

SYMBOLIC CIRCUIT ANALYSIS

Circuit simulation is an integral part of tools used in circuit design. The principle of symbolic analysis is to derive analytic (or symbolic) functions describing the circuit responses, using symbols representing (all or some) circuit parameters rather than their numerical values. These parameters may be of a different nature. In the case of most common alternating current (ac) analysis, element parameters and frequency are represented in symbolic form. Evaluation of these symbolic functions for specific values of symbols provides, if required, numerical responses of the circuit. In the symbolic approach, network functions are determined from the knowledge of network elements, their nature, and their connections (network topology). For that reason, symbolic analysis is sometimes called topological analysis.

The primary domains of symbolic analysis are direct analysis using either graph-based (topological) or matrix approach,

hierarchical analysis, and approximation techniques. Although the most traditional domain of symbolic computer-aided design (CAD) is ac analysis of lumped, linear, time-invariant networks, the symbolic approach is present in various domains of circuit design starting with the formulae for manual calculations or spreadsheets used in the design process. Different applications have been developed based on the symbolic approach. These include speed-up of calculation in domains like circuit optimization and statistical analysis; design automation; device characterization; and structural synthesis.

BASIC CONCEPTS

Most of the well-known programs of circuit simulation exist in the numerical version. It means that a numerical value should be given for any circuit parameter, and the simulated circuit functions are presented as numerical tables or plots. For instance, in ac analysis for each frequency point, the complex value of voltage or current is calculated. For symbolic analysis, there is no need to specify element values when analysis starts. Element parameters are present as symbols in simulation results and are valid for any numerical value of these parameters. The numerical results can thus be obtained by simple function evaluation for a specified set of parameter values.

Symbolic results of circuit analysis may be generated in different forms depending on whether all or only some of network elements are characterized by symbolic parameters.

For linear, lumped, and stationary (LLS) circuits, the symbolic network functions can be presented in the form of rational functions of complex frequency s

$$H(s) = \frac{N(s, x_1, \dots, x_n)}{D(s, x_1, \dots, x_n)} \quad (1)$$

where x_1, \dots, x_n are parameters of circuit elements.

In Eq. (1), coefficients of the polynomials N and D are the sums of products of circuit parameters and can be expressed as follows:

$$P(s, x_1, \dots, x_n) = \sum_{i=0}^m s^i \sum_{j=1}^{n_i} \prod_{k \in K_{ij}} x_k \quad (2)$$

This format is called the classical *sum-of-products* (SOP) form. If each product of symbols appears only once in this expression, we have the case of a canonic SOP form. Equation (2) can also be presented as a product of two or more polynomials. In this case, we have a nested form of symbolic results. A nested polynomial is called cancellation-free, if its expansion yields a canonic form. An example of canonic SOP and nested form of symbolic results follows.

$$\text{Canonic SOP form:} \quad s^2 C_1 C_2 + s C_1 G_2 + s C_2 G_1 + G_1 G_2 \quad (3)$$

$$\text{Canonic nested form:} \quad (s C_1 + G_1)(s C_2 + G_2) \quad (4)$$

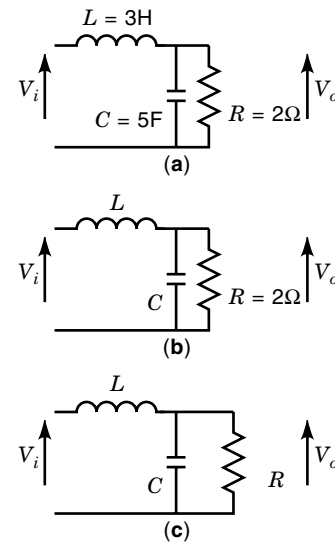


Figure 1. Three levels of symbolic analysis.

Example 1. Let us calculate the voltage transfer function V_o/V_i of circuits presented in Fig. 1(a–c). The small signal analysis gives following expressions.

1. For the circuit in Fig. 1(a), the complex frequency s is the only symbolic parameter. The transfer function is a rational function of s with real coefficients:

$$\frac{V_o}{V_i} = \frac{2}{30s^2 + 3s + 2} \quad (5)$$

2. In Fig. 1(b), capacitor C and inductance L are symbolic while $R = 2 \Omega$. This is the case of partially symbolic or semisymbolic network function:

$$\frac{V_o}{V_i} = \frac{2}{2s^2 CL + sL + 2} \quad (6)$$

3. In Fig. 1(c), all circuit parameters are represented by symbols. This is the case of fully symbolic network function:

$$\frac{V_o}{V_i} = \frac{R}{s^2 CLR + sL + R} \quad (7)$$

HISTORY OF SYMBOLIC METHODS

Methods of symbolic analysis started in the nineteenth century with Kirchhoff's work. Symbolic methods were intensively developed in the 1960s and 1970s in parallel with the growing popularity of computer programs of circuit analysis. Different computer programs for direct symbolic analyses were then developed. The limitation of direct methods came from the exponential growth of the number of terms in generated expressions with circuit size. To overcome this problem, decomposition techniques were proposed. Instead of analyzing the network as a whole, the network is cut into parts, each part is analyzed separately, and partial results are combined to form a description of the whole network. An-

