$\mathcal{H}$-Matrix-Based Finite-Element-Based Thermal Analysis for 3D ICs

HAI-BAO CHEN, Shanghai Jiao Tong University
YING-CHI LI, The University of Hong Kong
SHELDON X.-D. TAN and XIN HUANG, University of California, Riverside
HAI WANG, University of Electronic Science and Technology of China
NGAI WONG, The University of Hong Kong

In this article, we propose an efficient finite-element-based (FE-based) method for both steady and transient thermal analyses of high-performance integrated circuits based on the hierarchical matrix ($\mathcal{H}$-matrix) representation. $\mathcal{H}$-matrix has been shown to provide a data-sparse way to approximate the matrices and their inverses with almost linear-space and time complexities. In this work, we apply the $\mathcal{H}$-matrix concept for solving heating diffusion problems modeled by parabolic partial differential equations (PDEs) based on the finite element method. We show that the matrix from a FE-based steady and transient thermal analysis can be represented by $\mathcal{H}$-matrix without any approximation, and its inverse and Cholesky factors can be evaluated by $\mathcal{H}$-matrix with controlled accuracy. We then show and prove that the memory and time complexities of the solver are bounded by $\mathcal{O}(k_1 N \log N)$ and $\mathcal{O}(k_1^2 N \log^2 N)$, respectively, where k_1 is a small quantity determined by accuracy requirements and N is the number of unknowns in the system. The comparison with existing product-quality LU solvers, CSPARSE and UMFPACK, on a number of 3D IC thermal matrices, shows that the new method is much more memory efficient than these methods, which however prevents CPU time comparison with those methods on large examples. But the proposed method can solve all the given thermal circuits with decent scalabilities, which shows good agreement with the predicted theoretical results.

Categories and Subject Descriptors: J.6 [**Computer Application**]: Computer-Aided Engineering—*Computer-aided design (CAD)*

General Terms: Design, Algorithm

Additional Key Words and Phrases: Finite element method, integrated circuits, $\mathcal{H}$-matrix, thermal analysis

ACM Reference Format:
Hai-Bao Chen, Ying-Chi Li, Sheldon X.-D. Tan, Xin Huang, Hai Wang, and Ngai Wong. 2015. $\mathcal{H}$-matrix-based finite-element-based thermal analysis for 3d ICs. ACM Trans. Des. Autom. Electron. Syst. 20, 4, Article 47 (September 2015), 25 pages.
DOI: http://dx.doi.org/10.1145/2714563

1. INTRODUCTION

Continuous process scaling and aggressive device integration lead to rapid power density increase and adverse thermal effects. This problem becomes more severe as the VLSI technology scales to the nanometer ranges. Excessively high on-chip temperature can cause many severe problems such as reduced reliability of chips, thermal and

---

This work is supported in part by NSF grant under No. CCF-1017090, in part by NSF Grant under No. CCF-1255899, in part by NSF Grant under No. CCF-1527324, in part by Semiconductor Research Corporation (SRC) grant under No. 2013-TJ-2417, in part by the Hong Kong Research Grants Council under the GRF Projects 718711E and 718213E, in part by the Nature Science Foundation of China (NSFC) under No. 61404024, in part by 985 research funds from Shanghai Jiao Tong University and University of Electronic Science and Technology of China.
Corresponding author: Sheldon X.-D. Tan; email: stan@ece.ucr.edu.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2015 ACM 1084-4309/2015/09-ART47 $15.00
DOI: http://dx.doi.org/10.1145/2714563

run-away, and elevated cooling cost of the packaging [Gunther et al. 2001; Brooks and Martonosi 2001; Kornaros and Pnevmatikatos 2013; Lee and Huang 2013]. Thermal- and cooling-related design problems continue to be identified by the Semiconductor Industries Association Roadmap [ITRS 2012; Yu et al. 2009] as one of the five key challenges during the next decade to achieve the projected performance goals of the semiconductor industry. Thus, accurate and efficient thermal modeling and analysis are vital for thermal-aware VLSI design [Pedram and Nazarian 2006; Yu et al. 2008, 2009; Sai et al. 2013; Wang et al. 2013; Liu et al. 2014] to in terms of improve performance, reliability, power reduction as well as online temperature regulation techniques [Brooks and Martonosi 2001; Skadron et al. 2003; Liu et al. 2015b].

Traditional thermal analysis solves partial thermal diffusion equations using numerical approaches such as the finite difference method (FDM) [Ozisik 1994] and the finite element method (FEM) [Lewis et al. 2004]. FE discretization of thermal initial-value problems typically leads to a large sparse FE system, which can be solved by iterative and direct methods. The iterative methods are usually efficient; however, the convergence of the iteration highly depends on an appropriate preconditioner which is often problem dependent. For LU-based direct method, it is typically more expensive for both memory and computation with super-linear time complexity [Davis 2006]. Recently, parallelization of transient thermal analysis on many-core processors such as GPUs has been exploited [He et al. 2014; Liu et al. 2015a].

Recently, a mathematic framework called hierarchical matrix or $\mathcal{H}$-matrix has been proposed to solve many numerical problems modeled by partial differential equations [Hackbusch 1999; Grasedyck and Hackbusch 2003; Borm et al. 2003]. $\mathcal{H}$-matrix is shown to be well suited for data-sparse representation of dense or sparse matrices arising in FEM or for the approximation of the inverse to the FE matrices for elliptic partial differential equations (PDEs). These matrices are not necessarily sparse, but they are data sparse in the sense that these matrices can be described by fewer data. The significance of this $\mathcal{H}$-matrix technique is that those matrices can be stored by almost linear-space complexities. The resulting matrix operations such as addition, multiplication, and even inverse (or LU and Cholesky factorization) can be done in terms of almost linear-space complexities. Recently, the $\mathcal{H}$-matrix structure has been used in solving electromagnetic problems modeled by hyperbolic wave PDEs [Liu and Jiao 2010, 2009b; Chai and Jiao 2010; Wan et al. 2010].

In this article, the $\mathcal{H}$-matrix technique is applied to solve both steady and transient thermal analysis problems using FE-based approaches. For heat diffusion PDEs, the steady-state PDEs are elliptic, while the transient PDEs are parabolic [Ozisik 1994a]. Not many works have been reported for $\mathcal{H}$-matrix FEM solvers for parabolic PDEs, as most existing work focuses on either elliptic or hyperbolic PDEs. Our article mainly focuses on a new $\mathcal{H}$-based solver for parabolic equations used in thermal analysis of 3D IC. For this new solver, the *Gmsh* program[1] is used to create tetrahedron meshes for 3D IC with any type of geometry structure. We will prove that the thermal matrix obtained from the finite element method using the generated meshes has a hierarchical structure. For transient analysis, we need to solve ordinary differential equations while steady-state analysis solves algebraic equations. LU decomposition is required for solving a system of linear equations arising from the finite element discretization. With theoretical and numerical investigations, we will show that the $\mathcal{H}$-based LU decomposition has an obvious advantage if the size of the thermal matrix is large. Contributions include the following.

[1]A three-dimensional, finite-element mesh generator. `http://geuz.org/gmsh/`.

—We demonstrate that $\mathcal{H}$-matrix techniques can be applied to transient thermal equations which are described by parabolic PDEs. We show that the matrix generated from a steady and transient thermal analysis based on FEM can be represented by $\mathcal{H}$-matrix without approximation.
—We prove that the inverse and Cholesky factors of the thermal matrices generated from the parabolic PDEs of the FEM can also be approximated and represented using $\mathcal{H}$-matrix arithmetics.
—We prove that the memory and the time complexities of the solver are bounded by $\mathcal{O}(k_1 N \log N)$ and $\mathcal{O}(k_1^2 N \log^2 N)$, respectively, where k_1 is a small quantity determined by the accuracy requirements and N is the number of unknowns in the system.

The numerical results from a 3D IC demonstrate the almost-linear scalability of the proposed method in terms of both memory footprint and CPU time. The comparison with existing product-quality LU solvers, CSPARSE and UMFPACK, on a number of 3D IC thermal matrices, shows that the new method is much more memory efficient than these methods, which however prevents CPU time comparison with those methods on those large examples. The proposed method can solve all the given thermal circuits with decent scalabilities, which show good agreement with the predicted theoretical results. For a few very large examples, the existing solvers fail to deliver results, which prevent CPU time comparison on those large examples. But due to their superlinear time complexities, we expect decent speedup will occur on very large examples with the new solver over those solvers.

We also remark that 3D IC thermal structures with many complicated through silicon vias (TSVs) can lead to very large and sparse matrices to solve using the finite element methods, as shown here. As a result, solver efficiency in terms of both CPU time and memory becomes critical for such problems. Hence, the proposed linear thermal solver becomes attractive for attacking such problems. In light of such a backdrop, we select 3D ICs as the application of the proposed sovler. The $\mathcal{H}$ matrix-based solver for 3D ICs is based on the finite element method in which the tetrahedron meshes are created by a 3D finite element grid generator *Gmsh*. The *Gmsh* program provides a fast, light, and user-friendly meshing tool which has an advantage over the commercial tool COMSOL.[2] The software package COMSOL implements the finite element method for various physics and engineering applications. From the perspective of finite element analysis, our $\mathcal{H}$-matrix-based thermal analysis method has the advantage over COMSOL. The $\mathcal{H}$-matrix representation provides a data-sparse way to approximate the matrix generated from the finite element discretization of heat differential equation, which has almost linear-space and time complexities. As far as we know, there are no results to show that the software package COMSOL has a linear complexity for solving thermal equations. Besides, we will show that the relative errors of the proposed method are quite small compared with the CSPARSE solver.[3]

The article is organized as follows: Section 2 presents the basics for FE-based thermal analysis. Section 3 shows and proves that thermal matrices from FEM can be represented by $\mathcal{H}$-matrices without any approximation. Then, Section 4 shows that the inverse of thermal matrices can be represented by $\mathcal{H}$-matrices with controlled accuracy as well as the arithmetics to compute its inverse and its Cholesky factors. Section 5 shows the complexity analysis for the proposed method. Section 6 presents some numerical results, and Section 7 concludes this article.

[2]`http://www.comsol.com`.
[3]A concise sparse matrix package in c. `http://www.cise.ufl.edu/research/sparse/`.

2. FINITE-ELEMENT-BASED THERMAL ANALYSIS

In the circuit, package, and board levels, the heat transfer phenomena are governed by the following heat differential equation [Cheng et al. 2000; Bergman et al. 2011]:

$$\rho C_p \frac{\partial T(\vec{r}, t)}{\partial t} = \nabla \cdot [\kappa(\vec{r}, T) \cdot \nabla T(\vec{r}, t)] + g(\vec{r}, t), \tag{1}$$

subject to the following general thermal boundary condition (Robin's boundary condition):

$$\kappa(\vec{r}, T) \frac{\partial T(\vec{r}, t)}{\partial \mathbf{n}_i} = h_i(T(\vec{r}, t) - T_a). \tag{2}$$

In (1), $\vec{r}$ is the position, t is the time, T is the temperature (K), ρ is the density of the material (kg/m^3), C_p is the mass heat capacity (Jkg^{-1}K^{-1}), κ is the thermal conductivity (Wm^{-1}K^{-1}), and g is the heat energy generation rate (W/m^3). In (2), $\mathbf{n}_i$ is the outward direction normal to the boundary condition i, h_i is the heat-transfer coefficient (Wm^{-2}K^{-1}) (for the convective interface), and T_a is the ambient temperature surrounding the thermal systems. If $h_i = 0$, the boundary condition is adiabatic (isolated), otherwise it is convective. Note that the thermal conductivity κ differs for different materials and also depends on the temperature.

