January 2010

A Parallel Direct Solver for the Simulation of Large-Scale Power/Ground Networks

S. Cauley

V. Balakrishnan

Koh Cheng-Kok

Cauley, S.; Balakrishnan, V.; and Cheng-Kok, Koh, "A Parallel Direct Solver for the Simulation of Large-Scale Power/Ground Networks" (2010). *Department of Electrical and Computer Engineering Faculty Publications.* Paper 60.
http://dx.doi.org/http://dx.doi.org/10.1109/TCAD.2010.2042901

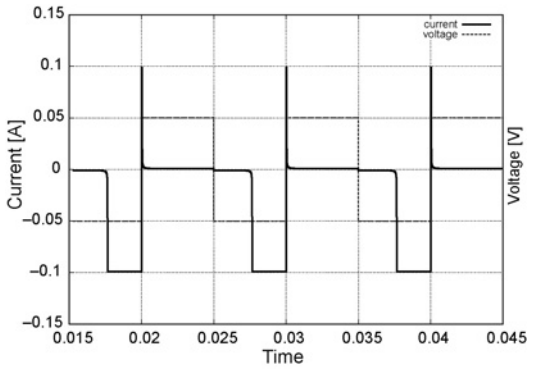

Fig. 9. Behavior of the model when square wave voltages were connected to its terminals.

to the Intel Core2 Duo 3 GHz CPU, using only a single core, corresponds to 45 000 clock cycles.

These simulation measurements show similar behavior to the results of the implemented memristor [2] by HP Labs.

IV. Conclusion

We presented a new SPICE macromodel of the recently implemented memristor. This macromodel could be a powerful tool for researchers and electrical engineers to design and experiment new circuits with memristors. Comparing the *IV* characteristics and also the time dependences of the state variables, our simulation results show the figures with qualitatively and quantitatively similar behavior to the lately published measurements of the physical implementation [2]. The functionality of our macromodel is demonstrated with computer simulations. The source code of our macromodel can be found in the appendix below.

Appendix A
SPICE Code of the Memristor Macromodel

```
.SUBCKT memristor 1 2 6
Eres   1   9        POLY(2)
+(8, 0) (11, 0) 0 0 0 0 1
Vsense 9   4        DC 0V
Fcopy  0   8        Vsense 1
Rstep  8   0        1K
Rser   2   4        10
Fmem   6   0        POLY(2) Vsense
+Ecopy -0.5E-10 0 1E-10 0 -1 0 0 0 1
Cmem   6   0        90nF
Rsp    6   0        1000Meg
Ecopy  7   0  0   6 1
Rc     7   0        1
Ecpy2 10   0  6   0 1
Vref ref   0        DC 1V
R1    10  11        100K
Ssat1 11   0  0  11 SWX
Ssat2 11 ref 11 ref SWX
.MODEL SWX SW(Ron=0.001, Roff=1000Meg,
+Vt=0.00001V, Vh=0.00001V)
.ENDS
```

Acknowledgment

The authors wish to thank Prof. Tamás Roska for his inspiration, discussions, and his suggestions.

References

[1] L. Chua, "Memristor: The missing circuit element," *IEEE Trans. Circuit Theory*, vol. 18, no. 5, pp. 507–519, Sep. 1971.
[2] D. B. Strukov, G. S. Snider, D. R. Stewart, and S. R. Williams, "The missing memristor found," *Nature*, vol. 453, no. 7191, pp. 80–83, May 2008.
[3] G. Snider, "Architecture and methods for computing with reconfigurable resistor crossbar," U.S. Patent 7 203 789, Apr. 10, 2007.
[4] G. Snider, "Molecular-junction-nanowire-crossbar-based neural network," U.S. Patent 7 359 888, Apr. 15, 2008.
[5] B. Mouttet, "Programmable crossbar signal processor," U.S. Patent 7 302 513, Nov. 27, 2007.
[6] B. Mouttet, "Operational amplifier with resistance switch crossbar feedback," U.S. Patent Application 2008/0307151, Dec. 11, 2008.
[7] Z. Biolek, D. Biolek, and V. Biolková, "SPICE model of memristor with nonlinear dopant drift," *Radioengineering*, vol. 18, no. 2, pp. 210–214, 2009.
[8] J. Tour and T. He, "Electronics: The fourth element," *Nature*, vol. 453, no. 7191, pp. 42–43, May 2008.
[9] P. Marks, "Missing memristor makes an appearance," *New Scientist*, vol. 198, no. 2654, p. 26, May 2008.
[10] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *Nature*, vol. 459, no. 7250, corrigendum, p. 1154, Jun. 2009.
[11] L. Chua and S. Kang, "Memristive devices and systems," *Proc. IEEE*, vol. 64, no. 2, pp. 209–223, Feb. 1976.

