MORE–Stress: Model Order Reduction based Efficient Numerical Algorithm for Thermal Stress Simulation of TSV Arrays in 2.5D/3D IC

Tianxiang Zhu[1], Qipan Wang[1,2], Yibo Lin[1,3,4*], Runsheng Wang[1,3,4], Ru Huang[1,3,4]

[1]School of Integrated Circuits, [2]Academy for Advanced Interdisciplinary Studies, Peking University, Beijing,
[3]Institute of Electronic Design Automation, Peking University, Wuxi,
[4]Beijing Advanced Innovation Center for Integrated Circuits
{txzhu, qpwang, yibolin, r.wang, ruhuang}@pku.edu.cn

Abstract—Thermomechanical stress induced by through-silicon vias (TSVs) plays an important role in the performance and reliability analysis of 2.5D/3D ICs. While the finite element method (FEM) adopted by commercial software can provide accurate simulation results, it is very time- and memory-consuming for large-scale analysis. Over the past decade, the linear superposition method has been utilized to perform fast thermal stress estimations of TSV arrays, but it suffers from a lack of accuracy. In this paper, we propose MORE-Stress, a novel strict numerical algorithm for efficient thermal stress simulation of TSV arrays based on model order reduction. Experimental results demonstrate that our algorithm can realize a 153–504× reduction in computational time and a 39–115× reduction in memory usage compared with the commercial software ANSYS, with negligible errors less than 1%. Our algorithm is as efficient as the linear superposition method, with an order of magnitude smaller errors and fast convergence.

I. INTRODUCTION

2.5D/3D ICs are one of the most promising technologies to achieve the increasingly demanding integration and performance targets [1]. Nevertheless, many reliability issues in 2.5D/3D ICs remain challenging, among which the thermomechanical stress induced by the mismatch in thermal expansion coefficients and the thermal load between the room temperature and fabrication temperature plays an important role. Severe thermal stress will incur mechanical cracking and damage, accelerate degradation, and affect the mobility of transistors, leading to degraded performance [2]–[5]. Generally, the thermal stress in 2.5D/3D ICs is evaluated by finite element method (FEM) in commercial software like ANSYS [6] for verification and design optimization [7]–[9].

However, 2.5D/3D ICs typically contain a significant number of local fine structures, such as through-silicon vias (TSVs), micro bumps, die-to-die interconnects, etc., as shown in Fig. 1. These numerous fine structures embedded in the large systems lead to the multi-scale nature of 2.5D/3D ICs and renders full-system thermal stress analysis by conventional FEM extremely time- and memory-consuming, which is one of the key numerical challenges [2]. Over the past decade, the linear superposition method has been widely used to perform fast estimations of thermal stress in 2.5D/3D ICs [3], [10]–[12]. Although this method greatly reduces the analysis time, it is not a convergent numerical method and suffers from low accuracy, especially when the adjacent fine structures are very close or the local variations in the background stress are very sharp. Advanced numerical algorithms that are efficient, accurate, and flexible are urgently needed to tackle the multi-scalability challenge in full-system thermal stress analysis for 2.5D/3D ICs.
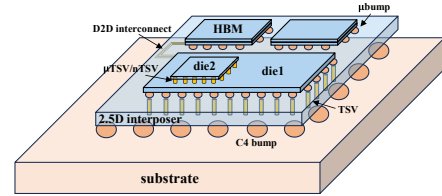
Fig. 1. Schematic of a typical 2.5D/3D IC with multi-scale character. Numerous local fine structures are embedded in the system, such as TSVs, micro bumps, die-to-die interconnects, etc. These fine structures render full-system thermal stress simulation by conventional FEM extremely expensive.

Fortunately, the most common fine structures in 2.5D/3D IC scenarios are generally configured in regular arrays for processing simplicity and reliability control, such as TSV arrays, micro bump arrays, C4 bump arrays, etc., where identical structures are repeated periodically [3], [7], [13], [14]. The periodicity allows us to perform model order reduction to significantly reduce the total number of DoFs in the system. Meanwhile, the integration with sub-modeling, a technique commonly applied to refining a small part of the solution, offers our algorithm with high flexibility in face of complex background stress.

Based on these novel ideas, we propose a strict numerical algorithm, named *MORE-Stress*, aiming at reducing the cost of thermal stress simulation of large-scale arrays of fine structures in 2.5D/3D ICs and overcoming the multi-scalability numerical challenge. In this paper, we apply our algorithm to the thermal stress simulation of TSV arrays, one of the most common and important fine structures in 2.5D/3D ICs, to demonstrate our algorithm's core principles and representative performances. It is worth noting that our algorithm is not limited by TSVs and is adaptable to other types of fine structures in 2.5D/3D ICs, such as micro bumps, pillars, direct bondings, etc., regardless of their geometries. The key contributions of this work are:

- We propose a novel strict numerical algorithm based on model order reduction, which remarkably improves the computational efficiency for thermal stress simulation of TSV arrays in 2.5D/3D ICs. The algorithm consists of a one-shot local stage, where reduced order models of the TSV structures are developed, and a global stage, where the thermal stress of TSV arrays with arbitrary array sizes and thermal loads is efficiently calculated.

- We design a procedure combined with the sub-modeling technique for calculating the thermal stress of TSV arrays embedded at arbitrary locations in a package system, which makes our algorithm highly flexible in various scenarios.

- Extensive experimental results show that our algorithm can realize a 153–504× reduction in computational time and a 39–115× reduction in memory usage compared with the commercial FEM software ANSYS, with negligible errors

less than 1%. Our algorithm is as efficient as the linear superposition method, with an order of magnitude smaller errors and fast convergence.

The rest of this paper is organized as follows. Section II reviews the relevant work. Section III formulates the studied problem. Section IV demonstrates the framework and principles of our algorithm. Section V provides the experimental setup and shows the results. Section VI concludes this paper. The source code has been released at [15].

