ELSEVIER

Engineering Analysis with Boundary Elements 28 (2004) 1035–1044

ENGINEERING ANALYSIS *with* BOUNDARY ELEMENTS

www.elsevier.com/locate/enganabound

Preconditioned multi-zone boundary element analysis for fast 3D electric simulation

Wenjian Yu*, Zeyi Wang, Xianlong Hong

EDA Laboratory, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

Received 25 November 2003; revised 19 February 2004; accepted 23 February 2004

Available online 28 March 2004

Abstract

For fast 3D electric simulation, the multi-zone collocation boundary element analysis (BEA) with iterative solver like GMRES algorithm is required. In this paper, we present a scheme of equation assembly with ordering unknowns and collocation points, and a matrix storing structure. A group of easily computed preconditioners based on the mesh neighbor method are then proposed to remarkably quicken the convergence of GMRES iteration, and demonstrate at least 30% time reduction than using the diagonal preconditioner. Compared with the equation assembly in Merkel et al. [Engng Anal Bound Elem 22 (1998) 183], where two unknowns on same interfacial node are arranged subsequently, and two storing schemes in Li et al. and Araujo et al. [Sys Engng Electron 21 (1999) 10; J Chin Inst Engrs 23 (2000) 269], the proposed method generates fewest non-zero blocks and facilitates the matrix–vector multiplication remarkably. Numerical experiments verify the analysis and show a fast iterative multi-zone BEA simulator for actual very large-scale integration interconnects with a large amount of zones.

Keywords: Boundary element; Multi-zone; Equation assembly; Iterative solution; Precondition; Electric simulation

1. Introduction

The boundary element method (BEM), based on the direct boundary integral equation (BIE) [1], has been widely employed to solve the Laplace or Poisson equation in scientific and engineering problems, because of its ability to reduce the dimensionality of problem. For a problem defined over a region which is only piecewise homogeneous, the multi-zone boundary element analysis (BEA) is usually used. The multi-zone BEA refers to the techniques of dividing the whole region into a series of homogeneous subregions (zones) and then discretizing boundary of each zone with elements (panels). Using the conditions of continuity and compatibility at nodes of the elements shared by two or more zones, the discretized BIEs of all zones can be coupled up, producing an overall system matrix that has a blocked and sparse character. The multi-zone collocation BEA strategy was briefly introduced in Ref. [1], and then developed by Kane and other authors [2–7].

A direct matrix block triangular factorization process, which exploits the substantial block sparsity present in multi-zone BEA, was described for equation solution in Refs. [2,3]. Kane and his colleagues developed an arbitrary condensation multi-zone approach, which is very suitable to many iterative-type problems with localized effects, such as the shape optimization analysis [4]. These works showed that the multi-zone collocation techniques could significantly extend the range of model shapes and substantively improve the computational efficiency in the overall analysis process. And for some problems, the multi-zone solution has shown its superiority to the solution of an equivalent single-zone problem [3–6].

Besides the direct solutions mentioned above, the Krylov iterative solvers were also investigated for multi-zone BEA, with different preconditioning techniques [3,6,7]. Since the 2D problems of elasticity area or stress analysis were mainly dealt with, and a relative error norm of 10^{-6} was used for high accuracy, the iterative solvers did not show dominating superiority to the direct solution in Refs. [3,6,7].

Fast 3D electric simulation is increasingly important in the area of very large-scale integration (VLSI) interconnects

* Corresponding author. Tel.: +86-10-6277-3440; fax: +86-10-6278-1489.

E-mail address: yu-wj@tsinghua.edu.cn (W. Yu).

0955-7997/$ - see front matter © 2004 Published by Elsevier Ltd.
doi:10.1016/j.enganabound.2004.02.006

and micro-electro-mechanical system (MEMS), under the current deep submicron technology. For example, the parasitic capacitance and resistance of the interconnect metal lines need to be computed with high speed and accuracy for verification of circuit performance, such as time delay, power consumption, etc. [8–11]. In the electric simulation, BEM is employed to solve the electrostatic Laplace equation and the constant discontinuous element is often used for required accuracy [9–11]. This avoids the usual difficulties associated with corners and multi-zone formulation. And in the existing solutions, the Krylov iterative solvers were used with a relative error norm of 10^{-2} or 10^{-3} as convergence criterion. The iterative solvers have shown fast convergence rate, and become indispensable in the 3D electric simulation.

For large-scale BEA with iterative solver, a quasi-multiple medium (QMM) technology was proposed recently, which was analyzed to have a nearly linear computational complexity for a simplified single-medium problem [12]. The QMM technology fully exploits the matrix sparsity present in the multi-zone BEA by decomposing the homogeneous dielectric into some fictitious subregions, and as the result, improves the computational efficiency for the original problem. The QMM has greatly reduced the CPU time and memory usage in the 3D electric analysis (10× speed-up for large cases) [11]. However, the QMM strategy leads to a large amount of zones (sometimes more than 100), and this causes some difficulty to the organization and solution of the discretized BIEs.

Since the orders of unknowns and collocation points in the discretized BlEs correspond to the arrangement of columns and rows in the system matrix, respectively, they determine the distribution of non-zero matrix entries and are the main concern in organizing the multi-zone BEA equations. While using a direct equation solver, the equation assembly is important to the computational time and memory due to the 'fill-in' phenomenon, and an ordering was proposed to make non-zero matrix blocks as close as possible to the main diagonal, minimizing the fill-in [2–5]. This order is formed by first simply listing all permutations of two zones, for a three-zone problem which is:

$$11\ 12\ 13\ 21^{*}\ 22\ 23\ 31^{*}\ 32^{*}\ 33. \tag{1}$$

For permutations where two digits are the same, unknowns merely in one zone are related; otherwise, two kind of unknowns on the interface between zone i and zone j are related with ij and ji^{*}, respectively [3]. Different from the multi-zone BEA with direct solution, the impact of equation assembly on computational efficiency of iterative solution is rarely investigated, except for Refs. [13–15]. In Ref. [13], it was observed that to order the unknowns and collocation points consistently can bring fast convergence rate for the iterative solver with the diagonal (Jacobi) or block diagonal preconditioner. And for multi-zone problem, subsequent numbers were assigned to the two kinds of unknowns on a same interfacial node. Li et al. [14] used almost the same disciplines as Merkel et al. [13] in 3D capacitance computation of VLSI interconnects. In Ref. [15], the equation assembly was not actually discussed; only a sparse storage of 'work vector' was used for the system matrix of multi-zone BEA, and its emphasis was to demonstrate the advantage of iterative solver with a Lanczos process to the direct solver using the condensation technique.

