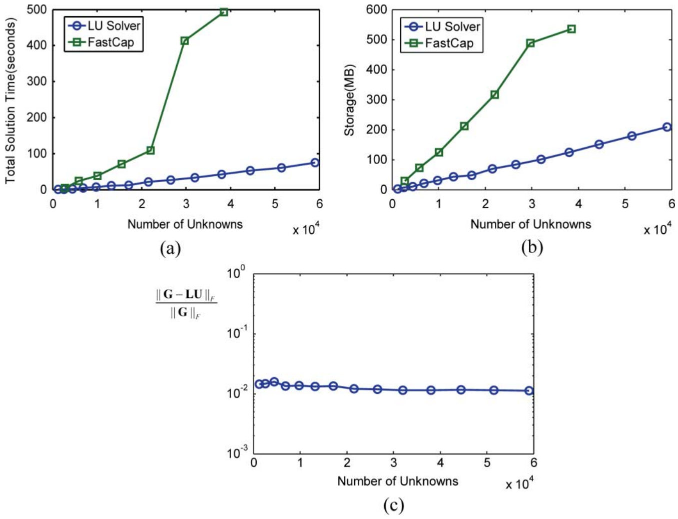

Fig. 4. Simulation of a bus structure in a uniform dielectric material. (a) CPU Time Comparison. (b) Memory Comparison. (c) Accuracy of the proposed LU factorization.

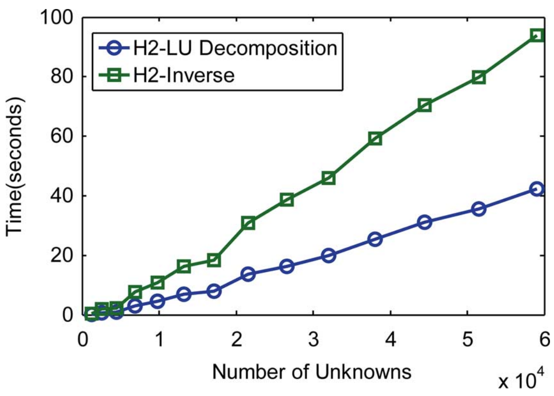

Fig. 5. Comparison between an inverse- and an LU-based direct solution.

of the proposed LU factorization, less than 2% error is observed in the entire range.

In Table I, we listed the solution error of the proposed direct solver with respect to the size of the bus structure, again, good accuracy is observed in the entire range. The solution error was defined as $\|\mathbf{G}q - v\|/\|v\|$, and used throughout the following examples.

For this example, we also implemented an $\mathcal{H}^2$-based direct inverse [8], and compared it with the proposed LU-based direct solver. As shown in Fig. 5, the LU-based direct solver costs less time. As mentioned earlier, for capacitance extraction, what is needed is $\mathbf{G}^{-1}v$ instead of $\mathbf{G}^{-1}$, and hence an LU-based solution is, in general, more efficient.

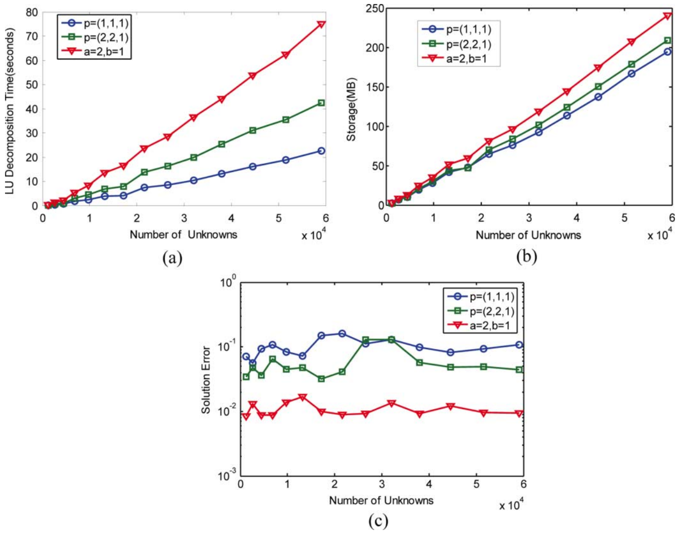

Fig. 6. Performance of the higher-order-scheme based LU solver. (a) LU Factorization time (b) Storage (c) Solution error.