II. RELATED WORK

For the past decade, the linear superposition method has been widely utilized to perform fast thermal stress estimations for 2.5D/3D ICs, especially for the TSV arrays [3], [11]. The idea of this method is to perform high-fidelity FEM simulation to get the stress tensors for each single structure and then superpose the stress tensors directly. This method greatly reduces the analysis time, but it overlooks the coupling between adjacent structures and the local variations in the global background stress, which leads to a lack of accuracy when the pitches are small or when the background stress changes sharply.

Several works are trying to improve the accuracy of the linear superposition method or find other solutions. Li et al. [10] proposed a two-stage semi-analytical method that considers the interactions between nearby TSVs to provide more accurate analysis results. However, this method cannot be generalized to more complicated structures, and there still exists certain errors because the authors only considered interactions between pairs of TSVs. Zhou et al. [12] proposed a novel adaptive strategy finite element method to better simulate the stress distribution of a single TSV and then utilize the naive linear superposition method to get the stress distribution of the whole TSV array. This work is still within the framework of the linear superposition method and does not make fundamental improvements. Besides, Wang et al. [16] have recently proposed a deep-learning-based method for the run-time thermal stress analysis of TSV arrays. However, the reliability of the deep-learning methods cannot be guaranteed and their model is limited to the training scenarios, so we do not consider similar deep-learning-based methods in this work.

III. PROBLEM FORMULATION

A. Governing Equation of Thermal Stress

The governing equation of thermal stress is as follows [17]:

$$-\nabla \cdot \sigma(\boldsymbol{u}) = \boldsymbol{f} \quad \text{in} \quad \Omega,$$
$$\sigma(\boldsymbol{u}) = \lambda \text{tr}(\epsilon(\boldsymbol{u})) \cdot \mathbf{1} + 2\mu\epsilon(\boldsymbol{u}) - \alpha(3\lambda + 2\mu) \cdot \Delta T \cdot \mathbf{1},$$
$$\epsilon(\boldsymbol{u}) = \frac{1}{2}(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T), \tag{1}$$

where Ω is the computational domain, $\boldsymbol{u}$ is the three-dimensional displacement vector field, $\epsilon(\boldsymbol{u})$ is the strain tensor field, $\sigma(\boldsymbol{u})$ is the stress tensor field, $\boldsymbol{f}$ is the body force vector, λ, μ are the Lamé parameters, α is the thermal expansion coefficient, ΔT is the thermal load between room temperature and package processing temperature, and $\mathbf{1}$ represents the second-order unit tensor. The displacement field $\boldsymbol{u}$, the strain tensor field ϵ, and the stress tensor field σ are all functions of the space coordinate $\boldsymbol{r} = (x, y, z)^T$. In short, the first equation is the equilibrium equation, the second equation is the constitutive law, and the third equation is the strain-displacement relation.

The Lamé parameters are related with the common Young modulus E and Poisson's ratio ν as follows:

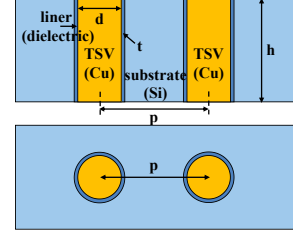$$\lambda = E\nu/(1+\nu)/(1-2\nu), \quad \mu = E/2/(1+\nu). \tag{2}$$

Fig. 2. Sectional view and top view of the TSV structure adopted in this paper [10]. It consists of a copper TSV body in the silicon substrate and a dielectric liner. d denotes the diameter of the TSV, h denotes the height, t denotes the thickness of the liner, and p denotes the pitch of adjacent TSVs.

B. FEM for Thermal Stress Simulation

The simulation of thermal stress belongs to the computational structural mechanics, which is generally and most suitably performed by FEM. Following the convention, our algorithm is based on FEM as well, so we briefly introduce the FEM formulation for thermal stress simulation in this section [18].

The first step of FEM is to convert the governing equation into the weak form, or the integral form. In the scenarios of integrated circuits, the gravity is ignored so the body force $\boldsymbol{f}$ is zero everywhere, and the boundary surfaces are generally assumed to be free or clamped so there are only Dirichlet boundary conditions. Therefore, the weak form of Eq. 1 is:

$$a(\boldsymbol{u}, \boldsymbol{v}) = L(\boldsymbol{v}), \tag{3}$$

where $\boldsymbol{u}$ is called the trial function, and $\boldsymbol{v}$ is called the test function. The bilinear form $a(u, v)$ is equal to:

$$a(\boldsymbol{u}, \boldsymbol{v}) = \int_\Omega (\lambda \text{tr}(\epsilon(\boldsymbol{u})) \cdot \mathbf{1} + 2\mu\epsilon(\boldsymbol{u})) : \epsilon(\boldsymbol{v}) \text{d}\boldsymbol{r}, \tag{4}$$

where the symbol : denotes the contraction of tensors. The linear form $L(\boldsymbol{v})$ is equal to:

$$L(\boldsymbol{v}) = \int_\Omega (\alpha(3\lambda + 2\mu) \cdot \Delta T \cdot \mathbf{1}) : \epsilon(\boldsymbol{v}) \text{d}\boldsymbol{r}. \tag{5}$$

After obtaining the weak form, the computational domain is discretized into a mesh, and $\boldsymbol{u}$, $\boldsymbol{v}$ are substituted by linear combinations of shape functions φ_i associated with each node i of the mesh with the coefficient α_i. By doing so, the weak form (Eq. 3) is assembled into a system of linear equations:

$$A\alpha = b, \tag{6}$$

where A is called the stiffness matrix and b is called the load vector. After solving Eq. 6, the final solution of the thermal stress problem expressed in displacement is:

$$\boldsymbol{u} = \sum_i \varphi_i \alpha_i, \tag{7}$$

and then the strain ϵ and the stress σ is calculated by Eq. 1.

If the system is large and the size of the mesh is small, the number of DOFs (the dimension of α) will be very large and the assembling and solving of Eq. 6 will be extremely time- and memory-consuming, which is the case for thermal stress simulation of TSV arrays in 2.5D/3D ICs. Advanced numerical algorithms have to be designed to decrease the number of DoFs to be solved and reduce the computational cost.

