A General Hierarchical Circuit Modeling and Simulation Algorithm

Sheldon X.-D. Tan, *Member, IEEE*

***Abstract*—This paper proposes a new hierarchical circuit modeling and simulation technique in s-domain for linear analog and interconnect circuits. The new method is based on a graph-based symbolic hierarchical circuit decomposition scheme. It can derive the exact or approximate admittances in the reduced circuit matrix and compute the circuit characteristics in rational function forms for very large linear analog and interconnect circuits. We show that the exact symbolic expressions of a circuit can be obtained by finding the cancellation-free expressions from the same circuit with hierarchical definitions. Some theoretical results are characterized for the presence and conditions of cancellations in the symbolic expressions from the subcircuit reduction. A novel decancellation strategy based on a graph-based hierarchical decomposition process is proposed and canceling terms are removed both symbolically and numerically to obtain the order-reduced circuit models. The proposed method can be used for modeling and simulation of any passive or active linear circuit, which makes our method very attractive for modeling both analog circuits and resistance–capacitance–inductance interconnect circuits in both frequency and time domain. An example *RC* circuit is illustrated and experimental results with some large analog and interconnects circuits are presented to validate the proposed method. Our experimental results also show that subcircuit (multiple-node) reduction scheme in general is better than one-node reduction methods such as Y–Δ transformation in terms of CPU time and memory usage.**

***Index Terms*—Behavioral modeling, circuit simulation, determinant decision diagrams, matrix determinant, model order reduction.**

NOMENCLATURE

Composite admittance	Circuit matrix element created during subcircuit reduction, which typically takes the form $(a^{BI}_{u,k_1}(s)\Delta^{II}_{k_2,k_1}(s)a^{IB}_{k_2,v}(s))/\det(A^{II})$ in (4) and (6).
Admittance order	Number of the basic admittances of circuit devices (like $1/R_1$, sC_2, etc.) in any product term in the denominators of those generated composite admittances [$a^{BB*}_{u,v}(s)$ and $b^{B*}_{u}(s)$ in (4) and (6)] in the reduced matrix.
Term cancellation	Cancellation between two symbolic terms, which contain the exact constituent symbols but opposite signs.
Common-factor cancellation	Cancellation between the numerator polynomial and the denominator polynomial of a symbolic rational expression due to a symbolic common factor between the two polynomials.
DDDs	Determinant decision diagrams where each DDD node represents an element in a circuit matrix.
YDDDs	Y (parameter) expanded determinant decision diagrams where each DDD node represents a stamped device parameter or a composite admittance.
s-expanded DDDs	Determinant decision diagrams, where each DDD node represents a device parameter and each DDD tree represents a coefficient of an s polynomial obtained from a determinant.
Subcircuit/ node reduction/ suppression	Reduction of circuit matrix via matrix Schur decomposition.
Internal matrix	Matrix to be reduced and does not include the boundary nodes. It is usually represented by A^{II} in this paper.

Manuscript received September 26, 2003; revised February 15, 2004 and June 14, 2004. The work was supported in part by the University of California (UC) Regent's Faculty Fellowship, in part by the UC Senate Research Fund, in part by Cadence Design Systems Inc., and in part by the Semiconductor Research Corporation CSR fund. This paper was recommended by Associate Editor S. Sapatnekar.

The author is with Department of Electrical Engineering, University of California, Riverside, CA 92521 USA (e-mail: stan@ee.ucr.edu).

Digital Object Identifier 10.1109/TCAD.2004.842815

I. INTRODUCTION

MODELING and simulation of linear analog circuits in both frequency (s-domain) and time domain are critical for top-down constraint-driven design methodology for mixed-signal system-on-a-chip (SoC) designs [9] and interconnect-centered physical design and optimizations [3], [12].

Due to the importance of on-chip global interconnects like power/ground grids, global signal nets, and clock trees, a number of projection-based model-order reduction-based techniques have been introduced [5]–[7], [14], [16], [23], [24] to analyze the transient behavior of interconnects. Asymptotic waveform evaluation (AWE) algorithm was first proposed [16], [17], where explicit moment matching was used to compute the dominant poles at low frequencies. However, the AWE method is numerically unstable for higher-order approximation. Thereafter, a number of other projection-based model-order reduction methods based on implicit moment matching (via Krylov subspace projection) were developed. Examples are: Padé via Lanczos (PVL) [5], Matrix PVL [6], the Arnoldi method [24], the Arnoldi transformation method [23], PRIMA [14], and the SyPVL algorithm [7]. Those projection-based algorithms mainly work for passive linear networks as the computation of moments and Krylov space base vectors requires a special partitioning of circuit matrices and solving of the partitioned circuit matrices iteratively, which may not be

possible for general active networks with controlled sources seen in many analog circuits. Also, projection-based methods are not efficient for circuits with many independent sources as its time complexity is dependent on the number of independent sources or external ports.

Another approach to circuit complexity reduction is by means of local node reduction. The main idea is to reduce the number of nodes in the circuits and approximate the newly added elements in the circuit matrix in reduced rational forms. The major advantage of those methods over projection-based methods is that the reduction can be done in a local manner and no overall solutions of the whole circuit are required (with some circuit realization/synthesis techniques), which makes those methods very amenable to attacking large linear networks. This idea has been explored by approximate Gaussian elimination for *RC* circuits [4], by the TICER program [20], which is also based on Gaussian elimination, but it only keeps first two moments, and by the extension of TICER method into resistance–inductance–capacitance (*RLC*) circuits [1]. The rational approximation is also explored by the direct truncation of the transfer function algorithm (DTT) [11] for tree-structured *RLC* circuits and by an extended DTT method for nontree structured *RLC* circuits [29].

Recently a more general topology based node-reduction method was proposed [18], [19], in which nodes are reduced one at a time (topologically, it is called Y-Δ transformation) and the generated admittance in the reduced network is represented as an order-reduced rational function of s. This method is equivalent to symbolic Gaussian elimination (s is the only symbol), but the reduction is done on circuit topologies only, which is equivalent to the nodal analysis (NA) formulation of a circuit only. The stability is enforced by Hurwitz polynomial approximation. But this method only works for linear circuits with limited element types (RCLK-VJ) and cannot be applied to reduce general linear circuits due to NA formulation requirement.

For both existing projection-based and node reduction-based algorithms, the major problem is that they mainly work for interconnect circuits modeled as passive *RLC* circuits with limited capacity for mutual inductors. For active linear analog circuits with controlled sources, those reduction methods cannot be applied in general. More importantly, as mutual inductive effects become more and more pronounced, more effective mutual inductance models, like the vector potential equivalent circuit (VPEC) model [15], the improved VPEC model [35], and wire duplication models [37] begin to emerge. Those new mutual inductor models contain controlled sources, which makes them difficult for reduction by existing model-order reduction techniques.

In this paper, we propose a new modeling and simulation technique via hierarchical symbolic analysis technique. The new algorithm can reduce the circuit complexities by applying general symbolic subcircuit reduction and keeping the generated admittances in the reduced circuit matrices in the exact or approximate rational function forms. The new method performs the node reduction directly on the circuit matrices and, hence, it is general enough to reduce any linear circuits formulated by more general modified NA (MNA) method. Furthermore, instead of reducing one node at a time, as done in [18], the new method allows elimination of multiple nodes simultaneously. This leads to a general hierarchical s-domain analysis technique as we can suppress subcircuits, which consist of a number of nodes, one at a time in a hierarchical and an independent way.

The major differences between the proposed method and the Y-Δ transformation are listed below.

1) **Reduction Methodology:** Both methods essentially are based on the Gaussian elimination process to perform the node reduction. The hierarchical method is a more general version of the Y-Δ reduction method as we can reduce multiple nodes at a time, which is shown to be more efficient than reducing one node at a time.
2) **Circuits Can Be Reduced:** The hierarchical reduction method can be applied to any linear circuit with any controlled and independent source. While Y-Δ method can only solve certain circuits with limited independence sources (like current sources). Circuit elements whose stamps can not be written in the admittance form cannot be used in the Y-Δ method due to the nodal formulation (NA) requirement.
3) **Circuit Formulation:** The hierarchical method is based on more general MNA formulation and works directly on circuit matrix. The Y-Δ method is based on NA and works on the circuit topology directly, which limits its applications.
4) **Implementations:** The hierarchical method is a general version of the Y-Δ method. It uses a special graph-based (DDD graphs) technique to compute the rational function of a determinant. It also allows the symbolic term decancellation to improve the accuracy of the reduction process during the computation process. In contrast, the Y-Δ method does not have to deal with determinant and can thus be implemented more easily. However, all the cancellations have to be done numerically in the Y-Δ method. The major implementation overhead of the hierarchical method over the Y-Δ method is the DDD graph related data structures and general subcircuit related data structures and corresponding operations.

The new method is thus able to exploit the naturally hierarchical structures inherent in many linear circuits. Such a hierarchical node reduction is made possible by means of a new graph-based hierarchical symbolic analysis technique for computing the new admittances and removing the canceling terms and common factors (decancellation) in determinants [21], [22]—the key operations in the new hierarchical modeling and simulation algorithm. Hurwitz polynomial approximation can be performed on the order-reduced transfer functions to enforce stability. The resulting algorithm can perform efficient s-domain and time-domain analysis on any linear active or passive network with very high accuracy.

This paper is organized as follows. Section III reviews the matrix-based node-reduction approach and concepts of DDDs and DDD-based hierarchical decomposition method for symbolic analog circuit analysis. Section IV introduces a new symbolic hierarchical decomposition method. Section V first illustrates

the cancellation problems by an *RC* circuit example and then presents some theoretical results for the conditions of the cancellation in subcircuit reduction processes. Section VI presents the new approach to the circuit complexity reduction on linear circuits. Experimental results are described in Section VII. Section VIII concludes the paper.

II. Subcircuit Reduction and DDD-Based Hierarchical Decomposition

A. Subcircuit Reduction

In this section, we briefly review how nodes in a subcircuit can be suppressed in matrix form. Consider a subcircuit with some internal structures and terminals. The circuit unknowns—the node-voltage variables and branch-current variables—can be partitioned into three disjoint groups x^I, x^B, and x^R, where the superscripts I, B, and R stand for, respectively, *internal* variables, *boundary* variables, and the *remaining* variables. Internal variables are those local to the subcircuit, boundary variables are those related to both the subcircuit and the rest of the circuit. Note that boundary variables include those variables required as the circuit inputs and outputs. With this, the system-equation set $Ax = b$, can be rewritten in the following form:

$$\begin{bmatrix} A^{II} & A^{IB} & 0 \\ A^{BI} & A^{BB} & A^{BR} \\ 0 & A^{RB} & A^{RR} \end{bmatrix} \begin{bmatrix} x^I \\ x^B \\ x^R \end{bmatrix} = \begin{bmatrix} b^I \\ b^B \\ b^R \end{bmatrix}. \tag{1}$$

The matrix A^{II} is the *internal* matrix associated with internal variable vector x^I. In the following discussions, we assume the submatrix A^{II} is not singular as $\det(A^{II})$ is computed. Otherwise, we have to redefine the internal matrix such that A^{II} is not singular.