A Parallel Direct Solver for the Simulation of Large-Scale Power/Ground Networks

Stephen Cauley, Venkataramanan Balakrishnan, and Cheng-Kok Koh

***Abstract*—An algorithm is presented for the fast and accurate simulation of power/ground mesh structures. Our method is a direct (non-iterative) approach for simulation based upon a parallel matrix inversion algorithm. The new dimension of flexibility provided by our algorithm allows for a more accurate analysis of power/ground mesh structures using resistance, inductance, capacitance, interconnect models. Specifically, we offer a method that employs a sparse approximate inverse technique to consider more reluctance coupling terms for increased accuracy of simulation. Our algorithm shows substantial computational improvement over the best known direct and iterative numerical techniques that are applicable to these large-scale simulation problems.**

***Index Terms*—Circuit simulation, IR drop, mesh simulation, parallel, power and ground networks.**

I. Introduction

The accurate and efficient modeling and simulation of power/ground networks has become a difficult problem for modern design. Increases to integration density have necessitated the use of large-scale power mesh structures, and with the scaling of voltages, the need for accurate simulation of

Manuscript received August 25, 2008; revised April 5, 2009 and December 8, 2009. Current version published March 19, 2010. This paper was recommended by Associate Editor, F. N. Najm.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-2035 USA (e-mail: stcauley@purdue.edu; ragu@purdue.edu; chengkok@purdue.edu).

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

Digital Object Identifier 10.1109/TCAD.2010.2042901

these structures is crucial. Previously employed direct methods for simulation of this problem have become impractical due to both extraordinary memory requirements and prohibitive simulation times. This has prompted several variations of iterative schemes [1]–[7] that attempt to meet these rising computational challenges. The convergence for each of these methods, and therefore, the simulation time, is problem dependent (i.e., both switching activity within the network and branch coupling will affect the simulation time).

Although most of these methods have been shown to be quite successful for large-scale simulations (millions of nodes) of resistor capacitor mesh structures, none have clearly demonstrated an efficient and scalable approach to deal with inductive coupling effects. This can be largely attributed to the fact that with the inclusion of inductive coupling, much of the locality for the problem is lost. Specifically, an iterative method that uses small independent or slightly overlapped subsets of the network in order to infer information about the global system dynamics will not converge quickly if there is significant coupling across different regions of the network. In addition, as was alluded to by the authors of [4], the conditioning of the underlying system matrices would degrade if the interconnects, which constitute the mesh structure, are modeled as resistance, inductance, capacitance (RLC). By employing a parallel matrix inversion technique for simulation, we offer a stable alternative that allows for the efficient simulation of networks with a large amount of branch coupling. The parallel method for solving block tridiagonal systems presented in this paper scales well with the inclusion of additional reluctive coupling effects, within an assumed block tridiagonal structure.

II. Simulation of RLC Mesh Structures

When RLC interconnect models are used for the simulation of mesh structures (see Fig. 1), the typical modified nodal analysis representation yields equations of the form

$$\mathcal{G}x + \mathcal{C}\dot{x} = \mathcal{B} \tag{1}$$

where

$$\mathcal{G} = \begin{pmatrix} G & A_l^T \\ -A_l & 0 \end{pmatrix} \quad \mathcal{C} = \begin{pmatrix} C & 0 \\ 0 & L \end{pmatrix} \quad x = \begin{pmatrix} v_n \\ i_l \end{pmatrix}$$

$$\mathcal{B} = \begin{pmatrix} A_i^T I_s \\ 0 \end{pmatrix} \quad G = A_g^T R^{-1} A_g \quad \text{and} \quad C = A_c^T \hat{C} A_c.$$

Here, R, $\hat{C}$, and L are the resistance, capacitance, and inductance matrices, respectively. The matrices A_g and A_c transform the conductances and capacitances into node based relationships. The matrices A_l and A_i link the node voltages and branch currents described by the state variable x. In addition, I_s is the current vector that dictates, through the matrix A_i, the relationship of the current sinks onto the nodes of the mesh. Considering a uniform discretization of the time axis with resolution h and using the notation $v_n^k = v_n(kh)$ to denote the voltage at the nth node, we may then solve for v^{k+1}

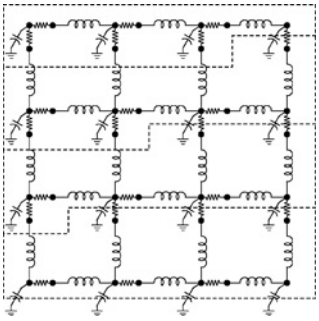

Fig. 1. 4×4 RLC mesh structure. The nodes of the mesh, depicted as black circles, are separated into disjoint groups.

in terms of v^k through the nodal analysis equations [8]

$$\underbrace{\left(G + \frac{2}{h}C + \frac{h}{2}S\right)}_{K} v_n^{k+1} = \underbrace{\left(-G + \frac{2}{h}C - \frac{h}{2}S\right)}_{H} v_n^k + A_i^T\left(I_s^{k+1} + I_s^k\right) - 2A_l^T i_l^k \tag{2}$$

$$2A_l^T i_l^{k+1} = 2A_l^T i_l^k + hS\left(v_n^{k+1} + v_n^k\right)$$

where $S = A_l^T L^{-1} A_l$. It is important to note that with the inclusion of inductance, for the modeling of the interconnects, we must now account for the effect of this additional susceptance term S.