IV. THE MORE-STRESS ALGORITHM

In this section, we elaborate on the proposed *MORE-Stress* algorithm, through the case of TSV arrays.

A. Overall Framework

Our algorithm consists of a **one-shot local stage** and a **global stage**. For a certain set of material and geometry parameters, the local stage only needs to be performed once, which typically finishes within minutes. After finishing the one-shot local stage, the thermal stress of TSV arrays with arbitrary array sizes, under arbitrary thermal loads, and at arbitrary locations in a package
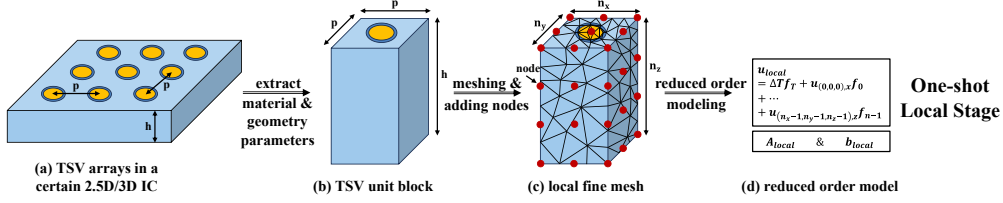
Fig. 3. Illustration of the one-shot local stage, which will be performed only once for a certain set of material and geometry parameters. (a)(b) The material and geometry parameters of the TSV structures to be studied in a certain 2.5D/3D IC are extracted and the unit TSV block is set up. (c) Lagrange interpolation points are added to the surface of the unit block for reduced order modeling, and a fine mesh is developed for the unit block to assemble the local problem. (d) Reduced order model of the unit TSV block is obtained and ready to be applied to the global stage.
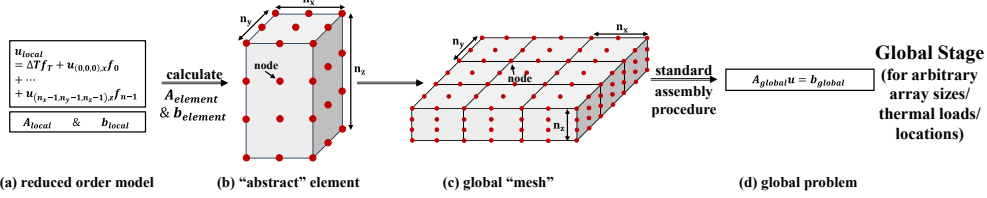

Fig. 4. Illustration of the global stage, which will performed every time given a new problem. Once the one-shot local stage is performed, thermal stress of TSV arrays with arbitrary array sizes, under arbitrary thermal loads, and at arbitrary locations in a package system can be efficiently calculated in the global stage. (a) The corresponding pre-calculated reduced order model is loaded. (b)–(d) A standard assembly procedure is applied to assemble the global problem.

system can be calculated with huge speedup and memory usage reduction during the global stage.

In the **one-shot local stage** (see Sec .IV-B), the TSV unit block is constructed based on the material and geometry parameters of the TSV structures to be studied, and a series of local problems associated with the unit block are solved to get the local basis functions for the global problem. In the **global stage** (see Sec. IV-C), the global stiffness matrix and global load vector are assembled using the local basis functions, the local stiffness matrix, and the local load vector calculated in the one-shot local stage. After assembling, the generated global problem is solved and then the detailed solution is calculated by a linear combination of the local basis functions.

In our algorithm, the reduction of computational cost is rooted in the reduction of the number of DoFs. The number of DoFs of the fine mesh for a unit block is reduced to the number of local basis functions. Because the TSV arrays are periodic and the TSV unit blocks are identical, the reduced order modeling can be performed just once in the local stage and applied to every unit block in the global stage to generate a model with much fewer DoFs, which is the philosophy of our algorithm. Obviously, the methodology can be generalized to any repeated, array-like fine structures in 2.5D/3D ICs, and the complexity of the geometry of the structure only affects the computational time of the one-shot local stage.

In the next subsections, we will detailedly explain the principles of the one-shot local stage and the global stage of our algorithm, and its combination with the sub-modeling technique.

B. The One-shot Local Stage

As discussed above, the complexity of the geometry of the local structure does not affect the overall performance of our algorithm. For brevity, we study a simplified TSV structure which consists of the copper TSV body and a dielectric liner as shown in Fig. 2 [10] in this work. The geometric parameters include the height h and the diameter d of the copper body, the thickness t of the liner, and the pitch p of adjacent TSVs.

A TSV array can be regarded as consisting of identical TSV unit blocks due to its periodicity, as shown by Fig. 3(a)(b). A TSV unit block consists of a single TSV structure in the middle of a silicon substrate cuboid, whose dimension is $p \times p \times h$. In this paper, we only study the 90° TSV arrays, while the unit blocks

of 120° TSV arrays can be chosen as a hexagonal prism with a TSV structure in the middle.

After defining the unit block, a group of equally spaced nodes is placed on the corners and surfaces of the unit block as shown in Fig. 3(c), serving as the Lagrange interpolation points for the displacement field of the unit block boundaries. The boundary displacement is approximated by the Lagrange interpolation functions defined on the nodes, which is the only source of error in our algorithm. Its convergence is guaranteed by the convergence of Lagrange interpolation.

The numbers of nodes along the three axes are denoted by (n_x, n_y, n_z). For the node (i, j, k) at coordinate (x_i, y_j, z_k), the Lagrange interpolation function $L_{3D}(\boldsymbol{r}; i, j, k)$ is written as:

$$L_{3D}(\boldsymbol{r}; i, j, k) = L_{1D}(x; i) \times L_{1D}(y; j) \times L_{1D}(z; k), \quad (8)$$

where $L_{1D}(x; i)$ is the 1D Lagrange interpolation function on node i along the x axis and the same is for $L_{1D}(y; j)$ and $L_{1D}(z; k)$. The boundary displacement of the unit block can then be approximated by the Lagrange interpolation functions L_{3D} and the displacement of the nodes $\boldsymbol{u}_{i,j,k}$:

$$
\begin{aligned}
u_{bc,x}(\boldsymbol{r}) &\approx \sum_{(i,j,k)} u_{(i,j,k),x} L_{3D}(\boldsymbol{r}; i, j, k), \\
u_{bc,y}(\boldsymbol{r}) &\approx \sum_{(i,j,k)} u_{(i,j,k),y} L_{3D}(\boldsymbol{r}; i, j, k), \\
u_{bc,z}(\boldsymbol{r}) &\approx \sum_{(i,j,k)} u_{(i,j,k),z} L_{3D}(\boldsymbol{r}; i, j, k),
\end{aligned}
\quad (9)
$$

where the x, y, and z components are listed separately for clarity.