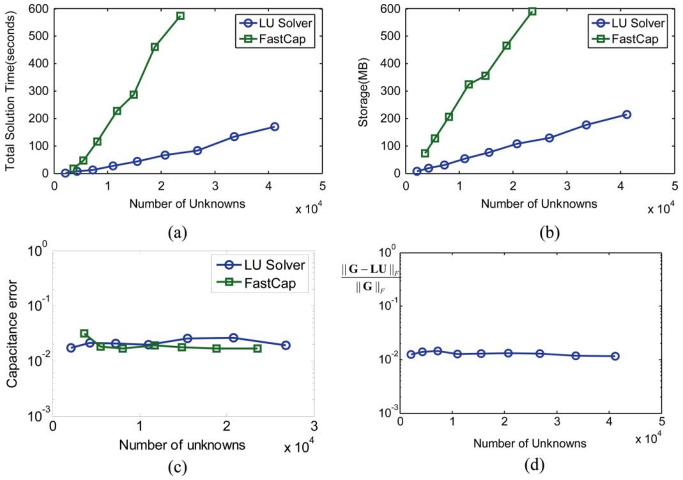

Fig. 7. Simulation of a bus structure embedded in a nonuniform dielectric. (a) CPU Time Comparison. (b) Memory Comparison. (c) Accuracy Comparison. (d) Accuracy of the proposed LU factorization.

Next, we tested the performance of the proposed higher-order scheme. The same example was simulated. The simulation parameters were chosen as *leafsize* $= 10$, $\eta = 1.2$. Three schemes were compared: a zeroth-order scheme having $p = 1 \times 1 \times 1$, a constant rank higher-order scheme having $p = 2 \times 2 \times 1$, and a variable-order-rank based high order scheme (17) having $\hat{a} = 2$, $\hat{b} = 1$. In Fig. 6, we plotted the CPU time, memory consumption and solution error of the three schemes. As can be seen clearly, all these three schemes have

TABLE II
SOLUTION ERROR (FROBENIUS NORM) FOR THE NONUNIFORM CASE

Bus Size	4×4	6×6	8×8	10×10	12×12
Error	3.12E-02	5.21E-02	5.26E-02	3.59E-02	7.01E-02
Bus Size	14×14	16×16	18×18	20×20	
Error	7.56E-02	5.68E-02	8.80E-02	6.60E-02	

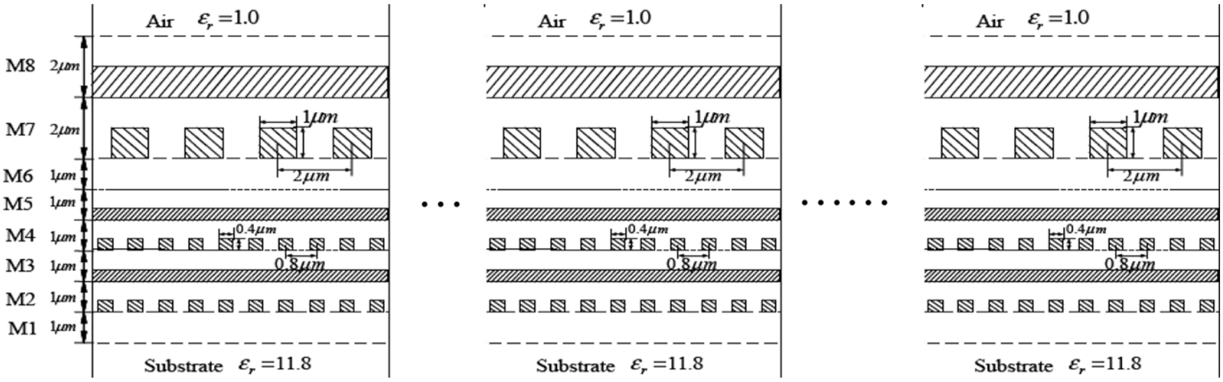

Fig. 8. On-chip interconnect having eight metal layers embedded in an inhomogeneous stack.

linear complexity in both CPU time and memory consumption, which agree very well with our theoretical complexity analysis. In addition, higher-order accuracy can be observed by the proposed higher-order schemes as can be seen in Fig. 6(c).