Subcircuit suppression (also called Schur decomposition) is used to eliminate all the variables in x^I, and to transform (1) into the following reduced set of equations:

$$\begin{bmatrix} A^{BB*} & A^{BR} \\ A^{RB} & A^{RR} \end{bmatrix} \begin{bmatrix} x^B \\ x^R \end{bmatrix} = \begin{bmatrix} b^{B*} \\ b^R \end{bmatrix} \tag{2}$$

where

$$A^{BB*} = A^{BB} - A^{BI}(A^{II})^{-1}A^{IB} \tag{3}$$

$$= A^{BB} - \frac{1}{\det(A^{II})}A^{BI}[\Delta^{II}_{u,v}]^T A^{IB} \tag{4}$$

where $[\Delta_{u,v}]^T$ is called the *adjoint* matrix of A, $\Delta_{u,v}$ is the first-order *cofactor* of $\det(A)$ with respect to $a_{u,v}$, and is defined as $\Delta_{u,v} = (-1)^{(u+v)}\det(A_{a_{u,v}})$ and matrix $A_{a_{u,v}}$ is the $(n-1)\times(n-1)$-matrix obtained from matrix A by deleting row u and column v. A^{BB*} is also called Schur complement [8]. We also have

$$b^{B*} = b^B - A^{BI}(A^{II})^{-1}b^I \tag{5}$$

$$= b^B - \frac{1}{\det(A^{II})}A^{BI}[\Delta^{II}_{u,v}]^T b^I. \tag{6}$$

Subcircuit suppression can be performed for all the subcircuits by visiting the circuit hierarchy in a bottom-up fashion. Suppose that the number of internal variables is m, and the number of boundary variables is l. Equations (4) and (6) can be written in the following expanded forms:

$$a^{BB*}_{u,v}(s) = a^{BB}_{u,v}(s) - \frac{1}{\det\left(A^{II}(s)\right)} \times \sum_{k_1,k_2=1}^{m} a^{BI}_{u,k_1}(s)\Delta^{II}_{k_2,k_1}(s)a^{IB}_{k_2,v}(s) \tag{7}$$

where $u, v = 1, \ldots, l$ and

$$b^{B*}_u(s) = b^B_u(s) - \frac{1}{\det\left(A^{II}(s)\right)}\sum_{k_1,k_2=1}^{m} a^{BI}_{u,k_1}(s)\Delta^{II}_{k_2,k_1}(s)b^I_{k_2}(s) \tag{8}$$

where $u = 1, \ldots, l$. From (7) and (8), we can observe that admittance $a^{BB*}_{u,v}$ and input stimuli b^{B*}_u at boundary nodes will become rational functions of s once the subcircuit is suppressed. The $(a^{BI}_{u,k_1}(s)\Delta^{II}_{k_2,k_1}(s)a^{IB}_{k_2,v}(s))/\det(A^{II})$ terms are called *composite admittances* as they are the admittances generated during subcircuit reduction.

In order to obtain the rational functions for $a^{BB*}_{u,v}(s)$ and $b^{B*}_u(s)$, we need to compute the rational function for a determinant whose elements may again be rational functions of s. This task can be achieved via determinant decision diagrams (DDD's), which were proposed originally for symbolic analysis of large analog circuits [21], [30]. We will show how the DDD graphs can be modified to compute the rational function for a determinant in the later sections.

An important issue is that if the symbolic expressions are kept (s-expressions are special symbolic expressions), the final expressions of the generated rational admittances are not free from cancellation as common factors between the numerator and denominator of the resulting rational admittances in $a^{BB*}_{u,v}(s)$ and $b^{B*}_u(s)$ will be generated (common-factor cancellation) or sum of two symbolic terms equals to zero (term cancellation). This has been observed already when nodes are reduced one at a time in $Y-\Delta$ transformation algorithme [18]. Fundamentally, we will show in Section V that all the cancellations are caused by subcircuit reductions. The conditions for common-factor cancellations and term cancellations will be explained in more details in Theorem 2 and Theorem 3.

Such s-expressions do not represent the real order of the system, which makes the synthesis or realization of the reduced system very difficult, if possible at all. Also, if we only keep a limited order of s for each rational function of s during the subcircuit reduction process, we may end up with nonphysical coefficients in the resulting rational admittance of s. As a result, it is important to obtain the cancellation-free rational admittance expressions. As we can see in the later sections of this paper, cancellation-free rational admittances are directly related to the exact admittance expressions computed from the flattened circuit matrix and are also very important to the new decancellation method.

To be more precise on cancellation-free expressions, we define *admittance order*, which is the number of the basic admittances of circuit devices (like $1/R_1$, sC_2 etc.) in any product term in the denominators of those newly generated admittance terms ($a_{u,v}^{BB*}(s)$ and $b_u^{B*}(s)$) in the reduced matrix. Actually according to the definition of a determinant, it can be seen that the number of device admittances equals the dimension of the determinant $\det(A^{II}(s))$. However, if all the nodes internal to the submatrix A^{II} are reduced by different subcircuits at different circuit hierarchical levels, the resulting admittance order in each new term ($a_{u,v}^{BB*}(s)$ or $b_u^{B*}(s)$) will be larger than the real admittance order if common-factors are not removed.

Therefore, how to remove those common factors (decancellation) in the rational functions of $a_{u,v}^{BB*}(s)$ and $b_u^{B*}(s)$ in general becomes a key issue for the subcircuit reduction process. Fortunately, such cancellation-free rational functions can be obtained by efficient DDD graph operations. The DDD concept will be briefly reviewed in the following subsection.

B. DDD Graph and DDD-Based Hierarchical Decomposition

DDDs [21] are compact and canonical graph-based representation of determinants. A DDD is a signed, rooted, directed acyclic graph with two terminal vertices, namely, the *0-terminal* vertex and the *1-terminal* vertex. Each nonterminal DDD vertex is labeled by a symbol in the determinant denoted by a_i, and a positive or negative sign denoted by $s(a_i)$. It *originates* two outgoing edges, called *1-edge* and *0-edge*. Each vertex a_i represents a symbolic expression $D(a_i)$ defined recursively as follows: $D(a_i) = a_i\ s(a_i)\ D_{a_i} + D_{\overline{a}_i}$, where D_{a_i} and $D_{\overline{a}_i}$ represent, respectively, the symbolic expressions of the vertices pointed by the 1-edge and 0-edge of a_i. The 1-terminal vertex represents expression 1, whereas the 0-terminal vertex represents expression 0. Such a DDD is called a complex DDD as each DDD node is a function of complex frequency variable s.

The idea of DDD-based hierarchical decomposition [30] is to represent all the determinants and cofactors in (7) and (8), as well as those determinants and cofactors in the final transfer functions of the desired circuit characteristics. Due to the compactness of DDD graphs, DDD-based hierarchical circuit decomposition was shown to be superior to the previous methods [10], [25].

For our problem, some elements (in admittance forms) of a circuit matrix will become a rational function of s during the hierarchical subcircuit reduction process. As a result, the construction of the rational function for such a determinant will be different from the method used for s-expanded DDDs [22]. A new DDD graph, called Y-expanded DDD (YDDD), is introduced for this task, which will be explained in Section VI.

III. New DDD-Based Hierarchical Decomposition

In this section, we give a new DDD-based hierarchical decomposition scheme. First, we introduce the following theorem [27].

Theorem 1: Assuming $det(A^{II}) \neq 0$, (7) can be written in the following form:

$$a_{u,v}^{BB*} = \frac{\det(A[1,\ldots,m,u|1,\ldots,m,v])}{\det(A^{II})} \tag{9}$$

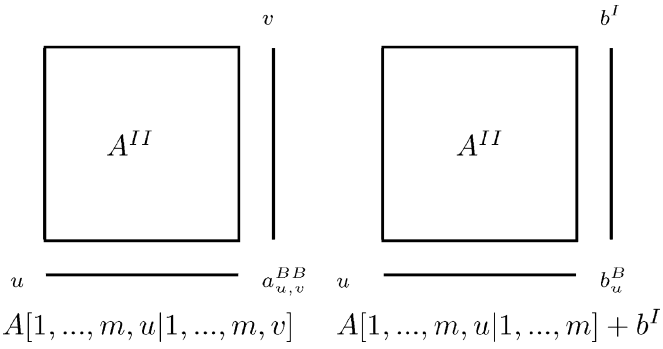


Fig. 1. Illustration of Theorem 1.

where $u, v = 1,\ldots,l$ and (8) can be written in the following form:

$$b_u^{B*} = \frac{\det(A[1,\ldots,m,u|1,\ldots,m]+b^I)}{\det(A^{II})} \tag{10}$$

where, $A[1,\ldots,m,u|1,\ldots,m,v]$ is the submatrix that consists of matrix A^{II}, which actually is $A[1,\ldots,m|1,\ldots,m]$, plus row u and column v of matrix A; $A[1,\ldots,m,u|1,\ldots,m]+b^I$ is the submatrix that consists of matrix A^{II} plus row u of matrix A and the right-hand side column b^I. The $A[1,\ldots,m,u|1,\ldots,m,v]$, and $A[1,\ldots,m,u|1,\ldots,m]+b^I$ are illustrated in Fig. 1.

Proof: Let us first take a look at (9). We can first develop $A[1,\ldots,m,u|1,\ldots,m,v]$ along row u to obtain a number of first-order cofactors of $A[1,\ldots,m,u|1,\ldots,m,v]$. We then develop those first-order cofactors along column v and we will end up with the same expression in (7). The consistent negative sign in each $a_{u,k_1}^{BI}\Delta_{k_2,k_1}^{II}a_{k_2,v}^{IB}/\det(A^{II})$ term in (7) is due to the fact that signs of first-order factors and the corresponding second-order cofactors are always opposite in such a development order. The same is true for (10).

We notice that (9) can also be derived from a general identity [2]

$$\det(H^{BB*}[i_1,\ldots,i_s|j_1,\ldots,j_s]) = \frac{det(A[1,\ldots,m,i_1,\ldots,i_s|1,\ldots,m,j_1,\ldots,j_s])}{det(A^{II})}$$

where H^{BB*} is an $s \times s$ submatrix of the Schur complement A^{BB*} and $A^{II} = A[1,\ldots,m|1,\ldots,m])$. When H^{BB*} becomes a 1×1 matrix, we obtain (9).

With Theorem 1, we only need to compute the rational function for the numerator, which is a determinant (as $\det(A^{II})$ shared by all the newly created matrix elements for each subcircuit) for each boundary-variable related element in the reduced circuit matrix instead of representing each individual first-order cofactor of $\det(A^{II})$ explicitly [30]. This leads to the new DDD-based hierarchical decomposition method. Such determinant-only DDD representation for all the involved matrix elements is more compact than the previous method [30] as first-order cofactors are not explicitly represented. Therefore, more sharing of common terms becomes possible among those cofactors and elements in row u, columns v and b^I. More importantly, such DDD-only representation of new admittances is more amenable to removing cancellation during subcircuit reduction process as shown in the next section.

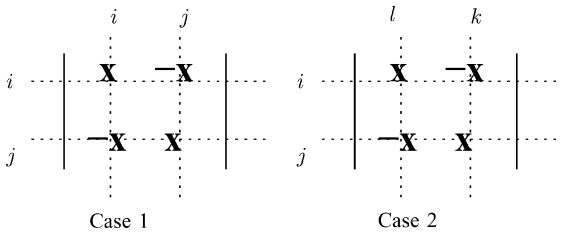


Fig. 2. Matrix patterns causing term cancellation.

IV. Cancellation Analysis for Subcircuit Reduction

In this section, we explain how cancellation happens due to circuit device and due to subcircuit reduction as well as their relationship in MNA formulation. Some theoretical results will be presented.

A. Cancellation due to Circuit Devices

The common factors will be introduced when the same circuit parameter will appear more than once in a circuit matrix in MNA formulation for a subcircuit. Two cancellation patterns are shown in Fig. 2.

Case 1 comes from the rectangular appearance of a floating resistor/conductor and case 2 reflects the pattern from a voltage controlled current source (VCCS) in MNA formulation [31]. As a result, all the product terms consisting of elements at (i, i) and at (j, j) will cancel all the terms consisting of elements at (i, j) and at (j, i) for case 1. The same is true for Case 2. Such cancellation can lead to numerical errors if they are carried out numerically in general. However, those errors may only affect very high-order terms based on our observations. In our algorithm, we have an better approach to this problem as term cancellations due to case 1 and case 2 in Fig. 2 and other general cases due to circuit reduction shown later can be symbolically removed during YDDD construction for a determinant [28]. A detailed description of the algorithm for general term decancellation is given in Section VI-C1.

As we can see later that the device-level cancellations coming from the NA or MNA formulation are essentially caused by subcircuit reduction discussed in the next section.

B. Cancellation due to Subcircuit Reduction

Once a subcircuit has been suppressed, new cancellation patterns are created. Let us illustrate this concept through a simple

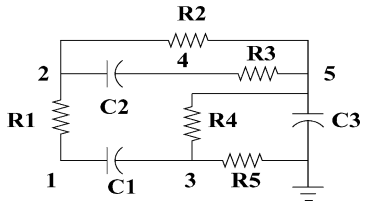


Fig. 3. Simple *RC* circuit.

