ICCAP: A Linear Time Sparse Transformation and Reordering Algorithm for 3D BEM Capacitance Extraction[*]

Rong Jiang [†], Yi-Hao Chang [‡], and Charlie Chung-Ping Chen [†]
[†] ECE Department, University of Wisconsin, Madison, WI 53706-1691
[‡] GIEE & EE Department, National Taiwan University, Taipei 106

ABSTRACT

This paper presents an efficient hierarchical 3D capacitance extraction algorithm – ICCAP. Most previous capacitance extraction algorithms introduce intermediate variables to facilitate the hierarchical potential calculation but still preserve the leaf panels as the basis. In this paper, we discover that those intermediate variables are fundamentally much better basis than leaf panels. As a result, we are able to explicitly construct the sparse potential coefficient matrix and solve it with *linear* memory in *linear* runtime. Furthermore, the explicit sparse formulation not only enables the usage of preconditioned iterative Krylov subspace methods but also the reordering technique. A new reordering technique is proposed to further reduce over 20% of memory consumption and runtime in comparison to no reordering techniques applied. Experimental results demonstrate the superior runtime and memory consumption of ICCAP over previous approaches while achieving similar accuracy.

Categories and Subject Descriptors: B.7.2 [Integrated Circuits]: Design Aids – simulation, verification.

General Terms: Algorithms.

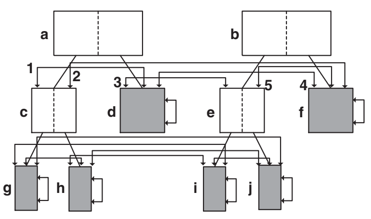
Keywords: Boundary element method, capacitance, parasitic extraction, interconnect, iterative methods.

1. INTRODUCTION

With the reduced feature size and the increased operation frequency, extracting self and coupling capacitances associated with on-chip interconnects and packages has become increasingly important for determining the functionality and performance of very large scale integration (VLSI) design [1].

Boundary element method (BEM) has been adopted as the main approach for 3D capacitance calculation. To numerically solve the integral equation associated with BEM, the conductor surfaces are discretized into much smaller panels, and surface charges on those most delicate panels (leaf panels) are assumed to be uniform. Therefore, a dense linear system $Pq = v$ is formulated, where $P \in \mathcal{R}^{n \times n}$ is referred to as the potential coefficient matrix and $q, v \in \mathcal{R}^n$ are leaf panel charges and potentials respectively.

HiCap [2] and PHiCap [3] propose a hierarchical algorithm, which can be represented by a multiple-tree structure as shown in Fig. 1. The root panel of each tree structure corresponds to a conductor surface or a dielectric interface. If the estimated potential coefficient between the two panels is larger than a threshold value P_ϵ, they are further divided into smaller panels. Otherwise, a link recording the potential coefficient is created between these two panels.



$$J = \begin{array}{c|cccccc} & d & f & g & h & i & j \\ \hline a & 1 & & 1 & 1 & & \\ b & & 1 & & & 1 & 1 \\ c & & & 1 & 1 & & \\ d & 1 & & & & & \\ e & & & & & 1 & 1 \\ f & & 1 & & & & \\ g & & & 1 & & & \\ h & & & & 1 & & \\ i & & & & & 1 & \\ j & & & & & & 1 \end{array}$$

Figure 1: Hierarchical BEM algorithms.

The multiple-tree structure can be fully described by a link matrix $H \in \mathcal{R}^{N \times N}$ and a structure matrix $J \in \mathcal{R}^{N \times n}$ [3], where N is the number of all panels and n the number of leaf panels. Each row of J corresponds to a panel, either leaf or non-leaf, and each column corresponds to a leaf panel. The (i, j) entry in J is 1 if panel i contains the leaf panel j, and is 0 otherwise [3]. For any two panels with no links in between, the corresponding entries in H are zero.

Since in every elementary tree, the parent panel charge is the sum of the charges on its two child panels, all panel charges can be represented by charges on leaf panels,

$$q_N = Jq, \tag{1}$$

where $q_N \in \mathcal{R}^N$ is the vector of all panel charges. Let $v_N \in \mathcal{R}^N$ denote the vector of potentials induced by links on individual panels, it is obtained that

$$v_N = Hq_N. \tag{2}$$

Since the potential on a parent panel distributes to its two children, the leaf panel potential vector is equal to

$$v = J^T v_N. \tag{3}$$

[*]This work was partially funded by Intel, TSMC, UMC, Faraday, SpringSoft, National Science Foundation under grants CCR-0093309 & CCR-0204468 and National Science Council of Taiwan, R.O.C. under grant NSC 92-2218-E-002-030.