In our work, we assume that κ is constant for each material. Then, (1) can be written as

$$\kappa \left[\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right] + g = \rho C_p \frac{\partial T}{\partial t}, \tag{3}$$

subject to the following boundary conditions:

$$T = T_b \text{ on } S_1, \tag{4}$$

$$\kappa \left[\frac{\partial T}{\partial x}\mathbf{n_1} + \frac{\partial T}{\partial y}\mathbf{n_2} + \frac{\partial T}{\partial z}\mathbf{n_3} \right] + q = 0 \text{ on } S_2, \tag{5}$$

$$\kappa \left[\frac{\partial T}{\partial x}\mathbf{n_1} + \frac{\partial T}{\partial y}\mathbf{n_2} + \frac{\partial T}{\partial z}\mathbf{n_3} \right] + h(T - T_a) = 0 \text{ on } S_3, \tag{6}$$

and initial condition $T = T_0$ at $t = 0$, where S_1, S_2, and s_3 are surfaces, $\mathbf{n_1}$, $\mathbf{n_2}$, and $\mathbf{n_3}$ are surface normals, q is the heat flux, and h is the heat transfer coefficient.

To solve PDE from the thermal diffusion Equation (3), FDM and FEM are among the popular methods. In this work, we focus on FE-based thermal analysis. FEM discretizes the solution region Ω into small elements Ω_e, such as triangles for 2D problems and tetrahedrons for 3D problems, by a specific set of shape function φ_i such that

$$T\,(x, y, z, t) \;=\; \sum_{i=1}^{m} \varphi_i\,(x, y, z)\,T_i\,(t) = [\Phi]\,[\mathbf{T}], \tag{7}$$

where m is the number of nodes on the element (e.g., four for a tetrahedron), $\Phi = [\varphi_1\ \varphi_2\ \cdots\ \varphi_m]$ is the shape function matrix, and $\mathbf{T} = [T_1\ T_2\ \cdots\ T_m]^{\mathrm{T}}$ is the vector of unknown temperature at corresponding nodes.

In this article, we use the tetrahedron elements as follows:

$$\varphi_i = \frac{1}{6V}\left(c_{i1} + c_{i2}x + c_{i3}y + c_{i4}z\right), \tag{8}$$

where

$$6V = \det \begin{bmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_4 & y_4 & z_4 \end{bmatrix}, \tag{9}$$

and c_{ij} is the (i, j) cofactor of the matrix in (9) with i and $j = 1, 2, 3, 4$.

Assume conductivity κ is independent of temperature and the heat flow due to radiation is negligible. The Galerkin representation of Eq. (3) for an element is

$$\int_{\Omega_e} \varphi_i \left[\kappa \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right) + g - \rho C_p \frac{\partial T}{\partial t} \right] d\Omega_e = 0. \tag{10}$$

By Green's theorem and integration by parts, the second-order derivatives can be rewritten in two parts:

$$\int_{\Omega_e} \varphi_i \left(k \frac{\partial^2 T}{\partial x^2} \right) d\Omega_e = \int_{S_e} \varphi_i \left(k \frac{\partial T}{\partial x} \right) \mathbf{n_1} dS_e - \int_{\Omega_e} k \frac{\partial \varphi_i}{\partial x} \frac{\partial T}{\partial x} d\Omega_e. \tag{11}$$

Substituting (11) and boundary condition Equations (5) and (6) into (10), we get

$$\begin{aligned} &\int_{\Omega_e} k \left[\frac{\partial \varphi_i}{\partial x} \frac{\partial T}{\partial x} + \frac{\partial \varphi_i}{\partial y} \frac{\partial T}{\partial y} + \frac{\partial \varphi_i}{\partial z} \frac{\partial T}{\partial z} \right] d\Omega_e \\ &- \int_{\Omega_e} \left[\varphi_i g - \varphi_i \rho C_p \frac{\partial T}{\partial t} \right] d\Omega_e + \int_{S_{2e}} \varphi_i q dS_{2e} + \int_{S_{3e}} \varphi_i h (T - T_a) dS_{3e} = 0. \end{aligned} \tag{12}$$

Rearranging (12) and substituting (7) into it, we have

$$\mathbf{C_e} \frac{\partial \mathbf{T}}{\partial t} + \mathbf{G_e} \mathbf{T} = \mathbf{f_e}, \tag{13}$$

where $\mathbf{f_e} \subset \mathbb{R}^{4\times 1}$ is the local thermal load (local power inputs), $\mathbf{C_e} \subset \mathbb{R}^{4\times 4}$ is the local heat capacity matrix, and $\mathbf{G_e} \subset \mathbb{R}^{4\times 4}$ is the local heat conductivity matrix. Matrices $\mathbf{C_e}$, $\mathbf{G_e}$, and the vector $\mathbf{f_e}$ can be given as follows:

$$\mathbf{C_e} = \int_{\Omega_e} \rho C_p \Phi^{\mathrm{T}} \Phi d\Omega_e, \tag{14}$$

$$\mathbf{G_e} = \int_{\Omega_e} \kappa \mathbf{B}^{\mathrm{T}} \mathbf{B} d\Omega_e + \int_{S_{3e}} h \Phi^{\mathrm{T}} \Phi dS_{3e}, \tag{15}$$

$$\mathbf{f_e} = \int_{\Omega_e} g \Phi^{\mathrm{T}} d\Omega_e - \int_{S_{2e}} q \Phi^{\mathrm{T}} dS_{2e} + \int_{S_{3e}} h T_a \Phi^{\mathrm{T}} dS_{3e}, \tag{16}$$

where

$$\mathbf{B} = \begin{bmatrix} \frac{\partial \varphi_1}{\partial x} & \frac{\partial \varphi_2}{\partial x} & \frac{\partial \varphi_3}{\partial x} & \frac{\partial \varphi_4}{\partial x} \\ \frac{\partial \varphi_1}{\partial y} & \frac{\partial \varphi_2}{\partial y} & \frac{\partial \varphi_3}{\partial y} & \frac{\partial \varphi_4}{\partial y} \\ \frac{\partial \varphi_1}{\partial z} & \frac{\partial \varphi_2}{\partial z} & \frac{\partial \varphi_3}{\partial z} & \frac{\partial \varphi_4}{\partial z} \end{bmatrix} = \frac{1}{6V} \begin{bmatrix} c_{12} & c_{22} & c_{32} & c_{42} \\ c_{13} & c_{23} & c_{33} & c_{43} \\ c_{14} & c_{24} & c_{34} & c_{44} \end{bmatrix} \tag{17}$$

is the gradient matrix of the shape functions.

The global thermal load $\mathbf{f}$, heat capacity matrix $\mathbf{C}$, and conductivity matrix $\mathbf{G}$ can be assembled by adding the values on corresponding nodes, hence we can get the representation of entire domain with the following equation:

$$\mathbf{C} \frac{\partial \mathbf{T}}{\partial t} + \mathbf{G} \mathbf{T} = \mathbf{f}. \tag{18}$$

In steady-stage analysis, there is no time-dependent term ($\frac{\partial \mathbf{T}}{\partial t} = 0$). Therefore, (18) can be reduced to

$$\mathbf{GT} = \mathbf{f}. \tag{19}$$

In transient analysis, FDM is applied to evaluate the result. Using the backward Euler method, the differential operator in (18) can be approximated as follow:

$$\frac{\partial \mathbf{T}}{\partial t} \approx \frac{\mathbf{T}^{n+1} - \mathbf{T}^{n}}{\Delta t}. \tag{20}$$

Substituting Eq. (20) into Eq. (18), we have

$$\left[\mathbf{G} + \frac{\mathbf{C}}{\Delta t}\right]\mathbf{T}^{n+1} = \mathbf{f}^{n+1} + \frac{\mathbf{C}}{\Delta t}\mathbf{T}^{n}. \tag{21}$$

3. $\mathcal{H}$-MATRIX REPRESENTATION FOR THE THERMAL MATRIX

3.1. Preliminaries of $\mathcal{H}$-Matrix

The hierarchical matrix format is a data-sparse representation with high accuracy for a special class of matrices which often arise in modeling of complex physical processes, such as heat differential equations and electromagnetic problems. In order to construct a general hierarchical matrix, we first need to introduce a tree structure. If t is a vertex of the tree $\mathcal{T}$, we denote by $\mathcal{S}(t)$ the set of sons of t. If $\mathcal{S}(t) = \emptyset$, t is a leaf of $\mathcal{T}$. For simplicity, $\mathfrak{L}(\mathcal{T})$ and $r(\mathcal{T})$ denote the set of leaves and the root of $\mathcal{T}$, respectively. Based on the index set $\mathcal{I}$, we define the cluster tree $\mathcal{T}_{\mathcal{I}}$ satisfying the following conditions: (i) $\mathcal{I} \in \mathcal{T}_{\mathcal{I}}$; (ii) if $t \in \mathcal{T}_{\mathcal{I}}$ has no leaf, then $t = \dot{\bigcup}_{s \in \mathcal{S}(t)} s$, where the notation $\dot{\bigcup}$ denotes the disjoint union.

For the index set $\mathcal{I} \times \mathcal{I}$, we define the block cluster tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ as follows: $r(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) := \mathcal{I} \times \mathcal{I}$, and

$$\mathcal{S}(r \times s) := \begin{cases} \{r' \times s' | r' \in \mathcal{S}(r), s' \in \mathcal{S}(s)\}, & \texttt{if } r = s; \\ \emptyset, & \texttt{otherwise}. \end{cases}$$

The depth of the block tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ is bounded by $\mathcal{O}(\log N)$, where N is the cardinality of index set $\mathcal{I}$. For simplicity, we denote by $|\mathcal{I}|$ the cardinality of set $\mathcal{I}$.

Given a positive integer number n_{min}, we denote by

$$\mathfrak{L}^{-}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) := \{r \times s \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}} |\, r \le n_{min} \texttt{ or } s \le n_{min}\},$$

and $\mathfrak{L}^{+}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) := \mathfrak{L}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) \backslash \mathfrak{L}^{-}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}})$ the set of small leaves and the set of large leaves, respectively. Based on the cluster tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$, we define the set of $\mathcal{H}$-matrix with block-wise rank $k \in \mathbb{Z}^{+}$ and minimal block size $n_{min} \in \mathbb{Z}^{+}$ as

$$\mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ k) := \{\texttt{M} \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|} | \forall r \times s \in \mathfrak{L}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}) :$$

$$\texttt{rank}(M|_{r\times s}) \le k \texttt{ or } r \le n_{min} \texttt{ or } s \le n_{min}\}.$$

The main idea of the $\mathcal{H}$-matrix is to partition a given sparse matrix M into submatrices $M|_{r\times s}$ that can be approximated as the following form

$$M|_{r\times s} \approx AB^{\mathrm{T}},\ A \in \mathbb{R}^{|r|\times k},\ B \in \mathbb{R}^{k\times|s|},$$

where k is the rank of the submatrix. These factors are rectangular with k much smaller than $|r|$ and $|s|$.

An $\mathcal{H}$-matrix $\mathbf{H} \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}$ is generally associated with an admissibility condition [Borm et al. 2003], where $\mathcal{I} = \{i_1, i_2, \ldots, i_N\}$ is the reordered index set containing the indices of the basis functions φ_i or nodes in global domain Ω with the total number of

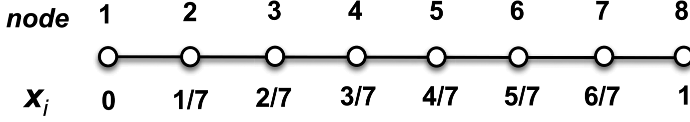

Fig. 1. One-dimensional FEM mesh.

N unknowns in the entire FEM model. If any two subsets r and $s \subseteq \mathcal{I}$ are admissible, the following admissibility condition holds:

$$\min\{\text{diam}(\Omega_r), \text{diam}(\Omega_s)\} \leq \eta \ \text{dist}(\Omega_r, \Omega_s), \tag{22}$$

where Ω_r and $\Omega_s \subseteq \Omega$ contain the supports of the basis functions, diam$(\cdot)$ is the Euclidean diameter of the subsets, dist$(\cdot,\cdot)$ is the Euclidean distance of two subsets, and η is a positive constant. In addition, the submatrix $\mathbf{H}_{r\times s}$ can be represented by a low-rank matrix with rank k and stored in a factorized form such that

$$\mathbf{H}_{r\times s} \approx \mathbf{AB}^{\mathrm{T}}, \tag{23}$$

where $\mathbf{A} \in \mathbb{R}^{|r|\times k}$ and $\mathbf{B} \in \mathbb{R}^{|s|\times k}$. The other submatrices in $\mathbf{H}$ corresponding to the inadmissible subsets in $\mathcal{I}$ will be represented by full matrices. In order to construct the suitable structure of $\mathbf{H}$, a block cluster tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ is necessary based on the geometric information of the FEM model.