A. Inductance Approximation Methods

We begin first with the construction of the coefficient matrix K from (2), given a regular power mesh topology. If all mutual inductive couplings are considered, both the reluctance matrix L^{-1} and coefficient matrix K will be dense. In this paper, we investigate the efficiency and accuracy of simulating power mesh structures when considering a block tridiagonal susceptance matrix. A matrix Y is block tridiagonal if it has the form

$$Y = \begin{pmatrix} A_1 & -B_1 & & & \\ -B_1^T & A_2 & -B_2 & & \\ & \ddots & \ddots & \ddots & \\ & & -B_{N_y-2}^T & A_{N_y-1} & -B_{N_y-1} \\ & & & -B_{N_y-1}^T & A_{N_y} \end{pmatrix} \tag{3}$$

where each $A_i, B_i \in \mathbb{R}^{N_x \times N_x}$. Thus $Y \in \mathbb{R}^{N_y N_x \times N_y N_x}$, with N_y diagonal blocks of size N_x each. We now introduce a flexible method by which reluctance values can be approximated to produce a sparse block tridiagonal susceptance matrix.

In [8], [9] the accuracy for simulation with interconnects was explored using *window* based techniques. In those papers, reluctive coupling was considered only to exist between neighbors in a given layer of parallel wires. In this paper, we consider the use of a sparse approximate inverse technique (SPAI) [10]. Given an inductance matrix L, the SPAI method can be used to form another matrix M that is constructed in an attempt to match the inverse of the inductance matrix under the Frobenius norm

$$\|LM - I\|_F^2 = \sum_{i=1}^{n} \|(LM - I)\, e_i\|_2^2 \tag{4}$$

where n in the number of columns of L and e_i is the ith euclidean basis vector. Therefore, we can solve n independent

TABLE I

ACCURACY COMPARISON FOR RELUCTANCE APPROXIMATION METHODS AGAINST FULL INDUCTANCE, MESH SIZES m = 16, 32, 48, 64, WITH ASSOCIATED NUMBER OF UNKOWNS GIVEN AS ''MATRIX SIZE''

	16 × 16 Matrix Size: 736			32 × 32 Matrix Size: 3008			48 × 48 Matrix Size: 6816			64 × 64 Matrix Size: 12160	
Data	NNZ	S (%)	RMSE	NNZ	S (%)	RMSE	NNZ	S (%)	RMSE	NNZ	S (%)
Window	6256	98.9	6.75E−04	26 320	99.7	1.30E−03	60 208	99.9	2.29E−03	107 920	99.9
$\tau = 0.94$	14 040	97.4	3.48E−04	60 280	99.3	5.83E−04	138 776	99.7	1.14E−03	249 528	99.8
$\tau = 0.95$	19 612	96.4	3.31E−04	84 956	99.1	4.47E−04	196 124	99.6	1.10E-03	353 116	99.8
$\tau = 0.96$	30 032	94.5	2.76E−04	132 736	98.5	3.92E−04	308 400	99.3	5.26E−04	557 024	99.6
$\tau = 0.97$	45 564	91.6	2.42E−04	206 156	97.7	3.76E−04	482 460	99.0	1.74E−04	874 476	99.4
$\tau = 0.98$	73 114	86.5	2.01E−04	346 558	96.2	1.36E−04	817 778	98.2	2.29E−04	1 499 314	99.0
$\tau = 0.99$	119 188	78.0	1.82E−04	649 208	92.8	1.24E−04	1 598 196	96.6	1.84E−04	1 758 736	98.8

The column labeled "NNZ" contains the number of non-zero entries in the coefficient matrix and the column labeled "S (%)" contains the percentage of the matrix whose entries are zero.

least squares problems in order to construct the approximate inverse matrix M. In this paper, we employ a threshold based approach to create a matrix whose entries must be significant with respect to the absolute maximum in a column

$$|M_{i,j}| > (1-\tau)\max_j |M_{i,j}| \tag{5}$$

where the diagonal entries $M_{i,i}$ are always included. If τ is close to zero, this criterion would prevent fill-in and result in a matrix that is very sparse. The value $\tau = 1$ would correspond to a matrix M where the entire pattern of L^{-1} will be considered.

We may form a block tridiagonal coefficient matrix K by evenly separating the nodes in the RLC mesh structure into groups. This block decomposition is illustrated in Fig. 1, where all nodes enclosed together are considered to be part of the same block. Given this decomposition, a sparse approximation to L^{-1} is formed so that S and K will be block tridiagonal. Table I examines the accuracy of simulation considering increases to the drop tolerance parameter τ. Next, we address the numerical challenges associated with using a direct (non-iterative) approach for the simulation of these structures.

B. Inverses of Block Tridiagonal Matrices

The inverse of a symmetric block tridiagonal matrix can be computed explicitly, as detailed in [11], [12]. Specifically, there exist two sequences of "ratio" matrices $\{R_i\}$, $\{S_i\}$ so that the inverse of a block tridiagonal matrix K can be written using a "compact" representation

$$K^{-1} = \begin{pmatrix} D_1 & D_1 S_1 & \cdots & D_1 \prod_{k=1}^{N_y-1} S_k \\ R_1 D_1 & D_2 & \cdots & D_2 \prod_{k=2}^{N_y-1} S_k \\ \vdots & \vdots & \ddots & \vdots \\ (\prod_{k=N_y-1}^{1} R_k) D_1 & (\prod_{k=N_y-1}^{2} R_k) D_2 & \cdots & D_{N_y} \end{pmatrix}. \tag{6}$$

Here, the diagonal blocks of the inverse, D_i, and the ratio sequences can be determined through a series of recursions [11], [12]. The time complexity associated with determining the parametrization of K^{-1} by the above approach is $O(N_x^3 N_y)$, with a memory requirement of $O(N_x^2 N_y)$.