So the potential coefficient matrix can be formulated as

$$P = J^T H J. \tag{4}$$

Therefore P is developed based on charges on leaf panels.

2. ICCAP ALGORITHM

Since every panel charge can be represented by a unique linear combination of charges on leaf panels as in Eq. 1, the set of leaf panel charges is a basis and leaf panels are corresponding basis panels.

However, since leaf panels interact with each other through links between themselves or their upper-level parent panels, every entry in the corresponding potential coefficient matrix P is non-zero, and hence the total number of fill-ins is n^2. Consequently, if we take all leaf panel charges as the basis, the corresponding potential coefficient matrix in Eq. 4 will be the densest one.

Fortunately, for a given panel refinement, there are many possible bases. For example, for the multiple-tree structure in Fig. 1, Fig. 2 shows another set of basis, which includes two non-leaf panels c and e. The corresponding structure matrix J' of the new basis is also shown in Fig. 2.

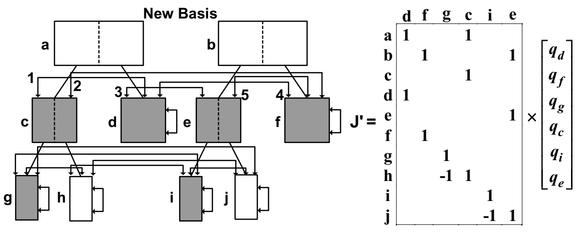


Figure 2: Another basis for a panel refinement.

Since each basis has its distinct structure matrices J', the related potential coefficient matrix $P' = J'^T H J'$ has different densities. Therefore, it's desirable that one can choose a basis so that its related potential coefficient matrix is sparse.

2.1 New Basis Panels

Theorem 1. *Assume the structure matrix and the potential coefficient matrix corresponding to a basis are J and P respectively. If the current basis contains two panels j and k, which are child panels of panel i, then arbitrarily eliminating one of them (say k) and adding their parent panel i to the basis gives another set of basis panels. The new structure matrix J' corresponding to the new basis can be obtained by*

$$\begin{aligned} J'_j &= J_j - J_k; \\ J'_i &= J_k. \end{aligned}$$

where J_i represents the column corresponding to panel i in J. And the new potential coefficient matrix P' is given by

$$P' = E^T P E,$$

where E is an elementary transformation matrix.

Theorem 2. *The basis which includes all root panels and all left-hand side panels will lead to a sparse potential coefficient matrix containing $O(n)$ non-zero entries.*

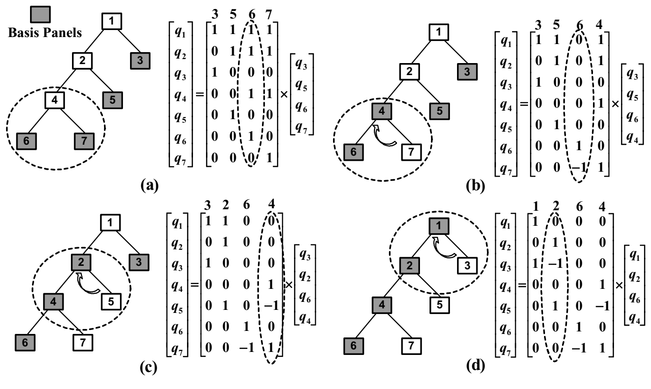


Figure 3: Generating new basis panels.

We use an example to gain a clear idea of those important theorems. As shown in Fig. 3.(a), leaf panels 6 and 7 are children of panel 4. Now, we move one basis panel from panel 7 to its parent panel 4 as shown in Fig. 3.(b). Apparently this movement results in a new basis since all panel charges still can be represented by charges on the new set of panels. The new J' matrix is shown on right hand side in Fig. 3.(b).