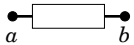

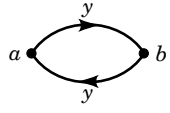
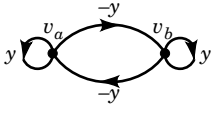
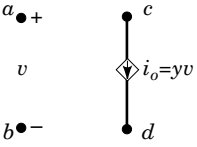
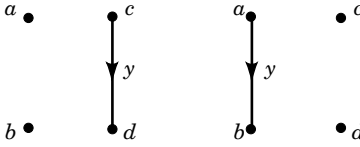
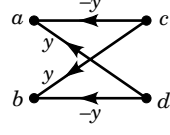
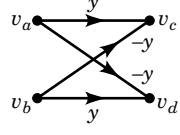
Element	Equations	Current voltage graphs	Directed graph	Coates graph
Admittance 	$i_a = y(v_a - v_b)$ $i_b = y(v_b - v_a)$			
VCCS 	$i_a = 0 \quad i_b = 0$ $i_c = y(v_a - v_b)$ $i_d = y(v_b - v_a)$			

Figure 2. Different graph representations of circuit elements.

other approach to overcoming the complexity of symbolic results is the technique of approximation of symbolic expressions. In this case, the less significant terms of the function are discarded. A controlled error of such truncated results is admitted as a cost of important simplification of symbolic expressions. With the progress in the methods of symbolic function generation, new application domains emerged simultaneously. To the primary advantages of the symbolic approach [e.g., better understanding of circuit operation (so-called insight into circuit behavior) or speed-up of circuit simulation in repetitive analysis] were added new ones like automated generation of analytic models, transistor sizing, exploration of new topologies, and various automated design methods based on the availability of symbolic description of circuit topology.

DIRECT SYMBOLIC METHODS

When symbolic functions are obtained directly from the representation of the whole network, we have the case of direct symbolic analysis, which is also called flat analysis. A different approach is presented with the hierarchical analysis of decomposed networks. This second approach is essentially oriented for the analysis of larger circuits.

Two basic approaches are used to derive symbolic functions: graph-based methods and algebraic methods. For simplicity, only networks having the admittance representation will be considered in the following presentation. The extension to any circuit topology can easily be done by introducing auxiliary circuit elements in much the same way as adding new variables allows modified nodal admittance (MNA). Different graph representations for two basic admittance elements of electrical networks are presented in Fig. 2.

A simple circuit representing a small signal model of bipolar transistor presented in Fig. 3 will be used as an example to illustrate different methods of topological analysis.

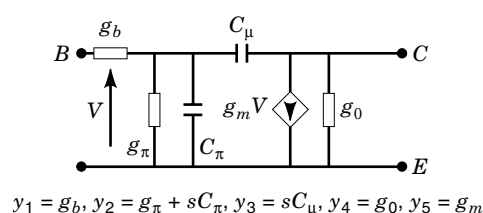


Figure 3. Small signal model of bipolar transistor.

Graph-Based Methods

In the graph-based (or topological) methods, symbolic functions are derived by topological operations on some graph representation of the circuits. These operations consist mainly in generation of specific subgraphs such as, for instance, spanning trees and multitrees or sets of paths and loops. Detailed description of graph-based methods and an important list of references is provided by Chen (1).

Pair of Conjugate Graphs. A current–voltage graph is the most common circuit representation using a pair of conjugate graphs.

For a circuit composed of admittance-type elements [passive admittance elements and voltage-controlled current source (VCCS) only] a *current–voltage pair of graphs* (G_I, G_V) is determined as follows:

1. The sets of nodes (identical for two graphs) correspond to circuit nodes.
2. The passive element with an admittance y connected between nodes i and j is represented by a pair of branches with weight y . In both graphs, these branches have the same direction [e.g., from node i to j (or from j to i)].
3. The VCCS source with the controlling voltage between nodes i (+) and j (–) and the current source between nodes k (outgoing) and p (incoming) is represented by a branch from i to j in the voltage graph and a conjugate branch from k to p in the current graph. The weight of this branch is equal to the transconductance controlling coefficient (g_m).

The pair of current–voltage graphs for the circuit from Fig. 3 is presented in Fig. 4.

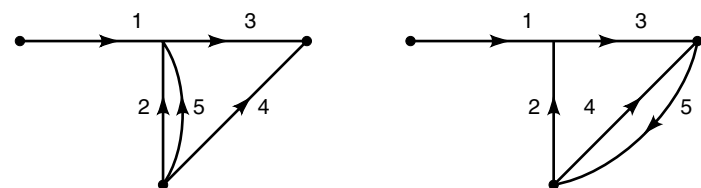


Figure 4. Pair of current–voltage graphs based on the bipolar transistor model.