In this paper, we build upon these ideas to create a scalable distributed framework for the transient simulation of power mesh structures. We begin by generalizing the method from [11] in order to compute all information necessary to determine a distributed compact representation of K^{-1}. It is important to note that the method in [11] was developed specifically to determine the diagonal entries for a matrix with structure similar to that of K^{-1}, but not the entire compact representation. The question then becomes what additional computation is necessary to find the compact representation for K^{-1}, as shown in (6). If we examine closely the block tridiagonal portion of K^{-1}, we find the following relations:

$$\begin{aligned} D_i S_i = Z_i \quad &\Longrightarrow \quad S_i = D_i^{-1} Z_i \qquad i = 1, \ldots, N_y - 1 \\ R_i D_i = Z_i^T \quad &\Longrightarrow \quad R_i = Z_i^T D_i^{-1} \quad i = 1, \ldots, N_y - 1 \end{aligned} \tag{7}$$

where Z_i denotes the ith off-diagonal block of K^{-1}. Thus, we would like to calculate both the diagonal and off-diagonal blocks of K^{-1} in order to formulate a compact representation for the matrix.

Our divide-and-conquer algorithm is illustrated in Fig. 2. If K is separated into p sub-matrices $\{\phi_i\}$ there will be $\log p$ combining levels with a total of $p-1$ combining steps needed to form K^{-1}. Here, $\phi_{i\sim j}^{-1}$ represents the result of any combining step through the use of the matrix inversion lemma. For example, $\phi_{1\sim 2}^{-1}$ is the inverse of a matrix comprised of the blocks assigned to both ϕ_1 and ϕ_2. It is important to note that using the matrix inversion lemma repeatedly to join sub-matrix inverses will result in a prohibitive amount of memory and computation for large simulation problems. This is due to the fact that at each combining step all entries would be computed and stored. For example, from Fig. 2 we can see that the final combining level would require the computation of half the entries from the matrix: $\left(N_x N_y\right)^2/2$. Thus, we introduce matrix mappings in order to avoid any unnecessary computation during the combining process. Specifically, the matrix maps only require the computation of $4N_x^2$ entries for each combining step [11], [12].

It has been shown in [11] and [12] that both the boundary block entries (first block row and last block column) and the block tridiagonal entries from any combined inverse $\phi_{i\sim j}^{-1}$ must be attainable (not necessarily computed) for all combining steps. Thus, we assign a total of 12 $N_x \times N_x$ matrix maps $M_{1-8;i}$ and $C_{1-4;i}$, for each sub-matrix ϕ_i. Fig. 3(a) illustrates the mapping dependencies for the first block row and last block row during the initial combining step, i.e., forming $\phi_{1\sim 2}^{-1}$. Fig. 3(b) illustrates the mapping dependencies for the block tridiagonal portion of K^{-1}. Here, we see that the maps $M_{5-8;i}$ are used to produce intra-domain information, while the "cross" maps $C_{1-4;i}$ are used to produce inter-domain information.

Given these governing responsibilities for the mappings, we must follow the process from Fig. 2 in order to "update" the maps to their correct state during each of the hierarchical combining steps. Although the updates to these maps will mimic the procedure of [11], in this paper we must also take into account interactions between neighboring divisions. This is due

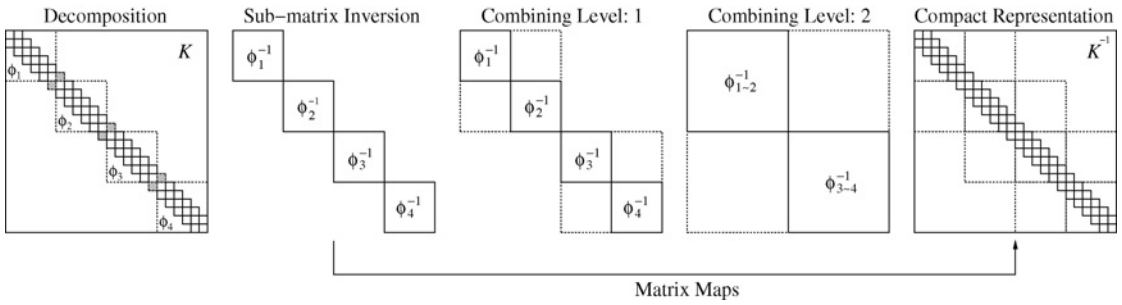

Fig. 2. Decomposition of block tridiagonal matrix K into four sub-matrices, where the shaded blocks correspond to the connectivity between domains. The two combining levels follow the individual sub-matrix inversions, where $\phi_{i\sim j}^{-1}$ represents the inverse of divisions ϕ_i through ϕ_j from the matrix K. Matrix mappings will be used to capture the combining effects and allow for the direct computation of the block tridiagonal portion of K^{-1}.