After discretizing the unit block with a fine mesh as shown in Fig. 3(c), the Dirichlet boundary problem (where boundary displacement is assigned) of the unit block is assembled as:

$$A_{local}\alpha_{local} = \Delta T b_{local}, \quad (10)$$

as Eq. 6, or in a detailed form:

$$
\begin{pmatrix} A_{local,f,f} & A_{local,f,bc} \\ A_{local,bc,f} & A_{local,bc,bc} \end{pmatrix} \begin{pmatrix} \alpha_{local,f} \\ \alpha_{local,bc} \end{pmatrix} = \Delta T \begin{pmatrix} b_{local,f} \\ b_{local,bc} \end{pmatrix},
\quad (11)
$$

where the subscript f denotes the free DoFs and the subscript bc denotes the boundary DoFs. To assign the boundary displacement to the boundary DoFs, a procedure called "lifting" is performed. The rows of the stiffness matrix corresponding to the boundary DoFs are set to zeros except that the diagonal elements are set to ones, and the elements of the load vector corresponding to the

boundary DoFs are set to the preassigned displacement. Finally, we get the lifted system of linear equations that determine the free DoFs:

$$A_{local,f,f}\alpha_f = \Delta T b_{local,f} - A_{local,f,bc}u_{local,bc}. \qquad (12)$$

Substituting Eq. 9 into Eq. 12, we can get the approximated, or the order-reduced local problem:

$$A_{local,f,f}\alpha_{local,f} \approx \Delta T b_{local,f} - A_{local,f,bc}L\times$$

$$\begin{pmatrix} u_{(0,0,0),x} & u_{(0,0,0),y} & u_{(0,0,0),z} & \cdots & u_{(n_x-1,n_y-1,n_z-1),z} \end{pmatrix}^T, \qquad (13)$$

where L is the matrix containing the geometric information of Lagrange interpolation functions, which is solved automatically and does not need to be explicitly calculated.

Using the Lagrange interpolation, we successfully reduce the order of Eq. 10 from the number of boundary DoFs to the number of interpolation nodes. The displacement field of the unit block can then be written as:

$$\boldsymbol{u}_{local} \approx \Delta T \boldsymbol{f}_T + u_{(0,0,0),x}\boldsymbol{f}_0 + u_{(0,0,0),y}\boldsymbol{f}_1 + u_{(0,0,0),z}\boldsymbol{f}_2 + \cdots +$$

$$u_{(n_x-1,n_y-1,n_z-1),z}\boldsymbol{f}_{n-1}, \qquad (14)$$

where $\boldsymbol{f}_i$ is the solution of the local problem setting the thermal load ΔT to zero, the corresponding components of nodal displacement to one and all other components to zeros, while $\boldsymbol{f}_T$ is the solution setting the thermal load ΔT to one, and all the components of nodal displacement to zeros. The total number of $\boldsymbol{f}_i$s, n, is equal to:

$$n = \{n_x \times n_y \times n_z - (n_x-2) \times (n_y-2) \times (n_z-2)\} \times 3. \qquad (15)$$

All of the $\boldsymbol{f}_i$s, together with the $\boldsymbol{f}_T$, constitute the set of local basis functions as shown in Fig. 3(d). The total number of the local basis functions is much smaller than the number of DoFs in the fine mesh of the unit TSV block, which is the root of model order reduction.

Because the stiffness matrix is the same for all of the local problems ($A_{local,f,f}$), the time-consuming LU or Cholesky decomposition needs to be performed only once and the intermediate results can be reused for all of the local problems, which greatly reduces the computational time of the one-shot local stage. Meanwhile, the calculation of local problems can be easily parallelized on the task level, which further reduces the time cost.

C. The Global Stage

Once the one-shot local stage is finished, the thermal stress of TSV arrays (with the same material and geometric parameters) with arbitrary array sizes and under arbitrary thermal loads can be quickly calculated in the global stage.

In the global stage, the unit TSV block can be regarded as an abstract "element" with its own nodal displacement components $u_{(0,0,0),x}, \cdots, u_{(n_x-1,n_y-1,n_z-1),z}$ being the element DoFs, while the TSV array to be simulated can be regarded as an abstract "mesh" made up by abstract "elements" with the entire nodal displacement components being the global DoFs, as shown in Fig. 4(b)(c). In this perspective, the global problem has no difference from a common finite element problem. The global stiffness matrix A_{global} and the global load vector b_{global} can be easily assembled through the standard assembly procedure [18].

The remaining work is to calculate the element stiffness matrix $A_{element}$ and element load vector $b_{element}$ which are requested by the standard assembly procedure. Notice that $\boldsymbol{f}_i$, the local basis function corresponding to the ith element DoF, can be expressed as the linear combination of the local shape functions $\boldsymbol{\varphi}_{local}$ as Eq. 7:

$$\boldsymbol{f}_i = \sum_p \boldsymbol{\varphi}_{local,p}\alpha_p. \qquad (16)$$

Considering Eq. 3 and 4, the (i,j)th entry of $A_{element}$ is:

$$A_{element,i,j} = a(\boldsymbol{f}_i, \boldsymbol{f}_j)$$
$$= \sum_p \sum_q \alpha_p \times a(\boldsymbol{\varphi}_{local,p}, \boldsymbol{\varphi}_{local,q}) \times \alpha_q = \alpha^T A_{local}\alpha, \qquad (17)$$

where $\alpha = \begin{pmatrix} \alpha_0 & \alpha_1 & \cdots & \alpha_n \end{pmatrix}^T$. Considering Eq. 3 and 5, the ith entry of $b_{element}$ is calculated by:

$$b_{element,i} = L(\boldsymbol{f}_i)$$
$$= \sum_p \alpha_p \times L(\boldsymbol{\varphi}_{local,i}) = \alpha^T \Delta T b_{local}. \qquad (18)$$

$A_{element}$ is an $n \times n$ dense array, and $b_{element}$ is an n-dimensional vector, where n is the number of element DoFs calculated by Eq. 15.