RC circuit shown in Fig. 3. Assume all the *R* and *C* devices have unit value, then the circuit matrix of the circuit is shown in (11)

$$\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{bmatrix} 1+s & -1 & -s & 0 & 0 \\ -1 & 2+s & 0 & -s & -1 \\ -s & 0 & 2+s & 0 & -1 \\ 0 & -s & 0 & 1+s & -1 \\ 0 & -1 & -1 & -1 & 3+s \end{bmatrix}.$$

Let us suppress node 1 (v_1) first, the resulting circuit matrix becomes

$$\begin{array}{c} v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{bmatrix} 2+s-\frac{1}{1+s} & -\frac{s}{1+s} & -s & -1 \\ -\frac{s}{1+s} & 2+s-\frac{s^2}{1+s} & 0 & -1 \\ -s & 0 & 1+s & -1 \\ -1 & -1 & -1 & 3+s \end{bmatrix}. \quad (11)$$

It can be seen that the product term of $-1/(1+s)$ and $-s^2/(1+s)$ will cancel the product term of $-s/(1+s)$ and $-s/(1+s)$. This is called *term cancellation* during the subcircuit reduction process in this paper. If we further reduce the node 2 (v_2), the resulting circuit matrix will become

$$\begin{array}{c} v_3 \\ v_4 \\ v_5 \end{array} \begin{bmatrix} 2+s-\frac{2s^2+s^3}{1+3s+s^2} & -\frac{s^2}{1+3s+s^2} & -1-\frac{s}{1+3s+s^2} \\ -\frac{s^2}{1+3s+s^2} & 1+s-\frac{s^2+s^3}{1+3s+s^2} & -1-\frac{s+s^2}{1+3s+s^2} \\ -1-\frac{s}{1+3s+s^2} & -1-\frac{s+s^2}{1+3s+s^2} & 3+s-\frac{1+s}{1+3s+s^2} \end{bmatrix}. \quad (12)$$

First, we notice that the denominators in the composite admittances actually are $\det(A^{II}) = (1+s)(2+s) - (+1)(+1) = 1 + 3s + s^2$, which is the determinant of the subcircuit matrix consisting of node 1 and 2 only [first two rows and columns in the (11)]. Also, all the numerators in those composite admittances agree with (7). As a result, we can rewrite (12) as in (13) at the bottom of the page.

Equation (13) tells us that the resulting admittances in the reduced circuit are the same regardless of how nodes are reduced (one-by-one or subcircuit-by-subcircuit). For this reduced matrix, cancellation becomes more complicated and both term cancellation and common-factor cancellation exist.

$$\begin{array}{c} v_3 \\ v_4 \\ v_5 \end{array} \begin{bmatrix} 2+s-\frac{(-s)\Delta_{11}^{II}(-s)}{\det(A^{II})} & -\frac{(-s)\Delta_{21}^{II}(-s)}{\det(A^{II})} & -1-\frac{(-s)\Delta_{21}^{II}(-1)}{\det(A^{II})} \\ -\frac{(-s)\Delta_{12}^{II}(-s)}{\det(A^{II})} & 1+s-\frac{(-s)\Delta_{22}^{II}(-s)}{\det(A^{II})} & -1-\frac{(-s)\Delta_{22}^{II}(-1)}{\det(A^{II})} \\ -1-\frac{(-1)\Delta_{12}^{II}(-s)}{\det(A^{II})} & -1-\frac{(-1)\Delta_{22}^{II}(-s)}{\det(A^{II})} & 3+s-\frac{(-1)\Delta_{22}^{II}(-1)}{\det(A^{II})} \end{bmatrix} \quad (13)$$

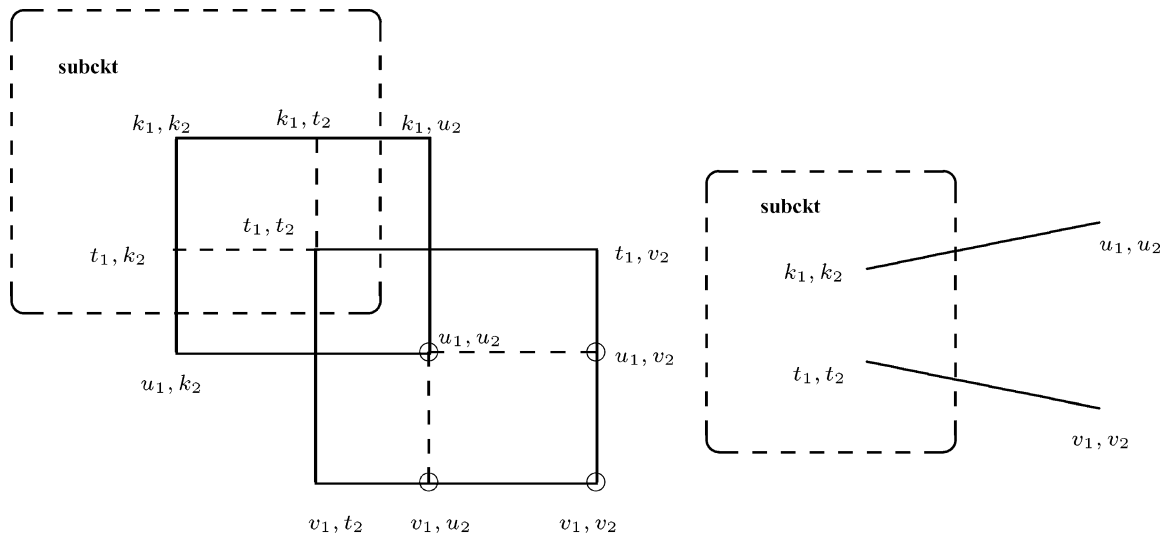


Fig. 4. Cancellation pattern due to subcircuit reduction.

For instance, if we look at the four composite admittances at rows v_3 and v_4 and columns v_3 and v_4 in (12), we will have

$$\begin{aligned}-\frac{2s^2+s^3}{\det(A^{II})}\times-\frac{s^2+s^3}{\det(A^{II})}-&-\frac{s^2}{\det(A^{II})}\times-\frac{s^2}{\det(A^{II})}\\ &=\frac{s^4(2+3s+s^2)-s^4}{\left(\det(A^{II})\right)^2}\\ &=\frac{s^4\det(A^{II})}{\det(A^{II})^2}=\frac{s^4}{\det(A^{II})}.\end{aligned}\tag{14}$$

So $\det(A^{II})$ becomes the common factor of the resulting rational admittance. This is called *common factor cancellation* in the paper.

In general, after a nonsingular subcircuit (whose $\det(A^{II})\neq 0$) is reduced both term cancellation and common-factor cancellation will be generated. For common-factor cancellation, it can be more clearly explained graphically in Fig. 4. Nodes at (k_1, k_2) and at (t_1, t_2) are the internal nodes in a subcircuit to be suppressed. Nodes at (u_1, u_2) and at (v_1, v_2) are boundary nodes. Their connections with the two internal nodes are shown in the right-hand side of Fig. 4. As a result, admittances a_{u_1,k_2}, a_{k_1,u_2}, a_{v_1,t_2} and a_{t_1,v_2} are not zero. According to (7) we have,

$$a'_{v_1,u_2}=a_{v_1,u_2}-\frac{a_{v_1,t_2}\Delta^{II}_{k_1,t_2}a_{k_1,u_2}}{\det(A^{II})}\tag{15}$$

$$a'_{u_1,v_2}=a_{u_1,v_2}-\frac{a_{u_1,k_2}\Delta^{II}_{t_1,k_2}a_{t_1,v_2}}{\det(A^{II})}\tag{16}$$

$$a'_{u_1,u_2}=a_{u_1,u_2}-\frac{a_{u_1,k_2}\Delta^{II}_{k_1,k_2}a_{k_1,u_2}}{\det(A^{II})}\tag{17}$$

$$a'_{v_1,v_2}=a_{v_1,v_2}-\frac{a_{v_1,t_2}\Delta^{II}_{t_1,t_2}a_{t_1,v_2}}{\det(A^{II})}\tag{18}$$

where, $a'_{x,y}$ is the new admittance at (x, y). $\det(A^{II})$ is the internal determinant of the subcircuit suppressed and $\Delta^{II}_{x,y}$ is the first-order cofactor of $\det(A^{II})$ with respect to element (x, y). Since the four new composite admittances appear in two rows and two columns, they will cause new cancellation when node (u_1, u_2) and (v_1, v_2) are suppressed as the internal nodes in the upper-level subcircuits. Specifically, we will have the following product terms in the internal determinant of the upper-level subcircuit according to the definition of a determinant (if those two nodes are suppressed at the same time):

$$\begin{aligned}a'_{u_1,u_2}a'_{v_1,v_2}-a'_{v_1,u_2}a'_{u_1,v_2}&=R_x+\frac{a_{u_1,k_2}a_{v_1,t_2}\Delta^{II}_{k_1,k_2}\Delta^{II}_{t_1,t_2}a_{k_1,u_2}a_{t_1,v_2}}{\det(A^{II})\det(A^{II})}\\ &\quad-\frac{a_{u_1,k_2}a_{v_1,t_2}\Delta^{II}_{k_1,t_2}\Delta^{II}_{t_1,k_2}a_{k_1,u_2}a_{t_1,v_2}}{\det(A^{II})\det(A^{II})}\end{aligned}\tag{19}$$

where R_x is the rest of other terms when the products of the new admittances are expanded. We have two scenarios to consider. First, if nodes (k_1, k_2) and (t_1, t_2) are the same node, i.e., $k_1=t_1$, $k_2=t_2$, the last two terms in (19) will cancel each other as $\Delta^{II}_{k_1,k_2}\Delta^{II}_{t_1,t_2}=\Delta^{II}_{k_1,t_2}\Delta^{II}_{t_1,k_2}$.

This is the term cancellation. Second, if those two nodes are different internal nodes, we will show that the combined last two terms in (19) has a common factor between the numerator and the denominator. Specifically,

$$\Delta^{II}_{k_1,k_2}\Delta^{II}_{t_1,t_2}-\Delta^{II}_{k_1,t_2}\Delta^{II}_{t_1,k_2}=\det(A^{II})\Delta_{k_1k_2,t_1t_2}.\tag{20}$$

As a result, we know that $\det(A^{II})$ is a common factor in the combined last two terms in (19), which will become

$$\frac{a_{u_1,k_2}a_{v_1,t_2}\Delta^{II}_{k_1,k_2,t_1,t_2}a_{k_1,u_2}a_{t_1,v_2}}{\det(A^{II})}$$

after the common factor $\det(A^{II})$ in the numerator and the denominator cancels. This actually is the common-factor cancellation.

Going back to the example in (14), we have

$$\begin{aligned}\Delta_{11}\Delta_{22}-\Delta_{21}\Delta_{12}&=(2+s)(1+2)-(+1)(+1)\\ &=\det(A^{II})\Delta_{11,22}\end{aligned}$$

where $\Delta_{11,22}=1$, which validates (20).

C. Theoretical Analysis of Cancellation Conditions

In general, if the composite admittances from the suppression of one subcircuit populate more than two rows and columns, their corresponding first-order cofactors Δ_{k_1,k_2} may also appear in more than two rows and columns in the submatrix A^{II}. We then have the following general theorem regarding the common-factor cancellation [26].

Theorem 2: Given a nonsingular matrix A and its $m \times m$ elements at rows $i_1, \ldots, i_m$ and columns $j_1, \ldots, j_m$ as shown in (21)

$$A = \begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & a_{i_1 j_1} & \cdots & a_{i_1 j_k} & \cdots & a_{i_1, j_m} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & a_{i_k j_1} & \cdots & a_{i_k j_k} & \cdots & a_{i_k, j_m} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & a_{i_m j_1} & \cdots & a_{i_m j_k} & \cdots & a_{i_m, j_m} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}. \quad (21)$$

We then have

$$\begin{vmatrix} \Delta_{i_1 j_1} & \cdots & \Delta_{i_1 j_k} & \cdots & \Delta_{i_1 j_m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \Delta_{i_k j_1} & \cdots & \Delta_{i_k j_k} & \cdots & \Delta_{i_k j_m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \Delta_{i_m j_1} & \cdots & \Delta_{i_m j_k} & \cdots & \Delta_{i_m j_m} \end{vmatrix} = \det(A)^{m-1} \Delta_{i_1 j_1, \ldots, i_m, j_m} \quad (22)$$

where $\Delta_{i_1 j_1, \ldots, i_m, j_m}$ is the mth-order cofactor with rows $i_1, \ldots, i_m$ and columns $j_1, \ldots, j_m$ removed from $\det(A)$ and is defined as

$$\Delta_{i_1 j_1, \ldots, i_m, j_m} = (-1)^{i_1 + \cdots + i_m + j_1 + \cdots + j_m} \det(A_{i_1 \ldots i_m, j_1 \ldots j_m}). \quad (23)$$

When all the rows and columns are removed from A, we have $\Delta_{11,\ldots,nn} = 1$.