Another popular circuit description leading to a two-graph representation is the *nullator–norator* network. Two additional singular elements (nullator and norator) are used in this circuit representation. *Nullator* is a two-terminal element with $i = 0$ and $v = 0$ and *norator* a two-terminal element with undefined i and v . A nullator–norator pair is called *nullor*. This type of representation is particularly convenient for circuits with ideal operational amplifiers, which can be represented as a nullator–norator pair. The pair of conjugate graphs is obtained by short-circuiting nullators and opening norators (one graph) and by short-circuiting norators and opening nullators (another graph). The analysis of nullator–norator networks was proposed by Davies (2).

Circuit functions can be expressed using determinants and cofactors of the admittance matrix Y . The following topological formulae can be used to calculate necessary determinants and cofactors of the circuit represented by a pair of conjugate graphs.

$$\begin{aligned}\Delta Y &= \sum_{t \in T} \text{sign}(t) \cdot \text{val}(t) \\ \Delta Y_{ii} &= \sum_{t \in T_{i,0}} \text{sign}(t) \cdot \text{val}(t) \\ \Delta Y_{ij} &= (-1)^{i+j} \sum_{t \in T_{ij,0}} \text{sign}(t) \cdot \text{val}(t)\end{aligned}\quad (8)$$

where T is the set of common trees of current and voltage conjugate graphs, $\text{sign}(t)$ is the sign of a common tree t and is a product of the signs of t in G_i and G_v , which are defined as determinants of tree incidence matrices, $\text{val}(t)$ is the product of all branch weights (admittances) from the tree t , $T_{X,Y}$ is the set of common 2-trees such that nodes from the set X and Y are in the disjoint parts, and 0 is the reference node.

Let us calculate the input admittance of the circuit from Fig. 3. Input admittance Y_i can be expressed using the following formula:

$$Y_i = \frac{\Delta Y}{\Delta Y_{11}} \quad (9)$$

According to Eq. (8), we should calculate trees and 2-trees of the pair of graphs in Fig. 4. The set of trees of the voltage graph is equal $\{(1, 2, 3), (1, 2, 4), (1, 5, 3), (1, 5, 4), (1, 3, 4)\}$. Because the tree $(1, 5, 4)$ is not a tree in the current graph, the set of common trees is equal to $\{(1, 2, 3), (1, 2, 4), (1, 5, 3), (1, 3, 4)\}$. The set of common 2-trees $T_{1,0}$ is equal to $\{(1, 3), (1, 4), (2, 3), (2, 4), (3, 4), (3, 5)\}$. All signs of trees and 2-trees are positive, including those with branches corresponding to active elements like $(1, 5, 3)$ and $(3, 5)$. With the Eqs. (8) and (9), the symbolic input admittance of the circuit in Fig. 3 can be presented as follows:

$$\begin{aligned}Y_{\text{in}} &= \frac{y_1(y_2y_3 + y_2y_4 + y_5y_3 + y_3y_4)}{y_1y_3 + y_1y_4 + y_2y_3 + y_2y_4 + y_3y_4 + y_3y_5} \\ &= \frac{g_b[(g_\pi + sC_\pi)sC_\mu + (g_\pi + sC)g_0 + sC_\mu g_m + sC_\mu g_0]}{g_b sC_\mu + g_b g_0 + (g_\pi + sC_\pi)sC_\mu + (g_\pi + sC)g_0 + sC_\mu g_m + sC_\mu g_0}\end{aligned}\quad (10)$$

Directed Graphs. The main goal of directed graph formalism was to eliminate cumbersome sign calculations necessary in the conjugate graph approach. In the directed graph

method, the network is represented with a single directed graph G_d and its determinant is calculated by enumeration of directed trees of this graph.

Directed graph G_d , for a given circuit, is determined as follows:

1. The set of graph nodes corresponds to the set of network nodes.
2. The passive element with an admittance y connected between the node i and j is represented by a pair of directed branches with the weight y (one from i to j and another from j to i).
3. The VCCS source with the controlling voltage between nodes i (+) and j (–) and the current source between nodes k (outgoing) and p (incoming) is represented by a set of four branches connected as follows: p to i and k to j with the weight g_m and k to i and p to j with the weight $-g_m$.

A subgraph T_d is said to be a *directed tree* of G_d , rooted at node r , if and only if:

- in T_d , node r has no outgoing branches and each remaining node has exactly one outgoing branch;
- the resultant undirected subgraph T is a tree of the resultant undirected graph G after all branch directions removed.

A subgraph $T_{i,j}$ is said to be a *directed 2-tree* of G_d , with roots i and j , if and only if:

- each component (possibly an isolated node) of $T_{i,j}$ is a directed tree with node i and j as the root;
- the resultant undirected subgraph is a 2-tree of the resultant undirected graph G after all branch directions removed.

If some nodes have to be in the same component as the root node, their symbols will be added to the root subscript.