The column corresponding to panel 4 in J' is identical to the column corresponding to panel 7 in J, since upper level panels originally gathering the charge on panel 7 still collect the charge on panel 4. So the column of panel 4 in J' "inherits" the column of panel 7 in J. On the contrary, since the charge on panel 4 is the sum of charges on panels 6 and 7, upper level panels now only need to gather the charge on panel 4 and do not need the charge on panel 6. So the column corresponding to panel 6 in J' is

$$J'_6 = J_6 - J_7. \tag{5}$$

Furthermore, Eq. 5 can be represented in a matrix form as $J' = JE$, where E is an elementary transformation matrix.

$$E = \left[\begin{array}{cccc} \ddots & & & \\ & 1 & 0 & \\ & -1 & 1 & \\ & & & \ddots \end{array} \right] \begin{array}{l} \\ \text{panel 6} \\ \text{panel 7} \\ \\ \end{array} \tag{6}$$

Consequently, the relation between the new potential coefficient matrix P' and P can be written as $P' = J'^T H J' = (JE)^T H (JE) = E^T P E$.

It is important to notice that this transformation only changes the column and row related to panel 6. P' is obtained by subtracting the column and row of panel 7 from the column and row of panel 6. Since links on upper level panels introduce identical fill-ins in columns and rows of panel 6 and 7, the subtraction cancels out identical terms and creates many zeros in P'.

Moving basis panels upward can be executed continuously as shown in Fig. 3.(c) and 3.(d). In each step, many zeros are created by eliminating identical terms and previously created zero entries will not be destroyed in the later steps. At the end, the result basis will include root panels and left-hand side (LHS) panels. So in the new potential matrix P', the number of non-zeros is comparable with the total number of links in the multiple tree structure, which has been proven to be $O(n)$ [2]. This property has also been observed in the experiment as shown in Fig. 4.

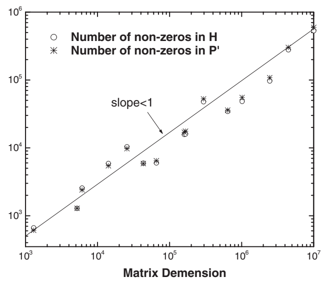


Figure 4: Comparison of non-zeros in H and P'.

The selection of basis panels is not unique since in each elementary operation, we can either eliminate right-hand side (RHS) panels or LHS panels. However, the construction of J' will be simplified by choosing the basis in Theorem 2.

2.2 Direct Formulation of J' in Linear Time

Theorem 3. *In the column J'_i corresponding to a basis panel i, each entry J_{ij} is 1 if panel i contains the right-hand side panel j. If panel i is not a root panel, then each entry J_{ij} is -1 if the parent of panel i contains the right-hand side panel j.*

The above theorem can be illustrated by a small example in Fig. 5.(a). Panel 2 is a LHS panel and has been included in the new basis. Panel 5 and 7 are its underlying RHS panels and hence the corresponding entries in J' are filled by 1. The parent of panel 2 contains RHS panel 3, so that the corresponding entry in J' is -1.

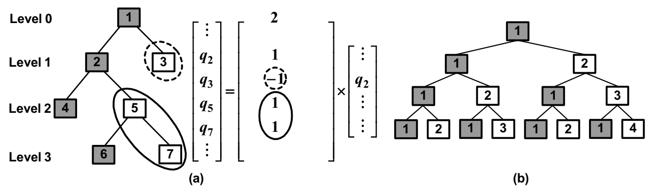


Figure 5: Efficient construction of J'.

Theorem 4. *The new structure matrix J' and the new potential coefficient matrix P' corresponding to the new basis in Theorem 2 have $O(n)$ entries.*

Assume a complete tree structure containing n leaf nodes and $m = lgn$ levels with root node in level 0. In level i, there are 2^{i-1} LHS panels. Each LHS panel in level 1 to level m introduces $2m-2i+2$ fill-ins. So the total number of fill-ins in J' is given by $m+\sum_{i=1}^{m} 2^{i-1}(2m-2i+2) = 4n-lgn-4$. So non-zeros in J' is $O(n)$. Similarly, we can prove that J contains $O(nlgn)$ non-zeros. That is the reason why PHiCap [3] has $O(nlogn)$ runtime and memory consumption.