In order to build $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$, a cluster tree $\mathcal{T}_{\mathcal{I}}$ for multidimensional meshes is constructed first. We find a bounding box that contains the full index of the basis functions then store into $\mathcal{I}$, split the box into two subsets based on the geometrical position of nodes, and repeat the splitting until each subset has no more than n_{min} (mathematic) indices which is a tree depth control parameter. $\mathcal{T}_{\mathcal{I}}$ is built in which each subset is a cluster and $\mathcal{I}$ is the root of the tree.

The block cluster tree $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$ is constructed from two cluster trees $\mathcal{T}_{\mathcal{I}}$ and $\mathcal{T}_{\mathcal{I}}$ in the thermal FEM analysis. The admissibility condition of each pair of clusters$(r, s) \in \mathcal{T}_{\mathcal{I}}$ is checked level by level, starting from the root and descending in the tree. If the cluster pair $(r, s) \in \mathcal{T}_{\mathcal{I}}$ is admissible, then make it a leaf of $\mathcal{T}_{\mathcal{I}\times\mathcal{I}}$, which corresponds to a disjoint partition of the index set $\mathcal{I} \times \mathcal{I}$. Otherwise, we repeat the procedure recursively for all pairs of its children.

Take one-dimensional structure as an example: the node index set $\mathcal{I} = \{1, \ldots, 8\}$ with corresponding domain $[0, 1]$ is shown in Figure 1 and Figure 2. The root of the block cluster tree is $\mathcal{I}\times\mathcal{I}$. No admissible clusters are found in level 0 and level 1. The clusters that are admissible in level 3 are linked and stored as leaves of the block cluster tree. The nodes

$$\begin{array}{lll} \{1,2\}\times\{3,4\}, & \{1,2\}\times\{5,6\}, & \{1,2\}\times\{7,8\}, \\ \{3,4\}\times\{1,2\}, & \{3,4\}\times\{5,6\}, & \{3,4\}\times\{7,8\}, \\ \{5,6\}\times\{1,2\}, & \{5,6\}\times\{3,4\}, & \{5,6\}\times\{7,8\}, \\ \{7,8\}\times\{1,2\}, & \{7,8\}\times\{3,4\}, & \{7,8\}\times\{5,6\}, \end{array}$$

are admissible because their corresponding domains satisfy

$$\text{diam}\left(\left[0, \frac{1}{7}\right]\right) \leq \text{dist}\left(\left[0, \frac{1}{7}\right], \left[\frac{2}{7}, \frac{3}{7}\right]\right), \text{diam}\left(\left[0, \frac{1}{7}\right]\right) \leq \text{dist}\left(\left[0, \frac{1}{7}\right], \left[\frac{4}{7}, \frac{5}{7}\right]\right),$$

$$\text{diam}\left(\left[0, \frac{1}{7}\right]\right) \leq \text{dist}\left(\left[0, \frac{1}{7}\right], \left[\frac{6}{7}, 1\right]\right), \text{diam}\left(\left[\frac{2}{7}, \frac{3}{7}\right]\right) \leq \text{dist}\left(\left[\frac{2}{7}, \frac{3}{7}\right], \left[0, \frac{1}{7}\right]\right),$$

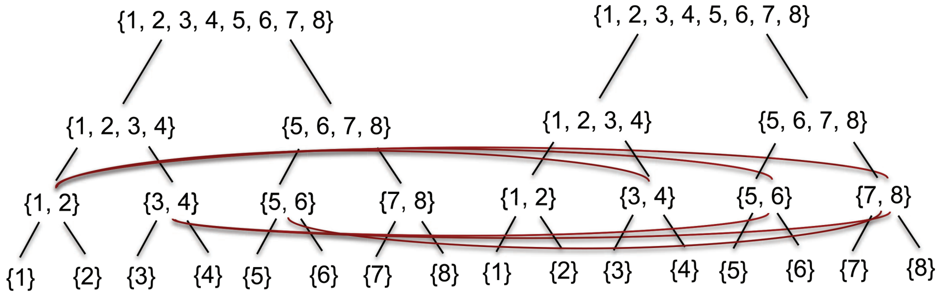

Fig. 2. An example for the block cluster tree.

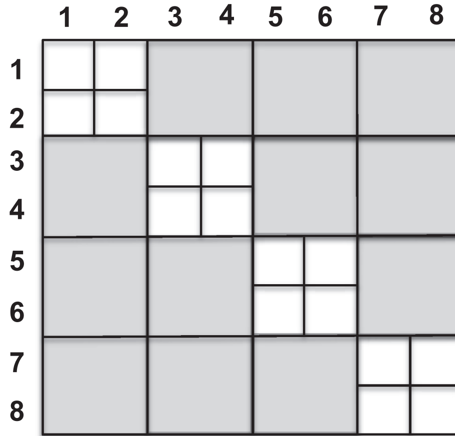

Fig. 3. $\mathcal{H}$-matrix structure of 1D example.

$$\operatorname{diam}\left(\left[\frac{2}{7},\frac{3}{7}\right]\right)\le\operatorname{dist}\left(\left[\frac{2}{7},\frac{3}{7}\right],\left[\frac{4}{7},\frac{5}{7}\right]\right),\ \operatorname{diam}\left(\left[\frac{2}{7},\frac{3}{7}\right]\right)\le\operatorname{dist}\left(\left[\frac{2}{7},\frac{3}{7}\right],\left[\frac{6}{7},1\right]\right),$$

$$\operatorname{diam}\left(\left[\frac{4}{7},\frac{5}{7}\right]\right)\le\operatorname{dist}\left(\left[\frac{4}{7},\frac{5}{7}\right],\left[0,\frac{1}{7}\right]\right),\ \operatorname{diam}\left(\left[\frac{4}{7},\frac{5}{7}\right]\right)\le\operatorname{dist}\left(\left[\frac{4}{7},\frac{5}{7}\right],\left[\frac{2}{7},\frac{3}{7}\right]\right),$$

$$\operatorname{diam}\left(\left[\frac{4}{7},\frac{5}{7}\right]\right)\le\operatorname{dist}\left(\left[\frac{4}{7},\frac{5}{7}\right],\left[\frac{6}{7},1\right]\right),\ \operatorname{diam}\left(\left[\frac{6}{7},1\right]\right)\le\operatorname{dist}\left(\left[\frac{6}{7},1\right],\left[0,\frac{1}{7}\right]\right),$$

$$\operatorname{diam}\left(\left[\frac{6}{7},1\right]\right)\le\operatorname{dist}\left(\left[\frac{6}{7},1\right],\left[\frac{2}{7},\frac{3}{7}\right]\right),\ \operatorname{diam}\left(\left[\frac{6}{7},1\right]\right)\le\operatorname{dist}\left(\left[\frac{6}{7},1\right],\left[\frac{4}{7},\frac{5}{7}\right]\right).$$

Therefore, the construction of block cluster tree defines the $\mathcal{H}$-matrix structure as shown in Figure 3, in which shaded matrix blocks are represented by $\mathbf{AB}^{\mathrm{T}}$ (which are admissible blocks), while others are represented by full matrices.

3.2. $\mathcal{H}$-Matrix Representation for the Thermal Matrices

We first show that the thermal matrices for both steady and transient FEM analyses can be converted to $\mathcal{H}$-matrices without any accuracy loss.

In the thermal FEM model, $\mathbf{G}$ and $\mathbf{C} \in \mathbb{R}^{|\hat{\mathcal{I}}|\times|\hat{\mathcal{I}}|}$ can be obtained from (21), where $\hat{\mathcal{I}} = \{1, 2, \ldots, N\}$ is the node indices set. If the indices of nodes belong to subset $r, s \subseteq \hat{\mathcal{I}}$

and are admissible, they must be physically disconnected. After applying the block cluster tree construction algorithm, $\hat{\mathcal{I}}$ is reordered into $\mathcal{I}$, and the global matrices $\mathbf{G}$ and $[\mathbf{G}+\frac{\mathbf{C}}{\Delta t}]$ can be then stored into $\mathbf{H}_{\text{steady}}$ and $\mathbf{H}_{\text{transient}}$, respectively, without any approximation.

THEOREM 1. *Let the heat capacity matrix* $\mathbf{C_e}$ *be defined in* (14)*, then it can be represented by an* $\mathcal{H}$*-matrix.*

PROOF. Given two admissible clusters $r,\ s \subseteq \mathcal{I}$, according to the structure of $\mathbf{C_e}$ we can see that there exist $\mathbf{A}_i$ and $\mathbf{B}_i$ $(i = 1,\ 2)$ for the matrix $\mathbf{C}_{\mathbf{e},\mathbf{r}\times\mathbf{s}}$ such that

$$\mathbf{C}_{\mathbf{e},\mathbf{r}\times\mathbf{s}} = \mathbf{A}_1\mathbf{B}_1^{\mathrm{T}} + \mathbf{A}_2\mathbf{B}_2^{\mathrm{T}} = [\mathbf{A}_1 \quad \mathbf{A}_2][\mathbf{B}_1 \quad \mathbf{B}_2]^{\mathrm{T}}.$$

From this expression, we know that

$$\mathtt{rank}(\mathbf{C}_{\mathbf{e},\mathbf{r}\times\mathbf{s}}) \leq \min\{\mathtt{rank}([\mathbf{A}_1 \quad \mathbf{A}_2]),\ \mathtt{rank}([\mathbf{B}_1 \quad \mathbf{B}_2])\},$$

which implies that the matrix $\mathbf{C_e}$ is an $\mathcal{H}$-matrix. This completes the proof. □

Since the local capacity matrix $\mathbf{C_e}$ is an $\mathcal{H}$-matrix, the heat capacity matrix $\mathbf{C}$ is also an $\mathcal{H}$-matrix. Similar to Theorem 1, we can prove that matrices $\mathbf{G}$ are an $\mathcal{H}$-matrix. From Eqs. (14) and (15), we can see that the structure of the matrix $[\mathbf{G}+\frac{\mathbf{C}}{\Delta t}]$ is identical to that of $\mathbf{G}$. Therefore, we have the following result.

THEOREM 2. $[\mathbf{G}+\frac{\mathbf{C}}{\Delta t}]$ *can be represented by an* $\mathcal{H}$*-matrix.*

PROOF. We first write the analytical solution of (3) as follows:

$$T(x,y,z,t) = U(x,y,z,t) * T_0 + \int_0^t U(x,y,z,t-\tau) * f(x,y,z,\tau)d\tau, \tag{24}$$

where the asterisk stands for convolution, and

$$f(x,y,z,t) = \frac{1}{\rho C_p} g(x,y,z,t),$$

$$U(x,y,z,t) = \frac{1}{(2a\sqrt{\pi t})^3} \exp\left(-\frac{x^2+y^2+z^2}{4a^2t}\right),$$

$$U(x,y,z,t) * T_0 = \frac{1}{(2a\sqrt{\pi t})^3} \iiint\limits_{\mathbb{R}^3} T_0 \exp\left(-\frac{(x-\xi)^2+(y-\eta)^2+(z-\zeta)^2}{4a^2t}\right) d\xi d\eta d\zeta,$$

$$U(x,y,z,t-\tau) * f(x,y,z,\tau) = \frac{1}{(2a\sqrt{\pi})^3} \iiint\limits_{\mathbb{R}^3} \frac{f(\xi,\eta,\zeta,\tau)}{(t-\tau)^{\frac{3}{2}}}$$

$$\times \exp\left(-\frac{(x-\xi)^2+(y-\eta)^2+(z-\zeta)^2}{4a^2(t-\tau)}\right) d\xi d\eta d\zeta,$$

in which $a = \sqrt{\frac{\kappa}{\rho C_p}}$. We discretize the solution region Ω into small elements $\Omega_l (l = 1, 2, \ldots, n)$. For the small element Ω_l, the corresponding tetrahedron elements are chosen as follows:

$$\varphi_{i,l}(x,y,z) = \frac{1}{6V_l}(c_{i1,l} + c_{i2,l}x + c_{i3,l}y + c_{i4,l}z), \tag{25}$$

where

$$V_l = \frac{1}{6}\det\begin{bmatrix} 1 & x_{1,l} & y_{1,l} & z_{1,l} \\ 1 & x_{2,l} & y_{2,l} & z_{2,l} \\ 1 & x_{3,l} & y_{3,l} & z_{3,l} \\ 1 & x_{4,l} & y_{4,l} & z_{4,l} \end{bmatrix}, \tag{26}$$

and c_{ij} is the (i, j) cofactor of the matrix in (26) with i and $j = 1, 2, 3, 4$.

