January 2010

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

S. Cauley

V. Balakrishnan

Koh Cheng-Kok

Follow this and additional works at: http://docs.lib.purdue.edu/ecepubs

http://dx.doi.org/http://dx.doi.org/10.1109/TCAD.2010.2042901
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 0 1
Vsense 9 4 DC 0V
Fcopy 0 8 Vsense 1
Rstep 8 0 1K
Rser 2 4 10
Pnum 6 0 POLY(2) Vsense
+Ecopp +0.5K-10 0 1K-10 0 -1 0 0 0 1
Cmem 6 0 90nF
Rap 6 0 10000Meg
Ecopp 7 0 0 6 1
Kc 7 0 0 1
Ecopp2 10 0 6 0 1
Vref ref 0 DC 1V
R1 10 11 100K
Saat1 11 0 0 11 SNX
Saat2 11 ref 11 ref SNX
.MODEL SNX PM(Von=0.001, Rooff=10000Meg,
+Vt=0.0001Y, VM=0.00001V)
.ENDS

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

Acknowledgment

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

REFERENCES


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/groud mesh structures. Our method is a direct (noniterative) 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 relucance 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 power and 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 power and ground networks.
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 modulated nodal analysis representation yields equations of the form

\[ \mathbf{G}x + \mathbf{C}x = \mathbf{B} \]  

(1)

where

\[ \mathbf{G} = \begin{pmatrix} G & A_{t}^T \\ -A_t & 0 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} C & 0 \\ 0 & L \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} \mathbf{v}_n \\ \mathbf{i}_n \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{A}_{t}^T L \end{pmatrix} \]

\[ \mathbf{A}_{t} = \mathbf{A}_t^T \mathbf{R}^{-1} \mathbf{A}_t \quad \text{and} \quad \mathbf{C} = \mathbf{A}_t^T \mathbf{C} \mathbf{A}_t. \]

Here, \( R, C, \) and \( L \) are the resistance, capacitance, and inductance matrices, respectively. The matrices \( A_t \) and \( A_{t} \) transform the conductances and capacitances into node based relationships. The matrices \( A_t \) and \( A_{t} \) link the node voltages and branch currents described by the state variable \( s \). In addition, \( I_i \) is the current vector that dictates, through the matrix \( A_t \), 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 \( v_{n}^i = v_{n}^i(h) \) to denote the voltage at the \( n \)th node, we may then solve for \( v_{n}^{i+1} \)

in terms of \( v_{n}^i \) through the nodal analysis equations \[ \left( G + \frac{2}{h} C + \frac{1}{h^2} S \right) v_{n}^{i+1} = \left( -G + \frac{2}{h} C - \frac{1}{h^2} S \right) v_{n}^i + A_t^T \left( h v_{n}^{i+1} + I_i \right) - 2 A_t^T \delta_l \]

(2)

where \( S = A_t^T L^{-1} A_t \). 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 \( R^{-1} \) 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

\[ Y = \begin{pmatrix} A_1 & -B_1 & 0 & \cdots & 0 \\ -B_{1}^T & A_2 & -B_2 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & -B_{N_i-1}^T & A_{N_i-1} & -B_{N_i-1} \\ 0 & \cdots & 0 & -B_{N_i}^T & A_{N_i} \end{pmatrix} \]

(3)

where each \( A_i, B_i \in \mathbb{R}^{N_i \times N_i} \). Thus \( Y \in \mathbb{R}^{N \times N \times N_i \times N_i} \), with \( N_i \) diagonal blocks of size \( N_i \) 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 (SPIA) [10]. Given an inductance matrix \( L \), the SPIA method can be used to form another matrix \( \hat{M} \) that is constructed in an attempt to match the inverse of the inductance matrix under the Frobenius norm

\[ \| \hat{M} \mathbf{v} - \mathbf{1} \|_F^2 = \sum_{i=1}^n \| (\hat{M} - I) \mathbf{e}_i \|_2^2 \]

(4)

where \( n \) is the number of columns of \( L \) and \( e_i \) is the \( i \)th euclidean basis vector. Therefore, we can solve \( n \) independent
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 $|M_{ij}| > (1 - \tau) \text{max}_{||} |M_{ij}|$ (5) where the diagonal entries $M_{ii}$ are always included. If $\tau$ 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 $\tau$. 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

$$K^{-1} = \left[ \begin{array}{cccc} D_1 & D_1 S_1 & \cdots & D_1 \prod_{j=1}^{p-1} S_j \\ R_1 D_1 & D_2 & \cdots & D_2 \prod_{j=1}^{p-2} S_j \\ \vdots & \ddots & \ddots & \vdots \\ \prod_{i=1}^{p-1} R_i D_{p-1} & \prod_{i=1}^{p-2} R_i D_{p-2} & \cdots & D_p \\ \end{array} \right]$$

(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 $O(N^2 N_x)$, with a memory requirement of $O(N^2 N_x)$.