In this paper, the techniques related to equation organization of multi-zone BEA are firstly discussed according to its effect to computational performance of iterative solution, especially under the situation with many zones. It is found that the disordered system matrix, produced by an equation assembly without serious consideration, would bring difficulty in storing the matrix effectively, and cause a lot of additional CPU time for locating non-zero entries in each matrix–vector multiplication.

Based on the permutation method of ordering unknowns in the direct multi-zone BEA [3], we present a scheme of equation assembly for the iterative multi-zone BEA, in which the unknowns and the collocation points are also arranged consistently and an efficient storing structure of length-varied 2D array is employed. This scheme produces a regular-structured system matrix with very fewer non-zero blocks, and can then greatly reduce the auxiliary manipulations in matrix–vector multiplication. Analysis and numerical results both show that our equation assembly is superior to other related approaches in Refs. [13–15], and is most suitable for the iterative BEA with a lot of zones. According to the structure of the sparse system matrix, a group of preconditioners based on the mesh neighbor (MN) method [17] are then proposed for the GMRES solver [16]. Computational results show these preconditioners have stable performance to reduce the iteration number, and outperform the Jacobi preconditioner by at least 30% reduction of the equation solution time. Finally our approaches make up of a fast solution scheme for the very sparse linear equation system in the multi-zone BEA with many zones, and have served as an important role in the 3D electric simulation using the QMM technology.

The rest of this paper is organized as follows. Section 2 outlines the multi-zone BEA for 3D electric problems. In Section 3, the efficient organization of the BEM equations is presented, compared with other existing approaches. The preconditioners for the GMRES solver are proposed in Section 4. The numerical results of 3D electric simulation are given in Section 5, demonstrating the advantage of the proposed approaches. Finally, conclusions are drawn in Section 6.

2. Multi-zone BEA for the 3D electric simulation

Fig. 1 shows a typical 3D structure of multi-layered interconnect conductors over a grounded substrate. There are five dielectric layers; the embedded conductors

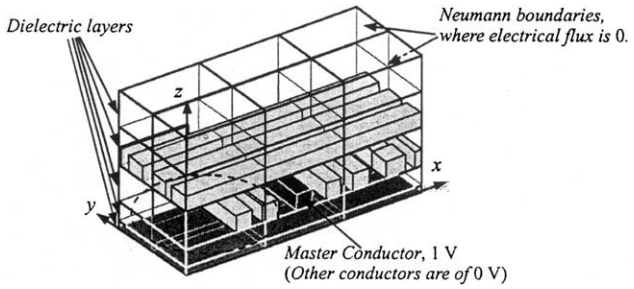

Fig. 1. A 3D multi-layered structure with interconnect conductors embedded in dielectrics, where the coupling capacitances between the master and others need to be computed.

touch dielectric interface with bottom. The simulated region is bounded by Neumann boundaries, and the electric potential on conductor surface is known, determined by the bias voltages. To calculate the coupling capacitances between the master conductor and others, the charge on each conductor must be obtained by solving the electrostatic field. With the QMM technology, each layer is cut into 3×2 subregions, and there are totally 30 subregions (see Fig. 1).

The whole simulated region is composed of piecewise homogeneous subregions, which is generally bounded by the Neumann boundary, conductor surface (Dirichlet boundary) and subregion interface. Within each subregion, the electrical potential u fulfills the Laplace equation. With the direct BEM, the Laplace equation is converted into the following collocation formula of BIE [1,3]:

$$c(\xi)u(\xi) + \int_{\partial\Omega_i} q^*(\xi,x)u(x)\mathrm{d}\Gamma(x) = \int_{\partial\Omega_i} u^*(\xi,x)q(x)\mathrm{d}\Gamma(x), \tag{2}$$

where $\partial\Omega_i$ stands for the boundary of the ith subregion, and $q = \partial u/\partial\mathbf{n}$ is the normal electrical intensity. $c(\xi)$ is a constant dependent on boundary geometry near to the collocation point ξ. $u^*(\xi,x)$ stands for the fundamental solution to Laplace equation, and q^* is the derivative of u^* along the outward normal direction of boundary $\partial\Omega_i$. Fig. 2 shows a subregion of the structure in Fig. 1, whose boundary is partitioned into quadrilateral elements.

Employing constant quadrilateral elements (of trapezoid shape generally), and evaluating the direct BIE (2) at

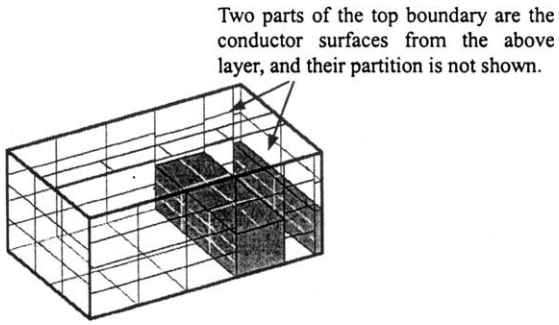

Fig. 2. A subregion on the third layer of the structure in Fig. 1 (indicated by a dashed circle), whose boundary is partitioned into quadrilateral elements.

collocation points, one for an element, the discretized BIEs for the ith subregion are got:

$$c_k u_k + \sum_{j=1}^{N_i}\left(\int_{\Gamma_j} q_k^*(x)\mathrm{d}\Gamma(x)\right)u_j = \sum_{j=1}^{N_i}\left(\int_{\Gamma_j} u_k^*(x)\mathrm{d}\Gamma(x)\right)q_j, \quad (k = 1,\ldots,N_i) \tag{3}$$

where N_i is the number of elements on boundary of subregion i, and Γ_j is the jth element. The evaluation of integrals in Eq. (3) is a time-consuming part of boundary element algorithms, in particular for 3D analysis. Classified as three types (singular, near singular and non-singular) and handled with different methods, respectively, these boundary integrals can be calculated with both high-speed and high-accuracy [11].