Next, we simulated the bus structure embedded in a nonuniform dielectric as that simulated for Fig. 3. The m was from 4 to 20. The simulation parameters were chosen as leafsize $=$ 10, $\eta = 1$, $p = 2\times2\times1$. In Fig. 7(a) and (b), we plotted the CPU time and memory consumption of the proposed LU-based direct solver in comparison with that cost by FastCap2.0. Again, a clear advantage of the proposed solver is observed. In addition, the time and memory complexity of the proposed solver are shown to be linear. In Fig. 7(c), we plot the error of the capacitance matrix generated by both solvers. The capacitance error is measured by $||\mathbf{C} - \mathbf{C}'||_F/||\mathbf{C}||_F$, where $\mathbf{C}$ is the capacitance matrix obtained from a full-matrix-based direct solver, and $\mathbf{C}'$ is that generated by the proposed solver or FastCap2.0. As can be seen clearly from Fig. 7(c), good accuracy of both solvers can be observed in capacitance matrix. In Fig. 7(d), we assessed the error of the proposed LU factorization, good accuracy is observed in the entire range. In Table II, we listed the solution error of the proposed solver with respect to the size of the bus structure, once again, good accuracy is observed in the entire range.

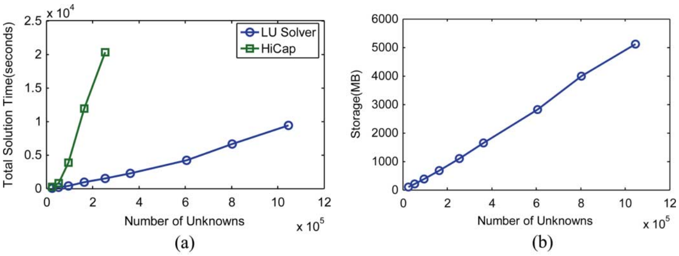

Fig. 9. Simulation of a 3-D on-chip interconnect from M1 to M8. (a) CPU time. (b) Storage.

TABLE III
SOLUTION ERROR (FROBENIUS NORM) FOR A 3-D M1–M8 ON-CHIP INTERCONNECTF

Num. of Conductor	Solution Error
25556	2.76E-02
53400	3.94E-02
94752	3.82E-02
164672	6.28E-02
253792	6.44E-02
362122	4.85E-02
605472	6.93E-02
802272	7.66E-02
1047236	6.48E-02

To test the performance of the proposed solver in simulating very large examples, we simulated a structure shown in Fig. 8 [3]. The relative permittivity is 3.9 in M1, 2.5 from M2 to M6, and 7.0 from M7 to M8. The disretization of the unit structure involving 48 conductors resulted in 25 556 unknowns. The unit structure was then duplicated horizontally, resulting in 72, 96, 120, 144, 192, 240, 288, and 336 conductors, which led to 1 047 236 unknowns. The simulation parameters were chosen as *leafsize* $= 10$, $\eta = 1$, $p = 2 \times 2 \times 1$. Fig. 9(a) shows the total solution time of the proposed solver. For comparison, the solution time of a HiCap [2], [3] based solver was also plotted, for which the number of conductors was simulated up to 144. We did not simulate more conductors using the HiCap based solver due to large run time, and also because the advantage of the proposed solver was already obvious. In Fig. 9(b), we show the linear storage of the proposed LU solver. In Table III, we list solution error versus the number of unknowns. Good accuracy is observed in the entire range. In addition, given any required

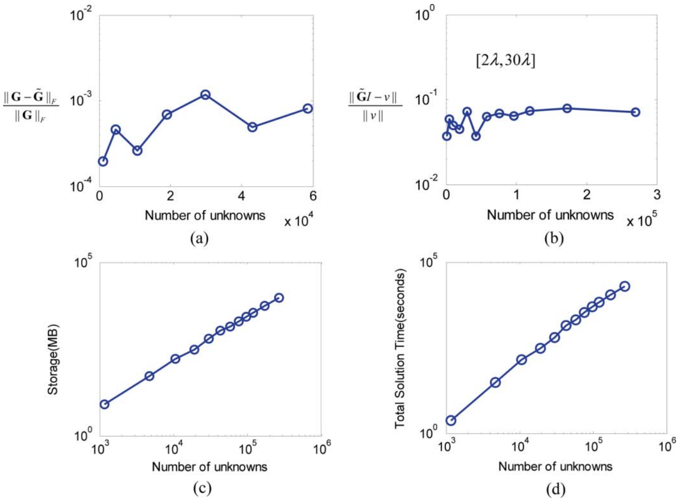

Fig. 10. Simulation of a plate from 2λ to 30λ. (a) Accuracy of $\mathcal{H}^2$ representation. (b) Solution error. (c) Storage. (d) CPU Time.

