Symbolic Moment Computation for Statistical Analysis of Large Interconnect Networks

Zhigang Hao, *Student Member, IEEE*, Guoyong Shi, *Senior Member, IEEE*, Sheldon X.-D. Tan, *Senior Member, IEEE*, and Esteban Tlelo-Cuautle, *Senior Member, IEEE*

***Abstract*— The shrinking technology feature size and dense large-scale integration make process variation a challenging issue directly confronting the latest design automation tools. Process variation causes severe variation in interconnect networks, including very large-scale integrated interconnect structures, such as clock trees, clock mesh, power-ground networks, and other wiring structures in 3-D integrated circuits. The traditional moment computation techniques are only partly useful for analyzing such variational problems, however, their computational efficiency cannot meet the quickly rising needs, such as statistical analysis. This paper presents a novel symbolic moment calculator (SMC) for variational interconnect analysis. The moment calculator is constructed in a regular data structure that incorporates binary decision diagrams for data storage and computation. Given an interconnect circuit, such a computation diagram has to be constructed only once and can be repeatedly invoked for computation of moments with varying parameter values. Also, the SMC is friendly to interconnect synthesis in that it can be incrementally modified according to the modifications made to the circuit structure. Applications of the SMC for fast moment computation, sensitivity analysis, and statistical timing analysis are addressed. Significant efficiency is demonstrated comparing to other existing methods.**

***Index Terms*— Binary decision diagram (BDD), clock mesh, incremental analysis, moment sensitivity, process variations, statistical analysis, symbolic moment.**

I. Introduction

THE progressively shrinking feature dimension of the latest integrated circuit (IC) technology has brought the process variation issue to the front-most scene. Process variation causes lots of issues in IC design and manufacturing that used to be ignorable. Modern manufacturing technology makes possible massive systems-on-chip integration, on which the most complicated structure is the on-chip interconnect system. Before the next generation network-on-chip technology becomes prevalent, the traditional metal interconnection system would remain dominant in the near-future manufacturing technology. Process variation has made the analysis and synthesis of the interconnection systems tougher by using the existing design automation tools.

Since the beginning of 1990's, circuit moment computation and its derivatives have been introduced for massive linear interconnect network analysis [1], [2]. A few years later, a swarm of publications making use of moments for reduced-order modeling of massive interconnect systems have appeared. Now, moment-based analysis has become a de facto standard tool for a variety of circuit analysis tasks in place of accurate SPICE simulation. As the variation issues arise in more advanced fabrication technologies, another category of publications have appeared addressing numerical moment computation techniques for variational analysis. Representative techniques include: 1) interval algebra in reduced-order modeling [3]; 2) symbolic model order reduction [4], [5]; 3) variational asymptotic waveform evaluation model by adjoint sensitivity [6], [7]; 4) moment methods for *statistical timing analysis* [8]–[10]; and 5) moment methods for *crosstalk noise analysis* [11], just to mention a few. Apart from the works based on moment computation, there exist other works applying implicit or explicit parametric moment matching for variational interconnect analysis. Techniques along this line are: 1) parameterized interconnect models using multiparameter moment-matching [12]; 2) parameterized model order reduction via a 2-D Arnoldi process [13]; and 3) parameterized interconnect order reduction with multiparameter moment matching [14], among others.

An interconnect network can take a variety of forms, being simple RC tree, capacitively and/or inductively coupled RLC trees or mesh networks involving resistive links and driven by multiple independent sources. These interconnection structures cover most interconnect networks in practice. The main objective of this paper is to develop a generic symbolic moment calculator (SMC) that is applicable to the variety of interconnect configurations.

Symbolic circuit analysis is traditionally considered a technique of academic interest because most of the existing techniques in the literature are based on *enumeration*. However, it is partly the purpose of this paper to demonstrate that the notion of *symbolic moment calculation* does not rely on term enumeration at all. Instead, the symbolic moments can be calculated from a computational structure directly

Manuscript received July 11, 2011; revised December 14, 2011; accepted April 12, 2012. Date of publication May 23, 2012; date of current version April 22, 2013. The work of Z. Hao and G. Shi was supported in part by the National Natural Science Foundation of China (NSFC), under Grant 60876089 and Grant 61176129, and the Doctoral Education Program from the Ministry of Education of China, under Grant 20090073110056. The work of S. X.-D. Tan was supported in part by the NSFC, under Grant 60828008 and the NSF under Grant CCF-1116882 and Grant OISE-1130402.

Z. Hao and G. Shi are with the School of Microelectronics, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zhigang.g.hao@gmail.com; shiguoyong@ic.sjtu.edu.cn).

S. X.-D. Tan is with the Department of Electrical Engineering, University of California, Riverside, CA 92521 USA (e-mail: stan@ee.ucr.edu).

E. Tlelo-Cuautle is with INAOE, Tonantzintla 72840, Mexico (e-mail: e.tlelo@ieee.org).

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/TVLSI.2012.2197835

constructed by a process of circuit decomposition whose computational complexity is of cubic polynomial. This substantial improvement in efficiency enables the possibility of applying the SMC for large-scale interconnect network synthesis.

The rest of this paper will present progressively the basic principles employed in symbolic moment computation. Section II reviews some well-established recursive moment formulas and reformulates them into binary decision diagram (BDD) data structure. Section III addresses tree networks with resistive links and multiple driving sources by applying the Kron's branch tearing technique. Some selected applications of the SMC are presented in Section IV. This paper is concluded in Section V.

Some preliminary results of this paper have been presented in [15]–[17]. The main contribution of this paper is to streamline the previously developed incomplete results and formulate a unified moment computation methodology in a single article. In addition, more validating evidences are provided in the experimental section.

II. BDD Formulation of Recursive Moment Computations

Linear circuits can be described by the following general linear differential equation in state-space form:

$$\mathbf{E}\frac{d\mathbf{x}}{dt} + \mathbf{A}\mathbf{x} = \mathbf{B}\mathbf{u} \tag{1}$$

$$\mathbf{y} = \mathbf{F}\mathbf{x} \tag{2}$$

where $\mathbf{E}$ and $\mathbf{A}$ are susceptance and conductance matrices, $\mathbf{B}$ and $\mathbf{F}$ are the selection matrices consisting of elements like 1, -1, and 0, $\mathbf{u}$ is a vector standing for the input voltage/current sources, $\mathbf{x}$ is the circuit state vector, including nodal voltages and branch currents, and $\mathbf{y}$ is the output vector. The frequency-domain transfer function of the above system from the input $\mathbf{u}$ to the output $\mathbf{y}$ is $H(s) = \mathbf{F}(\mathbf{E}s + \mathbf{A})^{-1}\mathbf{B}$. The moment analysis technique arises from the expansion of $H(s)$ in a series of s^i

$$H(s) = \sum_{k=0}^{\infty} (-1)^k m_k s^k = m_0 s^0 - m_1 s^1 + m_2 s^2 - m_3 s^3 + \cdots \tag{3}$$

where the coefficients m_k's are called the *moments* [1]. Since certain electrical information of a network can be derived from the moments, efficient moment computation procedures are of great interest in applications.

Traditionally, the moments are computed recursively by the following relations (assuming $\mathbf{A}$ is invertible):

$$\mu_0 = \mathbf{A}^{-1}\mathbf{B}, \quad \mu_k = \left(\mathbf{A}^{-1}\mathbf{E}\right)\mu_{k-1} \tag{4}$$

where $\mu_k, k = 0, 1, 2, \ldots$, are called the *state moments*. The input–output moments m_k are obtained from $m_k = \mathbf{F}\mu_k$. It is apparent that computation of the moments requires solving the inversion of $\mathbf{A}$, which is equivalent to a DC solution of the linear network given appropriate sources. It is well known that symbolically solving $\mathbf{A}^{-1}$ is of exponential complexity for a general matrix $\mathbf{A}$. Hence, it is generally not recommended to compute symbolic moments by inverting matrix A symbolically.

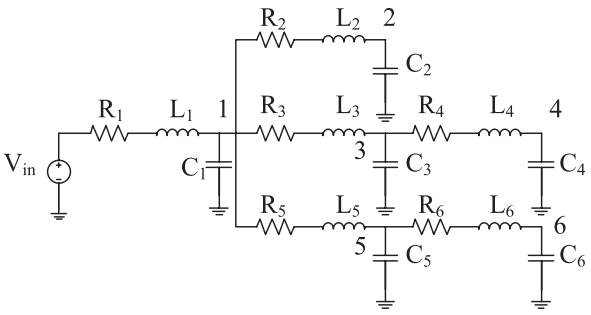


Fig. 1. RLC tree.

However, it has been recognized in the literature that moments of those special linear networks, such as tree networks or capacitively and inductively coupled tree networks etc., can be computed without solving matrices. For a tree-structured RLC network shown in Fig. 1, where every serial R-L branch has a connection to the ground by a capacitor (called a grounding capacitor), its state-space model is of the form in (1) and (2) where the coefficient matrices take the following forms:

$$\mathbf{E} = \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} 0 & A \\ -A^T & R \end{bmatrix} \quad \mathbf{B} = -\begin{bmatrix} 0 \\ I_N \end{bmatrix} \quad \mathbf{F} = [I_N, \quad 0]$$

where C and L are diagonal matrices comprised of the capacitances C_i and inductances L_i, respectively, and R is also a diagonal matrix comprised of the resistances R_i. The state vector $\mathbf{x}$ contains the nodal voltages as the first half for the nodes grounded via the capacitors, and contains the branch currents as the second half for the currents flowing through the serial R-L branches. N is the total number of nodes grounded by the C_i's (also equal to the total number of resistors and inductors). The nodal voltages are considered as the outputs. The matrix A is the incidence matrix of the serial R-L branches and the matrix I_N is the N-dimensional identity matrix. The input source vector $\mathbf{u}$ includes the independent current sources connected across the grounding capacitors.