Since constant discontinuous elements are employed, each node on interface only belongs to two zones. With the compatibility and equilibrium conditions at the interfacial node, the series of discretized BIEs of all subregions can be coupled up, resulting in an overall linear equation system. Substituting the Dirichlet and Neumann conditions, we can reorder the global linear system to become:

$$\mathbf{Ax} = \mathbf{f}, \tag{4}$$

where $\mathbf{x}$ is a vector comprising all discretized unknowns of u and q. The coefficient matrix $\mathbf{A}$ is a large unsymmetric one (usually has degree of more than 1000). A GMRES algorithm [16] with preconditioning is usually used to solve the linear system [9–11].

3. Organizing the equations in iterative multi-zone BEA

An efficient scheme of equation assembly is firstly presented. Then, comparison and discussion are given to show its advantage.

3.1. Efficient scheme for organizing and storing the BEM equations

Inheriting the above formulas for electric problem, we notate all discretized unknowns as the following three groups:

(1) Unknowns on the boundary except zone interface, denoted by v_{ii} $(i = 1,\ldots,M)$.
(2) u on the interface of zone i and zone j, denoted by u_{ij} $(i,j = 1,\ldots,M$, and $i < j)$.
(3) q on the interface of zone i and zone j, denoted by q_{ji} $(i,j = 1,\ldots,M$, and $i < j)$.

where M is the number of zones. These groups of unknowns correspond to all permutations of two zones,

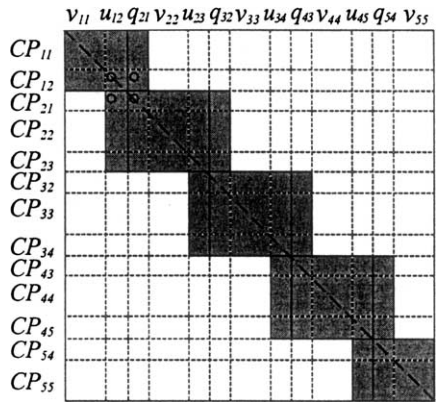

Fig. 3. Matrix population for a problem including five zones.

shown as Eq. (1) for 3-zone problem. Moreover, the order of permutations determines the order of the unknowns groups.

For the ordering of matrix rows, we first arrange collocation points in the first zone, then the second zone, until the Mth zone. In each zone the order of collocation points is consistent with that of unknowns, i.e. the collocation point and unknown of same no. is on the same boundary element. Therefore, the main diagonal entries of the matrix are the singular boundary integral (taken on the boundary element containing the collocation point) or the constant c_k in Eq. (3). Since they make the matrix diagonal entries have a large absolute value, the resulting equation (4) exhibits a good convergence behavior when using the simple Jacobi preconditioner [13,14].

With above arrangement, the matrix population for the five-dielectric-zone problem in Fig. 1 (without QMM cutting) is shown in Fig. 3. In Fig 3, the gray blocks stand for non-zero entries. And the types of unknowns are signed beside the corresponding columns. On the left of matrix rows, the types of collocation points are signed ('CP' means collocation point), which has the consistent subscript to that of unknowns. It should be pointed out, since some types of unknowns like u_{13} and $u_{24,}$ etc. do not exist due to the inexistent interfaces, they are not signed in the Figure.

For a problem involving 12 zones (2×2 QMM cutting performed on a three-layered structure), the matrix population is shown in Fig. 4.

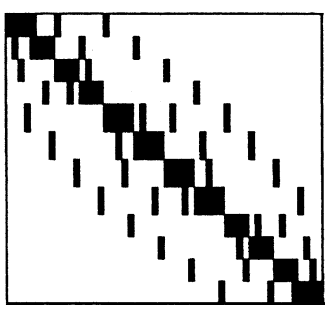

Fig. 4. Matrix population for a problem including $2 \times 2 \times 3$ zones.

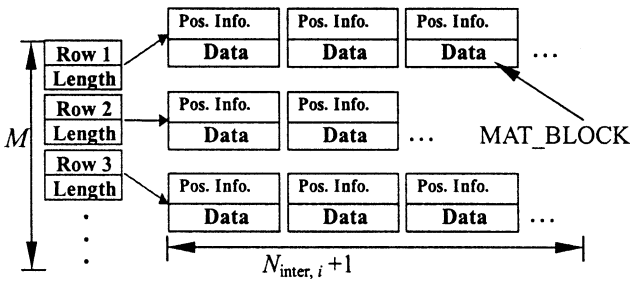

Fig. 5. Storing structure of the blocked sparse matrix.

From Figs. 3 and 4, it is found out that the non-zero coefficients are located the diagonal block, or some parallel strips. Only the non-zero matrix entries need to be stored, and we use a length-varied 2D array (see Fig. 5). This storing structure has M rows, and the cells in the ith row are one more than the interfaces related to zone i. Each cell is called MAT_BLOCK, corresponding to a non-zero matrix block.

In the MAT_BLOCK, the position information of a non-zero block and a 2D array to store its coefficients are included. This structure is also a variation of the block compressed row storage (BCRS), an efficient storage format for blocked sparse matrix [22].