level of accuracy, the proposed solver can achieve it with the proposed higher-order scheme.

B. Full-Wave Extraction

A square plate with electric size from 2λ to 30λ was simulated, where λ is wavelength. An electric field integral equation was employed [14]. The simulation parameters were chosen as $\eta = 1$, $leafsize = 20$, $\hat{a} = 4$, and $\hat{b} = 1$. The accuracy of the $\mathcal{H}^2$ representation was plotted in Fig. 10(a), which shows almost a constant order of accuracy in the entire electric size range. Fig. 10(a) was only plotted up to ~60 k unknowns because measuring the matrix error requires the storage of the original full matrix $\mathbf{G}$. In Fig. 10(b), we plotted the solution error in the entire range of electric sizes, which is measured by $\|\tilde{\mathbf{G}}I - v\|/\|v\|$. Good accuracy is observed. In Fig. 10(c) and (d), we plotted the storage and CPU time of the proposed direct solver, respectively. A linear scaling can be clearly seen.

VI. Conclusion

An LU-factorization based direct integral equation solver of linear complexity and higher-order accuracy was developed for large-scale interconnect extraction. Comparsions with state-of-the-art iterative and direct IE-based capacitance solvers have demonstrated its superior performance. It is kernel independent, and hence suitable for any integral equation based formulation. In addition, it is applicable to arbitrary structures and nonuniform materials.

References

[1] K. Nabors and J. White, "FastCap: A multipole accelerated 3-D capacitance extraction program," *IEEE Trans. Computer Aided-Design*, vol. 11, pp. 1447–1459, Nov. 1991.

[2] W. Shi, J. Liu, and N. Kakani, "A fast hierarchical algorithm for 3-D capacitance extraction," *IEEE Trans. Computer-Aided Design*, vol. 21, no. 3, pp. 330–336, Mar. 2002.

[3] S. Yan, V. Saren, and W. Shi, "Sparse transformations and preconditioners for hierarchical 3-D capacitance extraction with multiple dielectrics," in *Proc. DAC*, 2004, pp. 788–793.

[4] S. Kapur and D. E. Long, "IES3: A fast integral equation solver for efficient 3-dimensional extraction," in *Proc. 1997 ICCAD*, Nov. 1997, pp. 448–455.

[5] J. R. Phillips and J. White, "A precorrected FFT method for capacitance extraction of complicated 3-D structures," in *Proc. 1994 ICCAD*, pp. 268–271.

[6] Y. C. Pan, W. C. Chew, and L. X. Wan, "A fast multipole-method based calculation of the capacitance matrix for multiple conductors above stratified dielectric media," *IEEE Trans. Microw. Theory Tech.*, vol. 49, no. 3, pp. 480–490, Mar. 2001.

[7] D. Gope, I. Chowdhury, and V. Jandhyala, "DiMES: Multilevel fast direct solver based on multipole expansions for parasitic extraction of massively coupled 3-D microelectronic structures," in *Proc. 42th ACM/EDAC/IEEE Design Automation Conf. (DAC)*, 2005, pp. 159–162.

[8] W. Chai, D. Jiao, and C. C. Koh, "A direct integral-equation solver of linear complexity for large-scale 3-D capacitance and impedance extraction," in *Proc. 46th ACM/EDAC/IEEE Design Automation Conf. (DAC)*, Jul. 2009, pp. 752–757.

[9] S. Börm and W. Hackbusch, "$\mathcal{H}^2$-matrix approximation of integral operators by interpolation," *Appl. Numerical Math.*, vol. 43, pp. 129–143, 2002.