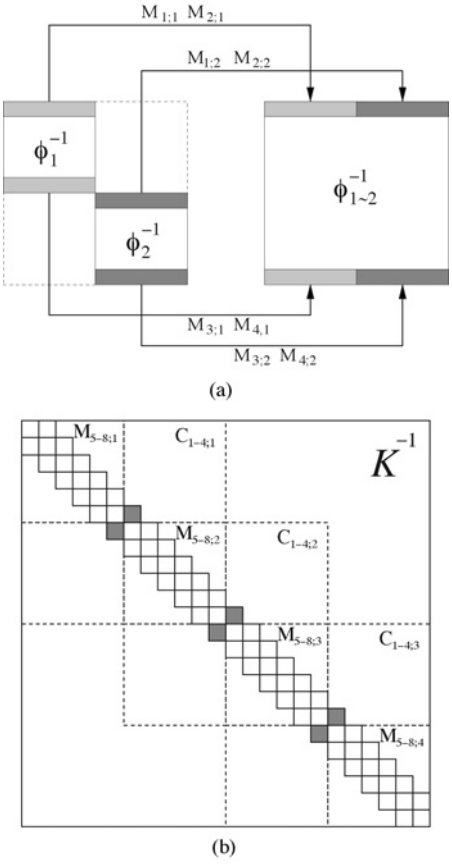

Fig. 3. Illustration of regions that can be generated through the use of matrix maps during combining process. (a) Boundary mapping dependence when combining ϕ_1^{-1} and ϕ_2^{-1} to form $\phi_{1\sim 2}^{-1}$. (b) Matrix map dependence for block tridiagonal reconstruction of K^{-1}, assuming $p = 4$.

to the inclusion of the inter-domain cross maps $C_{1-4;i}$. Under this generalized framework, matrix maps can be used to determine both the diagonal and off-diagonal block entries for K^{-1}. This subsequently allows for the computation of the ratio sequences for K^{-1}, via the relationships shown in (7), in a purely distributed fashion. The complete matrix inversion process, including the necessary update procedures, are derived in [12]. If the problem is distributed evenly across p CPUs, the time complexity of the algorithm presented is $O\left(N_x^3 N_y/p + N_x^3 \log p\right)$, with memory consumption $O\left(N_x^2 N_y/p + N_x^2\right)$.

C. Parallel Matrix–Vector Product

Given this distributed compact representation, we formulate recursions in order to compute a matrix-vector product for each step in the transient simulation, i.e., evaluate $x_k = K^{-1}c_k$ at each time step k. In [12] it is demonstrated that the elements of x_k can be found by through two sequences of vectors $\{W_l\}$ and $\{T_l\}$. The elements from each sequence can be computed through the following recursions:

$$\begin{aligned} &W_{N_y} = D_{N_y} c_{k;N_y} \quad W_l = D_l c_{k;l} + R_l W_{l+1} \quad l = N_y - 1, \ldots, 1 \\ &T_1 = S_1^T D_1 c_{k;1} \quad T_l = S_l^T (T_{l-1} + D_l c_{k;l}) \quad l = 2, \ldots, N_y - 1 \end{aligned} \tag{8}$$

where $c_{k;l}$ is the lth portion of the vector c_k (each of which has N_x elements). We can then solve for $x_{k;l} = W_{k;l} + T_{k;l-1}$, where $T_{k;0} = 0$. Although this multiplication procedure seems to be of a strictly recursive nature (and hence not readily parallel) we show that the vector x_k can in fact be computed efficiently in a distributed fashion. We begin by separating the matrix-vector into p sub-problems $x_k = K^{-1}(c_k^{(1)} + c_k^{(2)} \cdots + c_k^{(p)})$, where $c_k^{(i)} = 0$ outside the range of the sub-matrix ϕ_i. This decomposition of the matrix-vector multiply results in a set of operations that can be cascaded across the processors in a hierarchical fashion [12]. Therefore, if the problem is distributed evenly across the p CPUs, the total complexity of the process is $O(N_x^2 N_y/p + \log p N_x^2 N_y/p)$.

III. Numerical Results

There are two main categories of algorithms for solving sparse linear systems of equations: direct and iterative. In this section we will first demonstrate the advantages in scalability of direct methods for the *accurate* transient simulation of mesh structures. Here, the transient simulation time using both direct and iterative methods are compared for varying levels of reluctive coupling. Finally, the ability to trade-off computing resources for both increased mesh sizes and transient simulation time using the parallel inversion algorithm will be highlighted.