Theorem 2 actually is the classic Jacobi matrix identity [8].[1] In the sequel, we show how the Jacobi matrix identity is linked to the common-factor cancellation issue encountered in the new hierarchical node circuit reduction algorithm. After the subcircuit reduction, there are many composite elements in the reduced matrix (2) and each of them contains one first-order cofactor of the suppressed submatrix A^{II}. Theorem 2 basically says that if all those first-order cofactors form a $m \times m$ matrix whose associated matrix elements are coming from m rows and m columns in the reduced submatrix A^{II} shown in (21), $\det(A^{II})^{(m-1)}$ will become the common factor between the numerator and the denominator of the sum-of-product term, which consists of all the product terms with m first-order cofactors in each product term when the determinant of the reduced matrix is computed. Note that if the common-factor is not removed, the denominator of each product term is $\det(A^{II})^m$ as $\det(A^{II})$ is the denominator for every composite admittance.

However, does such $m \times m$ first-order cofactor matrix always exist in the reduced matrix? We have the following corollary answering this question. Let A^{II} be the nonsingular submatrix in (1). Assume there exists a product term in the determinant of the reduced matrix in (2) and the product term consists of m first-order cofactors of $\det(A^{II})$.

[1]The author rediscovered this matrix identity independently during this research.

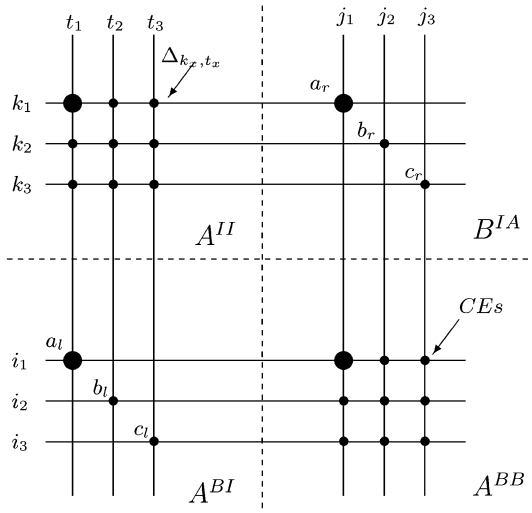


Fig. 5. Matrix pattern for a $m \times m$ full matrix of first-order cofactor.

Corollary 1: If all the first-order cofactors have unique row and column indices, there exists a full $m \times m$ first-order cofactor matrix as shown in (22) embedded in the reduced matrix where the product term is one of the terms in the determinant of the matrix. Also $\det(A^{II})^{(m-1)}$ will be the common factor between the numerator and the denominator of the sum-of-product rational expression of those product terms whose first-order cofactors form the $m \times m$ matrix.

Proof: First, we show that if there exists one product term that has m first-order cofactors of $\det(A^{II})$ and all the first-order cofactors have unique row and column indices that cover the m rows and m columns in the A^{II}, there exist $m! - 1$ other product terms that have also m first-order cofactors of $\det(A^{II})$. All the first-order cofactors in any of those product term have unique row and column indices that cover the same m rows and m columns in the A^{II} due to the symmetric distributions of boundary elements [a^{BI}_{u,k_1} in (7)].

This can be illustrated in Fig. 5, where $m = 3$ and matrix A^{II} and portions of matrix A^{BI}, A^{IB}, and A^{BB} are shown in (1). This figure gives a graphic view of the Schur decomposition. For instance, if a_l in A^{BI} and a_r in A^{IB} are not zero, then we will have new composite admittance generated at position (i_1, j_1) whose expression will be $a_l \Delta_{k_1,t_1} a_r / \det(A^{II})$. Δ_{k_1,t_1} is the first-order cofactor with respect to the element at row k_1 and column t_1, which are the row of a_r and the column of a_l respectively. The positions of a_l, a_r, $a_l \Delta_{k_1,t_1} a_r / \det(A^{II})$ and Δ_{k_1,t_1} are highlighted (larger solid circles) in the figure.

As a result, we know that there is a full 3×3 matrix of composite elements at the intersections of row i_1, i_2, i_3 and columns j_1, j_2, j_3 due to the existing of boundary elements a_l, b_l, c_l and a_r, b_r, c_r. Therefore, the first-order cofactors in those composite elements also form a full $m \times m$ matrix as shown in the A^{BB} section in Fig. 5. Such a symmetric distribution of boundary elements will exist if there exist one product term in which all those boundary elements are used and all its first order cofactors have unique row and column indices.

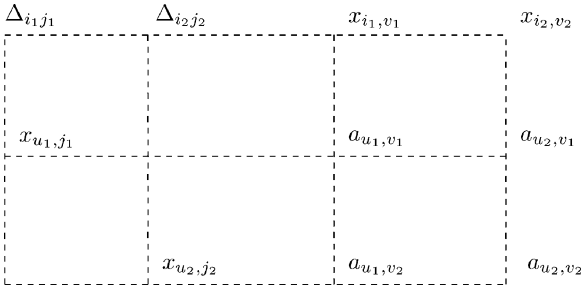


Fig. 6. Matrix patterns for first-order cofactors sharing the same row index.

Therefore, those first-order cofactors will form a full $m \times m$ determinant based on the definition of a determinant. Note that the symmetric property here does not require the circuit matrix to be symmetric as the rows and columns in the reduced matrix can be any m rows and m columns. Second, for a $m \times m$ first-order cofactor matrix, $det(A^{II})^{(m-1)}$ is the common factor of the rational expression of the sum of those product terms according to Theorem 2.

The corollary tells us that if there exists a term whose first-order cofactors come from different rows and columns in the submatrix A^{II}, there exist $m! - 1$ similar product terms and common-factor cancellation will always happen when those $m!$ terms are added together.

On the other hand, if there is a product term from a determinant of the reduced matrix with m first-order cofactor and whose row or column indices are not unique, this product term will eventually cancel out due to term cancellation. The following theorem gives the condition on the term cancellation in the reduction process [27].

Theorem 3: For a given product term from a determinant, which consists of m first-order cofactor $\Delta_{i_1,j_1} \ldots \Delta_{i_m,j_m}, m \geq 2$ of a subcircuit, if there are two first-order cofactors that share the same row index or column index, then there exists another product term that will cancel with this product term.

Proof: For the first-order cofactors $\Delta_{i_1,j_1} \ldots \Delta_{i_k,j_k} k \geq 2$, let us assume that Δ_{i_1,j_1} and Δ_{i_2,j_2} share the same row index without loss of generality, i.e., $i_1 = i_2$. As a result, Δ_{i_1,j_1} and Δ_{i_2,j_2} have the matrix pattern shown in Fig. 6.

The composite admittances a_{u_1,v_1} and a_{u_2,v_2} that contains those first-order cofactors can be written as follows:

$$a_{u_1,v_1} = -\frac{x_{u_1,j_1}\Delta^I_{i_1,j_1}x_{i_1,v_1}}{\det(A^I)} \tag{24}$$

$$a_{u_2,v_2} = -\frac{x_{u_2,j_2}\Delta^I_{i_2,j_2}x_{i_2,v_2}}{\det(A^I)}. \tag{25}$$

Due to the presence of admittances x_{u_1,j_1}, x_{i_1,v_1}, x_{u_2,j_2}, x_{i_2,v_2}, composite admittances at positions (u_1, v_2) and (u_2, v_1) must also exist and they are

$$a_{u_1,v_2} = -\frac{x_{u_2,j_2}\Delta^I_{i_2,j_2}x_{i_1,v_1}}{\det(A^I)} \tag{26}$$

$$a_{u_2,v_1} = -\frac{x_{u_1,j_1}\Delta^I_{i_1,j_1}x_{i_2,v_2}}{\det(A^I)}. \tag{27}$$

As a result, we know that

$$a_{v_1,u_1}a_{u_2,v_2} = -a_{u_1,v_2}a_{u_2,v_1}. \tag{28}$$

In other words, a product term containing $a_{v_1,u_1}a_{u_2,v_2}$ will cancel a product term containing $a_{u_1,v_2}a_{u_2,v_1}$. Therefore, those two terms will cancel out.

As a result, we can see that any higher admittance-order term coming from multiplications of composite admittances will eventually merge (to generate common cofactors) or cancel out with other same-order terms so that the resulting terms will have the correct admittance order. This is also consistent with the observation that no matter how internal nodes are reduced, the admittances in the reduced circuit should have the same admittance order given the same set of reduced nodes.

If we look at the example in Fig. 3 again, we notice that in (12), all the terms containing three first-order cofactors will cancel out eventually as some first-order cofactors will always share the same row or column indices for every product term. This also reflects the fact that the determinant consisting of all the first-order cofactors of all the composite admittances is a singular matrix as shown in

$$\begin{matrix} v_3 \\ v_4 \\ v_5 \end{matrix} \begin{vmatrix} \Delta_{11} & \Delta_{21} & \Delta_{21} \\ \Delta_{12} & \Delta_{22} & \Delta_{22} \\ \Delta_{12} & \Delta_{22} & \Delta_{22} \end{vmatrix} = 0. \tag{29}$$

As a result, we have the following result regarding the cancellation-free denominator of the rational admittance computed from the determinant of a reduced nonsingular circuit matrix.

Corollary 2: The $\det(A^{II})$ is the denominator of the cancellation-free rational admittances computed from a determinant of a reduced circuit matrix with A^{II} being the internal matrix of the reduced subcircuit.

Proof: Given a determinant with composite admittances coming from suppression of the subcircuit A^{II} (with A^{II} being internal matrix of the subcircuit), $\det(A^{II})$ will be the denominator of all the composite admittance. For every product term from the determinant, there are four scenarios we need to consider. 1) There is no first-order cofactor of $\det(A^{II})$ in the product term. This product will have $\det(A^{II})$ as the denominator when it is added with other terms with $\det(A^{II})$ as the denominator. 2) There is only one first-order cofactor of $\det(A^{II})$ in each product term. As a result, there is only one composite admittance involved and $\det(A^{II})$ is the dominator of the product term. 3) There are more than one first-order cofactor and all the first-order cofactors in the product terms have unique row and column indices. In this case, $\det(A^{II})^{m-1}$ is the common factor for all the product terms whose first-order cofactors form a $m \times m$ full matrix based on Corollary 1. So, the denominator of the resulting rational admittances from the addition of those product terms is $\det(A^{II})$, the cancellation-free denominator will be $\det(A^{II})$ again. 4) There are more than one first-order cofactors and at least two first-order cofactors share the same row or column indices in this product term. So the product term will cancel out based on Theorem 3. Combining the four scenarios, the corollary is proven.

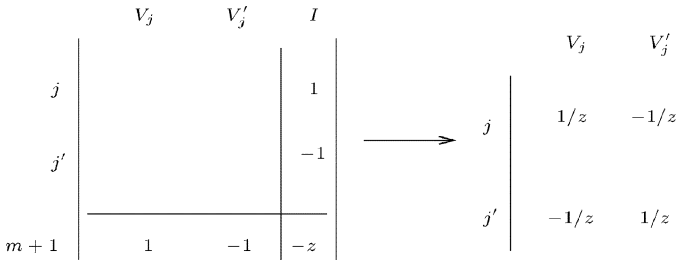


Fig. 7. Impedance stamp and the stamp in the reduced matrix.

Note that if the circuit has more than one immediate subcircuit (say k subcircuits), it can be easily shown that

$$\prod_{i=1}^{k} \det(A_i^{II}) \tag{30}$$

will be the denominator of the resulting rational admittance, where A_i^{II} is the internal matrix of subcircuit i.

D. Device-Level Cancellation From Subcircuit Reduction's Perspective

Device-level cancellation is actually caused by the subcircuit reduction. Circuit matrices formulated by MNA and NA methods can be obtained by reducing the branch current variables and branch voltage variables from circuit matrices formulated by using sparse tableau analysis (STA) method, which is cancellation-free as each device parameter appears only once in STA matrices. To see this, Fig. 7 shows the stamps for an impedance z (like a self-inductor) and the reduced matrix, where the current variable I is reduced.

So, the cancellation for an admittance device can be viewed as the cancellation due to the reduction of the current variable for an impedance element.