[10] S. Börm, L. Grasedyck, and W. Hackbusch, Hierarchical matrices Lecture Note 21 of the Max Planck Institute for Mathematics, 2003.

[11] S. Börm, "Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices," *Computing*, vol. 69, pp. 1–35, 2002.

[12] S. Börm, "$\mathcal{H}^2$-matrices—Multilevel methods for the approximation of integral operators," *Comput. Vis. Sci.*, vol. 7, pp. 173–181, 2004.

[13] S. Börm and L. Grasedyck, "Low-rank approximation of integral operators by interpolation," *Computing*, vol. 72, pp. 325–332, 2004.

[14] W. Chai and D. Jiao, "An $\mathcal{H}^2$-matrix-based integral-equation solver of reduced complexity and controlled accuracy for solving electrodynamic problems," *IEEE Trans. Antennas Propag.*, vol. 57, no. 10, pp. 3147–3159, Oct. 2009.

[15] W. Chai and D. Jiao, "$\mathcal{H}$ and $\mathcal{H}^2$-matrix-based fast integral-equation solvers for large-scale electromagnetic analysis," *IET Microwaves, Antennas Propag.*, 2010, to be published.

[16] S. Börm, "Approximation of integral operators by $\mathcal{H}^2$-matrices with adaptive bases," *Computing*, 2005.

[17] S. Börm, $\mathcal{H}^2$-matrix arithmetics in linear complexity 2005, Lecture Note of the Max Planck Institute for Mathematics.

Wenwen Chai received the B.S. degree from University of Science and Technology of China, in 2004, and the M.S. degree from the Chinese Academy of Sciences, in 2007, both in electrical engineering. She is currently working toward the Ph.D. degree at the School of Electrical and Computer Engineering and works in the On-Chip Electromagnetics Group at Purdue University, West Lafayette, IN.

Her research is focused on computational electromagnetics, high-performance VLSI CAD, fast and high-capacity numerical methods.

Ms. Chai received the IEEE Antennas and Propagation Society Doctoral Research Award for 2009–2010.

Dan Jiao (S'00–M'02–SM'06) received the Ph.D. degree in electrical engineering from the University of Illinois, Urbana-Champaign, in 2001.

She then worked at Technology CAD Division at the Intel Corporation until September 2005 as Senior CAD Engineer, Staff Engineer, and Senior Staff Engineer. In September 2005, she joined Purdue University, West Lafayette, IN, as an Assistant Professor in the School of Electrical and Computer Engineering. In 2009, she was promoted to Associate Professor with tenure. She has authored two book chapters and over 100 papers in refereed journals and international conferences. Her current research interests include computational electromagnetics, high-frequency digital, analogue, mixed-signal, and RF IC design and analysis, high-performance VLSI CAD, modeling of micro- and nano-scale circuits, applied electromagnetics, fast and high-capacity numerical methods, fast time domain analysis, scattering and antenna analysis, RF, microwave, and millimeter wave circuits, wireless communication, and bio-electromagnetics.

Dr. Jiao received Ruth and Joel Spira Outstanding Teaching Award in 2010. In 2008, she received NSF CAREER Award. In 2006, she received Jack and Cathie Kozik Faculty Start up Award, which recognizes an outstanding new faculty member in Purdue ECE. She also received an ONR award through Young Investigator Program in 2006. In 2004, she received the Best Paper Award from Intel's annual corporate-wide technology conference (Design and Test Technology Conference) for her work on generic broadband model of high-speed circuits. In 2003, she won the Intel Logic Technology Development (LTD) Divisional Achievement Award in recognition of her work on the industry-leading BroadSpice modeling/simulation capability for designing high-speed microprocessors, packages, and circuit boards. She was also awarded the Intel Technology CAD Divisional Achievement Award for the development of innovative full-wave solvers for high-frequency IC design. In 2002, she was awarded by Intel Components Research the Intel Hero Award (Intel-wide she was the tenth recipient) for the timely and accurate two- and three-dimensional full-wave simulations. She also won the Intel LTD Team Quality Award for her outstanding contribution to the development of the measurement capability and simulation tools for high-frequency on-chip cross-talk. She was the winner of the 2000 Raj Mittra Outstanding Research Award given her by the University of Illinois at Urbana-Champaign. She has served as the reviewer for many IEEE journals and conferences.