Determinant and cofactors of the admittance matrix can be calculated as follows:

$$\begin{aligned}\Delta Y &= \sum_{t \in T} \text{val}(t) \\ \Delta Y_{ii} &= \sum_{t \in T_{i,0}} \text{val}(t) \\ \Delta Y_{ij} &= (-1)^{i+j} \sum_{t \in T_{ij,0}} \text{val}(t)\end{aligned}\quad (11)$$

where T is the set of directed trees of the graph G_d , $\text{val}(t)$ is the product of all branch weights with their signs from the tree t , $T_{X,Y}$ is the set of directed 2-trees, and 0 is the reference node.

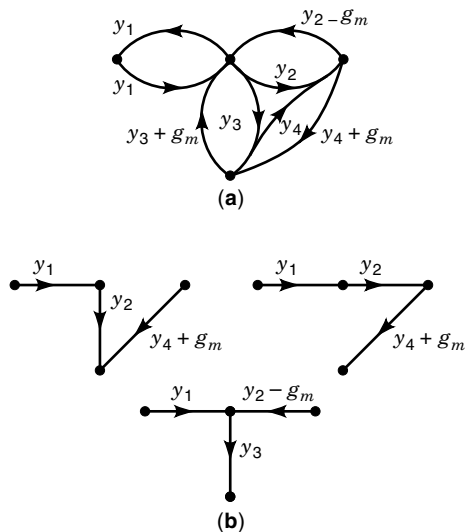


Figure 5. (a) Directed graph of bipolar transistor model and (b) its set of directed trees.

The directed graph for the circuit from Fig. 3 is presented in Fig. 5(a). The set of directed trees of that graph is presented in Fig. 5(b).

Signal-Flow Graph. Signal-flow graph (SFG) formalism was introduced by Mason (3) and is a popular technique to visualize and solve problems in various domains of engineering. Slightly modified equivalent representation was proposed by Coates (4). Both formalisms are used in topological analysis.

Let us consider a system of n linear equations in a form

$$X = AX + BZ \quad (12)$$

where X is the $n \times 1$ vector of unknowns, Z — $m \times 1$ vector of input variables, and A and B are the coefficient matrices $n \times n$ and $n \times m$, respectively.

Mason signal-flow graph G_m representing a system of Eq. (12) is a weighted directed graph built with the following rules:

1. Graph nodes represent variables X and Z (knowns and unknowns).
2. Each nonzero term a_{ij} is represented by a branch (j to i) with a weight a_{ij} .
3. Each nonzero term b_{ik} terms is represented by a branch (k to i) with a weight b_{ik} .

An n th-order loop in Mason graph is a set of n nontouching loops.

Mason Rule. Any transfer function $T_{ij} = x_i/z_j$ can be calculated as follows:

$$T_{ij} = \frac{\sum_k P_k(i, j) \Delta_k}{\Delta} \quad (13)$$

where $\Delta = 1 - (\text{sum of all loop weights}) + (\text{sum of all second-order loop weights}) - (\text{sum of all third-order loop weights})$

. . . , $P_k(i, j)$ is the weight of the k th path from the node z_j to x_i , and Δ_k is calculated as Δ without any loops touching the path $P_k(i, j)$.

The Coates graph G_c is associated with a square $n \times n$ matrix A according to the following rules:

1. The graph has n nodes associated with the matrix rows (unknowns).
2. Each nonzero element a_{ij} generates a branch from j to i with a weight a_{ij} .

Let us denote W as a set of pairs of nodes in the Coates graph $W = \{(v_1, r_1), \dots, (v_k, r_k)\}$ where $v_s \neq v_m$, $v_s \neq r_m$, $r_s \neq r_m$, for $s \neq m$.

k -connection (multiconnection) of graph G_c , defined by a set W , is a subgraph p_W composed of k node-disjoint directed paths and node-disjoint directed loops incident with all graph nodes. The initial node of i th path is v_i , and the terminal node is r_i (pairs of nodes from the set W). 0-connection (or connection) is denoted by p . When $v_i = r_i$, a multiconnection has an isolated node v_i .

The sign of k -connection $p \in P_W$ is equal to

$$\begin{aligned} \text{sign}(p) &= (-1)^{n+k+l_p} \text{ord}(v_1, \dots, v_k) \text{ord}(r_1, \dots, r_k) \\ &= (-1)^{l_p} \text{sign}(P_W) \end{aligned} \quad (14)$$

where

$$\text{ord}(x_1, x_2, \dots, x_k) = \begin{cases} 1 & \text{when the number of permutations} \\ & \text{ordering the set is even} \\ -1 & \text{otherwise} \end{cases}$$

$$\begin{aligned} n &= \text{number of graph nodes} \\ l_p &= \text{number of loops in } k\text{-connection } p. \end{aligned}$$

Note that for all k -connections of the type P_W , the sign can be calculated as a product of a common term $\text{sign}(P_W)$ and a term $(-1)^{l_p}$ depending on the number of loops in the k -connection.