It turns out that both term cancellation and common-factor cancellation can happen at device-level for NA formulation. Basically, NA formulation by reducing current variables works fine for *RLC* circuits, but it does not work well for mutual inductance as shown in our recent work [36]. There are two reasons. The first reason is that the susceptance matrix L^{-1}/s due to NA formulation can introduce incorrect dc paths, which are not exist in the original circuit. This is obvious: as s approaches zero, L^{-1}/s becomes an infinite admittance matrix. So, the low frequency response of NA formulated *RLCM* circuit will not be correct.

The second reason is that current variables in mutual inductors affect each other. From matrix reduction's perspective, this means that we have to reduce both current variables at the same time to get NA formulation. However, the submatrix, which corresponds to the mutually coupled two current variables, has more than one boundary nodes. As a result, common factors will be present in the resulting NA formulated circuit matrices for a mutual inductance. In other words, NA formulation for *RLCM* circuits are not common-factor free, but MNA formulation is common-factor free. If there are many mutually coupled inductors, we have an L matrix with nonzero off-diagonal elements. The L^{-1} is no longer a diagonal matrix. As a result the NA formulation, which contains L^{-1} (with mutual inductances), is no longer common-factor cancellation free. The reduced expressions will contain common factors even the reduction process does not introduce any new common factor as is the case for the new reduction algorithm.

Therefore, we use MNA formulation, which is common-factor cancellation free, to stamp both self and mutual inductances in the proposed hierarchical method, which is superior to $Y-\Delta$ transformation based on NA formulation.

E. Cancellation at Different Hierarchical Circuit Levels

If the nodes (u_1, u_2) and (v_1, v_2) in Fig. 4 are reduced in different subcircuits at different circuit hierarchical levels, cancellation will happen when those composite admittances from different subcircuits are merged. To see this, suppose that only node (u_1, u_2) is selected to be suppressed in subcircuit *III*. As a result, the new element value at node (v_1, v_2), which is not suppressed, becomes

$$a''_{v_1,v_2} = a'_{v_1,v_2} - \frac{a'_{v_1,u_2}\Delta^{III}_{u_1,u_2}a'_{u_1,v_2}}{\det(A^{III})}. \tag{31}$$

We also notice that

$$\det(A^{III}) = a'_{u_1,u_2}\Delta^{III}_{u_1,u_2} + R^{III}_{u_1,u_2}.$$

As a result, we have

$$\begin{aligned} a''_{v_1,v_2} &= a'_{v_1,v_2} - \frac{a'_{v_1,u_2}\Delta^{III}_{u_1,u_2}a'_{u_1,v_2}}{a'_{u_1,u_2}\Delta^{III}_{u_1,u_2} + R^{III}_{u_1,u_2}} \\ &= \frac{(a'_{u_1,u_2}a'_{v_1,v_2} - a'_{v_1,u_2}a'_{u_1,v_2})\Delta^{III}_{u_1,u_2} + a'_{v_1,v_2}R^{III}}{a'_{u_1,u_2}\Delta^{III}_{u_1,u_2} + R^{III}}. \end{aligned} \tag{32}$$

Notice that $(a'_{u_1,u_2}a'_{v_1,v_2} - a'_{v_1,u_2}a'_{u_1,v_2})$ is just (19). Therefore, we know that $\det(A^{II})$ is the common factor in the numerator and the denominator of the resulting rational function. After the common factor $\det(A^{II})$ is removed, a''_{v_1,v_2} will become cancellation-free.

The cancellation at different circuit hierarchical levels essentially can be viewed as the cancellation inside a determinant in one circuit level as discussed in Section V-B. Based on Theorem 1, we know that the numerator expression in (32) eventually forms a determinant if all the composite and noncomposite admittance terms are added together. So, the cancellation problems in the numerator are the cancellation problems in a determinant. More precisely, (32) only presents a specific common-factor cancellation when the numerator (which is a determinant $\det(A[1,\ldots,m,u|1,\ldots,m,v])$) is expanded along the row u and column v sequentially. If all the cancellations are carried out numerically, no special consideration for cancellation at different circuit hierarchical levels is required in the new hierarchical decomposition scheme. If we remove the term cancellations symbolically, we have to represent each composite admittance explicitly instead implicitly. The composite admittances from the reduction of different subcircuits will be added together in a cancellation-free manner.

V. New s-Domain Hierarchical Network Modeling and Simulation Algorithm

In this section, we detail the new network modeling and simulation algorithm based on cancellation rules discussed in the previous section.

A. Cancellation-Free Rational Admittance

First, we show that if we can manage to obtain symbolic cancellation-free rational admittances of the determinant of a reduced circuit matrix during the hierarchical decomposition, those cancellation-free rational admittances can lead to the exact symbolic expressions obtained from the determinant of original (flattened) circuit matrices. Such cancellation-free expressions can significantly simplify the reduction process.

To simplify the discussion, we first assume that for every circuit or subcircuit at a particular circuit hierarchical level, it has only one subcircuit. We can easily extend the result to multiple subcircuit case. Let A_i^{II} be the internal matrix of subcircuit i at level i, when all its subcircuit matrices are reduced, with subcircuit $i-1$ being its immediate children circuit and subcircuit $i+1$ being its immediate parent circuit. Also, let A_i be the internal matrix of the *flattened* subcircuit that includes all the internal nodes in subcircuit i as well as in its direct or indirect subcircuits in all the downstream hierarchical levels in one level. In other words, A_i represents the flattened subcircuit that includes all the suppressed internal nodes at subcircuit i level. Also we assume that all the subcircuits are not singular. Otherwise, the subcircuits need to be redefined.

Theorem 4: Let A_k^{II} be the matrix of the subcircuit at level k and A_k and A_{k-1} are the matrices of the flattened circuits at circuit level k and $k-1$. Then

$$\det(A_k^{II}) = \frac{\det(A_k)}{\det(A_{k-1})}. \tag{33}$$

Proof: First, we show (33) holds for circuit with two hierarchical levels.

1) For a linear system $Ax = b$ with the partitioning of the variables x in (1), after the reduction, we have

$$\det(A) = \begin{vmatrix} A_1^{II} & A^{IB} & 0 \\ 0 & A_2^{BB*} & A_2^{BR} \\ 0 & A_2^{RB} & A_2^{RR} \end{vmatrix} = \det(A_1^{II})\det(A_2^{II}) \tag{34}$$

where $\det(A_2^{II})$ is the matrix of the reduced circuit (at circuit level 2) in (2). As a result, we have $\det(A_2^{II}) = \det(A)/\det(A_1^{II}) = \det(A)/\det(A_1)$. So, (34) holds for the circuit with two hierarchical levels. Note that since A_1^{II} is a bottom-level circuit, $A_1^{II} = A_1$.

2) Assuming (34) holds for all circuits with less than k hierarchical levels, that is $\det(A_i^{II}) = \det(A_i)/\det(A_{i-1}), i < k$. Now, we want to show that this is also true for circuits with k hierarchical levels. For a circuit with k hierarchical levels, its system determinant $\det(A)$ can be expressed as follows based on (34):

$$\det(A) = \det(A_1^{II})\det(A_2^{II}) \times \cdots \times \det(A_k^{II}). \tag{35}$$

Use the hypothesis on $k-1, k-2, \ldots$, we have

$$\det(A) = \det(A_1)\frac{\det(A_2)}{\det(A_1)} \cdots \frac{\det(A_{k-1})}{\det(A_{k-2})}\det(A_k^{II}) \tag{36}$$

$$= \det(A_{k-1})\det(A_k^{II}) \tag{37}$$

Notice that $A = A_k$ as circuit k is the top level circuit. Therefore, (33) holds for $k, k > 1$. Combining the two hierarchical levels, the theorem is proven.

Note that if current circuit has more than one immediate subcircuit, (33) can be modified as

$$\det(A_k^{II}) = \frac{\det(A_k)}{\prod_{j=1}^{N} \det(A_{k-1,j})} \tag{38}$$

where N is the number of subcircuits of the current circuit.

The significance of Theorem 4 is twofold. First, by combining with Corollary 2, we can immediately have the following corollary.

Corollary 3: The numerator of a cancellation-free symbolic rational expression of a determinant of a reduced circuit matrix is the exact symbolic expression obtained from the determinant of the flattened circuit matrix.

As a result, we can obtain the *exact* symbolic expressions of the flattened matrix by obtaining the *cancellation-free* symbolic expressions during the hierarchical decomposition.

Second the reduction process is independent of reduction sequences if the symbolic rational expressions are kept cancellation-free all the time. In other words, we will arrive at the same cancellation-free symbolic expressions for each newly created rational admittance regardless of how internal nodes are suppressed (node-by-node or subcircuit by subcircuit). The reason is obvious. The cancellation-free rational expressions at a particular circuit level are identical to those obtained by reducing all the internal nodes all together in one subcircuit. As a result, we only need to worry about the cancellation between current circuit and its immediate subcircuits at any circuit hierarchical level. This will significantly simplify the decancellation process during the hierarchical decomposition process.

B. Y-Expanded DDDs

In order to efficiently compute the rational function for a determinant and handle cancellations, an new DDD graph called Y-parameter expanded DDDs (YDDD) is first introduced. Y-expanded DDDs are also DDD graphs, where each DDD node represents a device admittance or a composite admittance as shown in Fig. 8.

Note that some circuit parameter admittances are functions of the complex frequency variable s. This is different from the s-expanded DDD graphs, where s is explicitly extracted and represented [22]. The main purpose of the introduction of YDDDs is that we can easily handle both term cancellation and common-factor cancellation, as cancellation patterns can be easily detected by examining those device admittances or composite admittances. Similar to s-expanded DDDs [22], the YDDDs can be constructed from a complex DDD in linear time in the size of the original complex DDDs. The time complexity for constructing complex DDD depends on the circuit topology. Given the best vertex ordering, if the underlying circuit is a

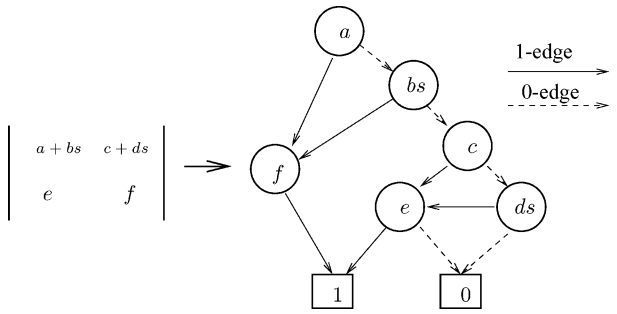


Fig. 8. Determinant and its YDDD.

```
YDDDCONST(D)
1   if ( D = 0 or D = 1)
2     return NULL
3   L0 = YDDDCONST(D.0)
4   L1 = YDDDCONST(D.1)
5   Presult = NULL
6   for i = 1 to m do
7     Ptmp = YDDDMULTIPLY(L1,D.xi)
8     Presult = YDDDUNION(Ptmp, Presult)
9   return YDDDUNION(Presult, L0)
```

Fig. 9. Y-expanded DDD construction.

ladder or tree circuit, $|DDD|$ is a linear function of the size of the circuit. For general circuits, the size of a DDD graph may grow exponentially in the worse case. However, like BDDs, with proper vertex ordering, the DDD representations are very compact for many real circuits [21], [22]. In Fig. 9, we present a version of the algorithm called YDDDCONST().

Function YDDDCONST() takes a complex DDD rooted at D and returns the resulting YDDD tree. $D.1$ and $D.0$ denote, respectively, the vertices pointed to by the 1-edge and 0-edge of vertex D. m is the number of devices admittance or composite admittances connected to a node and each is denoted by $D.x_i$. YDDDUNION(P_1, P_2) computes the union of two YDDDs, P_1 and P_2. YDDDMULTIPLY(P, v) computes the product of YDDD P and YDDD vertex v.

C. Computation of Cancellation-Free Rational Admittances

To construct a cancellation-free rational admittance (rational function of s) from a determinant represented by a DDD graph, whose elements are rational admittances too, we need to take care of both term cancellation and common-factor cancellation. The computation can be done in a bottom up fashion.