Substituting the following expression

$$T(x, y, z, t) = \sum_{i=1}^{4} \varphi_{i,l}(x, y, z)T_{i,l}(t),\ (x, y, z) \in \Omega_l,$$

into (24) yields

$$\int_{\Omega_l} \varphi_{j,l}(x, y, z) \sum_{i=1}^{4} \varphi_{i,l}(x, y, z)T_{i,l}(t)d\Omega_l = \psi_{j,l}(t), j = 1, 2, 3, 4, \tag{27}$$

where

$$\begin{aligned}\psi_{j,l}(t) = &\int_{\Omega_l} \varphi_{j,l}(x, y, z)(U(x, y, z, t) * T_0)d\Omega_l \\ &+ \int_{\Omega_l} \varphi_{j,l}(x, y, z)\left(\int_0^t U(x, y, z, t-\tau) * f(x, y, z, \tau)d\tau\right)d\Omega_l.\end{aligned}$$

For simplicity, let

$$\begin{aligned}\Phi_l(x, y, z) \quad &= [\varphi_{1,l}(x, y, z)\ \ \varphi_{2,l}(x, y, z)\ \ \varphi_{3,l}(x, y, z)\ \ \varphi_{4,l}(x, y, z)], \\ T_l(t) = &[T_{1,l}(t)\ \ T_{2,l}(t)\ \ T_{3,l}(t)\ \ T_{4,l}(t)]^{\mathrm{T}}, \\ \Psi_l(t) = &[\psi_{1,l}(t)\ \ \psi_{2,l}(t)\ \ \psi_{3,l}(t)\ \ \psi_{4,l}(t)]^{\mathrm{T}},\end{aligned}$$

then from (27), we get

$$\int_{\Omega_l} \Phi_l(x, y, z)^{\mathrm{T}}\Phi_l(x, y, z)d\Omega_l T_l(t) = \Psi_l(t). \tag{28}$$

It can be observed from (28) that the solution of Eq. (3) can be written as follows:

$$T(x, y, z, t) = \Phi_l(x, y, z)\left(\int_{\Omega_l} \Phi_l(x, y, z)^{\mathrm{T}}\Phi_l(x, y, z)d\Omega_l\right)^{-1}\Psi_l(t). \tag{29}$$

Note that the matrix $\Phi_l(x, y, z)(\int_{\Omega_l} \Phi_l(x, y, z)^{\mathrm{T}}\Phi_l(x, y, z)d\Omega_l)^{-1}$ is an $\mathcal{H}$-matrix because the matrix $\int_{\Omega_l} \Phi_l(x, y, z)^{\mathrm{T}}\Phi_l(x, y, z)d\Omega_l$ is hierarchical. From (21) and (29), we can see that the global matrix $[\mathbf{G} + \frac{\mathbf{C}}{\Delta t}]$ can be converted to an $\mathcal{H}$-matrix. This completes the proof. □

4. $\mathcal{H}$-MATRIX-BASED INVERSE AND CHOLESKY FACTORS OF FEM SYSTEM AND THEIR ACCURACY CONTROL

In this section, we look at the problem of $\mathcal{H}$-matrix representation of the inverse and Cholesky factors of the thermal matrices in both steady-state and transient FEM analysis.

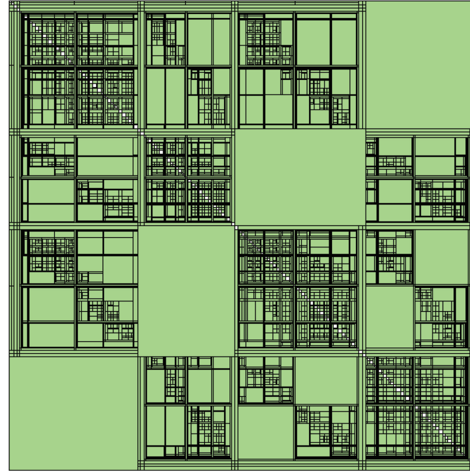

Fig. 4. An example of the structure of $\mathbf{H}$ and $\mathbf{H}^{-1}$.

4.1. Direct Inverse of $\mathcal{H}$-Matrix

Note that heat equation for steady-state analysis is an elliptic PDE Eq. (3), and the inverse of $\mathbf{H}_{\text{steady}}$ is also an $\mathcal{H}$-matrix [Bebendorf and Hackbusch 2003]. But the heat equation for transient analysis is a parabolic PDE of from Eq. (3) [Ozisik 1994], and not many studies have been reported for $\mathcal{H}$-matrix representation of FEM matrices from parabolic PDE so far. On the other hand, we observe that the structure of $[\mathbf{G} + \frac{\mathbf{C}}{\Delta t}]$ is identical to that of $\mathbf{G}$ at each time interval by comparing (14) and the second term of Eq. (15). Therefore, $\mathbf{H}_{\text{transient}}$ and $\mathbf{H}_{\text{steady}}$ have the same structure.

Consider the $\mathcal{H}$-matrix format of the matrix $\mathbf{H}$; it can be written as

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{21} & \mathbf{H}_{22} \end{bmatrix}. \tag{30}$$

The inverse of $\mathbf{H}$ can be computed by using the follow equation:

$$\mathbf{H}^{-1} = \begin{bmatrix} \mathbf{H}_{11}^{-1} + \mathbf{H}_{11}^{-1}\mathbf{H}_{12}\mathbf{S}^{-1}\mathbf{H}_{21}\mathbf{H}_{11}^{-1} & -\mathbf{H}_{11}\mathbf{H}_{12}\mathbf{S}^{-1} \\ \mathbf{S}^{-1}\mathbf{H}_{21}\mathbf{H}_{11}^{-1} & \mathbf{S}^{-1} \end{bmatrix}, \tag{31}$$

where $\mathbf{S} = \mathbf{H}_{22} - \mathbf{H}_{21}\mathbf{H}_{11}^{-1}\mathbf{H}_{12}$. This process only requires computing the inverse of $\mathbf{H}_{11}$ and $\mathbf{S}$, which is much faster than computing the whole inverse of $\mathbf{H}$.

During the inversion procedure, the $\mathcal{H}$-matrix format in $\mathbf{H}$ is preserved such that both $\mathbf{H}$ and $\mathbf{H}^{-1}$ have the same structure but not the same sparsity pattern. Figure 4 shows an example of the structure of $\mathbf{H}$. For the Cholesky factorization, the $\mathcal{H}$-matrix format is cut in half along the diagonal: Figure 5 shows the corresponding matrix structure of the resultant Cholesky factors. The results are stored in the form of $\mathbf{AB}^{\mathrm{T}}$ in Eq. (23) for admissible blocks in both inverse and Cholesky factors.

It is shown that we have the following result [Grasedyck and Hackbusch 2003]:

$$\mathtt{H}^{-1} \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ \tilde{k}),$$

where $\tilde{k} = k\mathtt{log}N$. This is quite a loose upper bound for the rank used for the inverse of $\mathbf{H}$. Our experimental results show that the rank $\tilde{k}$ in $\mathtt{H}^{-1}$ can be much smaller and may be different for different admissible blocks. Hence $\mathcal{H}$-matrix-based inverse and Cholesky factorization (matrix from (21) is symmetric) can also be applied to our problem, which is also called $\mathcal{H}$-arithmetics [Borm et al. 2003].

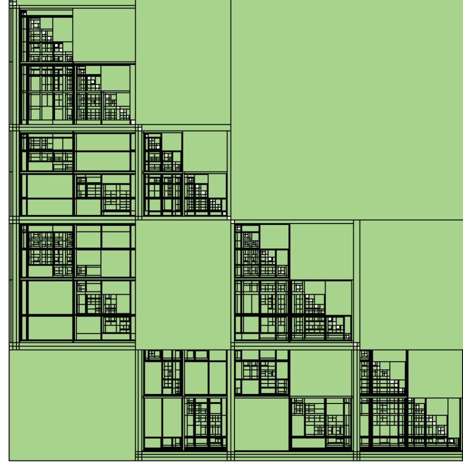

Fig. 5. An example of the structure of Cholesky factors.

Now assume that matrices $\mathbf{G}$ and $\mathbf{C}$ are nonsingular: their inverse matrices can be presented by the $\mathcal{H}$-matrix as we have previously show. Then we have the following result.

THEOREM 3. *Let* $\mathbf{G}^{-1} \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ k_2)$ *and* $\mathbf{C}^{-1} \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ k_3)$*, then we have*

$$\left(\mathbf{G} + \frac{\mathbf{C}}{\Delta t}\right)^{-1} \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ k_4),$$

where k_4 *is a certain positive integer.*

PROOF. According to the definition of the $\mathcal{H}$-matrix, we can assume that for the block $r \times s \subset \mathcal{I} \times \mathcal{I}$, matrices $\mathbf{G}^{-1}_{r\times s}$ and $\mathbf{C}^{-1}_{r\times s}$ have the following factorizations:

$$\mathbf{G}^{-1}_{r\times s} = \mathbf{G}_1\mathbf{G}_2^{\mathrm{T}},\ \mathbf{C}^{-1}_{r\times s} = \mathbf{C}_1\mathbf{C}_2^{\mathrm{T}}, \tag{32}$$

where $\mathbf{G}_1 \in \mathbb{R}^{|r|\times k_2}$, $\mathbf{G}_2 \in \mathbb{R}^{|s|\times k_2}$, $\mathbf{C}_1 \in \mathbb{R}^{|r|\times k_3}$, and $\mathbf{C}_2 \in \mathbb{R}^{|s|\times k_3}$.