A. Recursive Moment Calculation for Tree Circuits

For an RLC tree as given in Fig. 1, by node i we always refer to the node where the grounding capacitor C_i is connected. Let the tree be driven by a constant voltage $V_{in} \equiv 1V$ at the input. Let $m_{i,k}$ be the kth order output (*voltage*) moment at node i. It is known in the literature that any order moment at any node i (called *nodal voltage moment*) can be computed recursively by the following formulas [2], [18]:

$$m_{i,k} = \sum_{R_\ell \in P_i} R_\ell \sum_{j \in T_\ell} C_j \cdot m_{j,k-1} - \sum_{L_\ell \in P_i} L_\ell \sum_{j \in T_\ell} C_j \cdot m_{j,k-2} \tag{5}$$

for $k \geq 2$, where P_i represents the path from the tree root to node i, the summation index $R_\ell \in P_i$ means summing over all the resistors R_ℓ on the path P_i, T_ℓ denotes the subtree rooted at node ℓ, and the summation index $j \in T_\ell$ indicates

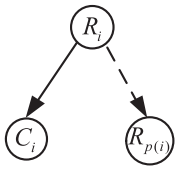


Fig. 2. Illustration of a BDD triple.

the summation over all nodes belonging to the subtree T_ℓ (including the root of T_ℓ). The initial two orders of moments are $m_{i,0} = 1$ for all i and

$$m_{i,1} = \sum_{R_\ell \in P_i} R_\ell \sum_{j \in T_\ell} C_j \tag{6}$$

for all i. Note that the first-order nodal moments $m_{i,1}$ are the Elmore delays from the input to the node i for all i.

We see that formula (5) involves two summation terms of the same form $\sum_{j \in T_\ell} C_j \cdot m_{j,k-1}$ with the two moment orders $(k-1)$ and $(k-2)$. For convenience, we denote such a summation term by

$$m^C_{\ell,k} = \sum_{j \in T_\ell} C_j \cdot m_{j,k-1} \tag{7}$$

and call it the *capacitor moment* because the C_j's are multiplied. Then the moment recursion (5) becomes

$$m_{i,k} = \sum_{R_\ell \in P_i} R_\ell \cdot m^C_{\ell,k} - \sum_{L_\ell \in P_i} L_\ell \cdot m^C_{\ell,k-1} \tag{8}$$

where both summations follow the path P_i from the tree root to node i. The circuit interpretation of $m^C_{\ell,k}$ is a "current" entering node i, which is the summation of currents of magnitudes $C_j \cdot m_{j,k-1}$ distributed at all capacitors C_i in the subtree T_ℓ. The term $\sum_{L_\ell \in P_i} L_\ell \cdot m^C_{\ell,k-1}$ is interpreted as the summation of voltage sources "generated" by the inductors along the path P_i. Hence, (8) is simply the Kirchhoff voltage law applied to the path from input to the observing point i with the driving voltage V_{in} shorted.

The zeroth-order moment is the DC solution of the original circuit. If there is an independent current source connecting to an internal node, the capacitor moment in (7) at that node is replaced by the value of the current source, then the zeroth-order moments of all nodes are calculated via (8). The independent sources do not appear again in computing the higher orders of moments.

It is interesting to note that a tree is a successively branching structure in that some computations propagating from the root to the tree terminals can be *shared*. Let $p(i)$ be the parent node of node i [i.e., node i is a fanout of node $p(i)$]. Then the moments computed from the tree root up to node $p(i)$ can be used for the computation of all moments at the nodes fanned out from node $p(i)$. Written in equation, the recursion formula (8) becomes

$$m_{i,k} = m_{p(i),k} + R_i \cdot m^C_{i,k} - L_i \cdot m^C_{i,k-1} \tag{9}$$

which means that the computation of all moments at any node i fanned out from the node $p(i)$ can reuse (or share) the moment $m_{p(i),k}$ which is computed at $p(i)$.

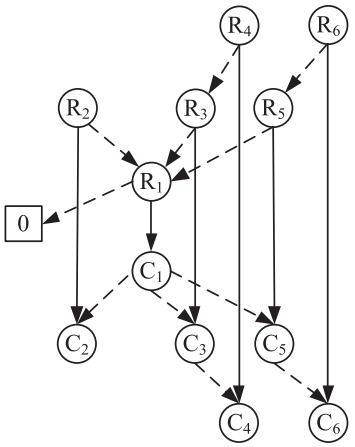


Fig. 3. BDD-tree.

Equation (9) can be decomposed into two parts

$$m^R_{i,k} := m^R_{p(i),k} + R_i \cdot m^C_{i,k} \tag{10}$$

$$m^L_{i,k} := m^L_{p(i),k} - L_i \cdot m^C_{i,k-1} \tag{11}$$

where both right-hand sides consist of one multiplication and one addition (subtraction). All such expressions can be graphically computed by three nodes connected as in Fig. 2, where the solid arrow indicates a multiplication and a dashed arrow indicates an addition. When computing the right-hand side of (11), we just substitute the top node R_i by $-L_i$. Putting together the two triple-node calculations gives the moment $m_{i,k} = m^R_{i,k} + m^L_{i,k}$.

The basic computation involving three graphical vertices (Fig. 2) can be programmed using a BDD [19], [20]. The BDD is different from a binary tree in that sharable subtree are never created twice. This property perfectly matches what we stated above in tree circuit moments where the moment at a parent node can be reused by the fanout nodal moments. Therefore, as the first contribution of this paper, we formulate a BDD-based moment computation scheme for tree circuits, which becomes a good foundation when we later deal with other more general interconnect networks.

The BDD comprising of the R_i vertices is hereafter referred to as an R-BDD. An analogous BDD that computes the terms involving $-L_i$ as given in (11) is hereafter referred to as an L-BDD.

Expression (7) shows that the capacitor moments $m^C_{\ell,k}$ are computed from the corresponding subtrees. Hence, it is convenient to implement a tree data structure topologically identical to the tree circuit, which saves all such capacitor moments in the tree nodes labeled by the C_i's. A bottom-up tree traversal would compute all capacitor moments. Such a data structure is referred to a *C-tree* hereafter.

Combining a *C-tree* with an R-BDD (and an L-BDD) forms a computational diagram illustrated in Fig. 3, where the bottom part consisting of the C_i nodes is the *C-tree* and the upper part containing the R_i nodes is the R-BDD. Each R_i node in the R-BDD has a dashed arrow pointing to its parent node $R_{p(i)}$ (for addition) and a solid arrow pointing at the C_i node in the *C-tree* (for multiplication). We point out that topologically the R-BDD has the

identical structure as the *C-tree* except for the opposite directions of the arrows and the algebraic meanings of the arrows. A computational diagram consisting of an $R(L)$-BDD and a *C-tree* is hereafter called a BDD-*C-tree* diagram.

The BDD-*C-tree* diagram as illustrated in Fig. 3 can recursively compute all triplets expressed in (10) and (11) by *cyclical* traversals. Such a diagram is called the SMC in this paper. The successive computation is executed as follows. The zeroth-order moment vector $m_0 = (m_{1,0}, \ldots, m_{N,0})^T$ is the DC solution of the tree circuit driven by unit independent sources. The first-order moment vector $m_1 = (m_{1,1}, \ldots, m_{N,1})^T$ is computed by traversing the SMC bottom-up with the *C-tree* nodes set to the values of $m^C_{i,1}$ (viewed as currents) and compute the values in the R-BDD vertices from bottom-up. The higher order moments $m_{i,k}$ $(k \geq 2)$ then are computed by repetition: the R-moments $m^R_{i,k}$ are computed by setting the *C-tree* nodes to the values of $m^C_{i,k}$ and traversing the R-BDD bottom-up, while the L-moments $m^L_{i,k}$ are computed by setting the *C-tree* nodes to the values of $m^C_{i,k-1}$ and traversing the L-BDD bottom-up. Adding the moments at the corresponding R-BDD vertices and L-BDD vertices produces the moments $m_{i,k}$ for $i = 1, \ldots, N$. If the tree network does not contain any inductors, i.e., an RC tree, the computation involving the L-BDD as stated above can be omitted.

For a tree network containing N branches of RLC segments, the memory complexity of constructing a SMC is $O(N)$. The time complexity for computing one order of moments is also $O(N)$. If the pth order moments are requested, the computation time complexity is $O(Np)$.

B. Recursive Moment Calculation for Coupled Trees

Signal integrity is a highly concerned issue in the current digital IC design practice. One general method for the signal integrity analysis is by modeling a set of interconnect circuits connected in the form of capacitively and inductively coupled RLC trees. A circuit connected in this form also can be analyzed by moments, where it is again valid to compute the moments recursively by traversing "coupled" BDD-*C-tree* diagrams. The basic principle for recursive moment computation of coupled tree networks was described in [21]–[23]. As before, the coupling capacitors are equivalent to *current sources* while the coupling inductors equivalent to *voltage sources*. Hence, in essence the moment computation of capacitively and inductively coupled tree circuits does not have drastic change in computational structures except for taking into account of more sources connected to the respective computational nodes.

Each individual tree in a set of coupled trees is labeled by a Greek superscript α, β, etc. Hence, T^α stands for the αth tree. Node n^α_j is the jth node in the tree T^α, whereas before a node refers to a tree intersection to which a grounding capacitor is connected. P^α_j refers to the path from the root of tree T^α (denoted $root(\alpha)$) to node n^α_j. A coupling capacitor connected between nodes n^α_i and n^β_j is denoted by $C^{\alpha,\beta}_{i,j}$. If two inductors L^α_i and L^β_j existing in two separated trees T^α and T^β are coupled, then the coupling is modeled by a mutual inductance $M^{\alpha,\beta}_{i,j} = K^{\alpha,\beta}_{i,j}\sqrt{L^\alpha_i \cdot L^\beta_j}$, where $K^{\alpha,\beta}_{i,j}$ is the mutual inductance coefficient. Fig. 4 shows an example of two coupled RLC trees.

The nodal voltage moments for trees with coupling can be expressed by the following recursive formulas similar to what we stated earlier in (8) and (7) [22]:

$$m^\alpha_{j,k} = \sum_{R^\alpha_\ell \in P^\alpha_j} R^\alpha_\ell \cdot m^{C,\alpha}_{\ell,k} - \sum_{L^\alpha_\ell \in P^\alpha_j} \times \left(L^\alpha_\ell \cdot m^{C,\alpha}_{\ell,k-1} + \sum_{L^\beta_{\ell'} \in \mathcal{L}^\alpha_\ell} M^{\alpha,\beta}_{\ell,\ell'} \cdot m^{C,\beta}_{\ell',k-1} \right) \quad (12)$$

$$m^{C,\alpha}_{j,k} = \sum_{n^\alpha_\ell \in T^\alpha_j} C^\alpha_\ell \cdot m^\alpha_{\ell,k-1} + \sum_{n^\alpha_\ell \in T^\alpha_j} \sum_{C^\beta_{\ell'} \in \mathcal{C}^\alpha_\ell} C^{\alpha,\beta}_{\ell,\ell'} \times \left(m^\alpha_{\ell,k-1} - m^\beta_{\ell',k-1} \right) \quad (13)$$

where $m^\alpha_{j,k}$ denotes the kth order nodal moment at node n^α_j and $m^{C,\alpha}_{j,k}$ denotes the kth order *capacitor moment* at node n^α_j. The notation $\mathcal{C}^\alpha_\ell$ in the summation index denotes the set of coupling capacitors connected at node n^α_ℓ and the notation $\mathcal{L}^\alpha_\ell$ denotes the set of inductors mutually coupled with the inductor L^α_ℓ. The other summation indices are self-evident. Because the physical meaning of moments $m^\alpha_{\ell,k-1}$ is voltage, the term $(m^\alpha_{\ell,k-1} - m^\beta_{\ell',k-1})$ in (13) is just the voltage difference across the coupling capacitor $C^{\alpha,\beta}_{\ell,\ell'}$, which generates a current from the node $n^\beta_{\ell'}$ to the node n^α_ℓ. The role of the coupling inductors as given by the term $M^{\alpha,\beta}_{\ell,\ell'} \cdot m^{C,\beta}_{\ell',k-1}$ generates a voltage in the path where the inductor L^α_ℓ exists.