Determinant and cofactors of the admittance matrix represented by a Coates graph G_c can be calculated as follows:

$$\begin{aligned} \Delta Y &= \sum_{p \in P} \text{sign}(p) \text{val}(p) = \text{sign}(P) \sum_{p \in P} (-1)^{l_p} \text{val}(p) \\ \Delta Y_{ii} &= \sum_{p \in P_{\{(i,i)\}}} \text{sign}(p) \text{val}(p) = \text{sign}(P_{\{(i,i)\}}) \sum_{p \in P_{\{(i,i)\}}} (-1)^{l_p} \text{val}(p) \\ \Delta Y_{ij} &= (-1)^{i+j} \sum_{p \in P_{\{(i,j)\}}} \text{sign}(p) \text{val}(p) = (-1)^{i+j+n+1} \text{sign}(P_{\{(i,j)\}}) \\ &\quad \sum_{p \in P_{\{(i,j)\}}} (-1)^{l_p} \text{val}(p) \end{aligned} \quad (15)$$

where P (P_W) is the set of connections (multiconnections) of the graph G_c , $\text{val}(t)$ is the product of all branch weights with their signs from the tree t , and l_p is the number of loops in a connection (multiconnection).

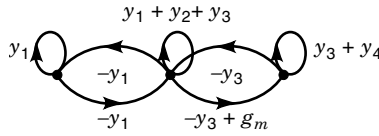


Figure 6. Coates graph of bipolar transistor model with reference node eliminated.

The Coates graph of the circuit in Fig. 3 is presented in Fig. 6.

Symbolic topological methods are closely linked with the graph theory, its methods, and its algorithms. In fact, after a graph representing the network is generated, the main difficulty of topological analysis consists in determining various subgraphs like trees, multitrees, paths, connections, and multiconnections. An important bibliography concerning this subject can be found in Refs. 1 and 5.

An important aspect of generation algorithms is the problem of term cancellations. The cancellation of terms can be of two types: *additive cancellation*, in which two equal terms with opposite sign cancel, or *multiplicative cancellation*, in which a numerator and a denominator factors cancel. In the direct symbolic approach, canceling terms can be eliminated with no loss of accuracy, but their generation and further treatment cause an unnecessary complication of the procedure. The main problem with canceling terms is thus the efficiency of the generation method, rather than precision, which is the main preoccupation in the case of the numerical approach. Anyway, the inherently cancellation-free methods are preferred in symbolic simulators.

Two-Port Analysis

Let us consider a two-port network. Any network transfer function can be expressed with the determinant and cofactors of the node admittance matrix. Topological Eqs. (8), (11), and (15) show how to calculate these functions for different topological representations of the network. This operation involves generating different sets of trees and multitrees (connections and multiconnections) of the graph. It is possible to simplify these calculations and obtain all necessary characteristic functions with one generation of trees or connections of so-called *augmented network*. This procedure avoids the separate generation of different kinds of trees and multitrees (respectively, connections and multiconnections) and in particular avoids the cumbersome calculation of the multitrees and multiconnections sign. An augmented network is created by adding some symbolic elements to the initial network. In Fig. 7 an admittance y'_s and VCCS g'_m have been connected to



Figure 7. Augmented network.

the two-port scheme. The admittance matrix Y' of the augmented network can be expressed as follows:

$$Y' = Y + \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 \\ \begin{matrix} 1 \\ 2 \\ \vdots \\ n \end{matrix} \end{matrix} \begin{bmatrix} y'_s & g'_m & -g'_m & 0 & \dots & 0 \\ 0 & 0 & 0 & & & \\ & & \ddots & & & \\ 0 & & & & & 0 \end{bmatrix} \end{matrix} \quad (16)$$

The evaluation of the determinant of such a matrix gives

$$\Delta Y' = \Delta Y + y'_s \Delta Y_{11} + g'_m (\Delta Y_{12} - \Delta Y_{13}) \quad (17)$$

where ΔY and ΔY_{ij} are determinant and cofactors of the initial admittance matrix Y .

A symbolic determinant of the augmented network is calculated keeping all added elements as symbols. Obtained terms can be sorted with respect to the added symbols, and the determinant and cofactors can easily be extracted. Other elements can also be added to calculate the different cofactors needed. Beside the simplification of the generation procedure, the augmented network approach also reduces the necessary calculation effort. In fact, a large part of generation is identical for different cofactors and is executed only once with the previously presented approach.

Algebraic Methods

Interpolation Method. The interpolation method is particularly convenient for the case when s is the only variable kept as a symbol. The admittance matrix determinant as well as cofactors necessary to describe network functions can be expressed as polynomial functions in s . The degree of these polynomials can be easily determined from the types of elements in the network. In the case of the polynomial of the degree n , $P(s) = a_0 + a_1s + \dots + a_ns^n$, polynomial coefficients a_0, a_1, \dots, a_n can be determined by evaluating the function P at $(n + 1)$ distinct values of s and then solving the system of equations:

$$\begin{bmatrix} 1 & s_0 & \dots & s_0^n \\ 1 & s_1 & \dots & s_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & s_n & \dots & s_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} P(s_0) \\ P(s_1) \\ \vdots \\ P(s_n) \end{bmatrix} \quad (18)$$

For small n ($n < 20$), real values of s can be chosen to simplify the calculation. In the case of greater n , the use of real s value leads to ill-conditioned equations. It was shown by Singhal and Vlach (6) that it is best to use complex values of s , uniformly distributed on a unit circle. Using the linear property of network polynomials toward every network parameter, the interpolation method can be extended to handle the problem when there are other symbolic variables beside s . Nevertheless, in this case, the same result can be obtained in a simpler manner with the parameter extraction method.