Since matrices $\mathbf{G}$ and $\mathbf{C}$ are both nonsingular, we have

$$\left(\mathbf{G} + \frac{\mathbf{C}}{\Delta t}\right)^{-1} = \mathbf{G}^{-1} - \mathbf{G}^{-1}\left(\left(\frac{\mathbf{C}}{\Delta t}\right)^{-1} + \mathbf{G}^{-1}\right)^{-1}\mathbf{G}^{-1}. \tag{33}$$

For simplicity, we define the following matrix;

$$\mathbf{F} = \left(\left(\left(\frac{\mathbf{C}}{\Delta t}\right)^{-1} + \mathbf{G}^{-1}\right)^{-1}\mathbf{G}^{-1}\right)^{\mathrm{T}}.$$

Thus, for the $r \times s$ subblock of the matrix $(\mathbf{G} + \frac{\mathbf{C}}{\Delta t})^{-1}$, we obtain from (33) that

$$\left(\mathbf{G} + \frac{\mathbf{C}}{\Delta t}\right)^{-1}_{r\times s} = \mathbf{G}^{-1}_{r\times s} - (\mathbf{G}^{-1}\mathbf{F}^{\mathrm{T}})_{r\times s}. \tag{34}$$

By the arithmetic process of matrix multiplication, we know that

$$(\mathbf{G}^{-1}\mathbf{F}^{\mathrm{T}})_{r\times s} = \mathbf{G}^{-1}_{r\times\mathcal{I}}\mathbf{F}^{\mathrm{T}}_{\mathcal{I}\times s}. \tag{35}$$

From (34), (35), and the first equality in (32), it yields

$$\left(\mathbf{G}+\frac{\mathbf{C}}{\Delta t}\right)^{-1}_{r\times s}=\mathbf{G}_1\mathbf{G}_2^{\mathrm{T}}-\mathbf{G}^{-1}_{r\times\mathcal{I}}\mathbf{F}^{\mathrm{T}}_{\mathcal{I}\times s}. \tag{36}$$

Furthermore, we set

$$\hat{\mathbf{G}}_1=\begin{bmatrix}\mathbf{G}_1 & \mathbf{0}_{r\times(\mathcal{I}\backslash k_2)}\end{bmatrix}\in\mathbb{R}^{|r|\times|\mathcal{I}|},\ \hat{\mathbf{G}}_2=\begin{bmatrix}\mathbf{G}_2 & \mathbf{0}_{s\times(\mathcal{I}\backslash k_2)}\end{bmatrix}\in\mathbb{R}^{|s|\times|\mathcal{I}|}. \tag{37}$$

It is easy to verify that $\hat{\mathbf{G}}_1\hat{\mathbf{G}}_2^{\mathrm{T}}=\mathbf{G}_1\mathbf{G}_2^{\mathrm{T}}$. Thus, we obtain from (36) that

$$\left(\mathbf{G}+\frac{\mathbf{C}}{\Delta t}\right)^{-1}_{r\times s}=\hat{\mathbf{G}}_1\hat{\mathbf{G}}_2^{\mathrm{T}}-\mathbf{G}^{-1}_{r\times\mathcal{I}}\mathbf{F}^{\mathrm{T}}_{\mathcal{I}\times s}=\begin{bmatrix}\hat{\mathbf{G}}_1-\mathbf{G}^{-1}_{r\times\mathcal{I}}\end{bmatrix}\begin{bmatrix}\hat{\mathbf{G}}_2 & (\mathbf{F}^{\mathrm{T}}_{\mathcal{I}\times s})^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}.$$

Therefore, it holds that

$$\texttt{rank}\left(\left(\mathbf{G}+\frac{\mathbf{C}}{\Delta t}\right)^{-1}_{r\times s}\right)\leq\min\left\{\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{G}}_1 & -\mathbf{G}^{-1}_{r\times\mathcal{I}}\end{bmatrix}\right),\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{G}}_2 & (\mathbf{F}^{\mathrm{T}}_{\mathcal{I}\times s})^{\mathrm{T}}\end{bmatrix}\right)\right\}, \tag{38}$$

which proves the assertion. □

Similar to the previous process, we can also prove that

$$\texttt{rank}\left(\left(\mathbf{G}+\frac{\mathbf{C}}{\Delta t}\right)^{-1}_{r\times s}\right)\leq\min\left\{\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{C}}_1 & -\mathbf{C}^{-1}_{r\times\mathcal{I}}\end{bmatrix}\right),\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{C}}_2 & (\mathbf{J}^{\mathrm{T}}_{\mathcal{I}\times s})^{\mathrm{T}}\end{bmatrix}\right)\right\}, \tag{39}$$

where $\hat{\mathbf{C}}_1=[\mathbf{C}_1\quad\mathbf{0}_{r\times(\mathcal{I}\backslash k_3)}]\in\mathbb{R}^{|r|\times|\mathcal{I}|}$, $\hat{\mathbf{C}}_2=[\mathbf{C}_2\quad\mathbf{0}_{s\times(\mathcal{I}\backslash k_3)}]\in\mathbb{R}^{|s|\times|\mathcal{I}|}$, and $\mathbf{J}=((\mathbf{G}^{-1}+(\frac{\mathbf{C}}{\Delta t})^{-1})^{-1}(\frac{\mathbf{C}}{\Delta t})^{-1})^{\mathrm{T}}$. Estimates from (38) and (39) for each block $r\times s\subset\mathcal{I}\times\mathcal{I}$ immediately lead to an estimate for k_4 due to

$$\begin{aligned}k_4\leq\max_{r\times s\subset\mathcal{I}\times\mathcal{I}}\min\Big\{&\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{G}}_1 & -\mathbf{G}^{-1}_{r\times\mathcal{I}}\end{bmatrix}\right),\\ &\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{G}}_2 & (\mathbf{F}^{\mathrm{T}}_{\mathcal{I}\times s})^{\mathrm{T}}\end{bmatrix}\right),\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{C}}_1 & -\mathbf{C}^{-1}_{r\times\mathcal{I}}\end{bmatrix}\right),\texttt{rank}\left(\begin{bmatrix}\hat{\mathbf{C}}_2 & (\mathbf{J}^{\mathrm{T}}_{\mathcal{I}\times s})^{\mathrm{T}}\end{bmatrix}\right)\Big\}.\end{aligned}$$

4.2. $\mathcal{H}$-Cholesky Decomposition

For most FEM applications, we needn't compute the whole inverse of the $\mathcal{H}$-matrix $\mathbf{H}$ to solve the system $\mathbf{HT}=\mathbf{f}$. Instead, it is effective to perform Cholesky decomposition of $\mathbf{H}$. In our the thermal FEM analysis, the matrix $\mathbf{H}$ is symmetric, positive semidefinite, and hierarchical. Therefore, an $\mathcal{H}$-Cholesky decomposition can be defined as the form $\mathbf{H}=\mathbf{L}\mathbf{L}^{\mathrm{T}}$, where matrix $\mathbf{L}$ is stored in the $\mathcal{H}$-form.

For simplicity, we assume that matrix $\mathbf{H}$ can be subdivided into 2×2 submatrices, that is, $\mathbf{H}$ has the form of (30). Then, $\mathbf{H}$ can be factorized into the following form:

$$\mathbf{H}=\begin{bmatrix}\mathbf{L}_{11} & \mathbf{0}\\ \mathbf{L}_{21} & \mathbf{L}_{22}\end{bmatrix}\begin{bmatrix}\mathbf{L}_{11}^{\mathrm{T}} & \mathbf{L}_{21}^{\mathrm{T}}\\ \mathbf{0} & \mathbf{L}_{22}^{\mathrm{T}}\end{bmatrix}=\begin{bmatrix}\mathbf{L}_{11}\mathbf{L}_{11}^{\mathrm{T}} & \mathbf{L}_{11}\mathbf{L}_{21}^{\mathrm{T}}\\ \mathbf{L}_{21}\mathbf{L}_{11}^{\mathrm{T}} & \mathbf{L}_{22}\mathbf{L}_{22}^{\mathrm{T}}\end{bmatrix}, \tag{40}$$

in which $\mathbf{L}=[\begin{smallmatrix}\mathbf{L}_{11} & \mathbf{0}\\ \mathbf{L}_{21} & \mathbf{L}_{22}\end{smallmatrix}]$. Comparing (40) with (30), the Cholesky factorization can be recursively computed as follows.

(1) Compute $\mathbf{L}_{11}$ by $\mathcal{H}$-Cholesky factorization $\mathbf{H}_{11}=\mathbf{L}_{11}\mathbf{L}_{11}^{\mathrm{T}}$.
(2) Compute $\mathbf{L}_{21}$ by solving $\mathbf{H}_{21}=\mathbf{L}_{21}\mathbf{L}_{11}^{\mathrm{T}}$.
(3) Compute $\mathbf{L}_{22}$ by $\mathcal{H}$-Cholesky factorization $\mathbf{H}_{22}=\mathbf{L}_{22}\mathbf{L}_{22}^{\mathrm{T}}$.

After obtaining factor $\mathbf{L}$, the FEM system can be solved in two steps.

(1) Compute $\mathbf{LX} = \mathbf{f}$ by $\mathcal{H}$-arithmetics.
(2) Compute $\mathbf{L}^{\mathrm{T}}\mathbf{T} = \mathbf{X}$ by $\mathcal{H}$-arithmetics.

The $\mathcal{H}$-arithmetic for solving the proceding equations follows.

$$\mathbf{LX} = \begin{bmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{L}_{21} & \mathbf{L}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix}, \tag{41}$$

solve $\mathbf{X}_1$ from $\mathbf{L}_{11}\mathbf{X}_1 = \mathbf{f}_1$, then solve $\mathbf{X}_2$ from $\mathbf{L}_{22}\mathbf{X}_2 = \mathbf{f}_2 - \mathbf{L}_{21}\mathbf{X}_1$.

The steps in the $\mathcal{H}$-Cholesky algorithm are analogous to those for the normal Cholesky decomposition. However, by using the hierarchical structure, the $\mathcal{H}$-Cholesky algorithm can effectively reduce the complexity of classical methods.

4.3. Adaptive Rank Scheme

Since the admissible blocks in $\mathbf{H}$ are stored in the form of low-rank matrices, as in (23), the accuracy of inversion and Cholesky factorization depends on the choice of rank k. However, there are many admissible blocks with different sizes. If a constant rank k is used, the accuracy in a large admissible block may not be guaranteed for a small k, or the efficiency of the computation may be lower for a large k. From Borm et al. [2003], adaptive rank scheme can be used instead of a constant rank.

Assume $\mathbf{H}_{t\times s}$ is in admissible block $\mathcal{T}_{t\times s}$; then it can be represented in the form of (23):

$$\mathbf{H}_{t\times s} \approx \mathbf{A}\mathbf{B}^{\mathrm{T}} = \hat{\mathbf{H}}_{t\times s}, \tag{42}$$

where $\mathbf{A} \in \mathbb{R}^{|t|\times k}$, $\mathbf{B} \in \mathbb{R}^{|s|\times k}$. The singular value decomposition can be used to represent a matrix in factorized form. Following the methodology in Borm et al. [2003], we can perform the singular value decomposition $\mathbf{H}_{t\times s} = \mathbf{U}\Sigma\mathbf{V}^{\mathrm{T}}$, where $\mathbf{U} \in \mathbb{R}^{|t|\times|t|}$, $\mathbf{V} \in \mathbb{R}^{|s|\times|s|}$ are unitary matrices, and Σ is a diagonal matrix with the diagonal entries

$$\Sigma_{11} \geq \Sigma_{22} \geq \cdots \geq \Sigma_{kk} \geq \Sigma_{k+1,k+1} = \cdots = \Sigma_{\min\{|t|,|s|\},\min\{|t|,|s|\}} = 0.$$

By introducing the relative truncation error ε, rank k for submatrices of the matrix $\mathbf{H}$ is determined by

$$k := \min\{\hat{k} \in \mathbb{N}_0 \mid \exists \hat{\mathbf{H}}_{t\times s} \in \mathcal{R}(\hat{k}, t, s)\}, \tag{43}$$

subject to

$$\|\mathbf{H}_{t\times s} - \hat{\mathbf{H}}_{t\times s}\| \leq \varepsilon \|\mathbf{H}_{t\times s}\|, \tag{44}$$

where

$$\mathcal{R}(k, s, t) := \{\mathbf{M} \in \mathbb{R}^{|s|\times|t|} \mid \mathrm{rank}(\mathbf{M}) \leq k\}. \tag{45}$$

For a fixed rank $\hat{k} < k$, we can get an approximation to $\hat{\mathbf{H}}_{t\times s}$ by performing the following steps.