Specifically, at each YDDD node P, we compute the rational function from the YDDD tree rooted at P recursively. Let $P_r = N_{P_r}/D_{P_r}$ denote the rational admittance that the YDDD node P *itself* represents. $P_1 = N_{P_1}/D_{P_1}$ and $P_0 = N_{P_0}/D_{P_0}$ represent the rational admittances for the YDDD subtrees rooted at nodes pointed by the 1-edge and 0-edge of P respectively. The rational admittance of the DDD tree rooted at P will become [21]

$$\begin{aligned} F_P &= \text{sign}(P)P_r(s)P_1(s) + P_0(s) \\ &= \text{sign}(P)\frac{N_{P_r}(s)}{D_{P_r}(s)}\frac{N_{P_1}(s)}{D_{P_1}(s)} + \frac{N_{P_0}(s)}{D_{P_0}(s)}. \end{aligned} \tag{39}$$

```
YDDDMULTIPLY(P, D.x)
1   for i = 0 to p − 1 do
2     for each Ly ∈ CL(D.x)
3       P[i] = REMAINDER(P[i], Ly)
4     P[i] = MULTIPLY(P[i], D.x)
5   return P
```

Fig. 10. Term-cancellation free YDDDMULTIPLY.

With (39), the rational function (admittance) at the root of the whole YDDD can be computed recursively. Since both multiplication and addition are involved, decancellation measures have to been taken for both operations as shown below.

1) Decancellation for Term Cancellation: Term cancellation happens when two identical symbolic product terms with opposite signs are added together. The cancellation can come from device admittances or from subcircuit reduction as shown in Sections V-A and V-B.

For canceling terms, generated by subcircuit reduction, we may have $\det(A^{II})^k, k > 1$ as their denominators, the new *constant-admittance-order* decancellation strategy (will be discussed below) will lead to unnecessary polynomial divisions, which add unnecessary computational costs at best and adversely introduce some numerical noises at worst. In our method, term cancellation can be removed during the construction of YDDD's for a determinant.

The idea is to remove all those canceling terms via DDD operations. We first introduce concept of a *canceling pair* for two admittances (represented by two DDD nodes). For instance, composite terms $-1/(1+s)$ and $-s^2/(1+s)$ in (11) is a canceling pair. If those two terms are in a product term, this term will cancel out with another term eventually. Since each DDD node has an index, we say the DDD index is a *canceling index* of the other for a canceling pair.

Before the construction of a YDDD for a determinant, we build a *canceling list* ($C_L(P)$) for each YDDD index, whose corresponding admittance is a composite admittance or device admittance. The use of a list is due to the fact that one DDD index can have more than one canceling DDD index. The criteria for two indices to be a canceling pair are based on Theorem 3 for composite admittances and the cancellation patterns in Fig. 2 for device admittances. To avoid duplication we require that $x \in C_L(P.index)$ and $x < P.index$. Once we have the canceling list for each DDD index, it will be used in YDDDMULTIPLY() shown in Fig. 9. The pseudo code for the term-cancellation-free YDDDMULTIPLY() in YDDDCONST() is shown in Fig. 10, where lines 2 to 3 are used to remove canceling terms for each canceling pair.

DDD operations REMAINDER(P, x) and MULTIPLY(P, x) are operations to find all terms which *do not* contain symbol x in P and to add symbol x to every term in P respectively. p is the number of the devices or composite admittances in the complex DDD vertex.

We notice that the $Y-\Delta$ algorithm cannot remove the term cancellation symbolically. This is another significant advantage of the new reduction algorithm over the $Y-\Delta$ algorithm.

2) Decancellation for Common-Factor Cancellation: Common-factors are more difficult to remove symbolically. Instead, we remove them numerically. We need to look at two situations where the common factors

are introduced into the numerators and denominators of the resulting rational functions. The first one comes from the *multiplication* of $P_r(s)$ and $P_1(s)$ according to Corollary 2. The second one is due to the *addition* of two rational functions in (39), with common factors in their denominators. For instance, let us consider the addition of two rational functions $a_1/b_1 + a_2/b_2 = (a_1 * b_2 + a_2 * b_1)/(b_1 * b_2)$. If there is a common term c in both $b_1 = x*c$ and $b_2 = y*c$, c will become a common factor in both the numerator and the denominator of the resulting rational function. Those two situations are handled separately in the new method.

1) *Cancellation due to multiplication*. According to Corollary 2, only $\det(A^{II})$ will be present in the denominator of the final rational admittances for each subcircuit from a circuit hierarchy. This leads to a very simple cancellation strategy: *whenever we see admittance order is increased due to multiplication of two terms that both have* $\det(A^{II})$ *in their denominators, we divide the resulting numerator by* $\det(A^{II})$. This way, the resulting admittance terms always have the correct admittance order and the cancellation-free numerators in the resulting expressions will be formed implicitly and numerically when all the resulting terms are added together.
2) *Cancellation due to addition*. Since common-factor cancellations are only caused by subcircuit reduction based on MNA formulation, we only need to consider the common factor $\det(A^{II})$ for each subcircuit. The idea is that when addition of two rational admittances takes place, we explicitly watch for the admittances such that $\det(A^{II})$ will not appear twice in the resulting new admittance. Specifically, this rule requires that only one $\det(A^{II})$ for each subcircuit from a circuit hierarchy exists in the denominator of the resulting rational admittance.

Now, we are ready to present the whole rational admittance computation method for a determinant represented by its YDDD tree based on the decancellation aforementioned strategy. For each YDDD node P, we keep two things: the numerator polynomial of the computed rational admittance for the subtree rooted at P, which are denoted by $N(P)$ and a set of YDDD indexes whose denominators are not trivial and unique, denoted by $S_D(P)$. In this way we do not need to keep the denominator polynomial for each YDDD node explicitly and it is also easy to watch the cancellation situations. $P(D)$ denotes the rational admittance for the DDD subtree rooted at D. $D.index$ is the index of the DDD node D. Following the convention of DDD operations, we use $D.1$ and $D.0$ to represent the DDD subtrees pointed by 1-edge and 0-edge of DDD vertex D, respectively. The resulting algorithm is shown Fig. 11.

The algorithm basically visits every YDDD node once in a bottom-up fashion and computes the new rational admittance for each subtree rooted at the YDDD node visited and performs the decancellation for both multiplication and addition operations. So, the time complexity of the algorithm is $O(|\mathrm{DDD}|k)$, where $|\mathrm{DDD}|$ is the number of YDDD nodes and k is the highest order of all the rational admittances.

```
COMPRAFUNC(P)
01  if ( P = zero or P = one or computed already)
02    return
03  COMPRAFUNC(P.0)
04  COMPRAFUNC(P.1)
05  N1 = N(P.1)
06  N0 = N(P.0)
07  if (P(P)'s denominator is not trivial and not exist in SD(P.1))
08    S1 = SD(P.1) ∪ P.index
09  else
10    S1 = SD(P.1)
11  S0 = SD(P.0)
12  N1 = sign(P) * N(Pr) * N1
13  if (both Pr and P(P.1) contain det(A^I))
14    N1 = N1/det(A^I)
15    N1 = N1/Ddvs
16  Compute N(P) from N1, N0, and SD(P) from S1 and S0
      such that det(A^I) and each device appear once in SD(P).
```

Fig. 11. Computation of the cancellation-free rational function from a YDDD.

D. Clustering Algorithm

To construct the circuit hierarchy and minimize the computation costs, efficient clustering algorithm is required. Since the hierarchical reduction process is equivalent to the block Gaussian elimination process, the reduction order should follow the minimum fill-in or minimum degree ordering. Our clustering algorithm consists of two steps: 1) finding a good node-reduction ordering and 2) performing the connectivity-based clustering based on the given node-ordering subject to node-size constraint and other user-specified constraints.

For the node ordering algorithm, the multiple minimum degree algorithm (MMD) [13] is employed to find the node reduction ordering. The basic idea of the MMD is to delay the updates of changing connectivity information (represented by the so-called elimination graphs to find the nodes with minimum degree) during the simulated Gaussian reduction process to speed up the ordering process.

Once the node order is found, clustering is performed based on the connectivity information and the given node ordering subject to a given node count limit, which specifies the maximum internal node count, to control the size of the internal matrix A^{II} for each subcircuit. Also nodes as inputs and outputs will be put into the top level circuit during the clustering process.

E. Overview of the Simulation and Reduction Algorithm

In this section, we give the overview of the new circuit modeling and simulation algorithm based on the new hierarchical decomposition framework and the decancellation algorithm given in early sections.

The basic idea is to reduce subcircuits and the right-hand side vector in a hierarchical and cancellation-free way. The new admittances coming from subcircuit suppressions shown in (9) and (10) are kept as rational functions of s in the exact or order-reduced form (with fixed order of s). Such a reduction can be repeated until we reach the top level circuit, which is typically small enough to be solved exactly and symbolically (s is still the only symbol). The resulting circuit unknown variables are rational functions of s, which can be converted to time domain for waveform evaluation [16]. From a simulation

perspective, once the parent circuit variables x^B are known, we can obtain the internal variables of subcircuit A^{II} by solving

$$A^{II}x^I = b^I - A^{IB}x^B. \tag{40}$$

In this way, we can obtain all the unknown variables in the rational form.

From a modeling perspective, we can easily obtain the network functions of the given circuits to compute other small-signal characteristics of analog circuits and second or higher order effects as nonlinearity (especial weakly nonlinear characteristics like distortions) can be estimated based on network functions of linearized circuits [33].

We assume that the circuit hierarchy has been constructed using the MMD algorithm. The modeling algorithm is shown below:

The General Hierarchical Network Modeling Algorithm

1. Build the complex DDD's for each subcircuit matrix and all the required cofactors/determinants for each newly created admittances in a bottom up way.
2. If the present circuit has subcircuits, perform the reduction on each subcircuit first.
3. In the present circuit, identify all the admittances, which are either from circuit parameters or are composite admittances due to subcircuit reduction.
4. Derive the YDDDs from complex DDDs for each new admittances shown in **(9)** and **(10)** and construct the cancellation-free rational admittances for each YDDD using CompRafunc().
5. If truncation is carried out and stability is required, perform Hurwitz approximation on the reduced transfer functions.

It was shown that YDDD representation is more compact than *sequence of expressions* [10], [30], whose complexity essentially represents the complexity of symbolic node-by-node Gaussian elimination process. Hence, time complexity of the new reduction algorithm is better than $Y-\Delta$ reduction algorithm [18], which is based on Gaussian elimination. Another advantage of the new algorithm over the existing $Y-\Delta$ transformation algorithm is that we need to remember fewer number of common-factors, which are the determinants of the reduced subcircuit matrices. If we reduce one node at a time, each node becomes a subcircuit, we end up with more common-factors to deal with. Also, there are more independent subcircuit hierarchies as each subcircuit is the smallest (consists of one node). As a result, we have to remember all the common-factors of the reduced circuits (nodes) that are connected to nodes yet to be reduced, which in turn leads to more memory usage compared to the new method. Our experimental results confirm those observations.

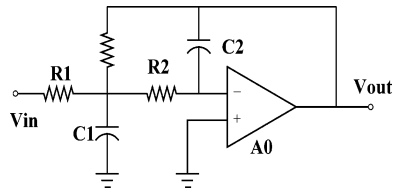


Fig. 12. Second-order active filter.

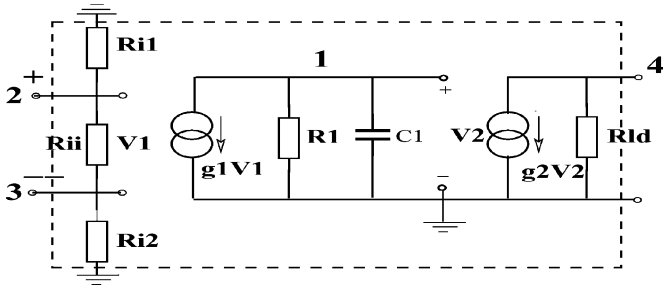


Fig. 13. Linear model of an Opamp circuit.

For large *RLC* interconnect circuits, truncations have to be carried out to keep limited orders of s in each rational admittance. The resulting admittances or transfer functions may not be stable. Hurwitz approximation can be carried out to enforce stability [18], [26], [34].

VI. Experimental Results

The proposed algorithm is implemented in C++. We perform the reduction on a number of linear(ized) analog circuits. All the experimental results are obtained on a Linux workstation with dual 1.6-GH AMD Athlon MP 2000+ CPUs with 2 GB of memory.