Parameter Extraction Method. Efficiency of direct graph-based methods does not depend on number of symbolic parameters but on the network complexity. Interpolation methods can handle large networks but are essentially limited to one symbolic parameter s . The combination of two methods

allows the analysis of larger networks with a reasonable set of parameters. A common point of different contributions on this subject is that the variable parameters are “extracted” from the networks determinant and cofactors. The extraction procedure can combine topological or algebraic (numerical) approach.

In the case of k symbolic parameters $\{x_1, x_2, \dots, x_k\}$, network polynomials can be presented in the following form:

$$P(s, x_1, x_2, \dots, x_k) = P_0(s) + [P_1(s)x_1 + P_2(s)x_2 + \dots + P_k(s)x_k] \\ + [P_{12}(s)x_1x_2 + P_{13}(s)x_1x_3 + \dots] \\ + \dots + P_{12\dots k}(s)x_1x_2\dots x_k \quad (19)$$

The extraction process consists of determining the polynomials $P_0, P_1, \dots, P_{12\dots k}$. For each polynomial, a corresponding cofactor can be identified by deleting some rows and columns in the admittance matrix and setting to zero some of the parameters. After this operation, the cofactor can be determined with numerical methods. This approach provides an important advantage when many network branches are characterized by numerical values. Of course, when more and more elements are represented by symbols, the process can attain similar or even greater complexity than topological methods.

The following conclusions can be obtained by comparing the direct methods presented here:

1. Topological methods are most suitable to obtain fully symbolic functions of small networks.
2. Parameter-extraction methods are most suitable for obtaining partially symbolic network functions where only a small part of the network elements are characterized by symbolic parameters.
3. Interpolation methods can be used for large networks when the complex frequency s is the only symbolic parameter.

DECOMPOSITION AND HIERARCHICAL ANALYSIS

Calculating symbolic results with the direct methods already presented is very often limited to the small and medium class of circuits or to the case when a number of parameters kept as symbols is limited. In fact, the number of terms in fully symbolic function grows exponentially with the size of the circuit characterized by a number of nodes and/or number of elements. The number of terms also grows exponentially with the number of symbolic elements. To overcome this difficulty and perform full symbolic analysis of a larger network, a decomposition (partitioning, tearing) technique was proposed. In much the same way as direct methods, analyzing a decomposed network can be based on topological or algebraic approach.

In the node decomposition, which is most often used in symbolic decomposition methods, the network (or its graph) is partitioned into edge-disjoint parts called *blocks*. Nodes common to two or more blocks are called *block nodes*. A decomposed graph can be represented as a *block graph*. A particular case of node decomposition is decomposition into two parts, which is called *bisection*. Any decomposition can be represented as a sequence of bisections. A graph decomposed into

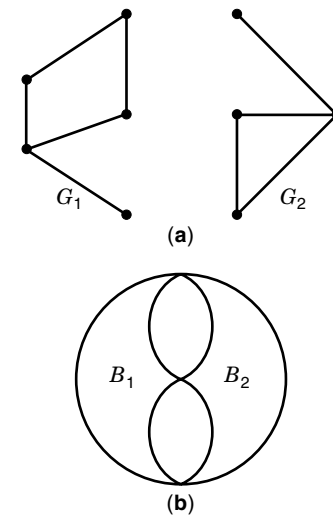


Figure 8. (a) Graph bisection and (b) its block graph.

two parts and the block graph of this bisection are presented in Fig. 8.

The complexity of symbolic analysis of decomposed network depends not only on the size of blocks, but also on the complexity of the block graph. In the case of larger networks, a *simple (one-level) decomposition* may be ineffective. It may result in either elementary blocks that are too large or a block graph that is too complex. When simple decomposition is applied to subgraphs, we deal with *hierarchical decomposition*.

An example of hierarchical decomposition is presented in Fig. 9. The graph G_1 was first decomposed into two parts G_2 and G_3 . In the following step, G_2 was decomposed into G_4, G_5 and G_6 , and G_3 into G_7 and G_8 .