Analogously to (9), the (12) also can be written recursively as

$$m^\alpha_{j,k} = m^\alpha_{p(j),k} + R^\alpha_j \cdot m^{C,\alpha}_{j,k} - L^\alpha_j \cdot m^{C,\alpha}_{j,k-1} - \sum_{L^\beta_{j'} \in \mathcal{L}^\alpha_j} M^{\alpha,\beta}_{j,j'} \cdot m^{C,\beta}_{j',k-1}. \quad (14)$$

This recursive formula indicates that the computation can be decomposed in successive triples as in (10) and (11), hence can be represented again by a BDD.

The construction procedure is as follows: 1) construct an *R(L)*-BDD-*C-tree* diagram for each individual tree and 2) add coupling links between the individual structures. Since the inductive coupling exists between inductors, the corresponding L-BDD nodes are linked. Similarly, the coupled *C-tree* nodes also are linked. Fig. 5 shows such a coupled data structure for moment computation of the coupled RLC tree circuit given in Fig. 4. The dash-dot lines in the middle represent the coupling existing between the two BDD-*C-tree* diagrams. The coupling links are used for computing the term of

$$\sum_{L^\beta_{j'} \in \mathcal{L}^\alpha_j} M^{\alpha,\beta}_{j,j'} \cdot m^{C,\beta}_{j',k-1}$$

in (14) and the term of

$$\sum_{C^\beta_{\ell'} \in \mathcal{C}^\alpha_\ell} C^{\alpha,\beta}_{\ell,\ell'} \left(m^\alpha_{\ell,k-1} - m^\beta_{\ell',k-1} \right)$$

in (13).

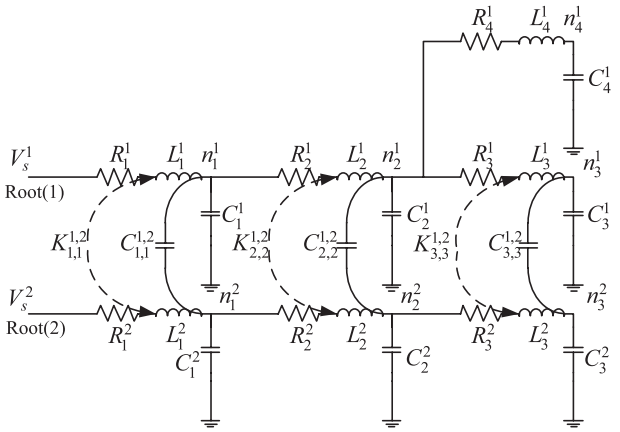


Fig. 4. Two RLC trees coupled.

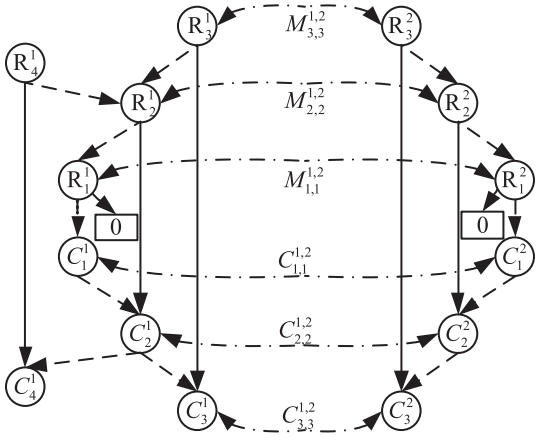


Fig. 5. Coupled R-BDDs and C-trees.

The initial zeroth-order moments of all nodes are the DC solution of the coupled circuit, which can be solved for each decoupled trees because the inductive and capacitive couplings have no effects for the DC solution. These DC moments are used for computing the coupling terms stated above for the next order moment computation. It is noted that no extra computational nodes have to be created for the inductive and capacitive couplings except for making such linkage between the computation diagrams constructed for the tree circuits.

III. BDD-Based Moment Computation for Mesh Circuits With Multiple Sources

The moment computation procedures and diagrams presented in the preceding section only work for analyzing RLC trees or capacitively/inductively coupled RLC trees. When resistive links exist between any tree nodes, new analysis procedures have to be developed. Since resistively linked circuits widely exist in digital IC, such as clock mesh or power-ground networks [24], [25], developing a symbolic moment computation method is of significance for application.

The objective of this section is to develop a unified computational diagram that can be applied for computing moments of RLC trees with resistive links, while the previously developed

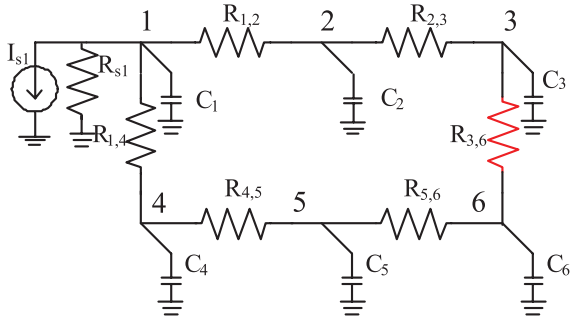


Fig. 6. Tree-type circuit with a resistive link $R_{3,6}$.

BDD-*C-tree* diagram can be reused. A fundamental technique that plays the key role in this regard is a technique known as the *Kron's branch tearing* [26]. The basic idea of Kron's tearing is to decompose a network with a resistor link R_{link} into two networks, one with R_{link} removed and the other with R_{link} replaced by a current source. If all existing R-links in a network are processed by successive decompositions, the original network would end up with a set of networks without resistive links but driven by current sources at different nodes. Such purely current-driven tree-structure networks can be analyzed symbolically by the means of moment computation already developed in Section II-A.

The Kron's branch tearing technique was first applied by Ratzlaff *et al.* [2] to handle resistor links for matrix inversion in the RICE work. Later Lee *et al.* [27], [28] introduced a BDD-based decomposition process that greatly improves the computational efficiency. The work [16] further enhances the computational efficiency by pointing out that the tree circuits resulted from tearing all links are simply current-driven RC-tree networks that can be analyzed in terms of moments by symbolic BDD-*C-tree* computational diagram. The next two sections contribute a systematic development on how a mesh circuit driven by multiple sources can be analyzed by the means of symbolic moments.

A. Kron's Tearing and Mesh Decomposition

The circuit shown in Fig. 6 is used to illustrate the basic notation and procedure involved with Kron's tearing. The circuit has a single driving current source and a resistive link $R_{3,6}$ connecting nodes 3 and 6. By removing the resistor link, the circuit becomes an RC tree.

According to the principle of Kron's tearing, solving the circuit shown in Fig. 6 is equivalent to solving the two RC-tree circuits shown in Figs. 7 and 8, where the circuit in Fig. 7 is just the circuit in Fig. 6 with the R-link $R_{3,6}$ removed, while the circuit in Fig. 8 is the result of removing the R-link $R_{3,6}$ while placing two opposite-polarity current sources of value I_R at the two terminals of $R_{3,6}$, which is equivalent to replacing $R_{3,6}$ by a current source of I_R.

Let R_{link} be the link resistor to be removed. The magnitude of the current source of I_R is computed by [26]

$$I_R = \frac{V^{(O)}_{R_{\text{link}}}}{R_{\text{link}} + R_{Th}} \tag{15}$$

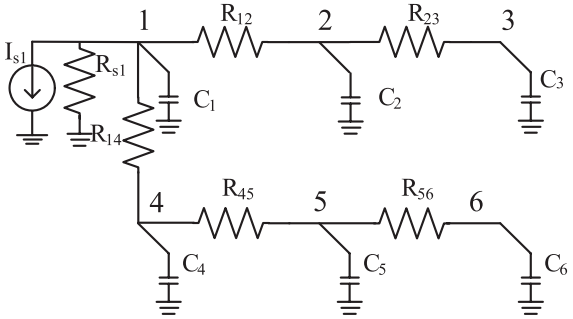


Fig. 7. Circuit with the branch $R_{3,6}$ removed.

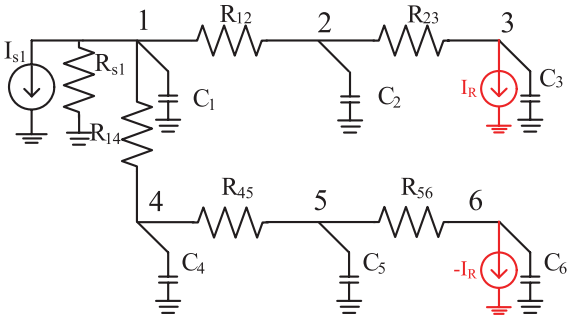


Fig. 8. Circuit with the branch $R_{3,6}$ replaced by two current sources I_R and $-I_R$.

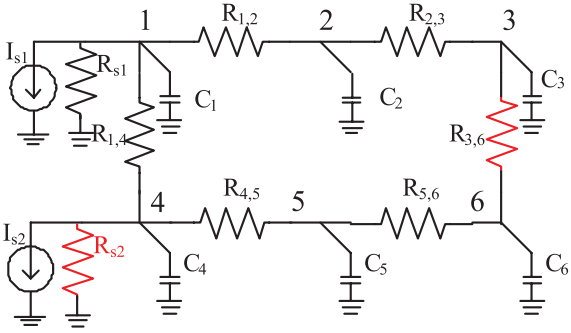


Fig. 9. Mesh circuit with multiple sources.

where $V_{R_{\text{link}}}^{(O)}$ is the cross voltage at the terminals of R_{link} with R_{link} removed and R_{Th} is the Thevenin equivalent resistance seen at the port of R_{link}. By definition, the Thevenin equivalent resistance R_{Th} is calculated by setting I_R to unit current while *turning off all other sources*, solving the DC nodal voltages $V_{p,q}^{(A)} = V_p^{(A)} - V_q^{(A)}$ gives the Thevenin resistance R_{Th}. Here, we use the superscript $^{(A)}$ to indicate that the voltages are computed when the two port currents are set to unit current (1A), i.e., $I_R = 1A$. Consequently, $R_{Th} = V_{p,q}^{(A)}$. With a BDD-*C-tree* diagram, the computation of R_{Th} is executed by setting in the *C-tree* $C_p = C_q = 1$ and all other capacitors to zero values. The calculated zeroth-order moments at nodes p and q give rise to $V_p^{(A)}$ and $V_q^{(A)}$.