The above arrangement of equations and storing structure are suitable for any problem with arbitrary complex topologic relationship of zones. And for a problem with M zones, the number of non-zero blocks (MAT_-BLOCK) produced by the above method is:

$$N_{\mathrm{nb}} = \sum_{i=1}^{M} (1 + N_{\mathrm{inter},i}) = M + 2N_{\mathrm{inter}}, \tag{5}$$

where $N_{\mathrm{inter},i}$ means the number of interfaces in the zone i, and N_{inter} is the total number of interfaces.

At the same time, we have:

$$N_{\mathrm{inter}} = \frac{1}{2} \sum_{i=1}^{M} N_{\mathrm{inter},i}. \tag{6}$$

For the capacitance problem with stratified layers and zones distributed regularly after QMM decomposition (like that in Fig. 1, and $M = m_x \cdot m_y \cdot m_z$), Eq. (5) can be transformed into:

$$\begin{aligned} N_{\mathrm{nb}} &= M + 2[m_x \cdot m_y \cdot (m_z - 1) + m_y \cdot m_z \cdot (m_x - 1) \\ &\quad + m_z \cdot m_x \cdot (m_y - 1)] \\ &= 7M - 2(m_x \cdot m_y + m_y \cdot m_z + m_z \cdot m_x), \end{aligned} \tag{7}$$

where $m_x m_y$ and m_z are the number of zones along x-, y-, z-axes, respectively.

3.2. Comparison and discussion

In Ref. [13], to deal with the multi-zone situation, subsequent numbers are assigned to the u and q unknowns

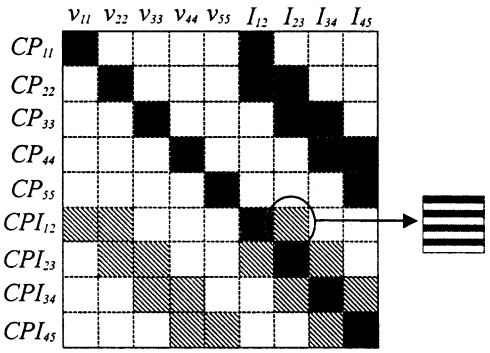

Fig. 6. Matrix population produced by an equation assembly obeying the rules in Ref. [13].

on a same interfacial node, in order to make all singular integrals adjacent to the main diagonal. Obeying this consideration and ordering the collocation points consistently, the matrix population can be drawn as Fig. 6, for the five-dielectric problem from Figs. 1 and 3.

In Fig. 6, the types of unknowns and collocation points are signed beside the matrix columns and rows, respectively. Since the unknowns u, q of same interface element are arranged sequently, we use I_{ij} to denote all unknowns on interface between zone i and j. To keep consistent order, two collocation points at an interfacial node are also performed one after another. Therefore, some matrix block in Fig. 6 (of downward diagonal pattern) is not fully non-zero, but interlined (with non-zero row and zero row one after another).

The number of non-zero matrix blocks under this arrangement is:

$$\begin{aligned} N'_{\mathrm{nb}} &= \sum_{i=1}^{M}(1+N_{\mathrm{inter},i}) + \sum_{1\le i<j\le M, I_{ij}\mathrm{exist}}(2+N_{\mathrm{inter},i} \\ &\quad + N_{\mathrm{inter},j}-1) \\ &= (M+2N_{\mathrm{inter}}) + N_{\mathrm{inter}} + \sum_{i=1}^{M}(N_{\mathrm{inter},i}\times N_{\mathrm{inter},i}) \\ &= M+3N_{\mathrm{inter}} + \sum_{i=1}^{M}(N_{\mathrm{inter},i})^2, \end{aligned} \tag{8}$$

where on the right hand of equal sign, the first item stands for non-zero blocks produced when the collocation points are not on interfaces, and the second item is associated with the collocation points on interfaces (see Fig. 6).

Comparing Figs. 3 and 6, we find the former includes much fewer non-zero blocks than the latter. More comparisons of the number of non-zero blocks are listed in Table 1 (calculated with Eqs. (7) and (8)). For the problem with 12 zones corresponding to Fig. 4, the equation assembly of Ref. [13] generates 208 non-zero blocks, much larger than 52 by our method.

Because the unknowns on a zone interface is divided into two categories (of type u_{ij} and type q_{ji}), and they are arranged separately, all collocation points for a same zone can be arranged together to persist a consistent order with unknowns. This makes the regular-shaped matrix in our method have the fewest non-zero blocks, and is the main difference between the equation assembly in Section 3.1 and that in Refs. [13,14]. Furthermore, the existence of not fully non-zero blocks (see Fig. 6) makes the latter worse, producing a very disordered matrix population for problem with a large amount of zones. It also forbids the use of the efficient BCRS format.

Table 1
Comparison of the number of non-zero matrix blocks produced by the two kinds of equation assembly

	N_{nb}	N'_{nb}
The five-dielectric problem (5 zones)	13	31
The three-dielectric problem with 2 × 2 QMM cutting (12 zones)	52	208
The five-dielectric problem with 3 × 2 QMM cutting (30 zones)	148	682
The five-dielectric problem with 7 × 3 QMM cutting (105 zones)	593	3175

Ref. [14] adopts almost the same discipline as Ref. [13], except that the arrangement of elements or unknowns is according to the input of 3D objects (conductor and dielectric), whereas zones and interfaces, such that a bit more non-zero blocks are resulted in. In its implementation, six arrays with multiple dimensions are used to store the non-zero entries, which are named by *MtoM, MtoI_u, MtoI_q, ItoM, ItoI_u* and *ItoI_q*. Under this organization, a transformation is necessary to give the global position of a non-zero matrix entry from its local position. Therefore, the shifting between local arrays (whose quantity is very huge) and locating the non-zero entries bring much more auxiliary manipulations, in a matrix–vector multiplication.