Our algorithm has been implemented, in C, and compared against standard algorithms unsymmetric multifrontal package (UMFPACK), MATLAB Sparse LU, and the conjugate gradient (CG) method using incomplete LU (ILU) preconditioner. All simulations were performed on a cluster of 32-bit 3 GHz Intel Xeon workstations with 2 GB of memory for each node. The simulations considered in this paper are for square power meshes of dimension $m \times m$. The sizes of the variables N_x and N_Y for the block tridiagonal representation of the coefficient matrix will depend on the amount of inductive coupling considered. All inductance values used as inputs for the windowing and SPAI approximation procedures were generated

TABLE II

SCALING TREND FOR RELUCTANCE APPROXIMATIONS WITH RESPECT TO THE NUMBER OF UNKNOWNS OR ''MATRIX SIZE'' AND THE NUMBER OF NON-ZEROS OR ''NNZ,'' GIVEN A MESH OF SIZE $m \times m$

m	128	192	256	384	512
Matrix Size	4.9E+04	1.1E+05	2.0E+05	4.4E+05	7.9E+05
	NNZ				
Window	4.4E+05	9.8E+05	1.8E+06	3.9E+06	7.0E+06
$\tau = 0.95$	1.4E+06	3.2E+06	5.7E+06	1.3E+07	2.3E+07
$\tau = 0.97$	3.5E+06	8.0E+06	1.4E+07	3.2E+07	5.7E+07

TABLE III

PERFORMANCE OF DIRECT ALGORITHMS ACROSS MESH SIZE

Data	Our Algorithm	LU	UMF
16×16 $\tau = 0.99$	2.01E−03	2.78E−03	2.61E−03
32×32 $\tau = 0.99$	1.45E−02	2.25E−02	1.90E−02
48×48 $\tau = 0.99$	4.43E−02	6.21E−02	6.03E−02
64×64 $\tau = 0.99$	8.73E−02	1.41E−01	1.48E−01
128×128 $\tau = 0.97$	7.29E−01	1.25E+00	7.71E−01
192×192 $\tau = 0.90$	1.50E+00	1.67E+00	–
256×256 Window	2.25E+00	–	–
384×384 Window	6.57E+00	–	–
512×512 Window	1.08E+01	–	–

Transient step solve times are shown in seconds; the lack of memory scalability for existing direct approaches is shown through an inability to perform the larger simulations.

TABLE IV

SIMULATION TIME FOR DIVIDE-AND-CONQUER ALGORITHM MESH SIZES $m = 16, 32, 48, 64$

Data	p	16×16 Total Time (s)	32×32 Total Time (s)	48×48 Total Time (s)	p	64×64 Total Time (s)
Window	1	1.86E+00	1.56E+01	5.60E+01	2	1.18E+02
Window	2	1.80E+00	1.10E+01	3.75E+01	4	7.40E+01
Window	4	–	2.71E+01	5.22E+01	8	5.22E+01
$\tau = 0.95$	1	3.70E+00	3.77E+01	1.95E+02	2	3.47E+02
$\tau = 0.97$	1	3.81E+00	3.89E+01	1.75E+02	2	3.56E+02
$\tau = 0.99$	1	4.00E+00	4.26E+01	1.85E+02	2	3.75E+02
$\tau = 0.95$	2	3.15E+00	3.00E+01	1.72E+02	4	2.13E+02
$\tau = 0.97$	2	3.18E+00	2.95E+01	1.74E+02	4	1.81E+02
$\tau = 0.99$	2	3.41E+00	3.23E+01	1.53E+02	4	2.40E+02
$\tau = 0.95$	4	–	2.36E+01	8.35E+01	8	1.34E+02
$\tau = 0.97$	4	–	2.39E+01	1.04E+02	8	1.37E+02
$\tau = 0.99$	4	–	2.50E+01	7.79E+01	8	1.54E+02

with the FastHenry extraction tool [13]. We consider V_{dd} pins to be placed at equally spaced positions throughout the grid. A random subset from the remaining nodes are considered current sinks with a square waveform triggered by a rising clock edge. All simulations consisted of 1500 time steps, given a step size of 0.1 ps and clock signal of 50 ps. All interconnects were assumed to be of uniform size: $1\,\mu\text{m} \times 2\,\mu\text{m} \times 100\,\mu\text{m}$.

A. Simulation Time

The total simulation time, given any level of inductance approximation, is dominated by the fixed time cost of inversion or factorization plus the variable time cost to multiply or solve at each time step. When considering transient simulations involving a large number of time steps, any speed-up seen in the variable time cost will dominate the fixed time cost.

1) *Comparison Against Direct Solvers:* In order to gain perspective for the computational limitations for each of the direct algorithms considered in this paper, several large-scale simulations $m = 128, 192, 256, 384$, and 512 were performed. Table II shows the size of coefficient matrix and the number of non-zero entries, considering different amounts of reluctance coupling. Table III shows the transient step solve times of the direct algorithms for these mesh sizes. It was determined that the UMFPACK algorithm was only able to handle up to a mesh size of $m = 192$ with $\tau = 0.85$. This case corresponded to a coefficient matrix with approximately 110k unknowns, but less than 412k non-zero entries. The memory consumption of the Sparse LU algorithm scaled slightly better, being able to perform the simulation for a mesh size of $m = 192$ with $\tau = 0.90$ which corresponds to over $2\times$ the number of non-zero entries as compared to $\tau = 0.85$. The divide-and-conquer method was clearly the most memory scalable of the direct algorithms, it was able to perform the largest example in this paper: $m = 512$ with window based approximation (involving 785K unknowns and 7M non-zero entries). The divide-and-conquer approach was able to perform the largest simulation $m = 512$ with window based approximation using $p = 32$ computers and the remaining cases using $p = 16$ or lower. For the case of $m = 512$, we are considering a compact representation for the inverse of the coefficient matrix, shown in (6), that would account for nearly 30 GB of memory. Next, in order to give a practical measure for the improvement of the divide-and-conquer approach we examine in more detail the effect of increasing the amount of reluctance coupling.