The hierarchical decomposition structure can be illustrated by a tree of decomposition. If a subgraph G_k was ob-

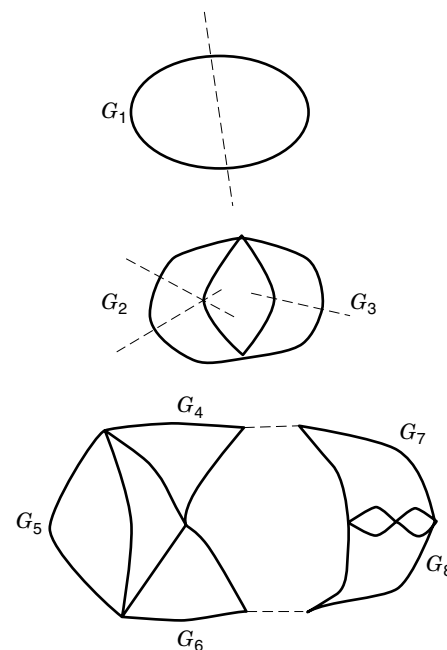


Figure 9. Hierarchical decomposition.

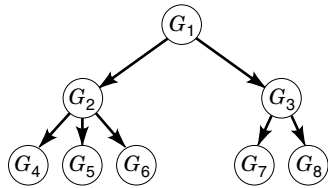


Figure 10. Tree of decomposition.

tained during decomposition of a subgraph G_i , then there is an edge from node G_i to node G_k in the decomposition tree. In the decomposition tree, we have one *initial node*, which is the root of the tree. *Terminal nodes* are leaves of the tree. All nodes that are not terminal nodes are *middle nodes*. For middle nodes, we determine the *decomposition level*, which is equal to the number of nodes in the path from the initial node to that node. The *range of hierarchical decomposition* is equal to the maximal decomposition level. Every middle node has its descendants, and every node except the initial one has its *ascendant*. Figure 10 shows the tree of the decomposition presented in Fig. 9.

Topological Hierarchical Analysis

The hierarchical analysis of decomposed network consists of two parts: terminal block analysis and middle block analysis. It starts with the initial node (first level of decomposition) and proceeds down to the next levels according to the connections in the decomposition tree. On each intermediate level, the type of necessary functions to be calculated at the next level are determined. At the terminal node, the enumeration of necessary trees and multitrees (or multiconnections) of the terminal block is realized. It is possible to organize the exploration of the decomposition tree in such a way that no multiple passes through the hierarchical structure are necessary. Symbolic results of terminal blocks have a classical “sum of products” form. Middle block analysis results are combinations of functions calculated at superior decomposition level. These functions are not developed and provide final network description in a nested form. The case of topological approach to the hierarchical analysis of decomposed Coates graph and associated algorithms were presented by Starzyk and Konczykowska (7).

Hierarchical Analysis with the Sequence of Expression Technique

Hierarchical analysis based on matrix approach was presented by Hassoun and Lin (8) and called the *sequence of expressions* approach. This methodology uses the Reduced Modified Nodal Analysis (RMNA) technique, which allows the network to be characterized in terms of only a small subset of the network variables (external variables). The decomposed network is represented by a binary decomposition tree. During the terminal block analysis, all internal variables are suppressed leaving only external variables necessary for the next level of processing. This operation, performed analytically, results in symbolic RMNA equations of the terminal block. For each middle block, the combination of results obtained from the descendant block produces RMNA equations for the new block. After this operation is realized, the new block is treated as a terminal block. As a final result of this procedure, charac-

teristic functions of the total initial network are expressed as a sequence of expressions (nested-form).

Properties of Hierarchical Analysis

The hierarchical decomposition approach drastically reduces the time of analysis and the complexity of symbolic results. This last aspect is crucial for the further utilization of symbolic expressions (e.g., function evaluation). The total number of operations (multiplications and additions) necessary for the evaluation of characteristic functions of the initial network serves as an evaluation criterion of the decomposition quality.

Another interesting aspect of the decomposition approach is that large parts of calculations can be realized independently. These parallel processing properties make the decomposition technique very suitable for implementation on multi-processor machines.

Closely related to the hierarchical analysis of a decomposed network is the problem of network or graph partitioning. The main criterion for evaluating the decomposition quality is the low complexity of obtained results. Although a reasonable analysis time should also be preserved, most often this post-processing time should be minimized after the results are generated and then exploited for multiple evaluations. In the case of hierarchical decomposition being a sequence of bisections, optimal results are theoretically obtained if each bisection divides the graph into two equal or similar in size parts. A decomposition tree should be regular (2^{n-1} blocks on n th level of the decomposition). The minimization of the number of common block nodes for each bisection is very important, especially in the case of the topological approach, where formulae are particularly simple in the case of two common block nodes. The number of decomposition levels depends on the size of the initial network and how sparse it is. An additional constraint for the decomposition procedure is the identification of blocks having identical structures. In the case when multiple block have an identical topology, the symbolic analysis and storage of results can be executed only for one of such isomorphic blocks. Of course, the automated generation of optimal decomposition is a final, unfortunately not yet achieved, goal. In practical cases, only heuristic strategies are known.

It is possible to realize symbolic hierarchical analysis without decomposition. Starzyk and Zou (9) proposed such an approach based on Coates graph representation. A node exploding technique locally simplifies the graph by node elimination. Iterative repetition of this step allows us to obtain result for the total network. Pierzchala and Rodanski (10) developed a similar method based on the algebraic approach. Internal variables are suppressed using Gaussian elimination. Locally optimal pivoting strategy allows us to minimize the number of symbolic operations of the final sequence of expressions.