Alternative storing scheme for the disordered sparse matrix is the compressed row storage (CRS), which is the most general format [22], and was adopted in Ref. [15]. Fig. 7 shows this format as a length-varied 2D array. The CRS format is not very efficient, needing an indirect addressing step for locating each matrix entry, in a matrix–vector product [22]. In contrast, the BCRS in Section 3.1 is modification of the CRS to exploit

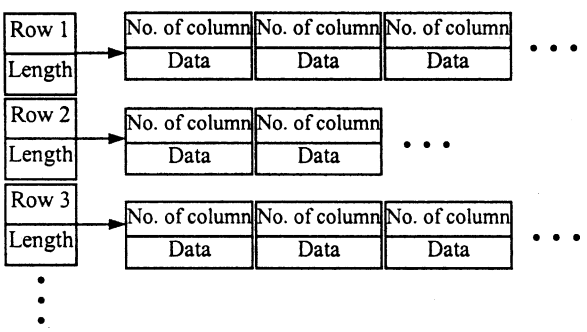

Fig. 7. A CRS format for storing the sparse matrix.

the block patterns, which saves memory for storing locations and reduces the time for indirect addressing [22], because the global position of a non-zero entry can be easily obtained by adding its local position to a fixed offset.

Additionally, to merge each diagonal blocks with its adjoined left or right non-zero blocks may reduce the N_{nb} further (see Fig. 3), in our equation assembly in Section 3.1. However, this merging is of less meaning and not generally available even by ordering the zone numbers (such as for the structure corresponding to Fig. 4). Thus, it is not considered in Table 1 and the following discussion.

4. Preconditioners for the GMRES Solution in multi-zone BEA

4.1. A brief overview

The GMRES algorithm is one of the most efficient iterative solutions for large-scale unsymmetric system of linear equations. In each step of iteration, its main manipulation is one product of matrix and vector. When dealing with sparse matrix, the advantage of GMRES over direct solution is more remarkable, since the zero entries can be omitted and the 'fill-in' need not be considered. For GMRES algorithm, the right preconditioning is always used to quicken the convergence rather than other preconditioning strategy, because it is found to be the most effective and provides the residual of the original equation directly for convergence criterion [3]. With the right preconditioning, solving Eq. (4) is equivalent to using GMRES to solve $\mathbf{APy} = \mathbf{f}$ for the unknown vector $\mathbf{y}$, from which the original unknown $\mathbf{x}$ is computed by $\mathbf{x} = \mathbf{Py}$. For iteration with k steps, the preconditioned GMRES adds $k + 2$ multiplications of $\mathbf{P}$ and vector, besides constructing the matrix $\mathbf{P}$.

The convergence of GMRES is governed by eigenvalues or pseudo-eigenvalues [13,18]. Therefore, a good preconditioner for GMRES should at first improve the distribution of eigenvalues. On the other hand, the expense on constructing $\mathbf{P}$ and multiplying it with a vector should be controlled not to overwhelm the benefit from the improvement of convergence rate. So, $\mathbf{P}$ has to be a sparse matrix explicitly or implicitly.

Some literatures are concerned with the preconditioning for boundary element analysis. In Refs. [17,19,20], a lot of preconditioning methods for the dense linear system derived from BEA were discussed, and classified into three types: operator splitting preconditioner (OSP), least squares approximate inverse (LSAI) and diagonal block approximate inverse (DBAI). Among them, OSP methods can cluster the eigenvalues of the original matrix, but maybe do not approximate the inverse well; the LSAI's effectiveness lies in a knowledge of a dominate sparsity pattern of the true inverse; DBAI methods have not a clear theoretic support, but many experiments show they often employ the advantages both of LSAI and OSP [20]. The diagonal (Jacobi), blocked diagonal preconditioners and the preconditioner based on incomplete LU decomposition (ILUD) were used in Refs. [3,6,7,13] for the multi-zone BEA, to perform comparison with the direct solvers.

Compared with the multi-zone problems discussed in Refs. [3,6,7,13], the GMRES iterative steps are fewer in the 3D electric problems. Furthermore, the coefficient matrix involved is very sparse because of using the QMM technology. Thus, the simplicity of preconditioner is very important for our concern. The existing preconditioner (except the Jacobi) for multi-zone analysis are relatively expensive, involving LU decomposition of BEM equations for each zone, and are not applicable to the electric problems. In the existing literatures, only the Jacobi perconditioner has high overall efficiency for the electric multi-zone BEA [11]. Below, we propose a group of easily computed preconditioners based on the mesh neighbor (MN) method [17], which is superior to the Jacobi perconditioner.

4.2. Mesh neighbor method [17]

Each row of preconditioner $\mathbf{P}$ is generated separately. Let the ith column of $\mathbf{P}^{\mathrm{T}}$ be denoted by $\mathbf{p}_i$, i.e. $\mathbf{P}^{\mathrm{T}} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N)$. Ideally, we would like to have:

$$\mathbf{PA} = \mathbf{I} \Leftrightarrow \mathbf{A}^{\mathrm{T}}\mathbf{p}_i = \mathbf{e}_i, \tag{9}$$

where $\mathbf{e}_i$ is the ith column of the identity matrix. Note that each column (or row of matrix $\mathbf{A}$ corresponds to a discretized unknown (or collocation point), and further to a boundary element. Therefore, we use the no. of row or column as the index of its corresponding collocation point, unknown and element. With some strategy we may determine a small set L of indices drawn from $\{1, 2, \ldots, N\}$, which denotes the unknowns having intensive impact on the current unknown i. Then, Eq. (9) can be reduced to

$$\bar{\mathbf{A}}^{\mathrm{T}}\bar{\mathbf{p}}_i = \bar{\mathbf{e}}_i, \tag{10}$$

where the bars over the variables indicate that all the rows and columns except those in L are deleted. After solving Eq. (10), we expand $\bar{\mathbf{p}}_i$ back to the corresponding entries in the ith row of $\mathbf{P}$. Repeating the above procedure for all rows; we get the whole sparse matrix $\mathbf{P}$. In each row of $\mathbf{P}$, there are only several non-zero entries, whose number is the same as the order of the corresponding set L.