TABLE V

SIMULATION TIME FOR MATLAB SPARSE LU AND UMFPACK MESH SIZES $m = 16, 32, 48, 64$

	Data	16×16 Total Time (s)	32×32 Total Time (s)	48×48 Total Time (s)	64×64 Total Time (s)
LU	Window	2.33E+00	1.74E+01	4.45E+01	9.75E+01
	$\tau = 0.95$	3.58E+00	2.57E+01	8.35E+01	1.92E+02
	$\tau = 0.97$	4.23E+00	3.32E+01	1.03E+02	2.24E+02
	$\tau = 0.99$	4.28E+00	3.54E+01	1.01E+02	2.34E+02
UMF	Window	2.07E+00	1.57E+01	1.39E+02	1.31E+03
	$\tau = 0.95$	2.97E+00	1.77E+01	5.05E+01	1.02E+02
	$\tau = 0.97$	3.52E+00	2.49E+01	6.98E+01	1.54E+02
	$\tau = 0.99$	4.03E+00	2.95E+01	9.40E+01	2.31E+02

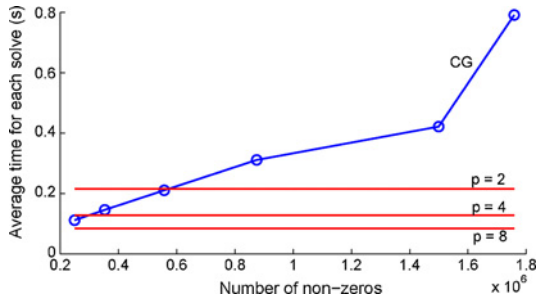

Fig. 4. Comparison of CG and divide-and-conquer approach. Average time is shown for each transient step of $m = 64$ power mesh using SPAI: $\tau = 0.94$–0.99.

Using several smaller mesh examples, $m = 16, 32, 48$, and 64, we examine the sensitivity of each algorithm to the inclusion of reluctance coupling terms, i.e., larger values of the parameter τ. Table IV shows the results for the divide-and-conquer method and Table V for the Sparse LU and UMFPACK algorithms. From Table V, first notice that although simulations using the UMFPACK algorithm with the windowing technique matched the accuracy seen across all other algorithms, the simulation time was often substantially larger than that seen using the more dense SPAI based approximation. This can be attributed to the fact that the UMFPACK algorithm was unable to properly decide on an efficient ordering for elimination given the windowing coefficient matrix.

2) *Comparison Against Iterative Solvers:* We now turn our attention to the performance of iterative methods for the transient simulation of power mesh structures. Specifically,

TABLE VI
SIMULATION TIME FOR CG USING ILU, 20 DROP TOLERANCES FROM 10^{-1} THROUGH 10^{-20}, FOR MESH SIZES m = 16, 32, 48, 64

Data	16×16 Total Time (s)	32×32 Total Time (s)	48×48 Total Time (s)	64×64 Total Time (s)	Avg # Iter
Window	5.94E+00	2.84E+01	7.49E+01	1.41E+02	1.51
τ = 0.95	1.43E+01	7.22E+01	2.20E+02	3.76E+02	1.54
τ = 0.97	2.86E+01	1.35E+02	4.10E+02	6.87E+02	1.50
τ = 0.99	6.19E+01	3.88E+02	7.88E+02	1.49E+03	1.50

we analyze the lack of scalability for the CG algorithm with respect to the addition of reluctance coupling. Although the CG method has the smallest memory consumption of any algorithm considered in this paper, we observe that the iterative CG method using ILU scaled the worst with respect to the inclusion of reluctance coupling. A comparison of transient step solve times for the example $m = 64$ are shown in Fig. 4. It is important to note that the times for the divide-and-conquer method do not change as all terms involved in the parallel matrix-vector multiply are dense. This property does not hold for any other algorithm considered in this paper.

Table VI shows the sensitivity of the CG algorithm to the inclusion of additional reluctance coupling terms, again using several smaller mesh examples. On average the time for CG was more than 11× slower when comparing the SPAI approximation with $\tau = 0.99$ to the basic windowing approach. If we use this fact, we can arrive at speed-up factors when comparing the dominant computational task for transient simulation. Specifically, if we consider the maximum scaling of the divide-and-conquer method for the cases $m = 256, 384,$ and 512, we calculate speed-up factors of 8.9×, 7.2×, and 9.2×, respectively, when considering transient solve times for a $\tau = 0.99$ quantity of reluctance coupling.