Here, we present the exact transfer function for a second-order linear filter shown in Fig. 12 obtained from the new reduction program. In this circuit, we have $R_1 = 1\ \text{k}\Omega$, $R_2 = 2\ \text{k}\Omega$, $R_3 = 5\ \text{k}\Omega$, $C_1 = 10\ \text{nF}$, $C_2 = 20\ \text{nF}$, $g_1 = 2S$, and $g_2 = 1.333S$. A_0 is an Opamp circuit and is represented by a linear model of 741 Opamp circuit shown in Fig. 13 which contains two voltage controlled current sources g_1 and g_2. The transfer function obtained by the new program is shown in the following:

$$H(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{\frac{-5.33\times10^{-7}+2.68\times10^{-15}s+3.72\times10^{-19}s^2}{1.02\times10^{-3}+2.66\times10^{-6}s}}{\frac{2.66\times10^{-7}+9.07\times10^{-11}s+5.46\times10^{-16}s^2+7.47\times10^{-23}s^3}{1.02\times10^{-3}+2.66\times10^{-6}s}}.$$

Note that both the numerator and the denominator become rational functions of s due to suppression of subcircuit A_0. But they share a common denominator which is actually the determinant of the internal matrix of the subcircuit A_0, $\det(A^{II}_{A_0})$. After the common denominator is removed, the transfer function is exactly the same one as we obtain from the flattened filter circuit. We also notice that the highest order of the transfer function of the second order filter becomes 3. However, if Opamp A_0 is an ideal voltage controlled current source with infinite gain, the highest order will be 2.

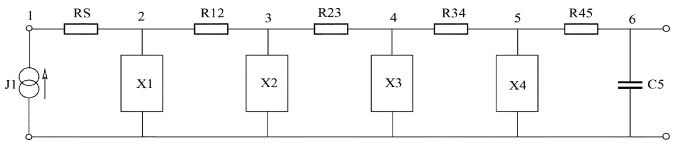


Fig. 14. Active low-pass filter.

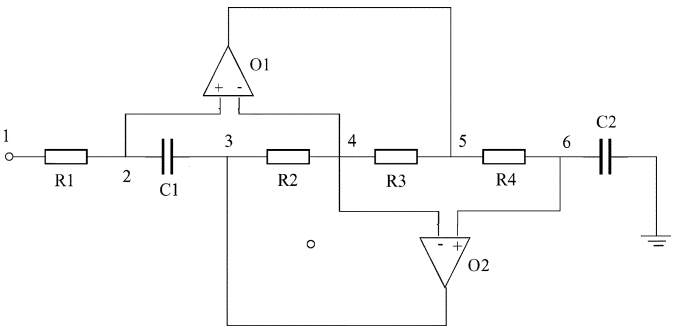


Fig. 15. FDNR subcircuit.

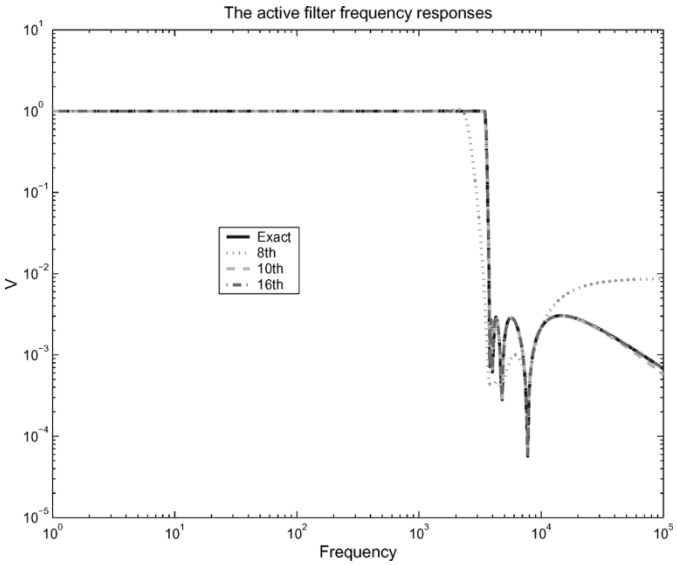


Fig. 16. Frequency response of the active filters (exact versus 8th-, 10th-, and 16th-order approximations).

Next, we derive the exact transfer function for a more complicated active low pass filter shown in Fig. 14, which also contain controlled current sources. It has four identical subcircuits, named *X*1 to *X*4. Fig. 15 shows the detailed structure of the subcircuit. Each subcircuit contains two Opamp subcircuits, which are also represented by the Opamp circuit shown in Fig. 13.

The exact transfer function, computed by the new method, is shown as the follows:

$$\frac{V_6(s)}{V_1(s)} = \frac{\frac{2.8\times10^{-18}}{1}}{\begin{array}{c}1.3\times10^{-57}+2.4\times10^{-61}s+2.8\times10^{-65}s^1\\ 2.3\times10^{-69}s^2+1.4\times10^{-73}s^3+7.6\times10^{-78}s^4\\ 2.9\times10^{-82}s^5+1.0\times10^{-86}s^6+2.0\times10^{-91}s^7\\ 4.3\times10^{-96}s^8+4.5\times10^{-102}s^9+2.4\times10^{-108}s^{10}\\ 8.3\times10^{-115}s^{11}+1.9\times10^{-121}s^{12}+3.0\times10^{-128}s^{13}\\ 3.3\times10^{-135}s^{14}+2.2\times10^{-142}s^{15}+7.5\times10^{-150}s^{16}\end{array}}{\begin{array}{c}4.6\times10^{-40}+5.0\times10^{-46}s+2.2\times10^{-48}s^1\\ 2.4\times10^{-54}s^2+3.9\times10^{-57}s^3+4.2\times10^{-63}s^4\\ 2.7\times10^{-66}s^5+2.9\times10^{-72}s^6+6.1\times10^{-76}s^7\\ 6.7\times10^{-82}s^8+3.6\times10^{-88}s^9+1.2\times10^{-94}s^{10}\\ 2.8\times10^{-101}s^{11}+4.6\times10^{-108}s^{12}+5.0\times10^{-115}s^{13}\\ 3.4\times10^{-122}s^{14}+1.1\times10^{-129}s^{15}\end{array}}.$$

It takes 1.8 s to obtain the transfer function from a SPICE netlist. The flattened circuit of this low pass filter has 41 nodes, which can not be analyzed by DDD symbolic analysis directly due to circuit-limitation problem. If we set the maximum order to eight, we will obtain the exact coefficients up to 8th order of the exact transfer function. The exact, 8th-, 10th-, and 16th-order approximate responses of the active filter are shown in Fig. 16. We also notice that the numerator and the denominator do not share the same denominator in this case. However, it should agree with the exact transfer function after the transfer function is transformed into a rational form.

We also notice that when subcircuits are very small, $(M < 5)$ with controlled sources. The internal matrix of some subcircuits may be singular $(det(A^{II}) = 0)$. To resolve this problem, we move the whole subcircuit into its parent circuits (actually to remove this subcircuit from the circuit netlist). But for the Y-Delta method, it does not work any more as it has to reduce one node at a time and require the subcircuit to be nonsingular for valid division operations.

Finally, we test the proposed algorithm on a number of *RC/RLC* interconnect circuits. Group one has tree or ladder structured *RC/RLC* circuits. Group two has mesh-structured circuits. In our experiments, 8th order is the highest order kept for the generated admittances. For each circuit, M is the user specified maximum internal node count limit. We report both the CPU times and memory usages for different M for each circuit. When $M = 1$, the reduction algorithm essentially becomes the $Y - \Delta$ algorithm [18] as we reduce one node at a time. The initial maximum node degree is three for tree/ladder circuits and four for all the mesh-structured circuits. We will compare the CPU time and memory usage for computing an order-reduced transfer function for different $M's$. The statistics for tree and ladder structured RC circuits are summarized in the Table I.

Columns $\#N$ and $\#B(R+C+L)$ are numbers of nodes and branches (resistors+capacitors+inductors) in each circuit respectively. Column M gives the *maximum internal node count limit* used for each circuit. Column $\#Level$ shows the deepest hierarchical level for each circuit.

From Table I, we can observe that when $M = 1$, which corresponds to the $Y - \Delta$ transformation, the reduction takes the longest CPU times and the largest memory usage for all the test cases. The memory usage of $M = 1$ can be more than two times larger than $M > 1$ cases for circuits 10×100_*. The memory saving can be significant as shown in circuits 10×2000_*. For even larger circuits (like 20×2000_1), when $M = 1$ the hierarchical reduction method runs out of memory. The difference can be mainly explained by the very compact DDD representation and the reduced number of common-factors to be remembered. The compact DDD representations also lead to smaller CPU times as shown in the table.

TABLE I
STATISTICS FOR TREE OR LADDER/TREE STRUCTURED *RLC* CIRCUITS

Ckt	#N	#B (R+C+L)	M	#Level	CPU (sec.)	Mem (peak)	SPICE(sec.)
3x30_1	90	95+87+0	1	31	0.30	10.21MB	0.28
3x30_5	90	95+87+0	5	8	0.22	3.71MB	0.29
3x30_10	90	95+87+0	10	5	0.21	3.13MB	0.27
5x50_1	250	259+245+0	1	52	0.71	28.31MB	0.76
5x50_5	250	259+245+0	5	12	0.53	9.96MB	0.77
5x50_10	250	259+245+0	10	7	0.61	8.35MB	0.75
5x50_20	250	259+245+0	20	5	0.82	8.68MB	0.76
10x100_1	1000	1019+990+0	1	105	3.42	112.84MB	3.66
10x100_5	1000	1019+990+0	5	25	2.10	38.36MB	3.76
10x100_10	1000	1019+990+0	10	15	2.34	30.27MB	3.69
10x100_20	1000	1019+990+0	20	10	3.10	27.26MB	3.68
10x2000_1	40001	20028+20000+20000	1	4016	115.5	916.6MB	289.95
10x2000_5	40001	20038+20000+20000	5	809	99.03	58.56MB	290.49
20x2000_1	80001	40058+40000+40000	1	4125	–	–	782.17
20x2000_5	80001	40058+40000+40000	5	811	143.42	340MB	798.05

TABLE II
STATISTICS FOR MESH STRUCTURED *RLC* CIRCUITS

Ckt	#N	#B (R+C+L)	M	#Level	Row x Col	CPU (sec.)	Mem (peak)	Spice (sec.)
3x30x3_1	90	99+87+0	1	18	3x3	0.42	10.53MB	0.31
3x30x3_5	90	99+87+0	5	8	3x3	0.22	4.60MB	0.32
3x30x3_10	90	99+87+0	10	6	3x3	0.32	3.73MB	0.34
5x50x5_1	250	275+245+0	1	23	5x5	1.12	30.62MB	0.79
5x50x5_5	250	275+245+0	5	10	5x5	0.90	15.20MB	0.82
5x50x5_10	250	275+245+0	10	8	5x5	1.5	16.32MB	0.80
5x50x5_20	250	275+245+0	20	6	5x5	2.74	19.32MB	0.77
10x100x10_1	1000	1100+990+0	1	40	10x10	5.61	142.91MB	3.91
10x100x10_5	1000	1100+990+0	5	17	10x10	5.32	71.67MB	3.90
10x100x10_10	1000	1100+990+0	10	13	10x10	20.12	127.04MB	3.84
10x100x10_15	1000	1100+990+0	15	10	10x10	22.40	136.19MB	3.41
rlcmesh80_1	16001	8397+8000+8000	1	913	80x3	69.1	741.2MB	138.40
rlcmesh80_5	16001	8397+8000+8000	5	78	80x3	52.2	371.1MB	139.40
rlcmesh160_1	32001	16796+16000+16000	1	313	160x3	91.64	916.42MB	322.35
rlcmesh160_5	32001	16796+16000+16000	5	138	160x3	58.35	741.42MB	320.25
rlcmesh300 1	60001	31496+30000+30000	1	674	300x3	–	–	772.48
rlcmesh300_5	60001	31496+30000+30000	5	243	300x3	316.14	916.41MB	769.48

We also notice that the smallest CPU time for all the circuits happens when $M = 5$ for most of the circuits. The reason is that if subcircuit sizes are too large (more than 20), DDD construction time will become significant for general unstructured circuits, the saving due to the reduced number of subcircuits will be offset by the increasing DDD construction time. On the other hand, if the subcircuit size is too small, the number of subcircuits will increase, so do reduction time and memory. The worst case in this direction is the $Y - \Delta$ reduction, where each node become a subcircuit.