After getting the element stiffness matrix $A_{element}$ and the element load vector $b_{element}$, a standard finite element assembly procedure can be performed to get the global counterparts A_{global} and b_{global}. The detailed process is omitted here for brevity. A_{global} is a sparse matrix because only nodes in the same elements (unit blocks) or shared by adjacent elements (unit blocks) can contribute to its entries.

With A_{global} and b_{global} prepared, the global problem is to be solved:

$$A_{global}u = b_{global}, \qquad (19)$$

as shown in Fig. 4(d), where u is the vector consisting of the nodal displacement components. Eq. 19 is better solved by iterative methods such as GMRES for a shorter computational time because we do not need to solve the same equation repeatedly in the global stage.

Finally, the displacement field $\boldsymbol{u}(\boldsymbol{r})$ of a certain unit block can be calculated using the corresponding entries of u and the local basis functions $\boldsymbol{f}_0, \boldsymbol{f}_1, \cdots, \boldsymbol{f}_{n-1}$ together with $\boldsymbol{f}_T$ following Eq. 15. Further the strain $\epsilon(\boldsymbol{r})$ and stress $\sigma(\boldsymbol{r})$ can be calculated by derivation of the displacement routinely.

D. Combination with Sub-modeling

Sub-modeling is a common technique in the engineering practice of structural mechanics [19] and is widely applied in commercial software like ANSYS. If a small part of a large system is of interest, a coarse mesh is first developed for the whole system and a coarse solution is obtained. Then the part of interest is cut out to form a sub-model. After that, a fine mesh is solely developed for the sub-model, and the coarse solution is applied to its boundaries as boundary conditions. This procedure avoids discretizing the whole system with the fine mesh and is proved to be accurate if the boundary of the sub-model is far away enough from the part of interest in engineering.

Our algorithm is naturally compatible with sub-modeling. The displacement values obtained from the coarse solutions can be directly assigned to the boundary nodes of the TSV arrays through the "lifting" procedure as described by Eq. 10, 11 and 12. Moreover, to satisfy the condition that the boundaries of the sub-model should be far away enough from the part of interest, arbitrary "dummy" unit blocks can be added to the periphery of the TSV array. A "dummy" unit block has the same dimension and nodal distribution as a TSV unit block, but it is a pure silicon cuboid without a TSV structure in the middle. An extra local stage has to be performed for the "dummy" unit block,
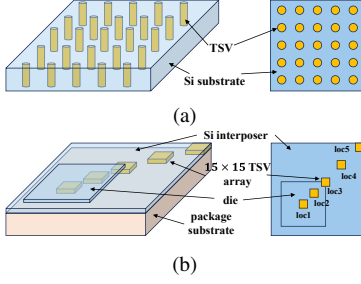
Fig. 5. (a) The first scenario. TSV arrays with array sizes ranging from 10×10 to 50×50 are studied. Except for efficiency and accuracy, this scenario is designed to test the scalability and convergence of our algorithm. (b) The second scenario is a 15×15 TSV array embedded at five different locations in a chiplet. This scenario is designed to test the combination of our algorithm with the sub-modeling technique.

and the corresponding $A_{element,dummy}$ and $b_{element,dummy}$ are calculated in the same way as Eq. 17 and 18. The standard assembly procedure can handle hybrid elements without difficulty [18].

The compatibility with sub-modeling implies that our algorithm can be easily integrated with other stress simulators, commercial or open-sourced, and can be treated as a module specializing in the local fine structures in 2.5D/3D ICs in a comprehensive simulation flow.

V. EXPERIMENTAL SETUP AND RESULTS

A. Implementation

Our algorithm is implemented based on an open-sourced FEM framework, FeniCSx [20]–[24], in Python. Besides, we use PETSc [25] as the linear algebra backend and use Gmsh [26] to mesh the unit TSV block in the local stage. The algorithm is designed to support parallelization, both for the one-shot local stage and the global stage. The experiments are conducted on a Linux server with an Intel Xeon Silver 4210R 2.40GHz processor (20 logical CPUs). To balance the threading overhead and the gain of parallelization, we set the number of threads to 16 for the one-shot local stage and the number of threads to 8 for the global stage hereafter.

B. Experimental Setup

Two representative scenarios are studied to test the performance of our algorithm, as shown in Fig. 5. The first scenario is a group of standalone TSV arrays with array sizes ranging from 10×10 to 50×50 and with their top and bottom surfaces clamped [11], [12]. This scenario is designed to test the convergence and scalability of our algorithm. The height h and diameter d of the copper via is set to be 50 μm and 5 μm, respectively, and the thickness of the liner t is set to be 500 nm, as shown in Fig. 2 [2], [11]. Two pitches p are tested in this scenario, which are 15 μm and 10 μm, respectively [11]. The thermal load ΔT is set to be -250 °C (annealing/reflow 275 °C $\rightarrow$ room temperature 25 °C) to represent a fabrication process [3], [9].