(1) Compute the QR-decompositions $\mathbf{A} = \mathbf{Q_A}\mathbf{R_A}$, $\mathbf{B} = \mathbf{Q_B}\mathbf{R_B}$, where $\mathbf{Q_A} \in \mathbb{R}^{t\times k}$, $\mathbf{Q_B} \in \mathbb{R}^{s\times k}$, $\mathbf{R_A}, \mathbf{R_B} \in \mathbb{R}^{k\times k}$.
(2) Compute the singular decomposition $\mathbf{R_A}\mathbf{R_B}^{\mathrm{T}} = \mathbf{U}'\Sigma\mathbf{V}'^{\mathrm{T}}$, where $\mathbf{U}', \Sigma, \mathbf{V}' \in \mathbb{R}^{k\times k}$.
(3) Compute $\tilde{\mathbf{U}} = \mathbf{Q_A}\mathbf{U}'$ and $\tilde{\mathbf{V}} = \mathbf{Q_B}\mathbf{V}'$.
(4) Compute $\hat{\mathbf{U}} = \tilde{\mathbf{U}}|_{t\times\hat{k}}$, $\hat{\mathbf{V}} = \tilde{\mathbf{V}}|_{s\times\hat{k}}$ and $\hat{\Sigma} = \mathtt{diag}(\Sigma_{11}, \ldots, \Sigma_{\hat{k}\hat{k}})$.

Let $\hat{\mathbf{H}}_{t\times s} = \hat{\mathbf{U}}\hat{\Sigma}\hat{\mathbf{V}}^{\mathrm{T}}$, then it is a best approximation in the sense of Frobenius norm, that is, we have

$$\|\mathbf{H}_{t\times s} - \hat{\mathbf{H}}_{t\times s}\| \leq \sqrt{\sum_{i=\hat{k}+1}^{\min\{|t|,|s|\}} \Sigma_{ii}^2}.$$

From the proceding inequality, we can see that the accuracy of the results can be controlled by choosing an appropriate ε. To perform the adaptive rank scheme, computing the QR-decompositions $\mathbf{A}$ and $\mathbf{B}$, the singular value decomposition $\mathbf{R_A}\mathbf{R_B}^{\mathrm{T}}$, $\tilde{\mathbf{U}}$, and $\tilde{\mathbf{V}}$, require $\mathcal{O}((|t|+|s|)k^2)$, $\mathcal{O}(k^3)$, $\mathcal{O}((|t|+|s|)k^2)$ flops, respectively. Therefore, the adaptive rank algorithm has almost linear complexity of $\mathcal{O}(k^2 \max\{|t|, |s|\})$. However, the complexity for computing the singular value decomposition of a general matrix $\hat{\mathbf{H}}_{t\times s}$ by a standard method in Golub and Loan [1996] is bounded by $\mathcal{O}(\min\{|t|, |s|\}(\max\{|t|, |s|\})^2)$.

5. COMPLEXITY ANALYSIS

Complexity analysis for general $\mathcal{H}$-arithmetics [Borm et al. 2003] and for FE-based electromagnetic analysis [Liu and Jiao 2009a] are well developed. Since FE-based thermal analysis is similar to that in electromagnetic, complexity analysis of the inverse [Liu and Jiao 2009a] can be adopted in this article.

During complexity analysis, an important parameter, sparsity constant C_{sp}, is introduced, which is defined as follows:

$$C_{sp} := \max\left\{\max_{t\in\mathcal{T}_{\mathcal{I}}}\{s \in \mathcal{T}_{\mathcal{I}} \mid s\times t \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}}\},\ \max_{s\in\mathcal{T}_{\mathcal{I}}}\{t \in \mathcal{T}_{\mathcal{I}} \mid s\times t \in \mathcal{T}_{\mathcal{I}\times\mathcal{I}}\}\right\}.$$

From Borm et al. [2003], for any $\mathcal{H}$-matrix $\mathbf{H}$ built from the block cluster tree $T_{\mathcal{I}\times\mathcal{I}}$ with tree depth $p = \log N$, minimum leave size n_{min}, and rank k, we can see that its storage complexity $\mathbf{H}_{St}(T, k)$ is bounded by

$$\mathbf{H}_{St}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, k) \leq 2C_{sp}(p+1)\max\{k, n_{min}\}N \sim \mathcal{O}(k_1 N\log N),$$

where $k_1 = \max\{k, n_{min}\}$. Since $\mathbf{H} \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ k)$, we know that $\mathbf{H}^{-1} \in \mathcal{H}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}},\ k\log N)$. Thus, the storage complexity $\mathbf{H}_{St}^{-1}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, k\log N)$ can be bounded by

$$\begin{aligned}
&\mathbf{H}_{St}^{-1}(\mathcal{T}_{\mathcal{I}\times\mathcal{I}}, k\log N)\\
&\quad\leq 2C_{sp}(k\log N+1)\max\{k\log N, n_{min}\}N\\
&\quad= 2C_{sp}(k\log^2 N + \log N)\max\{k, \tfrac{n_{min}}{\log N}\}N\\
&\quad\sim \mathcal{O}(k_2 N\log^2 N),
\end{aligned}$$

where $k_2 = k\max\{k, \frac{n_{min}}{\log N}\}$.

The complexity of computing the inverse is the sum of the complexity of $\mathcal{H}$-based multiplication and addition, while Cholesky factorization only depends on the complexity of multiplication. Both $\mathcal{H}$-based multiplication and addition for the entire cluster tree are bounded by the complexity of $\mathcal{O}(k_1^2 N\log^2 N)$ [Liu and Jiao 2009a]. Therefore, the complexity of computing inverse and $\mathcal{H}$-Cholesky factors are also bounded by the complexity of $\mathcal{O}(k_1^2 N\log^2 N)$. After $\mathcal{H}$-Cholesky factorization, solving Eq. (21) for each time interval only has a complexity of $\mathcal{O}(k_1 N\log N)$.

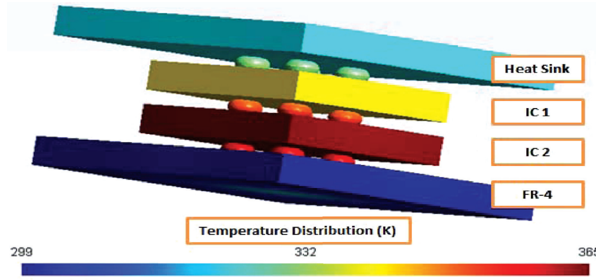

Fig. 6. A 3D-IC model with TSV.

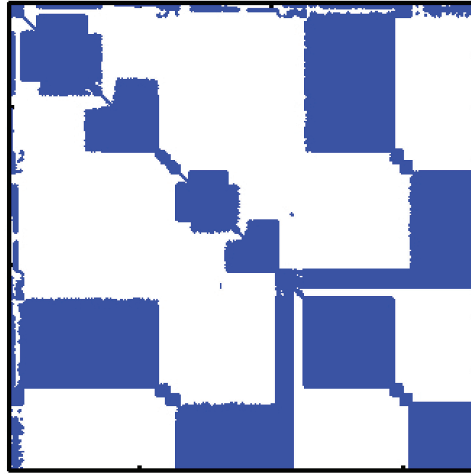

Fig. 7. The sparsity pattern of the transient thermal matrix $\mathbf{H}_{\text{transient}}$.

6. NUMERICAL RESULTS AND DISCUSSIONS

The proposed $\mathcal{H}$-based finite element direct solver is prototyped in C-based hierarchical matrices library.[4] The temperature distribution of the TSV (through silicon vias) is shown in Figure 6. The experiments are carried out on a Linux server with a 2.4GHz Intel Xeon quad-core CPU and 36GB memory.

The meshes are generated by the *Gmsh* program for the structure shown in Figure 6. To illustrate the proposed method, we create a 3D-IC structure which consists of two active layers with TSVs and micro-bumps connecting the layers, as shown in Figure 6. The ambient temperature is set to 300K. To demonstrate the effectiveness of the $\mathcal{H}$-based solver, we need to show the almost-linear complexity of $\mathcal{H}$-based Cholesky factorization for the FEM thermal matrices obtained in terms of both memory space and CPU time.

The 3D-IC structure is discretized into tetrahedral elements with four nodes in each element to generate 7,239 to 1,232,188 unknowns for the complexity analysis. Matrix $\mathbf{G}$ generated from 3D IC is directly transferred into $\mathbf{H}_{\text{steady}}$ for steady analysis, while the matrix with capacity matrix $[\mathbf{G} + \frac{\mathbf{C}}{\Delta t}]$ is transferred into $\mathbf{H}_{\text{transient}}$ for transient analysis. Figure 7 shows the sparsity pattern of the transient thermal matrix $\mathbf{H}_{\text{transient}}$. Figure 8 demonstrates the sparsity pattern of the hierarchical representation of the

[4]http://www.hlib.org.

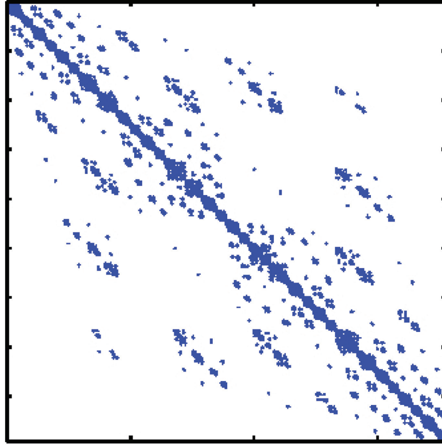

Fig. 8. The sparsity pattern of the $\mathcal{H}$-matrix representation of the transient thermal matrix.

Table I. $\bar{k}$ in Cholesky Factors

Matrix size	$\bar{k}$ in $\mathcal{H}$-Cholesky
7239	8.00
12300	8.55
31888	10.03
49687	10.81
76840	11.63
124344	12.61
177933	14.16
239028	16.56
354099	20.46
597436	28.67
825875	36.40
1232188	45.32

matrix $\mathbf{H}_{\text{transient}}$ by reordering. For these two figures, the x-axis represents the number of columns and the y-axis represents the number of rows. The dots in the figures represent the nonzero elements in the matrix. The simulation parameters of $\eta = 1$ and $n_{min} = 32$ are applied in both steady and transient scenarios. $\varepsilon = 1\text{e}{-9}$ is chosen to control the adaptive rank k in each tree level. Since the rank varies, the mean of rank $\bar{k}$ is used to evaluate the complexity such that $k_1 = \max\{\bar{k}, n_{min}\}$. The mean rank $\bar{k}$ of $\mathcal{H}$-Cholesky factorization is shown in Table I.

Note that the structure of $\mathbf{H}_{\text{transient}}$ at each time step is identical to the structure of $\mathbf{H}_{\text{steady}}$, and the storage complexity analysis of $\mathbf{H}_{\text{transient}}$ at any time interval can also apply to $\mathbf{H}_{\text{steady}}$. For CPU time complexity analysis, the solution time t_i for a time step i of $\mathbf{H}_{\text{transient}}$ is the same for every step; the total time for solving (21) is a multiple of t_i, which also has a time complexity bounded by $\mathcal{O}(k_1^2 N \log^2 N)$.