IV. CONCLUSION

Currently employed techniques for the simulation of mesh structures attempt to address the issue of increased problem sizes by trading off accuracy for simulation time via iterative schemes. Our algorithm is a direct solver that facilitates the simulation of large mesh structures through a divide-and-conquer approach. Due to the inherently parallel nature of the algorithm, computing resources can be flexibly allocated toward either speeding up the simulation of a problem of a given size, or solving problems of larger sizes in comparable time. The scalability of the divide-and-conquer method is clearly demonstrated by the absence of increases to time for the primary computational task for transient simulation, when considering the inclusion of these additional coupling terms. This attribute is not shared by any of the other methods analyzed in this paper. In addition, the divide-and-conquer method was able to show substantial computational improvement over the most widely used numerical techniques applicable for these large-scale simulations. Specifically, the divide-and-conquer approach allows for the the simulation of a 512×512 RLC mesh with a speed-up factor over 9× when compared to the CG method with ILU. Therefore, we conclude that the divide-and-conquer algorithm presented here offers a framework which can be built upon for the large-scale accurate simulation of power mesh structures.

REFERENCES

[1] H. Qian, S. R. Nassif, and S. S. Sapatnekar, "Power grid analysis using random walks," *IEEE Trans. Comput.-Aided Design Int. Circuits Syst.*, vol. 24, no. 8, pp. 1204–1224, Aug. 2005.

[2] K. Sun, Q. Zhou, K. Mohanram, and D. C. Sorensen, "Parallel domain decomposition for simulation of large-scale power grids," in *Proc. Int. Conf. Comput.-Aided Design*, 2007, pp. 54–59.

[3] G. Weikun, S. X.-D. Tan, Z. Luo, and X. Hong, "Partial random walk for large linear network analysis," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2004, pp. 173–176.

[4] M. Zhao, R. V. Panda, S. S. Sapatnekar, and D. Blaauw, "Hierarchical analysis of power distribution networks," *IEEE Trans. Comput.-Aided Design Int. Circuits Syst.*, vol. 21, no. 2, pp. 159–168, Feb. 2002.

[5] Y. Zhong and M. Wong, "Fast algorithms for IR drop analysis in large power grid," in *Proc. Int. Conf. Comput.-Aided Design*, 2005, pp. 351–357.

[6] J. Kozhaya, S. R. Nassif, and F. N. Najm, "A multigrid-like technique for power grid analysis," *IEEE Trans. Comput.-Aided Design Int. Circuits Syst.*, vol. 21, no. 10, pp. 1148–1160, Oct. 2002.

[7] H. Qian, S. R. Nassif, and S. S. Sapatnekar, "A hybrid linear equation solver and its application in quadratic placement," in *Proc. Int. Conf. Comput.-Aided Design*, 2005, pp. 905–909.

[8] T. H. Chen, C. Luk, H. Kim, and C. C.-P. Chen, "INDUCTWISE: Inductance-wise interconnect simulator and extractor," in *Proc. Int. Conf. Comput.-Aided Design*, 2002, pp. 215–220.

[9] G. Zhong, C.-K. Koh, and K. Roy, "On-chip interconnect modeling by wire duplication," in *Proc. Int. Conf. Comput.-Aided Design*, 2002, pp. 341–346.

[10] M. Grote and T. Huckle, "Parallel preconditioning with sparse approximate inverses," *Soc. Ind. Appl. Math. J. Sci. Comput.*, vol. 18, no. 3, pp. 838–853, 1997.

[11] S. Cauley, J. Jain, C.-K. Koh, and V. Balakrishnan, "A scalable distributed method for quantum-scale device simulation," *J. Appl. Phys.*, vol. 101, no. 123715, 2007.

[12] S. Cauley, C.-K. Koh, and V. Balakrishnan, "A parallel direct solver for the simulation of large-scale power/ground networks," Purdue Elect. and Comput. Eng. Tech. Rep. TR-ECE-09-12, 7 Dec. 2009.

[13] M. Kamon, M. J. Tsuk, and J. White, "FastHenry: A multipole-accelerated 3-D inductance extraction program," *IEEE Trans. Microw. Theory Tech.*, vol. 42, no. 9, pp. 1750–1758, Sep. 1994.

A Functional Unit and Register Binding Algorithm for Interconnect Reduction

Taemin Kim and Xun Liu

***Abstract*—This paper describes a simultaneous register and functional unit (FU) binding algorithm in high level synthesis. Our algorithm targets the reduction of multiplexer inputs, shortening the total length of global interconnects. Specifically, our algorithm maximizes the interconnect sharing among FUs and registers by considering flow dependences, common primary inputs, and common register inputs among operations. Experimental results have shown that our scheme achieves more than 20% multiplexer input count reduction, on average, over previously proposed algorithms. Our approach delivers a 18% wirelength reduction of global interconnects with minor area overhead.**

***Index Terms*—DSP synthesis, high level synthesis, interconnect, resource binding, synthesis.**

Manuscript received April 7, 2009; revised August 11, 2009. Current version published March 19, 2010. This paper was recommended by Associate Editor, R. Camposano.

T. Kim is with the Department of Computer Science, University of California, Los Angeles, CA 90095 USA (e-mail: tmkim76@gmail.com).

X. Liu is with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695 USA (e-mail: xunliu@unity.ncsu.edu).

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

Digital Object Identifier 10.1109/TCAD.2010.2042903