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:

$$D_i S_i = Z_i \Rightarrow S_i = D_i^{-1} Z_i \quad i = 1, \ldots, N_y - 1$$

(7)

where $Z_i$ denotes the $i$th 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 form 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, $\phi_{i+1}^{-1}$ represents the result of any combining step through the use of the matrix inversion lemma. For example, $\phi_0^{-1}$ is the inverse of a matrix comprised of the blocks assigned to both $\phi_0$ and $\phi_1$. 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: $(N_x N_y)/2$. 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 $\phi_i^{-1}$ must be attainable (not necessarily computed) for all combining steps. Thus, we assign a total of $N_x N_y$ matrix maps $M_{1:p}$ and $C_{1:p}$, for each sub-matrix $\phi_i$. Fig. 3(a) illustrates the mapping dependencies for the first block row and last block row during the initial combining step, i.e., forming $\phi_0^{-1}$. Fig. 3(b) illustrates the mapping dependencies for the block tridiagonal portion of $K^{-1}$. Here, we see that the maps $M_{1:p}$ and $C_{1:p}$ are used to produce intra-domain information, while the “cross” maps $C_{1:p}$ 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 combing 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
CAULEY et al.: A PARALLEL DIRECT SOLVER FOR THE SIMULATION OF LARGE-SCALE POWER/GROUND NETWORKS 639

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 resistive coupling. Finally, the ability to trade-off computing resources for both increased mesh sizes and transient simulation structures. Here, the transient simulation time using both direct and iterative methods are compared for varying levels of resistive coupling. Finally, the ability to trade-off computing resources for both increased mesh sizes and transient simulation structures. Here, the transient simulation time using both direct and iterative methods are compared for varying levels of resistive coupling. Finally, the ability to trade-off computing resources for both increased mesh sizes and transient simulation structures.

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 32-3 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 x m. The sizes of the variables N, and Np 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 SPAC approximation procedures were generated.
TABLE II  
SCALING TREND FOR RELUCTANCE APPROXIMATIONS WITH RESPECT TO THE NUMBER OF UNKNOBNS OR "NNZ." GIVEN A MESH OF SIZE m × m

<table>
<thead>
<tr>
<th>m × m</th>
<th>128</th>
<th>256</th>
<th>384</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>τ = 0.05</td>
<td>1.4E-06</td>
<td>3.1E-06</td>
<td>5.7E-06</td>
<td>1.1E-05</td>
</tr>
<tr>
<td>τ = 0.07</td>
<td>3.1E-06</td>
<td>6.3E-06</td>
<td>1.2E-05</td>
<td>2.3E-05</td>
</tr>
</tbody>
</table>

TABLE III  
PERFORMANCE OF DIRECT ALGORITHMS ACROSS MESH SIZE

<table>
<thead>
<tr>
<th>Mesh Size</th>
<th>16</th>
<th>32</th>
<th>48</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window</td>
<td>01</td>
<td>02</td>
<td>02</td>
<td>03</td>
</tr>
<tr>
<td>Time (s)</td>
<td>1.86E+00</td>
<td>3.32E+01</td>
<td>8.73E+00</td>
<td>3.54E+01</td>
</tr>
<tr>
<td>τ = 0.05</td>
<td>2.0E+05</td>
<td>3.89E+01</td>
<td>8.35E+01</td>
<td>2.40E+02</td>
</tr>
<tr>
<td>τ = 0.07</td>
<td>1.54E+02</td>
<td>1.74E+02</td>
<td>1.74E+02</td>
<td>1.54E+02</td>
</tr>
</tbody>
</table>

A. Simulation Time

The total simulation time, given any level of inductive 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 τ = 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 τ = 0.90 which corresponds to over 2x the number of non-zero entries as compared to τ = 0.85. The divide-and-conquer approach was able to perform the largest simulation 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.

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,
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 $m = 64$ is 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 $11 \times$ 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 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 simulation of a $512 \times 512$ RLC mesh with a speed-up factor over $9 \times$ 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 high accuracy, large-scale simulation of power mesh structures.

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: xuliu@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

TABLE VI

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>$m$</th>
<th>$T_{time}$ (s)</th>
<th>$Avg # Iter$</th>
<th>$Speed-up$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.10E-01</td>
<td>1.10E-01</td>
<td>1.10E-01</td>
<td>1.10E-01</td>
<td>1.10E-01</td>
</tr>
<tr>
<td>2.20E-02</td>
<td>2.20E-02</td>
<td>2.20E-02</td>
<td>2.20E-02</td>
<td>2.20E-02</td>
</tr>
<tr>
<td>3.30E-03</td>
<td>3.30E-03</td>
<td>3.30E-03</td>
<td>3.30E-03</td>
<td>3.30E-03</td>
</tr>
</tbody>
</table>

REFERENCES