In summary, the computation of both $V_{R_{\text{link}}}^{(O)}$ and R_{Th} in (15) can use the same BDD-*C-tree* diagram, which is constructed only once. With the quantities $V_i^{(O)}$, $V_i^{(A)}$, and I_R computed by the BDD-*C-tree* diagram, the nodal voltages of the original circuit, with the resistor link placed back, are computed by

$$V_i = V_i^{(O)} - I_R \cdot V_i^{(A)}. \tag{16}$$

The computation principle described above can be extended to the computation of higher order moments. Noticing that all moments being computed are just the coefficients in the Taylor expansion of the voltage transfer functions, we see that in general the kth order moment at node i for a network with one resistive link can be computed analogously by the following formulas:

$$R_{Th,k} = m_{\alpha,k}^{(A)} - m_{\beta,k}^{(A)} \tag{17}$$

$$I_{R,k} = \frac{m_{\alpha,k}^{(O)} - m_{\beta,k}^{(O)}}{R_{\text{link}} + R_{Th,k}} \tag{18}$$

and

$$m_{i,k} = m_{i,k}^{(O)} - I_{R,k} \cdot m_{i,k}^{(A)} \tag{19}$$

for $i = 1, \ldots, N$, where we assume that the link resistor R_{link} is connected between nodes α and β. Note that the superscripts $^{(O)}$ and $^{(A)}$ in the above equations reserve the same meaning as before.

The Kron's tearing essentially involves two operations, one is "*1A source substitution*" and the other is "*link opening*." Hence, BDD is a natural data structure of representing the binary decomposition [27]. If more than one link exists in the network, the tearing process can be repeated and a BDD can be constructed which is called a *tearing*-BDD. It was found later on that a mesh network driven by multiple independent sources can be treated by the same technique as well [17].

B. Moment Computation for Mesh Circuits With Multiple Sources

Severe process variation exists in the current nanometer fabrication technology. To avoid significant timing variation in the tree-structured clock networks, a new design technique called "clock mesh" has been introduced recently for balanced clock delivery [24]. More recently, interests in clock mesh analysis and synthesis arise [29]–[31]. A mesh network could be driven by many sources applied at a portion of mesh nodes. Synthesis and verification of such mesh systems require efficient mesh analysis methodology and algorithms. Running a SPICE simulator repeatedly for such purpose is considered too costly. Some work attempted to use model order reduction for mesh synthesis, but encountered difficulty in handling multiple driving sources [29], [30]. The second main contribution of this paper is to extend the link tearing procedure in the preceding section to multiple-source-driven mesh networks.

As a matter of fact, a nonideal voltage source driving any internal node of a mesh can be modeled by a driving current in parallel with a source resistance (i.e., the Norton equivalent). An example circuit is given in Fig. 9, which illustrates the possible configuration of a mesh circuit. A grounding resistor can be teared just as a link resistor by placing one unit current source directed from the tearing node to the ground. Following this idea, a unified tearing-based decomposition procedure can be formulated after introducing appropriate notations.

For the example given in Fig. 9 with two current sources, one current source is selected as the driving source for a

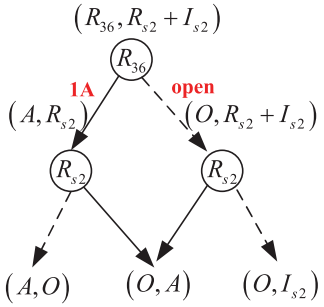


Fig. 10. Tearing-BDD for the circuit given in Fig. 9.

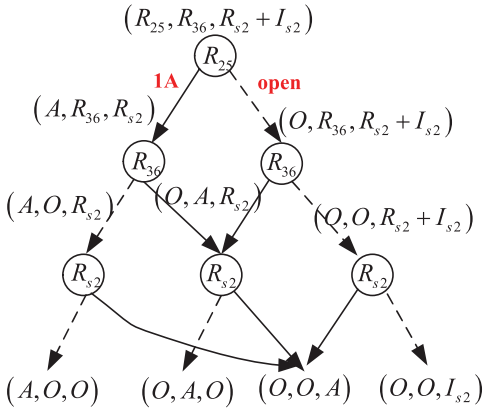


Fig. 11. Tearing-BDD resulting from adding a new resistor link $R_{2,5}$ to the circuit given in Fig. 9.

spanning tree network. By keeping the source I_{s1}, we shall tear the grounding resistor R_{s2} and the link resistor $R_{3,6}$ so that the network becomes a tree circuit and the BDD-*C-tree* diagram can be used for moment computation. The tearing procedure results in the BDD shown in Fig. 10, which keeps the record of the tearing process. Each BDD vertex is attached a tuple, which is a good notation for indexing the tearing sequence.

The tuple $(R_{3,6},\ R_{s2}+I_{s2})$ at the root indicates that the two resistors $R_{3,6}$ and R_{s2} are to be teared in the listed order. The reason of writing $R_{s2}+I_{s2}$ is because R_{s2} will be substituted with an open branch or a $1A$ current during tearing. In this sense, the role of R_{s2} is analogous to the role of current I_{s2}, which are added together to manifest the superposition principle.

The tearing order follows the sequence listed in the tuple (from the left to the right). Hence, the resistor $R_{3,6}$ is teared before the resistor R_{s2}. The tearing of the resistor $R_{3,6}$ results in the two derived circuits marked by the tuples $(A,\ R_{s2})$ and $(O,\ R_{s2}+I_{s2})$ labeled to the BDD vertices in the second row. The tuple $(A,\ R_{s2})$ means that the resistor $R_{3,6}$ is substituted by a unit $(1A)$ current while the current source I_{s2} must be turned off for the purpose of computing the Thevenin resistance. On the other hand, the tuple $(O,\ R_{s2}+I_{s2})$ simply means that the resistor $R_{3,6}$ is removed while the rest is retained.

In the next step, the resistor R_{s2} is teared. The two intermediate circuits at the middle layer would generate the three tree circuits at the bottom layer. The left-most tuple $(A,\ O)$ means that grounding resistor R_{s2} is removed. The middle tuple $(O,\ A)$ is pointed by two solid arrows from the preceding (middle) layer. The left arrow means that the resistor R_{s2} is substituted by a unit current source with *all other current sources switched off* while the right arrow carries the same meaning. Both arrows stand for the Thevenin resistance computation. The rightmost tuple $(O,\ I_{s2})$ results from removing R_{s2}. The existence of one vertex in a lower layer being shared by all solid arrows coming down from the preceding layer is a general property, which holds in all tearing-BDD layers (excluding the root). This important property essentially simplified the binary decomposition and makes the BDD representation an efficient means.

In addition to representing the sequence of Kron's tearing, each tearing-BDD vertex must perform the computation given in the formulas (17)–(19) for the zeroth-order moments. The moments marked with superscripts $^{(O)}$ and $^{(A)}$ are accessed from the two child vertices. When computing the higher order moments, all those tearing-BDD vertices marked with an (A) do not have to update the computation, because these circuits still are driven by a single unit current source, we do not need to substitute the capacitors by current sources (due to the Thevenin principle).

The decomposed circuits at the bottom layer of the tearing-BDD are an identical RC-tree circuit but driven by different current sources at the places marked with a single unit current source (A) or with the independent sources I_{sk}. Those tree circuits driven by a single unit current source have to be computed *only once* in the computation of the zeroth-order moments. Consequently, only those tree circuits driven by the *independent* sources have the need of recomputation with the capacitors substituted by appropriate current sources. This is another important property inherent in the tearing-BDD computational diagram.

Summarized in the following steps is the symbolic moment computation procedure for a mesh circuit driven by multiple sources.

C. General Symbolic Moment Computation Algorithm

Step 1: Select one primary source and find a spanning tree, which spans the original circuit and is rooted at the primary source.

Step 2: Construct a BDD-*C-tree* diagram for the spanning $R(L)C$-tree.

Step 3: Construct a tearing-BDD using the Kron's tearing process for all the link resistors and the grounding resistors.

Step 4: Evaluate the derived tree circuits at the bottom vertices of the tearing-BDD whose tuples include one single source (either an "A" or an independent current source) by invoking the BDD-*C-tree* diagram with one *C-tree* node substituted by the corresponding current source value.

Step 5: Evaluate the zeroth-order moments of the vertices in the tearing-BDD from bottom-up using the formulas (17)–(19). The moments superscripted with (A) [respectively, with (O)] are calculated

from the circuit at the child vertex pointed by the solid (respectively, dashed) arrow.

Step 6: Repeat Steps 4 and 5 to compute the moments of the next requested order. Recompute the moments of those tree circuits driven by the *independent* sources with the capacitors substituted by the previously computed moments and propagate upwards the tearing-BDD vertices affected.

For a mesh circuit with K resistor links (including the grounding links), the total number of vertices to be constructed in the tearing-BDD is $\sum_{t=1}^{K+1} t = 1/2(K+1)(K+2)$, which grows quadratically. At each tearing-BDD vertex, all nodal voltages of the network must be updated once using the formula (16). Suppose the mesh has N nodes and the maximum number of links does not exceed N. Then the total computational cost for computing the zeroth-order moments of such a mesh circuit is of the polynomial complexity $O(N^3)$. For higher order moments, this complexity is lower because only $K+1$ tearing-BDD vertices have to be updated.

Before ending this section, we point out that the order of tearing the resistor links does not affect the size of the tearing-BDD, because the tearing mechanism is based on linear superposition, which is commutable. This order-independent property can be made use of in incremental interconnect network synthesis, such as inserting an additional resistor link. For example, we would like to add a new resistor link $R_{2,5}$ into the circuit in Fig. 9, connecting nodes 2 and 5. If the tearing-BDD for the original circuit is already constructed as given in Fig. 10, then the new tearing-BDD does not have to be reconstructed, rather a slight modification applied to the existing tearing-BDD would result in a new tearing-BDD for the link inserted circuit, which is shown in Fig. 11. In the new tearing-BDD, we see that a new root vertex $R_{2,5}$ is created and following that one additional vertex is added to each tearing-BDD layer, to which the attached label indicates that the resistor $R_{2,5}$ is substituted by a unit current (A) (see the left-most vertex in each layer in Fig. 11).