Table II shows the comparison between the storage of the $\mathcal{H}$-matrix method and the storage of the CSPARSE solver[3], which is the industry-strength LU solver, as well as the comparison between the CPU time consumed by $\mathcal{H}$-arithmetics and the CPU time consumed by the CSPARSE solver. The $\mathcal{H}$-based solver is a direct method for solving large linear systems. The $\mathcal{H}$-matrix uses data-sparse approximations in order to reduce the storage requirement. The CSPARSE method is also a direct solver for

Table II. Numerical Results for the 3D-IC Model with TSV

Matrix size	Storage for $\mathcal{H}$-matrix (MB)	Storage for CSPARSE (MB)	Memory saved (times)	CPU time for $\mathcal{H}$-matrix	CPU time for CSPARSE	Speedup (times)
7239	38.35	10.11	0.26	17.13	0.81	0.05
12300	87.07	44.25	0.50	45.92	2.46	0.05
31888	177.27	161.68	0.91	82.57	13.35	0.16
49687	246.48	303.85	1.23	105.98	32.86	0.31
76840	377.87	496.32	1.31	150.65	67.61	0.45
124344	622.55	957.55	1.53	258.09	189.73	0.74
177933	942.88	1492.29	1.58	391.55	358.20	0.91
239028	1319.50	2485.72	1.88	551.74	880.83	1.60
354099	2061.79	4197.67	2.04	846.16	1857.14	2.19
597436	3742.66	8398.34	2.24	1458.51	5129.64	3.52
825875	5754.54	Failed	N/A	2379.21	Failed	N/A
1232188	9249.30	Failed	N/A	3774.71	Failed	N/A

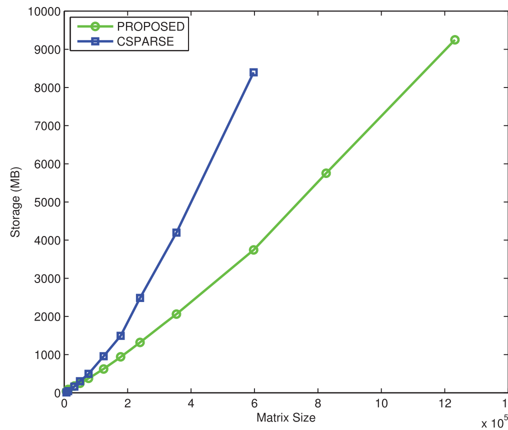

Fig. 9. Memory scalability for computing Cholesky factors.

solving sparse linear systems. The CSPARSE method uses the compressed sparse row format, which is a popular, general-purpose sparse matrix representation, to explicitly store column indices and nonzero values in arrays of indices and data. According to the table, memory required for storing Cholesky factors in the form of $\mathcal{H}$-matrix is much less than that of the CSPARSE solver. In addition, the $\mathcal{H}$-matrix-based Cholesky factorization requires less computation time compared with the LU solver if the size of matrix is large. The advantage becomes obvious if the size of matrix is beyond $49{,}687 \times 49{,}687$ for Cholesky factorization. The LU solver fails to work when the size of the thermal matrix is beyond 825,875, but the $\mathcal{H}$-based solver can still work for this case and for even larger matrices.

Figure 9 and Figure 10 show the scalability of the proposed method and LU-based methods in terms of CPU time and memory complexity. It shows the memory and CPU time complexities for computing the Cholesky factors of $\mathbf{H}_{\text{transient}}$. From those figures we can clearly see that the storage complexities and CPU time required for Cholesky factor match very well with the theoretical trends, $\mathcal{O}(k_1 N \log N)$ and $\mathcal{O}(k_1^2 N \log^2 N)$, which are almost linear.

We then study the accuracy in the proposed method. Figure 11 shows the transient pulse responses computed by the proposed $\mathcal{H}$-matrix based Cholesky factorization

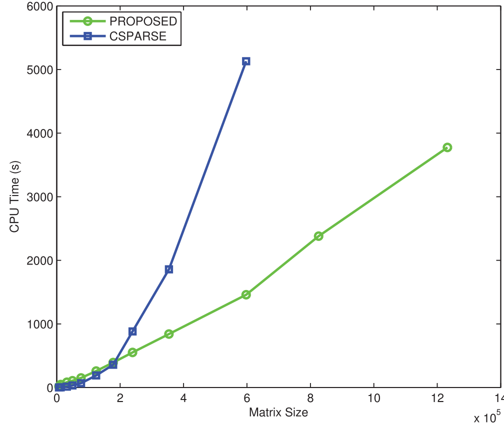

Fig. 10. CPU time scalability for computing Cholesky factors.

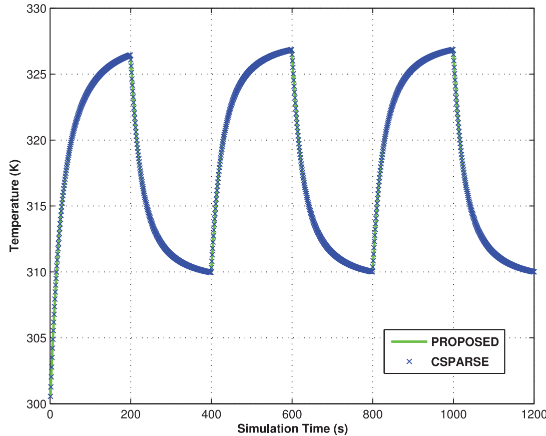

Fig. 11. Thermal transient responses by the $\mathcal{H}$-matrix and CSPARSE methods.

method and the LU method. Figure 12 shows the relative error of the proposed method compared with the LU method. It can be seen that errors are quite small (less than 0.025%) and is well acceptable for the practical application.

In addition to CSPARSE, we also did a comparison with UMFPACK, which is a right-looking multifrontal solver.[5] Our numerical simulation results show that for the given thermal matrices in Table II, UMFPACK indeed is slightly faster than the $\mathcal{H}$-based solver. But the UMFPACK solver fails when the thermal matrix size is greater than 597,436 due to memory failure. This shows the memory efficiency and scalability of our proposed method. UMFPACK is a highly optimized LU solver and it has very small overhead compared to the $\mathcal{H}$-based solver. But as the sizes of the problem increase, the proposed method will be faster due to its almost linear scalability and amortization of the its overhead over large problem sizes. We remark that CSPARSE and UMFPACK fail to work on some large examples on the given workstation, which prevents CPU time comparisons on those large examples.

[5]http://www.cise.ufl.edu/research/sparse/umfpack/.

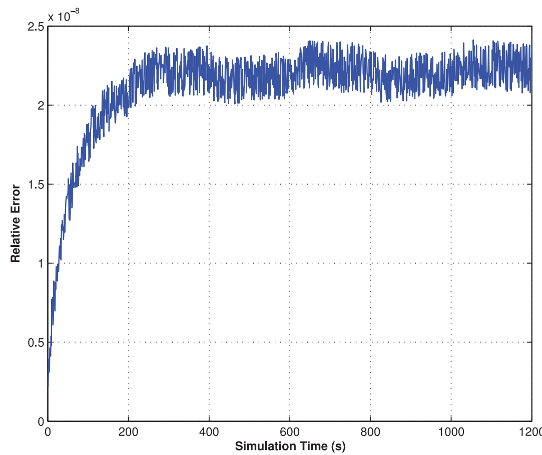

Fig. 12. Relative error of the $\mathcal{H}$-method over the CSPARSE method.

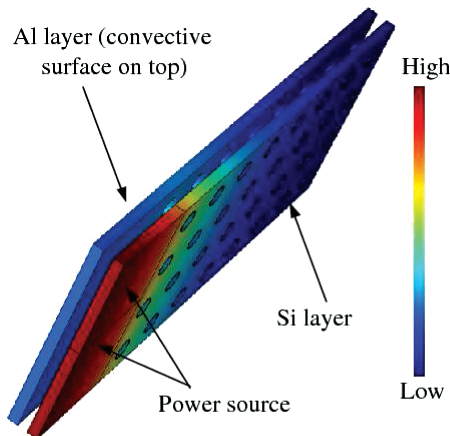

Fig. 13. Two-layer stack structure with a TSV array.

In addition to the two solvers, another well-known solver is CHOLMOD[6], which is a Cholesky factorization solver. We do not perform the comparison, as the proposed $\mathcal{H}$-based solver is the general nonsymmetric solver, and therefore it would be unfair to compare them directly. It is shown that the 3D ICs with microchannel-based liquid cooling technique can lead to nonsymmetric matrices, as the microchannel is modeled as voltage-controlled current sources[7], [Sridhar et al. 2010]. Hence CHOLMOD will not work for general thermal matrices.

In the following, we further demonstrate that the $\mathcal{H}$-based solver has the almost linear scalability in terms of both memory footprint and CPU time from a TSV array model [Liu et al. 2013]. In the experiment, a two-layer stack structure is connected by a TSV array. The structure is shown in Figure 13, which represents part of a 3D stacked chip structure that can be meshed by the *Gmsh* program. The 2D bottom view

[6]Supernodal sparse cholesky factorization and update/downdate. `http://www.cise.ufl.edu/research/sparse/cholmod/`.
[7]Interlayer cooling technology of 3D packages. `http://www.zurich.ibm.com/st/cooling/integrated.html`.

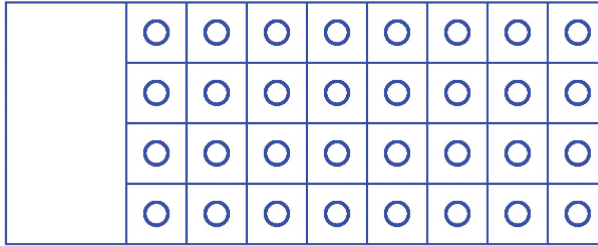

Fig. 14. 2D bottom view of the two-layer stack structure with TSVs.

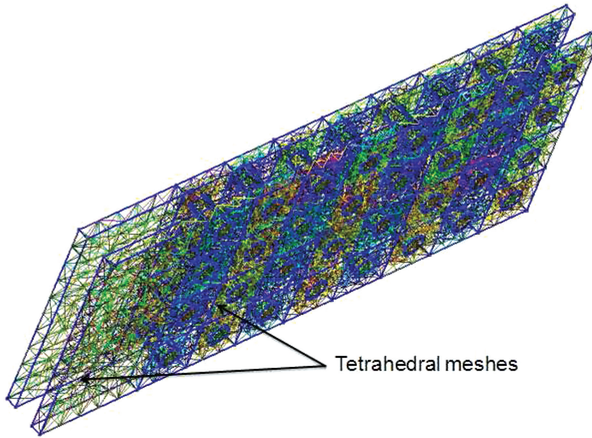

Fig. 15. 3D mesh for the two-layer stack structure with a TSV array.

of the two-layer chip structure is shown in Figure 14. Both the lateral dimensions of the Si layer and the Al layer are $200\mu m$ by $500\mu m$, and the thickness of the Si layer is $10\mu m$, while the thickness of Al layer is $20\mu m$. The two layers are connected by a TSV array consists of 32 TSVs (4×8 array), in total, as shown in Figure 13. The radius of each TSV is $10\mu m$, the insulation linear thickness is $1\mu m$, and the space between two adjacent TSVs is $30\mu m$, for nowadays, there are many works focusing on TSV models with the TSV radius at 10 micron level or even larger [Lu et al. 2009; Ni et al. 2010; Li et al. 2013]. It should be pointed out that for TSV radius size, this is just for illustration purposes of the proposed analysis algorithm. The proposed algorithm can work for any TSV structures. In this experiment, the step power sources of $1e6W/m^2$ are placed at the bottom of the Si layer, where there is no TSV, as shown in Figure 13 and Figure 14. The convective coefficient of the top surface of the Al layer shown in Figure 13 is set to $10{,}000W/m^2K$. For this TSV array, the thermal resistance model has been built [Liu et al. 2013].

As previously mentioned, we use the *Gmsh* program to generate the meshes for the structure shown in Figure 13. A tetrahedron mesh structure for the 3D-TSV array is shown in Figure 15. For the complexity analysis, the two-layer chip structure is discretized into tetrahedral elements with four nodes in each element to generate 5,452 to 886,154 unknowns. To demonstrate the effectiveness of the proposed solver, we show the complexity of $\mathcal{H}$-based Cholesky factorization for the FEM thermal matrices in steady state. The comparison of the complexities of the $\mathcal{H}$-matrix method and the CSPARSE solver in terms of both the storage requirement and the CPU time is shown in Table III. It can be seen from the table that the required memory for storing Cholesky

Table III. Numerical Results for the TSV Array Model

Matrix size	Storage for $\mathcal{H}$-matrix (MB)	Storage for CSPARSE (MB)	Memory saved (times)	CPU time for $\mathcal{H}$-matrix	CPU time for CSPARSE	Speedup (times)
5452	17.19	3.61	0.21	2.78	0.37	0.13
11115	50.32	18.62	0.37	14.34	1.99	0.14
22439	108.32	67.16	0.62	47.35	8.05	0.17
32525	159.13	138.44	0.87	97.02	25.23	0.26
70496	301.61	343.84	1.14	139.23	48.73	0.35
129445	561.19	774.44	1.38	210.27	128.26	0.61
215850	1103.15	1776.07	1.61	493.22	646.12	1.31
486260	2971.20	5318.45	1.79	1135.56	2191.63	1.93
886154	5860.69	Failed	N/A	2487.04	Failed	N/A

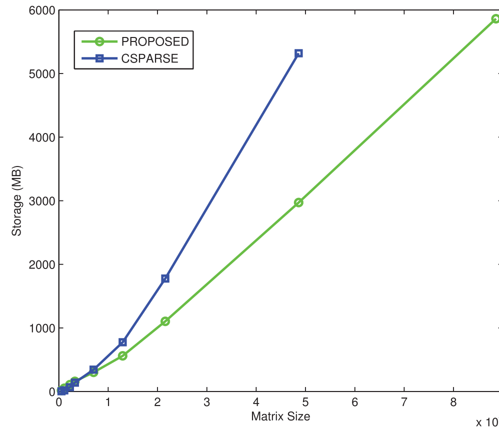

Fig. 16. Memory scalability for computing Cholesky factors.

factors in the form of an $\mathcal{H}$-matrix is much less than that of the CSPARSE solver, the obvious advantage being that less memory and CPU time are required when the size of the thermal matrix exceeds $70{,}496 \times 70{,}496$. In addition, the CSPARSE solver failes to work for the $886{,}154 \times 886{,}154$ thermal matrix, but the proposed $\mathcal{H}$-based solver can still be effective for this case. The complexities of the $\mathcal{H}$-based solver for the 3D-TSV array model from Table III are also demonstrated by Figure 16 and Figure 17. We can see from these figures that the complexities for the storage requirement and the CPU time are almost linear.