For example, if the set L has three indices, and the first one is the current row i (i.e. $I_1 = i$), then Eq. (10) becomes:

$$\begin{pmatrix} a_{l_1 l_1} & a_{l_2 l_1} & a_{l_3 l_1} \\ a_{l_1 l_2} & a_{l_2 l_2} & a_{l_3 l_2} \\ a_{l_1 l_3} & a_{l_2 l_3} & a_{l_3 l_3} \end{pmatrix} \begin{pmatrix} p_{i l_1} \\ p_{i l_2} \\ p_{i l_3} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \tag{11}$$

where a_{ij} and p_{ij} means the entry on the position of ith row and jth column in **A** and **P**, respectively.

4.3. Extended Jacobi and MN(n) preconditioner

Two strategies for selecting the set L are proposed to construct our preconditioners. The first one is called extended Jacobi (EJ) preconditioner. Actually, the Jacobi preconditioner using the $L = \{i\}$ for each row can also be attributed to the MN method. However, in the Jacobi preconditioner, not all singular integrals in multi-zone BEA is considered. For a node ξ_l on the interface between zone i and j, the two unknowns on it are denoted by $u_{ij}(\xi_l)$ and $q_{ji}(\xi_l)$. Note that the collocation point on ξ_l presents twice in the matrix **A**, for the discrete BIE of zone i and zone j, respectively. Therefore, there are four singular integrals (or the constant c_k) related to the element containing ξ_l, existing in the matrix **A** (for example, the circles in Fig. 5 denote the four occurrences). Two of them are not on the main diagonal. The EJ preconditioner originates from the above observation. In it, L contains two indices of the row itself and the other occurrence of the collocation point, for a row corresponding to an interfacial node; otherwise, L only contains the index of the current row. The EJ preconditioner is a small extension to the Jacobi (for some rows, a 2×2 equation is solved), but it accelerates the convergence remarkably.

In the EJ preconditioner, no 'neighbor' boundary element is considered. To bring more faster convergence to GMRES iteration, the MN(n) preconditioner is then proposed, where n stands for the number of selected neighbor elements. The geometry distance of two elements need not to calculate, since the matrix **A** is stored explicitly and its entry value can be used to judge the neighborhood. In a matrix row, the absolute value of an entry reflects the interaction between two related elements. We use the maximum absolute value of entries for same element (two entries are related with an interface element) to measure neighborhood of the element to current collocation point. Thus, we can select n elements that have the maximum neighboring measurement by comparing all non-zero entries in row i. These n elements are then considered as the neighbors to row i's source element, and the indices of their unknowns are added to L. Because the index for the current element must be selected and one element may contain two unknowns, the L has $2(n+1)$ items at most in the MN(n) preconditioner. Only manipulation of comparing float digits is needed to generate the set L, which is of the complexity $O(nN_{\text{non-zero}})$ where $N_{\text{non-zero}}$ is the number of non-zero matrix entries. The construction of **P** includes solving N reduced Eq. (10), whose degree varies from $(n+1)$ to $2(n+1)$.

The difference of our MN(n) preconditioner to the existing MN-type preconditioners (such as that in Ref. [17]) is that we used the explicitly stored matrix entries to judge the neighborhood to avoid calculating the 3D spatial distance between elements, which involves relatively complex manipulations such as multiplication and evaluation of square root. So, our method has less computational consumption for a little n and adapts well to the 3D electric multi-zone BEA.

In the iterative multi-zone BEA, another large part of memory is used for the orthogonal basis vectors in GMRES algorithm, besides the memory for storing the coefficient matrix. Since this memory usage is proportional to the number of iterations, the total memory usage can therefore be reduced while using a better preconditioning technology.

5. Numerical results

The experiments for electric multi-zone BEA are all carried on a SUN Ultra Enterprise 450 Server with 248 MHz. The program is written with C++ language, and compiled with an '-O' (optimal) option. The stop criterion $\varepsilon = \|\mathbf{f} - \mathbf{A}\mathbf{x}_j\|/\|\mathbf{f}\|$ in the GMRES solver is set to be 10^{-3}.

5.1. Experiments for the equation assembly approach

Two cases from actual VLSI layout are simulated for capacitance. The first one is the structure shown in Fig. 1. The simulation region is about $10.7 \times 4.3 \times 5.8$ (in μm). The second case is larger, with the dimensions of $11.3 \times 14.2 \times 5.1$ (in μm), and there are 16 conductors embedded in five dielectric layers. These structures with different QMM decomposition are analyzed, while using three approaches of equation organization. The first one uses our approach described in Section 3.1. The second approach obeys the equation assembly of Ref. [13], and stores the matrix in many multiple-dimensional arrays as that in Ref. [14]. Although this implementation is not the best among those obeying the rules in Ref. [13], it is still valid for comparison with the first approach. The third approach uses the general CRS format to store the matrix [15], which is suitable to any arrangement of BEA equations. The related computational results are shown in Table 2.

In all computations, the Jacobi preconditioner is used. (QMM $a \times b$) stands for imposing $a \times b$ QMM cutting. For these structures a strategy of non-uniform density partitioning is used to achieve high accuracy with fewer elements, and more details can be found in Ref. [11]. The column of 'FDM' is the capacitance results of RAPHAEL, a famous commercial software for parasitic capacitance extraction employing a finite difference solver with advanced non-uniform meshing. The results of RAPHAEL under very dense mesh are listed as standard values, to show the accuracy of our electric BEA.