IV. Performance Evaluation and Applications

So far we have presented the main construction procedures for a SMC. This computation scheme is based on the recursive computation of a tree-structured circuit and successive tearing of a set of resistive branches. Different orders of moments can be computed by repeatedly traversing the same hierarchically linked computation diagram, which consists of a *C-tree* at the bottom, an $R(L)$-BDD in the middle, and a tearing-BDD at the top. All nodes (tree nodes or BDD vertices) in the computation diagram have correspondence to the circuit elements. Hence, as soon as the circuit elements change their values, these values can substitute the SMC node values instantly and a re-execution of SMC computes the new moments of different orders.

The SMC has many applications, with the main application targeted at statistical interconnect analysis and synthesis, such as fast statistical computation of interconnect timing/crosstalk metrics, sensitivity analysis, and topological interconnect synthesis like wire sizing and adding/removing interconnect segments. To demonstrate the potential of the SMC applications, this section is dedicated to an investigation of the following three aspects.

1) We show via experiments that the SMC is a much faster symbolic analyzer than one of the best traditional BDD-based analog circuit simulators [32]. We also show that the SMC has better computational efficiency in repeated computations than HSPICE.
2) We outline an important application of SMC in the moment sensitivity analysis, which can easily be implemented based on the traceable symbolic data structure in SMC.
3) We provide an experimental study on the application of SMC in statistical timing analysis, where we discuss the efficiency and accuracy issues as well.

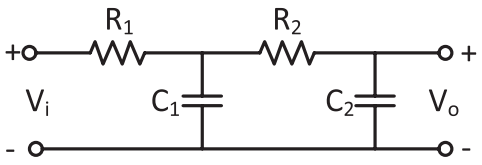


Fig. 12. Buffer model [35].

A. SMC Efficiency Evaluation

We have implemented a C++ program of SMC. All the test data reported in this section were collected from a computer with Intel Quad 3G CPU and 16GB memory, running a Redhat Enterprise Linux 4 operating system. We tried to use realistic interconnect models as close as possible to what industry uses.

The interconnect resistance and capacitance models are borrowed from [33]

$$R = \rho \cdot \frac{l}{hw} \tag{20}$$

$$C = l \cdot \left[\frac{2\pi \varepsilon_0 \varepsilon_r}{\ln(t_{ox}/h)} + \frac{(w - 0.5 \cdot h)\varepsilon_0 \varepsilon_r}{t_{ox}} \right] \tag{21}$$

where l, w, h, t_{ox}, ε_0, and ε_r are, respectively, the interconnect length, width, height, inter-layer dielectric thickness, dielectric constant, and relative permittivity. The interconnect inductance model is from the predictive technology model [34]

$$L = \frac{\mu_0 \cdot l}{2\pi} \left[\ln\left(\frac{2l}{w+h} \right) + \frac{1}{2} + \frac{0.22\,(w+h)}{l} \right] \tag{22}$$

with the parameters $l = 50\ \mu$m, $w = 0.28\ \mu$m, $h = 0.32\ \mu$m, $t_{ox} = 0.5\ \mu$m, $\varepsilon_0 = 8.85418 \times 10^{-12}$ F$\cdot$ m^{-1}, $\varepsilon_r = 3.9$, and $\mu_0 = 4\pi \times 10^{-7}$ H/m.

A large interconnect system may have buffers inserted. The buffer model used in this paper is a two pole approximation shown in Fig. 12 [35]. The parameter values of R_1, R_2, C_1, C_2 are obtained, using HSPICE optimize functions from the real gate delay characteristics of the Nangate 45-nm open cell library [36].

Nine industrial-level interconnect circuits were used for testing the SMC efficiency, six of them were purely tree structure circuits while three of them were mesh circuits containing different numbers of resistor links. The scales of the test results are listed in the third to fifth columns of Table I.

TABLE I
EFFICIENCY TEST OF THE SMC

Ckt #	Circuit type	No. elem.	No. drivers	No. links	SMC constr. (s)	Moment eval. (s)	Sens. eval. (s)
1	RC tree	1404	1	0	0.06	0.007	0.01
2	RC tree	8404	1	0	0.29	0.04	0.08
3	RLC tree	2104	1	0	0.08	0.01	0.06
4	RLC tree	12 606	1	0	0.32	0.04	0.32
5	RC coupled	3006	3	0	0.07	0.02	0.07
6	RLC coupled	3506	2	0	0.06	0.02	0.06
7	RC mesh	1209	30	104	11.12	0.10	0.91
8	RC mesh	3586	63	143	62.38	0.43	6.77
9	RC mesh	7973	130	298	599.5	0.88	30.05

Listed in the column *SMC Constr.* are the SMC construction time, which is required for only once for each circuit. As expected, the construction time measures for those mesh circuits are much larger than those for the tree circuits. Recall that the complexity of the tearing-DDD construction in SMC is cubic polynomial. The computation time measures for moments from the zeroth order up to the fourth order at all circuit nodes are listed in the column *Moment Eval.* It is noted that the moment evaluation time is proportional to the constructed SMC size, because the evaluation of each order of moments requires a traversal of the whole SMC. Clearly, the evaluation time measures are generally fractions of the construction time measures. The last column lists the sensitivity evaluation time, which will be explained later in the next section where the sensitivity computation is discussed.

For the purpose of efficiency comparison, we further performed some experiments of using the existing symbolic tools for moment computation. The software used for comparison all were run on the same computer where the SMC was run.

Equation (4) indicates that general moments can be computed by matrix inversion. If we directly use a general-purpose symbolic toolbox for matrix inversion, the computation efficiency would be very bad. For example, we generated a modified nodal analysis (MNA) matrix for a small mesh circuit with 1209 elements (Ckt #7 in Table I) and used the commercial MATLAB symbolic toolbox to compute its matrix inversion. It took the toolbox over 8 h to generate the symbolic inverse matrix. In general, it is not recommended to use any general-purpose symbolic toolbox for large-scale circuit analysis.

Another published symbolic tool which is dedicated to circuit analysis is the determinant decision diagram (DDD) algorithm [32], which is selected for comparison because it is one of the best performance symbolic simulators in the literature. The key technical component of DDD is to compute the symbolic solution of the form $A^{-1}b$ by the Cramer's rule, which is constructed symbolically by determinant expansion. For better efficiency, the determinant expansion is programmed using a BDD, which has the benefit of sharing all common sub-determinants (minors). The BDD sharing mechanism is the most crucial mechanism that improves the symbolic analysis efficiency of DDD.

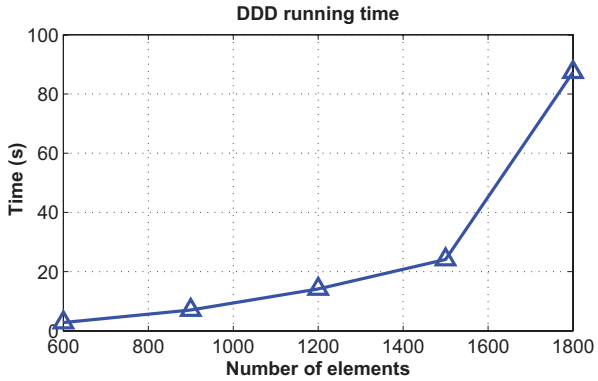


Fig. 13. DDD computation time for one frequency response for a set of smaller tree circuits.

However, even with the help of an advanced BDD data structure, the best complexity of DDD is still exponential for most practical circuits, unlike the polynomial complexity with SMC. The critical difference is that DDD computes the circuit DC solution via determinant expansion, while SMC computes the circuit DC solution with a circuit-based decomposition, whose complexity has been proven to be cubic polynomial. The theoretical complexity estimates can be justified by numerical tests on circuits.

We used a set of small RC tree circuits containing the number of elements from 600 to 1800 for comparing the runtime speeds of DDD and SMC. Fig. 13 shows the quick exponentially rising computation time. For comparison, we used another set of *much larger* RC-tree circuits containing the number of elements from 200K to one million (1M) to test the SMC runtime speed. Fig. 14 shows that not only the computation time grows linearly, but the absolute computation time is also much less than that needed by the DDD solver in solving those smaller circuits. For example, the SMC computation time for the 1M element circuit was only about 8 s, while the DDD time for the circuit containing only 900 elements took about 7 s. In this experiment, we tuned the DDD to a fast mode where the symbolic transfer function $H(s)$ was constructed in the s-expanded form [37].

Although both DDD and SMC use the BDD for the symbolic data representation, there exist some critical differences in addition to the complexity estimates mentioned above. The first critical difference is that DDD normally constructs a symbolic transfer function for one input and one output. If multiple output nodes are considered (such as solving all nodal voltages of a tree circuit), then DDD has to be invoked for the number of times equal to the number of outputs. Although one may create a multiroot BDD to enforce data sharing, the complexity increase would normally degrade the overall DDD performance in dealing with multioutput circuits. In contrast, the SMC by construction can automatically compute all nodal moments of the same order by one traversal of the SMC computational diagram, which is more efficient than DDD.

The second critical difference between DDD and SMC is related to the requirement of *symbol ordering*. When DDD

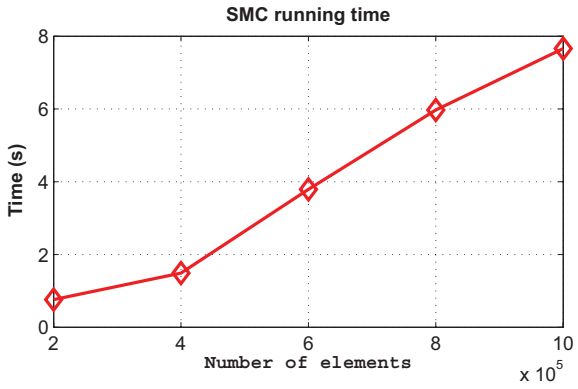


Fig. 14. SMC computation time for four orders of moments for another set of larger tree circuits.