We remark that the speedup of the proposed method over the LU-based method by the CSPARSE solver is not very impressive for small thermal examples, which can be solved by the LU solver. For small cases, the proposed method may be even slower, the reason being that the $\mathcal{H}$-matrix solver has pretty high up-front overheads, as it needs to build the hierarchical matrix first from the existing thermal matrices. Our implementation is also not well optimized due to limited student programming experiences. In tcontrast, the CSPARSE is well designed and an industry-strength solver. The LU solver actually is very fast if the memory is not limited, as demonstrated by a recent power grid analysis contest, where all the wining solvers are LU-based solver [Yu and Wang 2012; Yang et al. 2012].

But for large examples, the proposed method starts to deliver increasing speedups over the LU solver, as shown in Table II. The CSPARSE solver can not work for very large examples in our server with 32GB memory. Hence the key difference between the

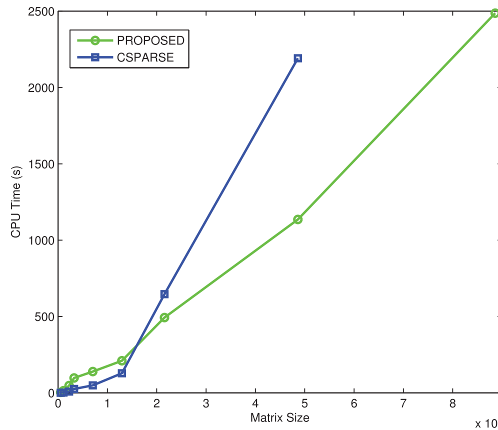

Fig. 17. CPU time scalability for computing Cholesky factors.

two methods is the growth rate of the CPU time and memory storage with the problem sizes, which makes the $\mathcal{H}$-matrix solver much more attractive for attacking very large problem sizes despite the limited computing resources available to us.

7. CONCLUSION AND FUTURE WORKS

In this article, we propose a new finite-element-based transient thermal analysis technique for high-performance integrated circuits based on the hierarchical matrix ($\mathcal{H}$) method. We showed that the $\mathcal{H}$-based techniques can be applied to the finite-element-based transient thermal analysis, which is described by the parabolic differential equations. We prove that the matrices from a finite-element-based steady and transient thermal analysis can be represented by $\mathcal{H}$-matrices without any approximation, and its inverse and Cholesky factors can be computed by $\mathcal{H}$-matrix arithmetics with controlled accuracy. Numerical results match the predicted theoretical scalability: that memory and time complexities of the solver are bounded by $\mathcal{O}(k_1 N \log N)$ and $\mathcal{O}(k_1^2 N \log^2 N)$ for large-scale thermal systems. The numerical results from two 3D-IC models demonstrate the almost-linear scalability of the proposed method in terms of both memory footprint and CPU time. The comparison with existing product-quality LU solvers, CSPARSE and UMFPACK, on a number of 3D-IC thermal matrices clearly shows the memory space advantage of the new method over these methods.

One way to further improve the $\mathcal{H}$-based solver is by means of parallel implementation on multicore or GPU platforms, which will be further investigated. The key problem is how to compute the Cholesky decompositions for the admissible matrix blocks in parallel process.

REFERENCES

M. Bebendorf and W. Hackbusch. 2003. Existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients. *Numerische Mathematik* 95, 1, 1–28.

T. Bergman, A. Lavine, F. Incropera, and D. DeWitt. 2011. *Fundamentals of Heat and Mass Transfer*. John Wiley & Son.

S. Borm, L. Grasedyck, and W. Hackbusch. 2003. Hierarchical matrices. Lecture notes 21 of the Max Planck Institute for Mathematics in the Sciences.

D. Brooks and M. Martonosi. 2001. Dynamic thermal management for high-performance microprocessors. In *Proceedings of the International Symposium on High-Performance Computer Architecture*. 171–182.

W. Chai and D. Jiao. 2010. $\mathcal{H}$- and $\mathcal{H}^2$-matrix-based fast integral-equation solvers for large-scale electromagnetic analysis. *IET Microwav. Antennas Propag.* 4, 10, 1583–1596.

Y. Cheng, C. Tsai, C. Teng, and S. Kang. 2000. *Electrothermal Analysis of VLSI Systems*. Kluwer Academic Publishers.

T. Davis. 2006. *Direct Methods for Sparse Linear Systems*. SIAM, Philadelphia.

G. Golub and C. V. Loan. 1996. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD.

L. Grasedyck and W. Hackbusch. 2003. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing* 70, 4, 295–334.

S. Gunther, F. Binns, D. Carmean, and J. Hall. 2001. Managing the impact of increasing microprocessor power consumption. *Intel Technol. J.* 5, 1.

W. Hackbusch. 1999. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices. *Computing* 62, 2, 89–108.

K. He, T. Yu, S. X.-D. Tan, H. Wang, and H. Tang. 2014. GPU-accelerated sparse matrix-vector multiplication with application on transient thermal analysis. In *Proceedings of the International Symposium on Quality Electronic Design (ISQED)*.

ITRS. 2012. International technology roadmap for semiconductors (ITRS), 2012 update. http://public.itrs.net.

G. Kornaros and D. Pnevmatikatos. 2013. A survey and taxonomy of on-chip monitoring of multicore systems-on-chip. *ACM Trans. Des. Autom. Electron. Syst.* 18, 2, 1–38.

Y. Lee and P. Huang. 2013. An efficient method for analyzing on-chip thermal reliability considering process variations. *ACM Trans. Des. Autom. Electron. Syst.* 18, 3, 1–32.

R. Lewis, P. Nithiarasu, and K. Seetharamu. 2004. *Fundamentals of the Finite Element Method for Heat and Fluis Flow*. John Wiley & Son.

D. Li, J. Kim, and S. Memik. 2013. Integrating thermocouple sensors into 3D ICS. In *Proceedings of the IEEE International Conference on Computer Design (ICCD)*. 221–226.

H. Liu and D. Jiao. 2009a. A direct finite-element-based solver of significantly reduced complexity for solving large-scale electromagnetic problems. In *Proceedings of the IEEE MTT-S International Microwave Symposium Digest.* 177–180.

H. Liu and D. Jiao. 2009b. Performance analysis of the $\mathcal{H}$-matrix-based fast direct solver for finite-element-based analysis of electromagnetic problems. In *Proceedings of the Antennas and Propagation Society International Symposium (APSURSI'09)*. IEEE, 1–4.

H. Liu and D. Jiao. 2010. $\mathcal{H}$-matrix-based fast direct finite element solver for large-scale electromagnetic analysis. *ECE Techn. Rep. Purdue University*.

X. Liu, K. Zhai, Z. Liu, K. He, S. X.-D. Tan, and W. Yu. 2015a. Parallel thermal analysis of 3D integrated circuits with liquid cooling on CPU-GPU platforms. *IEEE Trans. VLSI Syst.* 23, 3, 575–579.

Z. Liu, S. X.-D. Tan, X. Huang, and H. Wang. 2015b. Task migrations for distributed thermal management considering transient effects. *IEEE Trans. VLSI Syst.* 23, 2, 397–401.

Z. Liu, S. Swarup, S. X.-D. Tan, H.-B. Chen, and H. Wang. 2013. Compact lateral thermal resistance model of TSVs for fast finite-difference based thermal analysis of 3D stacked ICs. *IEEE Trans. Comput. Aid. Des. Integr. Circuits Syst.* 33, 10, 1490–1502.

Z. Liu, S. X.-D. Tan, H. Wang, Y. Hua, and A. Gupta. 2014. Compact thermal modeling for packaged microprocessor design with practical power maps. *Integr. VLSI J.* 47, 1, 71–85.

K. Lu, X. Zhang, S. Ryu, J. Im, R. Huang, and P. Ho. 2009. Thermo-mechanical reliability of 3D ICS containing through silicon vias. In *Proceedings of the Electronic Components and Technology Conference (ECTC)*. 630–634.

M. Ni, Q. Su, Z. Tang, and J. Kawa. 2010. An analytical study on the role of thermal TSVS in a 3DIC chip stack. In *Proceedings of the Design Automation and Test in Europe (DATE)*. 137–141.

M. Ozisik. 1994. *Finite Difference Methods in Heat Transfer*. Taylor & Francis, Inc.

M. Pedram and S. Nazarian. 2006. Thermal modeling, analysis, and management in VLSI circuits: Principles and methods. *Proc. IEEE* 94, 8, 1487–1501.

N. P. D. Sai, H. Yu, Y. Shang, C. S. Tan, and S. K. Lim. 2013. Reliable 3D clock-tree synthesis considering nonlinear capacitive TSV model with electrical-thermal-mechanical coupling. *IEEE Trans. Comput.-Aid. Des. Integr. Circuits Syst.* 32, 11, 1734–1747.

K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. 2003. Temperature-aware microarchitecture. In *Proceedings of the International Symposium on Computer Architecture*. 2–13.

A. M. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza. 2010. 3D-ICE: Fast compact transient thermal modeling for 3D-ICs with inter-tier liquid cooling. In *Proceedings of the International Conference on Computer Aided Design (ICCAD)*. IEEE, 463–470.

T. Yu and M. D. F. Wang. 2012. PGT_SOLVER: An efficient solver for power grid transient analysis. In *Proceedings of the International Conference on Computer Aided Design (ICCAD)*. 647–652.

T. Wan, X. Q. Hu, and R. S. Chen. 2010. An $\mathcal{H}$-matrix direct method with improved solution for finite element analysis of scattering problems. In *Proceedings of the International Conference on Microwave and Millimeter Wave Technology (ICMMT)*.

H. Wang, S. X.-D. Tan, D. Li, A. Gupta, and Y. Yuan. 2013. Composable thermal modeling and simulation for architecture-level thermal designs of multicore microprocessors. *ACM Trans. Des. Autom. Electron. Syst.* 18, 2, 28:1–28:27.

J. Yang, Z. Li, Y. Cai, and Q. Zhuo. 2012. PowerRush: Efficient transient simulation for power grid analysis. In *Proceedings of the International Conference on Computer Aided Design (ICCAD)*. 653–659.

H. Yu, J. Ho, and L. He. 2009. Allocating power ground vias in 3D ICs for simultaneous power and thermal integrity. *ACM Trans. Des. Autom. Electron. Syst.* 14, 3, 1–31.

H. Yu, Y. Shi, L. He, and T. Karnik. 2008. Thermal via allocation for 3D ICs considering temporally and spatially variant thermal power. *IEEE Trans. VLSI Syst.* 16, 12, 1609–1619.

Received May 2014; revised September, December 2014; accepted January 2015