Compared with approach 2, the first approach shows remarkable time reduction in the matrix–vector multiplication, as well as the whole solution phase. For two cases where 100 or more zones are involved, the CPU time of solving the equation with the approach 1 is even less than

Table 2
Comparisons of CPU time of each matrix–vector multiplication and whole solution phase, for different equation organization approaches

Case	No. of zones	No. of elements	No. of non-zero entries	Iterative number	CPU time (s)						BEM, cap.[a]	FDM, cap.[a]
					Approach 1		Approach 2		Approach 3			
					MV.[b]	Solution	MV.[b]	Solution	MV.[b]	Solution		
1	5	1800	132.5×10^4	22	0.11	2.54	0.13	3.25	0.17	4.06	795	800
1 (QMM3 × 2)	30	2080	47.7×10^4	24	0.04	1.21	0.06	1.76	0.06	1.67	797	800
1 (QMM7 × 3)	105	2528	27.5×10^4	24	0.03	0.83	0.10	2.73	0.04	0.98	814	800
2	5	2574	297.8×10^4	25	0.24	6.41	0.30	8.21	0.40	10.45	1597	1637
2(QMM3 × 3)	45	2810	88.6×10^4	29	0.07	2.49	0.12	3.98	0.12	3.61	1618	1637
2(QMM4 × 5)	100	3399	58.6×10^4	32	0.05	2.26	0.16	5.76	0.08	2.86	1635	1637

[a] Cap., capacitance value (unit in 10^{-18} farad).
[b] MV., one matrix–vector multiplication.

half of that for the second approach. This is because a lot of auxiliary time for dealing with the very disordered matrix by the equation assembly of Ref. [13] is involved. The third approach performs better than the second approach for problem with many zones. But it is still not as good as the first approach. Furthermore, more memory for storing positions and more manipulations for organizing the non-zero coefficients are needed for the CRS format, compared with the BCRS format used in the first approach.

It should also be pointed out, that the computational results are the same for different equation organization, because their corresponding coefficient matrix can be transformed to each other by symmetric transpositions of rows and columns, which do not influence the procedure of GMRES iteration except for the order of unknowns in the result vector. Compared with RAPHAEL, the BEAs with three approaches all give results with error less than 3%.

In Table 2, the number of non-zero matrix entries $N_{\text{non-zero}}$ in each computation is listed, as well as the number of iterations k. For these six computations, Fig. 8 shows GMRES solution time versus the product of $N_{\text{non-zero}}$ and k, while using the approach 1 for equation organization. From it, we can see a nearly linear relationship:

$$T_{\text{sol}} \propto N_{\text{non-zero}} \cdot k \tag{12}$$

where T_{sol} stands for the CPU time of equation solution. It demonstrates that the approach 1 is suitable for the multi-zone BEA with many zones, and ensures a very efficient solution phase for the sparse coefficient matrix. By the way, the equation solution time accounts for about 32–46% of the whole BEA time while using approach 1.

5.2. Experiments for GMRES preconditioners

The computational results of three capacitance problems and one resistance problem, with four preconditioners, are listed in Table 3. Two capacitance problems are the same as that used in Table 2, and with different QMM cutting. The last capacitance problem includes complex conformal dielectrics in some layers, and the original structure involves 11 dielectric layers (with 3 × 3 QMM cutting, the zone number increases to 99). The resistance problem is shown in Fig. 9, where the resistances between one 1 V port to other 0 V ports are to be computed. Since the potential in conductor body is also determined by the Laplace equation [21], the resistance problem is analogous to the capacitance problem except that the electric current penetrating the port is wanted instead of electric charge on conductor. With suitable QMM cutting, the resistance problem includes 25 zones. In the following computations, the proposed scheme of equation assembly of Section 3.1 is used.

For each case, the iteration number decreases gradually for the preconditioners in order: Jacobi, EJ, MN(1) and MN(2). The EJ shows much reduction in iteration number, compared with the Jacobi. At the same time, the CPU time spent in the whole solution phase has been reduced by about 30%. Because more time is spent in construction and multiplication of the preconditioner, the MN(1) and MN(2) show no advantage to the EJ for the first two capacitance problems. While for problem 3 with 3 × 3 QMM cutting and the resistance problem, the MN(1) and MN(2) perform better because these problems converge slower than others, so that the iteration reduction brought by them shows more important effect. In Table 3, the memory usages of whole computation are also listed. From it we can see that EJ and MN(1) preconditioner reduce the memory us age markedly,

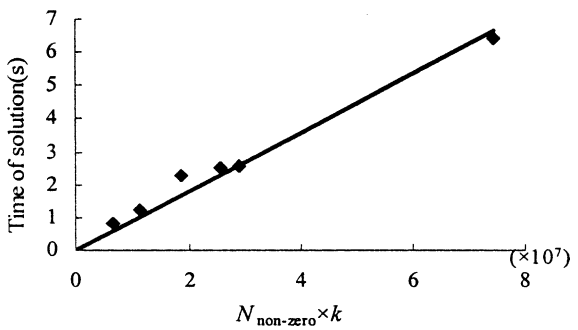

Fig. 8. CPU time of the GMRES solution versus the product of the number of non-zero matrix entries and the iterative number.

Table 3
Comparisons of four preconditioners for three capacitance problems and one resistance problem

Case	No. of zones	No. of unknowns	Jacobi			Extended Jacobi			MN(1)			MN(2)		
			Iter.[a]	Sol.[b]	Mem.[c]	Iter.[a]	Sol.[b]	Mem.[c]	Iter.[a]	Sol.[b]	Mem.[c]	Iter.[a]	Sol.[b]	Mem.[c]
1	5	2023	22	2.54	6.44	16	1.86	6.44	15	2.18	6.45	15	2.24	6.51
1 (QMM3 × 2)	30	2608	24	1.21	3.96	15	0.76	3.91	14	0.93	3.93	14	0.99	3.99
2	5	3049	25	6.41	13.38	17	4.40	13.34	17	5.21	13.38	17	5.27	13.46
2 (QMM4 × 5)	100	4793	32	2.26	6.80	22	1.56	6.66	21	1.79	6.71	21	2.18	6.82
3	11	4501	51	19.33	19.14	26	9.86	18.48	23	10.00	18.47	21	9.44	18.51
3 (QMM3 × 3)	99	5378	56	8.30	11.83	31	4.41	11.05	26	4.24	10.95	25	4.38	11.03
4 (resistance)	25	2763	107	10.83	4.85	62	5.22	4.25	46	3.54	4.11	45	3.55	4.11

[a] Iter., number of iterations.
[b] Sol., CPU time of equation solution (unit in s), including preconditioner construction.
[c] Mem., memory usage of the whole computation (unit in MB).