The second scenario is a 15×15 TSV array embedded at five different locations in a chiplet, as shown in Fig .5(b), which is designed to test the integration of our algorithm with the sub-modeling technique [2], [3]. The stress of the TSV array couples with the global warpage stress of the chiplet in this scenario [27]. The geometric parameters and the thermal load are the same as those in the first scenario. The pitch is set to be 15 μm or 10 μm, as well. The chiplet model consists of a composite substrate, a silicon interposer, and a silicon die. The TSV array is in the interposer. We first develop and solve a coarse model of the chiplet in ANSYS and then extract the sub-model and the displacement on the boundaries of the sub-model from the

TABLE I
SUMMARIZED COMPUTATIONAL TIME, MEMORY USAGE, AND COMPUTATIONAL ERRORS OF ANSYS, LINEAR SUPERPOSITION, AND OUR ALGORITHM FOR THE FIRST SCENARIO (FIG. 5(A)). THE COMPUTATIONAL ERRORS DO NOT APPLY TO ANSYS BECAUSE ITS RESULTS ARE THE GROUND TRUTH. THE IMPROVEMENT IN COMPUTATIONAL TIME AND MEMORY USAGE OF OUR ALGORITHM OVER ANSYS AND THE IMPROVEMENT IN ACCURACY OF OUR ALGORITHM OVER THE LINEAR SUPERPOSITION METHOD ARE ALSO LISTED.

$p = 15$ μm						
array size		10×10	20×20	30×30	40×40	50×50
ANSYS	time	391 s	1296 s	2952 s	5392 s	8612 s
	memory	12.2 G	53.1 G	119.7 G	212.7 G	330.1 G
Linear superposition [3], [11]	time	2.4 s	3.9 s	9.6 s	13.4 s	20.5 s
	memory	0.24 G	0.47 G	0.81 G	1.23 G	1.72 G
	error	5.36%	6.57%	7.27%	7.69%	7.98%
Ours	time	2.5 s	4.4 s	9.5 s	13.1 s	17.1 s
	memory	0.31 G	0.67 G	1.23 G	1.85 G	2.91 G
	error	0.93%	0.87%	0.80 %	0.72 %	0.67 %
Improve. over ANSYS	time	156×	295×	311×	412×	504×
	memory	39×	79×	97×	115×	113×
Improve. over Linear superposition	accuracy	5.8×	7.6×	9.1×	10.7×	12.0×
$p = 10$ μm						
array size		10×10	20×20	30×30	40×40	50×50
ANSYS	time	352 s	1201 s	2840 s	5304 s	8508 s
	memory	12.6 G	48.0 G	108.2 G	196.8 G	312.9 G
Linear superposition [3], [11]	time	2.3 s	4.0 s	9.6 s	13.3 s	20.7 s
	memory	0.24 G	0.46 G	0.81 G	1.23 G	1.71 G
	error	7.40%	10.33%	12.37%	13.60%	14.43%
Ours	time	2.3 s	4.5 s	11.4 s	13.9 s	18.5 s
	memory	0.31 G	0.67 G	1.22 G	1.86 G	2.90 G
	error	0.98 %	0.92 %	0.85 %	0.79 %	0.74 %
Improve. over ANSYS	time	153×	267×	250×	382×	460×
	memory	41×	72×	87×	106×	108×
Improve. over Linear superposition	accuracy	7.6×	11.2×	14.6×	17.2×	19.5×

coarse solution. Two columns and two rows of "dummy" unit blocks are added to the edge of the TSV array to keep enough distance between the boundaries and the TSV array, as explained in Sec. IV-D.

Following the convention, we calculate the gridded von Mises stress on the cut plane crossing the half height of the TSV arrays for comparison [3], [11], [12]. The number of grid points in a single TSV unit block is set as 100×100 to capture full details of stress distribution. For each case, the **mean absolute error** (MAE) between the result provided by our algorithm and the ground truth is calculated and normalized by the maximum von Mises stress because the stress is in direct proportion to the thermal load. We record the runtime of the global stage as the computational time of our algorithm, because the local stage only needs to be performed once for the same material and geometry parameters, and its typical runtime is within minutes, which is very short for large-scale analysis. For memory usage, we record the maximum memory usage during computation.

All simulations are performed in the commercial software ANSYS on a cloud server with an Intel Xeon Platinum 8350C 2.60GHz processor (24 logical CPUs). Because the models studied have large numbers of DoFs, we set the solver type to "iterative" in ANSYS during simulations.

C. Experimental Results and Discussion

1) Efficiency & Accuracy.: The efficiency and accuracy of our algorithm in the two scenarios are detailedly studied in this section. We set the number of interpolation nodes on the surface of the unit block to be $(4, 4, 4)$ to balance efficiency and accuracy in the following experiments. For the first scenario, the one-shot local stage costs 301.6 seconds and 287.4 seconds for the $p = 15$

µm case and $p = 10$ µm case, respectively. The computational time and memory usage of ANSYS, the linear superposition method, our algorithm, and the computational errors (normalized MAEs) are summarized in Table I. Our algorithm realizes a great reduction in computation time and memory usage compared with ANSYS. The speedup of our algorithm ranges from 153–504×, and the reduction in memory usage ranges from 39–113×. The improvement increases with the size of the TSV array because there are some fixed costs like the I/O time, which implies that our algorithm can realize more than 500× reduction in simulation time and more than 100× reduction in memory usage if the code is carefully optimized. The errors of our algorithm are less than 1%, which is negligible considering the huge improvement in efficiency. Compared with the linear superposition method, our algorithm is as efficient (and even slightly more efficient when the task scale is large), with an order of magnitude smaller errors. The linear superposition method works badly for the $p = 10$ µm case because it overlooks the coupling between adjacent TSVs, while the accuracy of our algorithm is basically not affected. Moreover, the computational errors of our algorithm decrease with the size of the TSV array because the errors concentrate near the boundaries. On the contrary, the errors of the linear superposition method increase with the size of the TSV array because its errors are distributed over the whole computational domain. As the part of interest is the TSV array itself, this is another manifestation that our algorithm prevails over the linear superposition method in accuracy.