TABLE II
EFFICIENCY COMPARISON BETWEEN SMC AND HSPICE FOR NUMERICAL MOMENT COMPUTATION

Ckt #	Circuit Type	Moment Eval. (s)	HSPICE Eval. (s)
1	RC tree	0.007	0.08
2	RC tree	0.04	0.52
3	RCL tree	0.01	0.31
4	RCL tree	0.04	1.98
5	RC couple	0.02	0.47
6	RCL couple	0.02	8.03
7	RC mesh	0.10	0.20
8	RC mesh	0.43	0.53
9	RC mesh	0.88	1.57

expands a determinant and constructs a BDD, the order of expansion (selecting one matrix entry before another) in general affects the resulting size of BDD, which further affects the BDD evaluation time. However, the optimal order for a minimum BDD is unknown in general. In contrast, the SMC is built directly from decomposing the circuit topology, in which the constructed BDD size is independent of the order of tearing the links. This is another critical factor that differentiate the performance of SMC from DDD for large-scale circuits.

Remark 1: Although, the SMC we proposed has substantially improved the computational efficiency comparing to the traditional MATLAB symbolic toolbox and the advanced DDD solver, we are aware of the fact that the cubic polynomial complexity is still high for extremely large interconnect mesh networks appearing in the contemporary IC. Further performance enhancement of SMC must be developed by introducing hierarchical schemes, for which the tearing mechanism we applied in this paper can be explored. The technical details will be presented in another work.

The last efficiency test is to compare the numerical computation performance between SMC and HSPICE (version 2007.09). As we observed in (4), the nodal moments of an RLC circuit are a sequence of DC solutions of the same circuit by substituting those inductors and capacitors with appropriate voltage sources and current sources whose values are determined by the moments computed for the previous order [1]. Although HSPICE does not provide directly the moment computation functionality, we used it for experiment purpose to measure the DC simulation time if some given orders of moments are to be computed by HSPICE. We assumed a scenario of interconnect synthesis where the interconnect sizes are to be adjusted. When the RLC values change, the LU factorization must be performed repeatedly in HSPICE, which also invokes the repeated setup of the MNA matrix. We compared the HSPICE time involved in such a scenario of computation needs to the SMC repeated computation time, excluding the one-shot SMC construction time which is not comparable to HSPICE. The time measurements are listed in Table II, where up to the fifth order of moments was computed. We see that the SMC is superior to HSPICE simulator in repeated numerical computations, which is a commonly requested analysis in statistical interconnect analysis and synthesis.

B. Symbolic Moment Sensitivity

Symbolic moment computation is different from numerical moment computation in that a regular data structure is constructed and preserved in the computer memory throughout the tool running duration. The maintenance of a computational data structure is beneficial to both repeated numerical evaluation and sensitivity analysis. In this section, we explain how the sensitivity analysis can be implemented on a SMC. The sensitivity of moment with respect to a set of selected parameters is calculated based on the derivative chain rule. The SMC linked data structure can facilitate the implementation of the chain rule provided that appropriate extra memory is allocated for storing the intermediate derivative values at the C-tree nodes and the BDD vertices. Using other computational techniques like automatic differentiation could be an option. However, an implementation based on an existing SMC is more efficient and straightforward.

There exist numerical sensitivity computation techniques in the literature [7], [38], [39]. When repeated sensitivity computations are required, such as for wire sizing [40], a symbolic implementation is more advantageous.

The moment sensitivity can be computed by numerically evaluating the following gradient vectors of moment with respect to the selected resistors or capacitors

$$\nabla_{\vec{R}} m_{i,k} := \left[\partial m_{i,k}/\partial R_1, \ldots, \partial m_{i,k}/\partial R_{q_1}\right]^T$$
$$\nabla_{\vec{C}} m_{i,k} := \left[\partial m_{i,k}/\partial C_1, \ldots, \partial m_{i,k}/\partial C_{q_2}\right]^T$$

where $\vec{R}$ and $\vec{C}$ are the vectors containing the selected R_i's and C_i's, q_1 and q_2 are, respectively, the total numbers of R_i's or C_i's selected for sensitivity analysis. In the case of a tree circuit, taking the gradient of the moments with respect to $\vec{R}$ or $\vec{C}$ in the (9) (ignoring inductances) gives

$$\nabla_{\vec{R}} m_{i,k} = \nabla_{\vec{R}} m_{p(i),k} + \nabla_{\vec{R}} \left(R_i \cdot m^C_{i,k}\right) \quad (23)$$
$$\nabla_{\vec{C}} m_{i,k} = \nabla_{\vec{C}} m_{p(i),k} + \nabla_{\vec{C}} \left(R_i \cdot m^C_{i,k}\right) \quad (24)$$

where by (7) the gradients $\nabla_{\vec{R}}(R_i \cdot m_{i,k}^C)$ and $\nabla_{\vec{C}}(R_i \cdot m_{i,k}^C)$ can be written, respectively, as

$$\nabla_{\vec{R}}\left(R_i \cdot m_{i,k}^C\right) = \sum_{j \in T_i} \left(\vec{e}_i C_j m_{j,k-1} + R_i C_j \nabla_{\vec{R}} m_{j,k-1}\right) \quad (25)$$

$$\nabla_{\vec{C}}\left(R_i \cdot m_{i,k}^C\right) = \sum_{j \in T_i} \left(R_i \vec{e}_j m_{j,k-1} + R_i C_j \nabla_{\vec{C}} m_{j,k-1}\right) \quad (26)$$

where $\vec{e}_i$ is the ith basis vector in the q_1 or q_2 dimensional space. Equations (23) and (24) are the basic equations for the computation of *resistive* sensitivity and the *capacitive* sensitivity, respectively. Also, we can compute the second-order derivatives as listed below

$$\nabla_{\vec{R}}^2 m_{i,k} = \nabla_{\vec{R}}^2 m_{p(i),k} + \sum_{j \in T_i} \left(\vec{e}_i C_j \nabla_{\vec{R}} m_{j,k-1} + R_i C_j \nabla_{\vec{R}}^2 m_{j,k-1} + \vec{e}_i C_j m_{j,k-1}\right) \quad (27)$$

$$\nabla_{\vec{C}}^2 m_{i,k} = \nabla_{\vec{C}}^2 m_{p(i),k} + \sum_{j \in T_i} \left(2R_i \vec{e}_j \nabla_{\vec{C}} m_{j,k-1} + R_i C_j \nabla_{\vec{C}}^2 m_{j,k-1}\right) \quad (28)$$

$$\nabla_{\vec{R}} \nabla_{\vec{C}} m_{i,k} = \nabla_{\vec{R}} \nabla_{\vec{C}} m_{p(i),k} + \sum_{j \in T_i} \left(\vec{e}_i \vec{e}_j m_{j,k-1} + R_i \vec{e}_j \nabla_{\vec{R}} m_{j,k-1} + \vec{e}_i C_j \nabla_{\vec{C}} m_{j,k-1} + R_i C_j \nabla_{\vec{R}} \nabla_{\vec{C}} m_{j,k-1}\right). \quad (29)$$

The higher order moment derivatives can be calculated analogously by continuing the chain rule. In most applications using derivatives up to the second order would be sufficient. If inductors are involved, the moment derivatives with respect to inductances can be established analogously.

The moment gradients given in (23) and (24) indicate that the gradient computation can be implemented based on an existing SMC because the fundamental sequence of computation has not changed, except for the requirement of certain extra gradient vectors attached to the SMC vertices for saving the intermediate gradient values. The bottom-up propagation of the intermediately computed gradient vector still follows the SMC data structure, as seen from the (23) and (24).

In the case of a mesh circuit with resistor links, the gradient vector computation would have to take in account of the special computation defined by (19) for each tearing-BDD vertexes (see Fig. 10). Taking gradient operation of (19) leads to

$$\nabla_x m_{i,k} = \nabla_x m_{i,k}^{(O)} - \nabla_x I_{R,k} \cdot m_{i,k}^{(A)} - I_{R,k} \cdot \nabla_x m_{i,k}^{(A)} \quad (30)$$

where x is any parameter selected for sensitivity. Since $I_{R,k}$ in the above expression is given by

$$I_{R,k} = \frac{m_{\alpha,k}^{(O)} - m_{\beta,k}^{(O)}}{R_{\text{link}} + m_{\alpha,k}^{(A)} - m_{\beta,k}^{(A)}} \quad (31)$$

$\nabla_x I_{R,k}$ can be computed in terms of $\nabla_x m_{i,k}^{(O)}$ and $\nabla_x m_{i,k}^{(A)}$, which are in turn computed and saved with the descendent tearing-BDD vertices.

As far as the computational complexity is concerned for the sensitivity, there is no substantial complexity increase except for extra memory required for saving the intermediate gradient vectors at the associated SMC vertices. As the number of the sensitivity parameters increases, the total sensitivity computation time would increase proportionally to the number of SMC vertices.

One special advantage of the SMC-based symbolic sensitivity calculation is that the moment sensitivity can be computed simultaneously with the moment values during the bottom-up traversal, which is clear from the derivative formulas derived above.

The experimental efficiency evaluation of the SMC sensitivity is given in the last column of *Sens. Eval.* in Table I, where both first- and second-order sensitivities of moments up to the fourth order were computed. Six parameters were chosen in the sensitivity gradient calculations.

C. Application to Statistical Timing Analysis

The purpose of statistical timing analysis is to extract the approximate timing distribution as fast as possible given probability distributions of the variational interconnect parameters. In general, this is a computation-intensive problem because the accurate relationships between the interconnect parameters and the timing metrics are very complicated and usually must be measured by numerical simulations. Digital IC synthesis requires fast timing metric evaluation routines to enable iterative optimizations. In the statistical scenario, a fast statistical timing mechanism is the enabling technology for an efficient synthesis flow.

A variety of techniques have been proposed in the literature for the statistical interconnect timing analysis. The majority of them use certain metrics measured by moments, typically up to order two. A few timing metrics that have been demonstrated useful are for example the D2M metric [41] that calculates delay using two moments, and the S2M metric [42] that calculates slew rate also using two moments. Whenever the timing metrics are computed in terms of moments, the SMC developed in this paper can greatly improve the efficiency of repeated computation and can help to establish the timing distribution with a very fast computation strategy.

Some other design metrics in digital IC, such as crosstalk and power etc. also can be computed in terms of moments. For a focused demonstration, we only address in this section the application of SMC to statistical timing analysis, while the other design metrics also can be treated with a similar strategy.

The empirical D2M metric was proposed by Alpert *et al.* in [41] for computing the delay at all nodes of an RC-tree circuit, which was demonstrated to be more accurate than what Elmore delay could predict. The D2M metric is given by the following equation:

$$t_{D2M,i} = \ln(2) \frac{m_{i,1}^2}{\sqrt{m_{i,2}}} \quad (32)$$

where $m_{i,1}$ and $m_{i,2}$ are the first- and second-order moments at node i. It is evident that the statistics of the delay metric $t_{D2M,i}$ can be computed as long as the statistics of the two moment are computed.