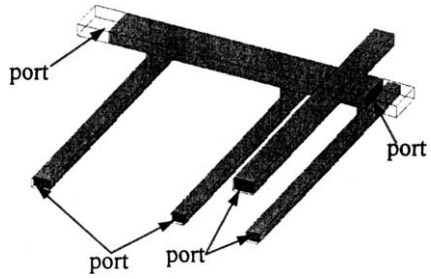

Fig. 9. A 3D structure of VLSI interconnects, where resistances between ports need to be computed.

although the reduction is often a little because the memory for storing the orthogonal basis vectors dose not accounts for a dominative part in the whole memory consumption.

More than 100 structures of VLSI interconnects have been computed. The four preconditioners demonstrate gradual decrease in the iteration number for all tested cases. With respect to the total computational time involving construction of the preconditioner, the EJ and MN(1) perform the best, which reduce about 30% or more solution time than the Jacobi, on average. For the problem with larger order (10,000 or more), more experiments show that MN(1) have better performance than EJ.

6. Conclusions

The impact of equation organization on iterative solution of multi-zone BEA is discussed, which is of special prominence for fast solution of large-scale problem with many zones, It should be the most complete investigation on this topic. Because the involved many zones lead to much disordering or much more non-zero blocks in the system matrix, a non-serious organization would make the matrix–vector product inefficient, spending much time on locating the non-zero matrix entries. Analysis and experiments have shown the presented equation assembly based on the ordering in Ref. [3] is most efficient for the iterative multi-zone BEA. Thanks to the particularity of fast electric simulation, i.e. the preconditioner must not be complex, a group of preconditioners based on the mesh neighbor method [17] are proposed. Among them, the EJ, MN(1) and MN(2) preconditioners all show gradual reduction in iteration number of GMRES algorithm, in comparison with the Jacobi preconditioner. While in respect to the total CPU time including constructing the preconditioner, EJ and MN(1) both have demonstrated at least 30% reduction than the Jacobi preconditioner. On the aspects of equation assembly, storage and solution, here we have presented an efficient scheme for the iterative multi-zone BEA, especially suitable for the fast 3D electric simulation of VLSI interconnects.

References

[1] Brebbia CA, Telles JCF, Wrobel LC. Boundary element techniques: theory and applications in engineering. Berlin: Springer; 1984.
[2] Crotty JM. A block equation solver for large unsymmetric matrices arising in the boundary element method. Int J Numer Meth Engng 1982;18:997–1017.
[3] Kane JH. Boundary element analysis in engineering continuum mechanics. Englewood Cliffs, NJ: Prentice-Hall; 1994.
[4] Kane JH, Kashava BLK. An arbitrary condensing, noncondensing solution strategy for large scale, multi-zone boundary element analysis. Comput Meth Appl Mech Engng 1990;79:219–44.
[5] Rigby RH, Aliabadi MH. Out-of-core solver for large, multi-zone boundary element matrices. Int J Numer Meth Engng 1995;38:1507–33.
[6] Kane JH, Keyes DE, Prasad Guru K. Iterative solution techniques in boundary element analysis. Int J Numer Meth Engng 1991;31:1511–36.
[7] Prasad Guru K, Kane JH. Preconditioned Krylov solvers for BEA. Int J Numer Meth Engng 1994;37:1651–72.
[8] Walker MG. Modeling the wiring of deep submicron ICs. IEEE Spectrum 2000;37(3):65–71.
[9] Nabors K, White J. Multipole accelerated capacitance extraction algorithm for 3-D structures with multiple dielectrics. IEEE Trans Circuits Sys I 1992;39(11):946–54.

[10] Bachtold M, Korvink JG, Baltes H. Enhanced multipole acceleration technique for the solution of large Poisson computations. IEEE Trans Comput-Aided Des 1996;15(12):1541–6.
[11] Yu W, Wang Z, Gu J. Fast capacitance extraction of actual 3-D VLSI interconnects using quasi-multiple medium accelerated BEM. IEEE Trans Microwave Theory Tech 2003; 51(1):109–19.
[12] Yu W, Wang Z. A fast quasi-multiple medium method for 3-D BEM calculation of parasitic capacitance. Comput Math Appl 2003;45(12): 1883–94.
[13] Merkel M, Bulgakov V, Bialecki R, Kuhn G. Iterative solution of large-scale 3D-BEM industrial problems. Engng Anal Bound Elem 1998;22:183–97.
[14] Li Y, Hou J, Wang Z. Calculating capacitance with the multiple dielectrics using the direct BEM and accelerating with single dielectric. Syst Engng Electron 1999;21(6):10–15. in Chinese.
[15] Araujo FC, Belmonte GJ, Freitas MSR. Efficiency increment in 3D multi-zone boundary element algorithms by use of iterative solvers. J Chin Inst Engrs 2000;23(3):269–74.
[16] Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM, J. Sci Stat Comput 1986;7(3):856–69.
[17] Vavasis SA. Preconditioning for boundary integral equations. SIAM, J Matrix Anal Appl 1992;13(3):905–25.
[18] Nachtigal NM, Reddy S, Trefethen N. How fast are nonsysmmetric matrix iterations. SIAM J Matrix Anal Appl 1992;13(3): 778–95.
[19] Chen K. On a class of preconditioning methods for dense linear systems from boundary element. SIAM, J Sci Comput 1998;20(2): 684–98.
[20] Chen K. An analysis of sparse approximate inverse preconditioners for boundary integral equations. SIAM, J Matrix Anal Appl 2001; 22(4):1058–78.
[21] Wang Z, Wu Q. A two-dimensional resistance simulator using the boundary element method. IEEE Trans Comput-Aided Des 1992; 11(4):497–504.
[22] Barrett R, Berry M, Chan TF, et al. Templates for the solution of linear systems: Building blocks for iterative methods. Philadelphia, PA: SIAM; 1994. . Available at http://www.netlib.org/.