Also the total number of fill-ins in P' is $O(n)$. Denote the maximum number of links on one panel to be C_{max}. C_{max} must be a constant, otherwise the total number of links in H will not be $O(n)$. Thus for links on each individual panel, we can calculate the maximum number of fill-ins created in P'. For example, for links on a non-basis panel in level 1, such as the one shown in Fig. 6.(b), the maximum number of non-zeros they will create is $2 * C_{max}$. For a complete tree, the total number of panels that introduce $i * C_{max}$ fill-ins is $n/2^{i-1}$, and hence the total number of fill-ins in P' is less than $C_{max} n \sum_{i=0}^{lgn}(i+1)/2^i = C_{max}(4n - lgn - 3)$. Therefore, the total number of non-zeros in P' is $O(n)$.

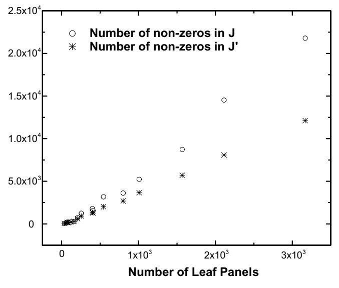


Figure 6: Comparison of non-zeros in J and J'.

2.3 Extracting E from J'

We have shown in Theorem 1 that $P' = E^T PE$. By substituting it into $P'q' = v'$, we get

$$E^T PEq' = v'. \quad (7)$$

Also we know that the original system is $Pq = v$. So these two equations can be satisfied by setting

$$v' = E^T v, \quad (8)$$
$$q = Eq'. \quad (9)$$

From $q = Eq'$, we can see that E is the coefficient matrix when leaf panel charges are represented by charges on new basis panels. Since all panel charges can be expressed by $q_N = J'q'$, so that E has been included in J'.

Since P' only contains $O(n)$ entries, the new linear system can be solved much more efficiently by preconditioned iterative matrix solvers, such as PCG and GMRES. Also notice that since the new basis includes all root panels, after solving q', root panel charges are already contained in q'.

2.4 Potential Coefficient Matrix Reordering

The distribution of non-zeros in P' affects the number of fill-ins in preconditioners produced by incomplete Cholesky or LU factorization. Directly applying minimum degree reordering (MMD) may still be expensive for large-scale design applications. So we propose a heuristic cost-free reordering method called Level-Oriented Reordering (LOR).

According to the selection of the new basis, it is reasonable to expect that columns and rows related to lower level basis panels contain more zeros than upper level basis panels, since fill-ins introduced by links on upper level panels can mostly be eliminated. So the basic idea of LOR is to assign basis panels in upper levels with larger indexes, thus most non-zeros will be in the lower right-hand side corner. LOR can be easily implemented by a bottom-up breadth-first search after the panel refinement process.

2.5 Complexity Analysis

The first step of ICCAP to select n basis panels based on Theorem 2 can be done by scanning all $N = 2n - 1$ panels to determine which are roots and LHS panels and hence takes O(n) time. The second step of constructing J' is equivalent to inserting $O(n)$ non-zeros in J' and hence is also $O(n)$. E is contained in J' and does not require extra time. P' has been proved to contain $O(n)$ non-zeros, so that the construction of $P' = J'^T H J'$ can also be done in $O(n)$.

3. EXPERIMENTAL RESULTS

ICCAP is implemented in $C++$ language. All experiments are executed on Sun-Blade 2000 with one 1.2 GHz UltraSPARC-III processor, 1G RAM and OS Sorlaris 8.

First we test the Level-Oriented Reordering (LOR) method embedded in the panel refinement process by using the bus 4×4 benchmark. Original lower and upper triangular factors from incomplete LU factorization contain 29017 and 24546 non-zeros respectively. By adopting LOR, the number of fill-ins is dramatically reduced by 30%. The result is comparable with directly applying MMD which in this case results in 22129 and 19633 fill-ins in L and U.

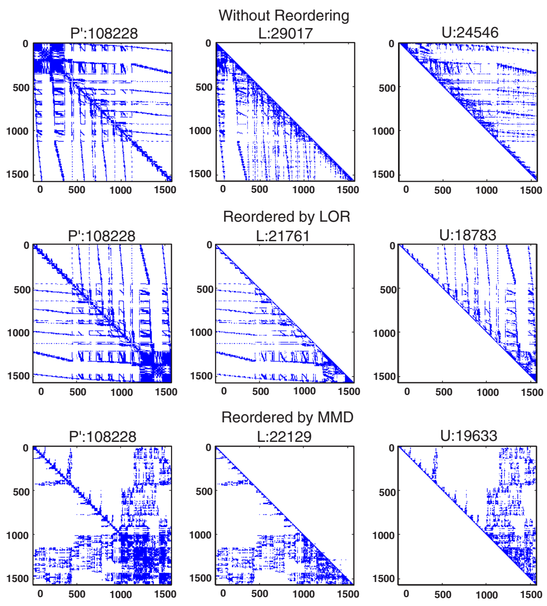


Figure 7: preconditioners from incomplete LU factorization with different reordering schemes ($RelativeResidue = 0.01$).