TABLE III
ACCURACY OF DELAY COMPUTED BY THE D2M METRIC WITH SMC

Case	Type	Rel. ave. err.
1	RC tree	0.19%
2	RC tree	0.39%
5	RC coupled	0.35%
7	RC mesh	0.72%
8	RC mesh	0.95%
9	RC mesh	1.12%

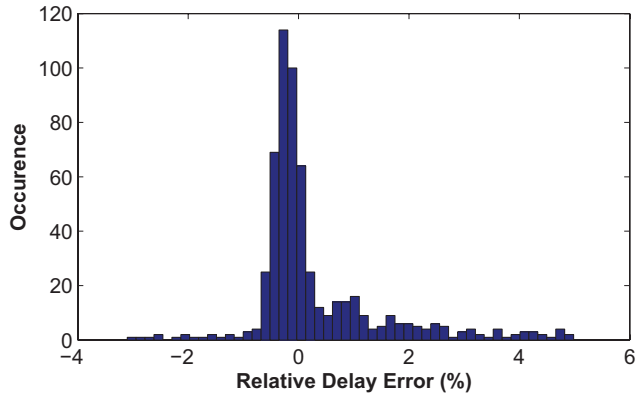


Fig. 15. D2M delay metric relative error of all nodes in case 1.

Nine test circuits listed in Table III were used for validating the proposed computation mechanism. We first demonstrate that the D2M metric is sufficiently accurate for the timing analysis of all nine test circuit. The two orders of moments in the D2M metric were computed by SMC. The accuracy of the delays estimated by the D2M metric was verified by the HSPICE transient simulation results and the following formula:

$$\text{Rel. Error} := \frac{\left|\bar{t}_{\text{D2M}} - \bar{t}_{\text{SPICE}}\right|}{\bar{t}_{\text{SPICE}}} \tag{33}$$

was used to calculate the relative errors. For each test case, the relative errors of all nodes were averaged over the number of nodes. Listed in Table III are the averaged D2M relative delay errors. The accuracy of the D2M estimation seems acceptable. Since the D2M delay might have different errors at different nodes in a circuit, we provide the histogram plot in Fig. 15, which shows the counts of nodes whose delay falls within a bin of relative delay error. It seems that the majority of node delays are accurately estimated by the D2M metric.

The original D2M was proposed for delay estimation given an ideal driving step input. If the driving signal is a ramp signal, then the D2M metric should be modified for better accuracy. Reference [43] proposed the following formula for the 50% delay estimation:

$$t_{d,r} = (1-\alpha)m_1 + \alpha \cdot t_{D2M} \tag{34}$$

where

$$\alpha = \left(\frac{2m_2 - m_1^2}{2m_2 - m_1^2 + t_{sl,in}^2/12}\right)^{\frac{5}{2}} \tag{35}$$

TABLE IV
STATISTICAL TIMING ANALYSIS PERFORMANCE

Case	Ckt Type	Rel. error		SMC	Speedup over
		Mean	Std. Var.	Time (s)	HSPICE MC
1	RC tree	0.54%	0.27%	1.1	5620
2	RC tree	0.51%	0.55%	5.0	4456
5	RC coupled	0.52%	0.33%	2.0	5290
7	RC mesh	1.33%	1.41%	12.6	518
8	RC mesh	1.52%	1.48%	68.8	171
9	RC mesh	1.87%	1.65%	622.1	50

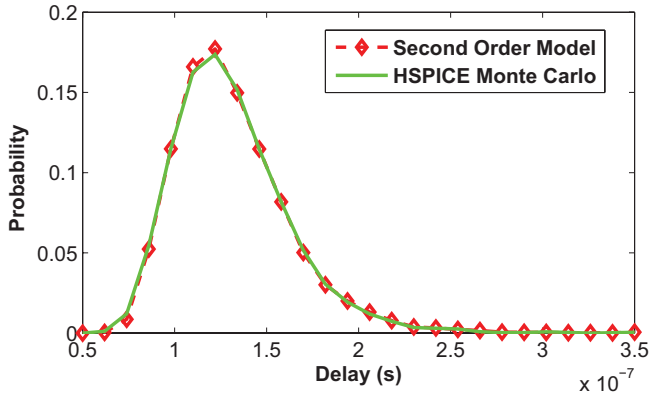


Fig. 16. Delay distribution computed by the ramped D2M metric (34) with the second-order parametric approximation (36) and the accurate delay computed by HSPICE for one test case. The delay was measured at one node.

m_1 is the Elmore delay, t_{D2M} is the ideal D2M delay, and $t_{sl,in}$ is the input slew time. This ramped delay metric is used in the following experiment to validate the application of moment sensitivities computed by the SMC for speedy characterization of timing distribution.

The buffer model shown in Fig. 12 is used for a statistical timing test. The buffer model parameters R_1, R_2, C_1, and C_2 are considered random variables. Also the interconnect width w and height h are considered random variables as well. Suppose all the random variables are subject to Gaussian distribution. In our experiment, the random variables were sampled with the 3σ variation equal to 30% perturbation. The input slew was chosen to be $t_{sl,in} = 30ps$.

The moment sensitivity we developed before can be used for computing an approximate polynomial expression for variational delay characterization. Similar approximation techniques were used in [10], [44], and [45] as well. Let x be the vector of parameter deviations from their nominal values. Let the ramped D2M delay expression in (34) be approximated by the following second-order Taylor expansion:

$$t_{d,r}(x) = \hat{t}_{d,r}(0) + t'_{d,r}(0)x + x^T t''_{d,r}(0)x \tag{36}$$

where $\hat{t}_{d,r}(0)$ is the ramp-input D2M delay obtained at the nominal parameter values, $t'_{d,r}(0)$ and $t''_{d,r}(0)$ are, respectively, the first- and second-order derivatives of the ramped delay evaluated at the nominal parameter values. They are computed via the moment sensitivity by taking derivatives of (34).

Experiment shows that this second-order approximation is adequately accurate for characterizing statistical delay distribution, while the computation speed is significantly enhanced. Shown in Table IV is the moment-based computational speed performance compared to HSPICE Monte Carlo computation running on 10 000 samples, with the six parameters mentioned above chosen for variational sampling. The speedup measure lowers for the three mesh circuits, which indicates more computational cost with SMC for mesh circuits. As an illustration, the probability density function computed by the proposed method is also adequately close to the Monte Carlo accurate computation result, as seen from one snapshot in Fig. 16.

Remark 2: Note the computational complexity for full mesh circuits is in order $O(N^3)$. This complexity is still too high for very large-scale mesh synthesis in the current technology. For practical use of the proposed symbolic moment computation, a hierarchical strategy is a potential solution. The solution developed in this paper for multiple-source-driven circuits can be utilized for developing a hierarchical construction. The limitation of this paper length does not allow us to get into the details in this regard further.

V. Conclusion

Since the proposal of *symbolic model order reduction* in [4], little progress has been made on this subject mainly because several key steps in the standard model order reduction procedures, such as matrix inversion and vector orthogonalization, etc., cannot be solved symbolically without running into exponential complexity. Despite the existing difficulties, a notable progress made in this paper is the proposal of a *symbolic moment* computation scheme, which implements the symbolic construction from the original circuit, rather than from mathematical formulations, such as modified nodal analysis matrices. It turned out the methodological innovation is able to completely reduce an *exponential complexity* problem into a polynomial complexity problem, making the symbolic approach to interconnect network synthesis a tractable problem. Although, a cubic complexity algorithm for full mesh circuits is still not handy for practical clock mesh synthesis, it was expected that a little further research effort would finally come up with a set of applicable symbolic synthesis methodology.

References

[1] L. Pillage and R. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 9, no. 4, pp. 352–366, Apr. 1990.

[2] C. Ratzlaff and L. Pillage, "RICE: Rapid interconnect circuit evaluation using AWE," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 13, no. 6, pp. 763–776, Jun. 1994.

[3] J. Ma and R. Rutenbar, "Fast interval-valued statistical modeling of interconnect and effective capacitance," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 4, pp. 710–724, Apr. 2006.

[4] G. Shi, B. Hu, and C.-J. Shi, "On symbolic model order reduction," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 7, pp. 1257–1272, Jul. 2006.

[5] Y. Shi, L. He, and C.-J. Shi, "Scalable symbolic model order reduction," in *Proc. IEEE Int. Workshop Behavioral Model. Simul.*, Sep. 2008, pp. 112–117.

[6] J. Y. Lee, X. Huang, and R. A. Rohrer, "Pole and zero sensitivity calculation in asymptotic waveform evaluation," *IEEE Trans. Comput.-Aided Design*, vol. 11, no. 5, pp. 586–597, May 1992.

[7] X. Ye, F. Liu, and P. Li, "Fast variational interconnect delay and slew computation using quadratic models," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 8, pp. 913–926, Aug. 2007.

[8] Y. Ismail and C. Amin, "Computation of signal-threshold crossing times directly from higher order moments," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 8, pp. 1264–1276, Aug. 2004.

[9] L. Zhang, W. Chen, Y. Hu, J. Gubner, and C.-P. Chen, "Correlation-preserved non-Gaussian statistical timing analysis with quadratic timing model," in *Proc. 42nd Design Autom. Conf.*, Jun. 2005, pp. 83–88.

[10] K. Agarwal, M. Agarwal, D. Sylvester, and D. Blaauw, "Statistical interconnect metrics for physical-design optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 7, pp. 1273–1288, Jul. 2006.

[11] L. Chen and M. Marek-Sadowska, "Closed-form crosstalk noise metrics for physical design applications," in *Proc. Conf. Design Autom. Test Eur.*, Mar. 2002, pp. 812–819.

[12] L. Daniel, O. C. Siong, L. Chay, K. H. Lee, and J. White, "A multi-parameter moment-matching model-reduction approach for generating geometrically parameterized interconnect performance models," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 5, pp. 678–693, May 2004.

[13] Y.-T. Li, Z. Bai, Y. Su, and X. Zeng, "Model order reduction of parameterized interconnect networks via a two-directional Arnoldi process," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 9, pp. 1571–1582, Sep. 2008.

[14] X. Li, P. Li, and L. Pileggi, "Parameterized interconnect order reduction with explicit-and-implicit multi-parameter moment matching for inter/intra-die variations," in *Proc. Int. Conf. Comput. Aided Design*, Nov. 2005, pp. 806–812.