TABLE II
SUMMARIZED RESULTS FOR THE SECOND SCENARIO (FIG .5(B)).

		loc1	loc2	loc3	loc4	loc5
$p = 15$ µm						
location		loc1	loc2	loc3	loc4	loc5
ANSYS	time	1137 s	1078 s	1006 s	1020 s	1012 s
	memory	37.6 G	38.2 G	37.4 G	37.6 G	38.0 G
Linear superposition [3], [11]	time	3.4 s	3.3 s	3.4 s	3.5 s	3.5 s
	memory	0.39 G	0.40 G	0.40 G	0.39 G	0.40 G
	error	5.62%	5.60%	7.23%	5.70%	7.40%
Ours	time	3.5 s	3.5 s	3.6 s	3.5 s	3.7 s
	memory	0.46 G	0.47 G	0.46 G	0.46 G	0.46 G
	error	0.62%	0.62%	0.71%	0.71%	0.72%
Improve. over ANSYS	time	325×	308×	279×	291×	273×
	memory	82×	81 ×	81×	82×	83×
Improve. over Linear superposition	accuracy	9.1×	9.0×	10.2×	8.0×	10.3×
$p = 10$ µm						
location		loc1	loc2	loc3	loc4	loc5
ANSYS	time	1086 s	1042 s	964 s	980 s	963 s
	memory	36.4 G	37.2 G	37.0 G	36.8 G	37.1 G
Linear superposition [3], [11]	time	3.3 s	3.4 s	3.4 s	3.4 s	3.5 s
	memory	0.39 G	0.40 G	0.39 G	0.40 G	0.39 G
	error	6.62%	6.60%	8.23%	6.70%	8.79%
Ours	time	3.6 s	3.5 s	3.7 s	3.6 s	3.7 s
	memory	0.45 G	0.46 G	0.46 G	0.45 G	0.46 G
	error	0.67%	0.68%	0.75%	0.73%	0.78%
Improve. over ANSYS	time	302×	298×	261×	272×	260×
	memory	81×	81×	80×	82×	81×
Improve. over Linear superposition	accuracy	9.9×	9.7×	11.0×	9.2×	11.3×

For the second scenario, the computational time, memory usage, and computational errors (normalized MAEs) of ANSYS, our algorithm, and linear superposition method are summarized in Table II. The one-shot local stage has been performed when studying the first scenario, and it does not need to be performed again because the geometry does not change. When the TSV array is close to or at the locations where the background stress changes sharply, such as the corner of the chip (loc3)

and the corner of the interposer (loc5) as shown in Fig 5(b), the errors of the linear superposition method are large. On the contrary, the accuracy of our algorithm is almost unaffected by this phenomenon because it follows the standard sub-modeling procedure, and the local variations in the global background stress are captured by the displacement boundary conditions assigned to the boundary nodes.

TABLE III
CONVERGENCE OF OUR ALGORITHM. n REFERS TO THE NUMBER OF ELEMENT DoFs (EQ. 16). THE SIMULATION TIME OF ANSYS FOR THIS CASE IS 1296 S.

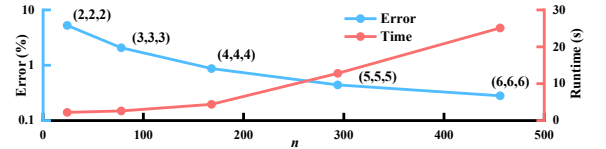| (n_x, n_y, n_z)	$(2,2,2)$	$(3,3,3)$	$(4,4,4)$	$(5,5,5)$	$(6,6,6)$
n	24	78	168	294	456
one-shot local stage runtime	147.0 s	204.1 s	301.6 s	431.8 s	603.2 s
global stage runtime	2.2 s	2.6 s	4.4 s	12.8 s	25.1 s
error	5.25%	2.07%	0.87%	0.44%	0.28%

Fig. 6. Computational errors and runtime (of the global stage) plotted against numbers of element DoFs, n. (n_x, n_y, n_z)s are labelled beside the corresponding points. The vertical axis of error is in log scale.

2) Convergence.: Next, we study the convergence of our algorithm on a 20×20 standalone TSV array with $p = 15$ µm. The simulation time of ANSYS for this case is 1296 s. We set the numbers of nodes used in the one-shot local stage from $(2, 2, 2)$ to $(6, 6, 6)$. The corresponding numbers of element DoFs n, computational errors (normalized MAEs), and computational time are summarized in Table III. Computational errors and runtime (of the global stage) are also plotted against n in Fig. 6. It is clear that as the number of element DoFs n increases, the computational errors decrease rapidly, implying that our algorithm enjoys fast convergence. This is because as the number of Lagrange interpolation nodes increases, the interpolation functions can better fit the surface displacement of the unit block. On the contrary, there is no concept of convergence for the linear superposition method because its accuracy is not influenced by any algorithm parameters.

VI. CONCLUSION

In this paper, we propose a novel strict numerical algorithm *MORE-Stress*, aiming at accelerating thermal stress simulation of large-scale arrays of fine structures in 2.5D/3D ICs and overcoming the multi-scalability numerical challenge. Our algorithm is based on FEM, and utilizes the periodicity of local fine structures to realize a great reduction in the number of DoFs through model order reduction. Meanwhile, combination with the sub-modeling technique makes our algorithm highly flexible. In this work, we focus on and apply our algorithm to the thermal stress simulation of TSV arrays in various scenarios. Extensive experimental results demonstrate that our algorithm can realize a 153–504× reduction in simulation time and a 39–115× reduction in memory usage compared with the commercial FEM software ANSYS, with negligible errors less than 1%. Our algorithm is as efficient as the linear superposition method, with an order of magnitude smaller errors and fast convergence. The proposed algorithm is general and adaptable to different types of fine structures in 2.5D/3D ICs, such as mircro bumps, pillars, direct bondings, etc., regardless of their geometries. In the future, we plan to apply it to more types of local fine structures and extend it to more complicated scenarios.

REFERENCES

[1] Y.-W. Chang, "Physical design challenges in modern heterogeneous integration," in *Proceedings of the 2024 International Symposium on Physical Design*, 2024, pp. 125–134.

[2] V. Moroz, X. Xu, A. Svizhenko, X.-W. Lin, S. Popov, H. Sheng, and K. Larsen, "3dic system-technology co-optimization with a focus on the interplay of thermal, power, timing, and stress effects," in *2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 2024, pp. 1–2.