We also compare the new algorithm with Spice3f4 for ac analysis by sampling 600 frequency points for each circuit. We notice that if more frequency points are sampled, Spice3f4 will take longer time, while the CPU time for the new algorithm will almost be the same as the new algorithm performs the reduction in s-domain and is independent of the number of sampling frequency points.

Table II gives the statistics for some mesh-structured circuits. Here the circuit $3 \times 30 \times 3$ means that there are three rows and each row has 30 *RC/RLC* ladder sections, and the circuit contains three resistive columns which connect all the rows. The column $Row \times Col$ gives the numbers of rows and columns in the meshed *RC* circuits. There are also many *RC* ladder sections between the vertical columns at each row. It can be seen that when $M = 1$, the reduction still has the largest peak memory usage for all the cases. However, the CPU times for $M > 1$ may become worse than $M = 1$ for some circuits. The reason is that when internal matrix A^{II} gets larger, it will take longer to construct the DDD graphs. For mesh-structured circuits, the internal matrix may become very dense, such a matrix is computationally expensive for DDD constructions when the size of the matrix is larger than 15. For circuit $10 \times 100 \times 10\_15$, we only try $M = 15$ instead of $M = 20$ as $M = 20$ will result in a very long DDD construction time.

As with tree and ladder structured circuits, when $M = 5$, we still have best results in terms of both CPU time and memory usage for larger circuits. We also notice that our current implementation of the new reduction algorithm is not very memory efficient for large mesh-structured circuits. More efficient implementation using techniques like garbage collections, dynamic vertex ordering [32], are required to significantly reduce memory usage.

Fig. 17 gives the responses for different orders of Hurwitz approximations for a *RLC* circuit. We compare the approximated results with exact response from SPICE. As the approximation order increases, the approximation gets better.

VII. CONCLUSION

In this paper, we have proposed a general hierarchical linear network simulation and modeling technique in s-domain. The

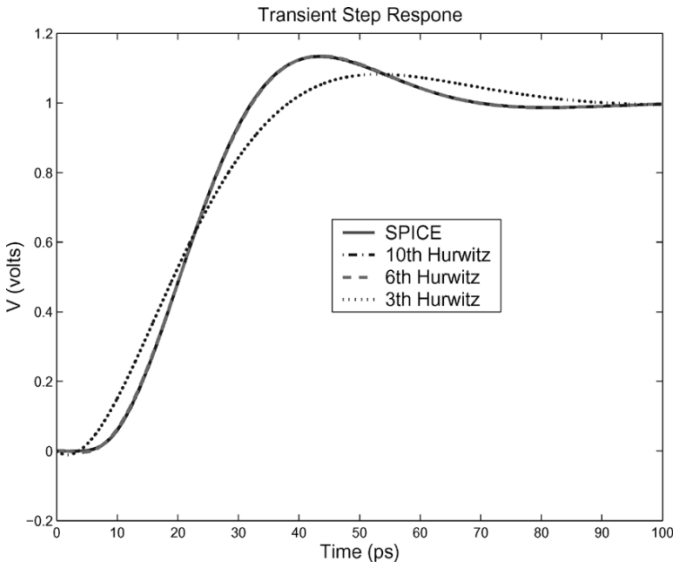


Fig. 17. Transient step responses given by the new reduction algorithm for different order Hurwitz approximation for a couple *RLC* circuit.

simulation and modeling are done by subcircuit suppression in a hierarchical way and by rational function approximation, which generates exact or order-reduced cancellation-free admittances in the reduced network matrices. The new method works on circuit matrices formulated by MNA formulation and can be applied to any linear circuit. On the theoretical side, we studied how common factors are generated in the general subcircuit reduction process and presented some theoretical results. On the practical side, we proposed a novel decancellation strategy based on determinant decision diagrams to derive the cancellation-free rational functions for determinants generated from subcircuit reduction. The resulting method can be used for modeling of both linear(ized) analog circuits, which typically are active circuits with controlled sources, and passive interconnect circuits. Experimental results validate the proposed method on some linear analog circuits and large *RLC* interconnect circuits. Our results also show that subcircuit (multiple-node) reduction scheme in general is better than one-node reduction method such as $Y-\Delta$ transformation in terms of CPU time and memory usage.

ACKNOWLEDGMENT

The author would like to thank Z. Qin and C.-K. Cheng for discussions on the $Y-\Delta$ transformation algorithm, W. Guo for implementing the MMD-based clustering algorithm, and J. Yang for implementing the Hurwitz approximation algorithm.

REFERENCES

[1] C. S. Amin, M. H. Chowdhury, and Y. I. Ismail, "Realizable *RLCK* circuit crunching," in *Proc. Design Automation Conf.*, 2003, pp. 226–231.

[2] R. Brualdi and H. Schneider, "Determinantal identities: Gauss, Schur, Cauchy, Sylvester, Kronecker, Jacobi, Binet, Laplace, Muir, and Cayley," *Linear Algebra Applicat.*, vol. 52/53, pp. 769–791, 1983.

[3] J. Cong, L. He, C.-K. Koh, and P. Padden, "Performance optimization of VLSI interconnect layout," *Integration, VLSI J.*, vol. 21, no. 1/2, pp. 1–94, 1996.

[4] P. Elias and N. van der Meijs, "Including higher-order moments of RC interconnections in layout-to-circuit extraction," in *Proc. Eur. Design Test Conf.*, 1996, pp. 362–366.

[5] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by pade approximation via the Lanczos process," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 14, no. 5, pp. 639–649, May 1995.

[6] ——, "Reduced-order modeling of large linear subcircuits via block Lanczos algorithm," in *Proc. Design Automation Conf.*, 1995, pp. 376–380.

[7] R. W. Freund and P. Feldmann, "Reduced-order modeling of large linear subcircuits by means of the sypvl algorithm," in *Proc. Int. Conf. Computer-Aided Design*, 1996, pp. 280–287.

[8] F. R. Gantmacher, *The Theory of Matrices*. New York: Chelsea, 1977, vol. 1.

[9] G. Gielen and R. Rutenbar, "Computer-aided design of analog and mixed-signal integrated circuits," *Proc. IEEE*, vol. 88, no. 12, pp. 703–717, Dec. 2000.

[10] M. M. Hassoun and P. M. Lin, "A hierarchical network approach to symbolic analysis of large scale networks," *IEEE Trans. Circuits Systems I: Fundam. Theory Appl.*, vol. 42, no. 4, pp. 201–211, Apr. 1995.

[11] Y. Ismail and E. G. Friedman, "DTT: Direct truncation of the transfer function – An alternative to moment matching for tree structured interconnect," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 21, no. 2, pp. 131–144, Feb. 2003.

[12] J. Lillis, C. Cheng, S. Lin, and N. Chang, *Interconnect Analysis and Synthesis*. New York: Wiley, 1999.

[13] J. W. Liu, "Modification of the minimum degree algorithm by multiple elimination," *ACM Trans. Math. Software*, vol. 11, pp. 141–153, 1985.

[14] A. Odabasioglu, M. Celik, and L. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 17, no. 6, pp. 645–654, Jun. 1998.

[15] A. Pacelli, "A local circuit topology for inductive parasitics," in *Proc. Int. Conf. Computer-Aided Design*, 2002, pp. 208–214.

[16] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 10, no. 4, pp. 352–366, Apr. 1990.

[17] L. T. Pillage, R. A. Rohrer, and C. Visweswariah, *Electronic Circuit and System Simulation Methods*. New York: McGraw-Hill, 1994.

[18] Z. Qin and C.-K. Cheng, "RCLK-VJ network reduction with Hurwitz polynomial approximation," in *Proc. Asia South Pacific Design Automation Conf.*, Jan. 2003, pp. 283–291.

[19] Z. Qin and C. Cheng, "Realizable parasitic reduction using generalized $Y-\Delta$ transformation," in *Proc. Design Automation Conf.*, 2003, pp. 220–225.

[20] B. N. Sheehan, "TICER: Realizable reduction of extracted *RC* circuits," in *Proc. Int. Conf. Computer-Aided Design*, 1999, pp. 200–203.

[21] C.-J. Shi and X.-D. Tan, "Canonical symbolic analysis of large analog circuits with determinant decision diagrams," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 19, no. 1, pp. 1–18, Jan. 2000.

[22] ——, "Compact representation and efficient generation of s-expanded symbolic network functions for computer-aided analog circuit design," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 20, no. 7, pp. 813–827, Apr. 2001.

[23] M. Silveira, M. Kamon, I. Elfadel, and J. White, "A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of *RLC* circuits," in *Proc. Int. Conf. Computer-Aided Design*, 1996, pp. 288–294.

[24] M. Silveira, M. Kamon, and J. White, "Efficient reduced-order modeling of frequency-dependent coupling inductances associated with 3-D interconnect structures," in *Proc. Design Automation Conf.*, Jun. 1995, pp. 376–380.

[25] J. A. Starzky and A. Konczykowska, "Flowgraph analysis of large electronic networks," *IEEE Trans. Circuits Syst.*, vol. 33, no. 3, pp. 302–315, Mar. 1986.

[26] S. X.-D. Tan, "A general s-domain hierarchical network reduction algorithm," in *Proc. Int. Conf. Computer-Aided Design*, 2003, pp. 650–657.

[27] S. X.-D. Tan, Z. Qi, and H. Li, "Hierarchical modeling and simulation of large analog circuits," in *Proc. Eur. Design Test Conf.*, Feb. 2004, pp. 740–741.

[28] S. X.-D. Tan and C.-J. Shi, "Efficient DDD-based interpretable symbolic characterization of large analog circuits," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E86-A, no. 12, pp. 3112–3118, 2003.

[29] S. X.-D. Tan and J. Yang, "Hurwitz stable model reduction for nontree structured *RLCK* circuits," in *Proc. IEEE Int. System-on-Chip Conf.* , 2003, pp. 239–242.

[30] X.-D. Tan and C.-J. Shi, "Hierarchical symbolic analysis of large analog circuits via determinant decision diagrams," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 19, no. 4, pp. 401–412, Apr. 2000.

[31] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*. New York: Van Nostrand Reinhold, 1995.

[32] J. Yang and S. X.-D. Tan, "Behavioral modeling of analog circuits by dynamic semisymbolic analysis," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2004, pp. V105–108.

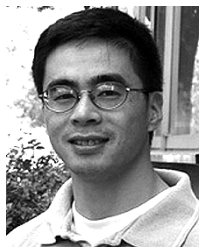
[33] ——, "An efficient algorithm for transient and distortion analysis of mildly nonlinear analog circuits," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2004, pp. V129–132.

[34] X. Yang, C.-K. Cheng, W. Ku, and R. Carragher, "Hurwitz stable reduced order modeling for *RLC* interconnect trees," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2000, pp. 222–228.

[35] H. Yu and L. He, "Vector potential equivalent circuit based on PEEC inversion," in *Proc. Design Automation Conf.*, 2003, pp. 781–723.

[36] H. Yu, S. X.-D. Tan, and L. He, "rVPEC: Realizable Model for *RLCM* Network Based on Hierachical Circuit Reduction," Tech. Rep. UCR, Dept. Elect. Eng., Univ. California, Riverside, 2004.

[37] G. Zhong, C. Koh, and K. Roy, "On-chip interconnect modeling by wire duplication," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 22, no. 11, pp. 1521–1532, Nov. 2003.



Sheldon X.-D. Tan (S'96–M'99) 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 an Assistant Professor in the Department of Electrical Engineering, University of California, Riverside. He was a faculty member in the Electrical Engineering Department, Fudan University from 1995 to 1996. He was with Monterey Design Systems Inc., CA, from 1999 to 2001, and the Altera Corporation, CA, from 2001 to 2002. He coauthored the book *Symbolic Analysis and Reduction of VLSI Circuits* (Norwell, MA: Springer/Kluwer, 2005). His research interests include several aspects of design automation for VLSI integrated circuits: modeling, analysis, and optimization of mixed-signal/RF/analog circuits, high-performance and intelligent embedded systems, signal integrity issues in VLSI physical design, high-performance power/ground distribution network design and optimization.

Dr. Tan received a Best Paper Award at the 1999 IEEE/ACM Design Automation Conference. He served on the Technical Program Committees of several international conferences, such as ISCAS'04, ISCAS'05, and ASPDAC'05.