[15] Z. Hao and G. Shi, "Sensitivity approach to statistical signal integrity analysis of coupled interconnect trees," in *Proc. IEEE Midwest Symp. Circuits Syst.*, Aug. 2009, pp. 212–215.

[16] Z. Hao and G. Shi, "Symbolic techniques for statistical timing analysis of RCL mesh networks with resistor loops," in *Proc. Int. Symp. Integr. Circuits*, Dec. 2009, pp. 470–473.

[17] Z. Hao and G. Shi, "A fast symbolic computation approach to statistical analysis of mesh networks with multiple sources," in *Proc. Asia South Pacific Design Autom. Conf.*, Jan. 2010, pp. 383–388.

[18] Q. Yu and E. Kuh, "Exact moment matching model of transmission lines and application to interconnect delay estimation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 3, no. 2, pp. 311–322, Jun. 1995.

[19] S. Akers, "Binary decision diagrams," *IEEE Trans. Comput.*, vol. 27, no. 6, pp. 509–516, Jun. 1978.

[20] R. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. Comput.*, vol. 35, no. 8, pp. 677–691, Aug. 1986.

[21] Q. Yu and E. Kuh, "Moment computation of lumped and distributed coupled RC trees with application to delay and crosstalk estimation," *Proc. IEEE*, vol. 89, no. 5, pp. 772–788, May 2001.

[22] H.-J. Lee, C.-C. Chu, and W.-S. Feng, "Moment computations of nonuniform distributed coupled RLC trees with applications to estimating crosstalk noise," in *Proc. Int. Symp. Qual. Electron. Design*, Mar. 2004, pp. 75–80.

[23] H.-J. Lee, C.-C. Chu, M.-H. Lai, and W.-S. Feng, "Moment computations of distributed coupled RLC interconnects with applications to estimating crosstalk noise," *IEICE Trans. Electron.*, vol. 88-C, no. 6, pp. 1186–1195, 2005.

[24] P. Restle, T. G. McNamara, D. A. Webber, P. J. Camporese, K. F. Eng, K. A. Jenkins, D. H. Allen, M. J. Rohn, M. P. Quaranta, D. W. Boerstler, C. J. Alpert, C. A. Carter, R. N. Bailey, J. G. Petrovick, B. L. Krauter, and B. D. McCredie, "A clock distribution network for microprocessors," *IEEE J. Solid-State Circuits*, vol. 36, no. 5, pp. 792–799, May 2001.

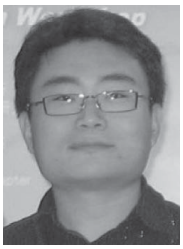
[25] A. Rajaram, J. Hu, and R. Mahapatra, "Reducing clock skew variability via crosslinks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 6, pp. 1176–1182, Jun. 2006.

[26] R. Rohrer, "Circuit partitioning simplified," *IEEE Trans. Circuits Syst.*, vol. 35, no. 1, pp. 2–5, Jan. 1988.

[27] H.-J. Lee, M.-H. Lai, C.-C. Chu, and W.-S. Feng, "Moment computations for RLC interconnects with multiple resistor loops using ROBDD techniques," in *Proc. IEEE Asia Pacific Circuits Syst. Conf.*, vol. 1. Dec. 2004, pp. 525–528.

[28] H.-J. Lee, M.-H. Lai, C.-C. Chu, and W.-S. Feng, "Applications of tree/link partitioning for moment computations of general lumped RLC networks with resistor loops," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 1. May 2004, pp. 713–716.

[29] H. Chen, C. Yeh, G. Wilke, S. Reddy, H. Nguyen, W. Walker, and R. Murgai, "A sliding window scheme for accurate clock mesh analysis," in *Proc. Int. Conf. Comput. Aided Design*, Nov. 2005, pp. 939–946.
[30] X. Ye, P. Li, M. Zhao, R. Panda, and J. Hu, "Analysis of large clock meshes via harmonic-weighted model order reduction and port sliding," in *Proc. Int. Conf. Comput. Aided Design*, Nov. 2007, pp. 627–631.
[31] A. Rajaram and D. Pan, "MeshWorks: An efficient framework for planning, synthesis and optimization of clock mesh networks," in *Proc. Asia South Pacific Design Autom. Conf.*, Mar. 2008, pp. 250–257.
[32] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 1, pp. 1–18, Jan. 2000.
[33] M. Golzar, N. Masoumi, and A. Atghiaee, "An efficient simulation CAD tool for interconnect distribution functions," in *Proc. 12th IEEE Workshop Signal Propag. Interconnects*, May 2008, pp. 1–4.
[34] *Predictive Technology Model*. (2005, Sep.) [Online]. Available: http://ptm.asu.edu/
[35] M. Shao, M. D. F. Wong, H. Cao, Y. Gao, L.-P. Yuan, L.-D. Huang, and S. Lee, "Explicit gate delay model for timing evaluation," in *Proc. Int. Symp. Phys. Design*, 2003, pp. 32–38.
[36] *Nangate 45 nm Open Cell Library*. (2008, Mar.) [Online]. Available: http://www.nangate.com/
[37] S.-D. Tan, "A general hierarchical circuit modeling and simulation algorithm," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 24, no. 3, pp. 418–434, Mar. 2005.
[38] J. Leeds and G. Ugron, "Simplified multiple parameter sensitivity calculation and continuously equivalent networks," *IEEE Trans. Circuit Theory*, vol. 14, no. 2, pp. 188–191, Jun. 1967.
[39] X. Ye, P. Li, and F. Liu, "Exact time-domain second-order adjoint-sensitivity computation for linear circuit analysis and optimization," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 57, no. 1, pp. 236–248, Jan. 2010.
[40] S. Pullela, N. Menezes, and L. T. Pillage, "Moment-sensitivity-based wire sizing for skew reduction in on-chip clock nets," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 16, no. 2, pp. 210–215, Feb. 1997.
[41] C. Alpert, A. Devgan, and C. Kashyap, "RC delay metrics for performance optimization," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 5, pp. 571–582, May 2001.
[42] K. Agarwal, D. Sylvester, and D. Blaauw, "A simple metric for slew rate of RC circuits based on two circuit moments," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 9, pp. 1346–1354, Sep. 2004.
[43] C. Kashyap, C. Alpert, F. Liu, and A. Devgan, "Closed-form expressions for extending step delay and slew metrics to ramp inputs for RC trees," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 4, pp. 509–516, Apr. 2004.
[44] M. Becer, D. Blaauw, R. Panda, and I. Hajj, "Early probabilistic noise estimation for capacitively coupled interconnects," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 3, pp. 337–345, Mar. 2003.
[45] J.-K. Zeng and C.-P. Chen, "Deep submicron interconnect timing model with quadratic random variable analysis," in *Proc. Conf. Design Autom. Test Eur.*, Mar. 2008, pp. 1091–1094.



Zhigang Hao (S'08) received the B.S., M.S., and Ph.D. degrees in electronics engineering from Shanghai Jiao Tong University, Shanghai, China, in 2005, 2008, and 2012, respectively.

He is currently a Staff Engineer with MediaTek Singapore, Singapore. He was a Visiting Student with the Department of Electrical Engineering, University of California, Riverside, from January 2010 to June 2011, and with the Graduate School of Information, Production and Systems, WASADA University, Tokyo, Japan, from August 2011 to September 2011. His current research interests include symbolic interconnect analysis, symbolic analog circuit analysis, and statistical power analysis.

Guoyong Shi (S'99–M'02–SM'11) received the Bachelor's degree in applied mathematics from Fudan University, Shanghai, China, the M.S. degree in electronics and information science from the Kyoto Institute of Technology, Kyoto, Japan, and the Ph.D. degree in electrical engineering from Washington State University, Pullman, in 1987, 1997, and 2002, respectively.

He is currently a Professor with the School of Microelectronics, Shanghai Jiao Tong University, Shanghai. Before joining Shanghai Jiao Tong University in 2005, he was a Post-Doctoral Research Scientist with the Department of Electrical Engineering, University of Washington, Seattle. He is the author or co-author of over 60 technical articles in the areas of systems, control, and integrated circuits. His current research interests include the development of design automation tools for analog, mixed-signal, and radio-frequency integrated circuits and systems.

Dr. Shi was the co-recipient of the Donald O. Pederson Best Paper Award from the IEEE Circuits and Systems Society in 2007.

Sheldon X.-D. Tan (S'96–M'99–SM'06) received the B.S. and M.S. degrees in electrical engineering from Fudan University, Shanghai, China, in 1992 and 1995, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Iowa, Iowa City, in 1999.

He is a Professor with the Department of Electrical Engineering, University of California, Riverside (UC Riverside). He is the Associate Director of the Computer Engineering Program with the Bourn College of Engineering, UC Riverside. He has co-authored three books. His current research interests include statistical modeling, simulation and optimization of mixedsignal, radio frequency, analog circuits, fast thermal analysis, modeling, and dynamic thermal management for microprocessors and platform systems, parallel circuit simulation techniques based on graphics processing unit and multicore systems, and embedded system designs based on field-programmable gate array platforms.

Dr. Tan currently serves as an Associate Editor for three journals, the *ACM Transaction on Design Automation of Electronic Systems, Integration,* the *VLSI Journal*, and the *Journal of VLSI Design*. He was a recipient of the NSF CAREER Award in 2004, the Best Paper Award from the IEEE International Conference on Computer Design (ICCD) in 2007, and the Best Paper Award from the IEEE/ACM Design Automation Conference in 1999. He served as a Technical Program Committee Member for DAC, ICCAD, ASPDAC, ICCD, ISQED, BMAS, and ASICON.

Esteban Tlelo-Cuautle (S'94–M'96–SM'04) received the B.Sc. degree from the Instituto Tecnologico de Puebla, Puebla, Mexico, in 1993, and the M.Sc. and Ph.D. degrees from the Instituto Nacional de Astrofisica, Optica y Electronica, Tonantzintla, Mexico, in 1995 and 2000, respectively.

His current research interests include systematic synthesis and behavioral modeling and simulation of linear and nonlinear circuits and systems, chaotic oscillators, symbolic analysis, multi-objective evolutionary algorithms, and analog and RF and mixed-signal design automation tools.

Dr. Tlelo-Cuautle serves as a reviewer in more than 20 high impact-factor journals, among them the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN of Integrated Circuits and Systems, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, and *Circuits, Systems and Signal Processing*. He has been a member of program committees in six prestigious international conferences and a reviewer for more than 20 internationally recognized conferences, among them IEEE ISCAS, IEEE APCCAS, IEEE SMACD, and IEEE MWSCAS.