[3] M. Jung, D. Z. Pan, and S. K. Lim, "Chip/package co-analysis of thermo-mechanical stress and reliability in tsv-based 3d ics," in *Proceedings of the 49th Annual Design Automation Conference*, 2012, pp. 317–326.

[4] B. Chen, G.-C. Lyu, H.-G. Wang, L. Zheng, Y.-K. Deng, and X.-P. Zhang, "A combined simulation and experimental study on cracking and delamination behavior at the cu/polyimide interface of rdls in chiplet package subjected to thermo-mechanical loads," in *2023 IEEE 73rd Electronic Components and Technology Conference (ECTC)*. IEEE, 2023, pp. 2046–2053.

[5] C.-L. Liang, Y.-S. Lin, C.-L. Kao, D. Tarng, S.-B. Wang, Y.-C. Hung, G.-T. Lin, and K.-L. Lin, "Electromigration reliability of advanced high-density fan-out packaging with fine-pitch 2-/2-μm l/s cu redistribution lines," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 10, no. 9, pp. 1438–1445, 2020.

[6] "Ansys," https://www.ansys.com/products/ansys-workbench.

[7] M. Pathak, J. Pak, D. Z. Pan, and S. K. Lim, "Electromigration modeling and full-chip reliability analysis for beol interconnect in tsv-based 3d ics," in *2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2011, pp. 555–562.

[8] J. Pak, S. K. Lim, and D. Z. Pan, "Electromigration-aware routing for 3d ics with stress-aware em modeling," in *Proceedings of the International Conference on Computer-Aided Design*, 2012, pp. 325–332.

[9] J. H. Lau, M. Li, L. Yang, M. Li, I. Xu, T. Chen, S. Chen, Q. X. Yong, J. P. Madhukumar, W. Kai *et al.*, "Warpage measurements and characterizations of fan-out wafer-level packaging with large chips and multiple redistributed layers," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 8, no. 10, pp. 1729–1737, 2018.

[10] Y. Li and D. Z. Pan, "An accurate semi-analytical framework for full-chip tsv-induced stress modeling," in *Proceedings of the 50th Annual Design Automation Conference*, 2013, pp. 1–8.

[11] M. Jung, J. Mitra, D. Z. Pan, and S. K. Lim, "Tsv stress-aware full-chip mechanical reliability analysis and optimization for 3d ic," *Communications of the ACM*, vol. 57, no. 1, pp. 107–115, 2014.

[12] H. Zhou, H. Zhu, T. Cui, D. Z. Pan, D. Zhou, and X. Zeng, "Thermal stress and reliability analysis of tsv-based 3-d ics with a novel adaptive strategy finite element method," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 7, pp. 1312–1325, 2018.

[13] J. H. Lau, "Recent advances and trends in multiple system and heterogeneous integration with tsv-less interposers," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 12, no. 8, pp. 1271–1281, 2022.

[14] J. H. Lau, "Recent advances and trends in multiple system and heterogeneous integration with tsv interposers," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 13, no. 1, pp. 3–25, 2023.

[15] T. Zhu, *et al.*, "MORE-Stress." https://github.com/txzhu2050/MORE-Stress.

[16] H. Wang, T. Xiao, D. Huang, L. Zhang, C. Zhang, H. Tang, and Y. Yuan, "Runtime stress estimation for three-dimensional ic reliability management using artificial neural network," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 6, pp. 1–29, 2019.

[17] R. B. Hetnarski, M. R. Eslami, and G. Gladwell, *Thermal stresses: advanced theory and applications*. Springer, 2009, vol. 41.

[18] M. G. Larson and F. Bengzon, *The finite element method: theory, implementation, and applications*. Springer Science & Business Media, 2013, vol. 10.

[19] J.-H. Wong, N. Wu, W.-H. Lai, D.-L. Chen, T.-Y. Chen, C.-H. Chen, Y.-H. Wu, Y.-s. Chang, C.-L. Kao, D. Tarng *et al.*, "Warpage and rdl stress analysis in large fan-out package with multi-chiplet integration," in *2022 IEEE 72nd Electronic Components and Technology Conference (ECTC)*. IEEE, 2022, pp. 1074–1079.

[20] I. A. Barrata, J. P. Dean, J. S. Dokken, M. Habera, J. HALE, C. Richardson, M. E. Rognes, M. W. Scroggs, N. Sime, and G. N. Wells, "Dolfinx: The next generation fenics problem solving environment," 2023.

[21] I. A. Baratta, J. P. Dean, J. S. Dokken, M. Habera, J. S. Hale, C. N. Richardson, M. E. Rognes, M. W. Scroggs, N. Sime, and G. N. Wells, "DOLFINx: the next generation FEniCS problem solving environment," preprint, 2023.

[22] M. W. Scroggs, J. S. Dokken, C. N. Richardson, and G. N. Wells, "Construction of arbitrary order finite element degree-of-freedom maps on polygonal and polyhedral cell meshes," *ACM Transactions on Mathematical Software*, vol. 48, no. 2, pp. 18:1–18:23, 2022.

[23] M. W. Scroggs, I. A. Baratta, C. N. Richardson, and G. N. Wells, "Basix: a runtime finite element basis evaluation library," *Journal of Open Source Software*, vol. 7, no. 73, p. 3982, 2022.

[24] M. S. Alnaes, A. Logg, K. B. Ølgaard, M. E. Rognes, and G. N. Wells, "Unified form language: A domain-specific language for weak formulations of partial differential equations," *ACM Transactions on Mathematical Software*, vol. 40, 2014.

[25] S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. Gropp *et al.*, "Petsc users manual," 2019.

[26] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities," *International journal for numerical methods in engineering*, vol. 79, no. 11, pp. 1309–1331, 2009.

[27] M.-Y. Tsai and Y.-W. Wang, "A theoretical solution for thermal warpage of flip-chip packages," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 10, no. 1, pp. 72–78, 2019.