Table 1 compares the performance of three algorithms: FastCap, HiCap, and ICCAP. The convergence tolerance is set to 0.01 and error is calculated with respect to FastCap. Compared to FastCap, ICCAP is $40 - 80$ times faster and with much less memory. Compared to HiCap, for the bus 12×12 benchmark, ICCAP exhibits nearly 5 times speedup. HiCap represents P as a block matrix and hence the real storage of P is $O(n)$. All H, J, and P' in ICCAP contain $O(n)$ non-zeros, so that the memory consumptions of ICCAP and HiCap are in the same order. Also it shows that HiCap and ICCAP have comparable accuracy.

We do not have access to PHiCap [3] and cannot compare to it explicitly. Published results show PHiCap is $2 - 3$ times faster than HiCap for the testing benchmarks in Table 2. Based on the comparison with HiCap, we can expect ICCAP to be faster than PHiCap as well. Also for testing cases in Table 2, normally ICCAP converges in less than 2 iterations while PHiCap needs about 3 iterations. Also, the main disadvantage of PHiCap is its memory consumption due to the explicit formulation of transformation matrix while ICCAP directly formulates the sparse matrix P'. Also [3] shows that PHiCap has lower accuracy than HiCap. So ICCAP can be superior to PHiCap in terms of memory and accuracy.

Also we use ICCAP and HiCap to test large files containing more conductors. The result is shown in Table 2. For these test files, ICCAP can converge within four iterations and shows more significant speedup compared to HiCap.

4×4 Bus, Unit List: Time(Sec), Memory(MB)

Algorithm	Time	Iteration	Memory	Error	Panels
FastCap	8.03	18.63	26.27	–	2736
HiCap	0.49	8.7	0.99	0.60%	2176
ICCAP	0.20	1.12	0.581	0.76%	2182

6×6 Bus, Unit List: Time(Sec), Memory(MB)

Algorithm	Time	Iteration	Memory	Error	Panels
FastCap	35.55	14.4	65.19	–	5832
HiCap	1.33	14.5	1.73	1.42%	3168
ICCAP	1.04	1.08	1.54	1.50%	3468

8×8 Bus, Unit List: Time(Sec), Memory(MB)

Algorithm	Time	Iteration	Memory	Error	Panels
FastCap	67.4	12	114.5	–	10080
HiCap	3.60	13.4	3.07	1.63%	4224
ICCAP	1.44	1.43	2.94	1.91%	4656

12×12 Bus, Unit List: Time(Sec), Memory(MB)

Algorithm	Time	Iteration	Memory	Error	Panels
FastCap	357.99	18.1	297.8	–	22032
HiCap	18.13	15.1	3.95	1.08%	6240
ICCAP	4.24	1.41	3.68	1.18%	6874

Table 1: Simulation results comparison.

Cond Num	32		36	
Algorithm	HiCap	ICCAP	HiCap	ICCAP
Time	48.76	10.42	81.14	12.35
Iteration	15.8	2.58	18.6	3.26
Memory	6.79	5.93	8.58	8.17
Panels	8448	8536	9504	9658

Table 2: Comparison with HiCap for large cases.

4. CONCLUSION

This paper presents a fast 3-D capacitance extraction algorithm, ICCAP. ICCAP proposes a novel technique for sparsifying and reordering the potential coefficient matrix. The sparse transformation is performed by simply switching basis from leaf panels to a new set of panels. Thus cost-efficient preconditioners can be easily constructed to greatly speedup iterative matrix solvers.

5. REFERENCES

[1] K. Nabors and J. White. Fastcap: a multipole accelerated 3d capacitance extraction program. *IEEE Trans. on CAD*, pages 1447–1459, 1991.

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

[3] S. Yan, V. Sarin, and W. Shi. Sparse transformations and preconditioners for hierarchical 3-d capacitance extraction with multiple dielectrics. *Proc. DAC*, pages 788–793, 2004.