APPROXIMATION TECHNIQUES

The hierarchical approach provides a solution for the symbolic analysis of large networks. The exact results are obtained in the nested form. Nevertheless, for some applications, this form of results is not adequate. In many design problems, identifying dominant elements and their role in global performance is essential. The interpretation of results in nested form is not easy. Exact symbolic network functions

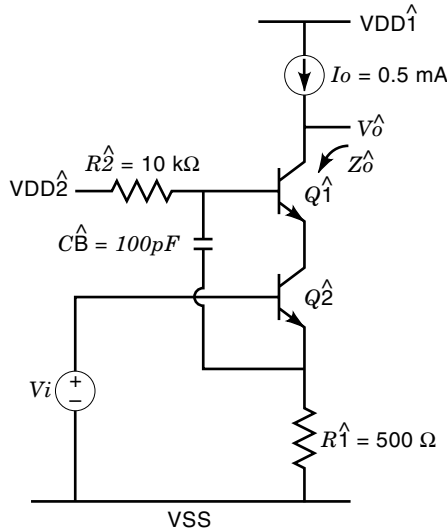


Figure 11. Bipolar cascode stage.

in expanded (flat) form are complex expressions characterized by a large number of terms even in the case of medium size networks. Approximation techniques propose to tackle this problem.

It can be observed that for exact symbolic expressions of large network, considering given frequency range and order of magnitude (or range) of element values, the magnitude of terms can vary significantly. The symbolic approximation technique consists of discarding smaller terms while keeping the error (difference between exact and simplified function) under control. In this approach a trade-off between accuracy and simplicity of results must be realized. Note that this kind of strategy is a common practice for the manual design of analog circuits.

Example 2 [proposed by Gielen and Sansen (11)]. Let us consider a bipolar cascode stage with bootstrap capacitor C_B as presented in Fig. 11. A plot of output impedance for a given

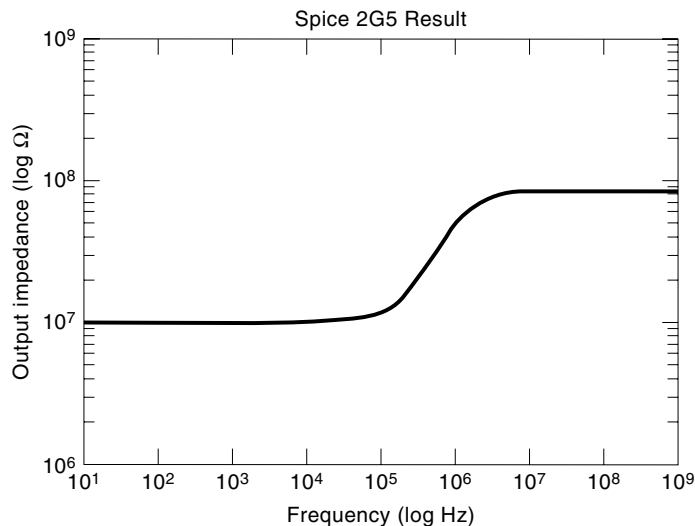


Figure 12. Output impedance of bipolar cascode stage—results of numerical simulation.

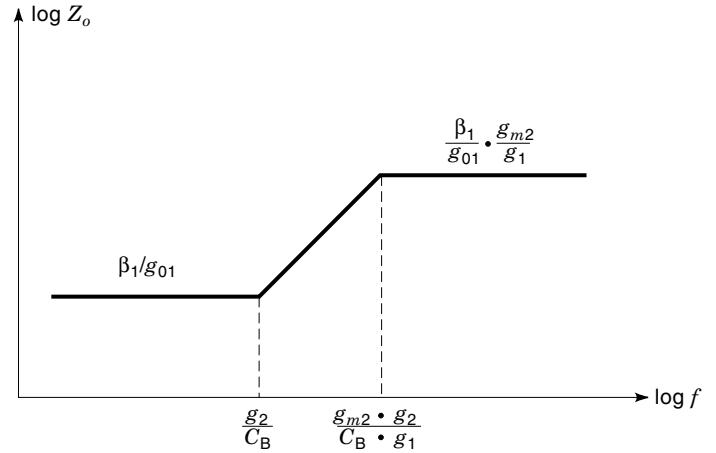


Figure 13. Output impedance of bipolar cascode stage—interpretation of symbolic results.

set of numerical values, as obtained with numerical simulation, is presented in Fig. 12 (bipolar transistors are modeled with r_π , g_m , and g_o elements only). Although overall behavior of the function can be apprehended, it is difficult to predict the influence of various elements and in particular of the capacitor C_B on the output impedance even in the case of such a simple circuit. Fully symbolic analysis of this network produces a result in the form of the rational function with 36 symbolic terms in the numerator and 14 terms in denominator. Interpretation of such function is not easy. If a 10% maximal error (see next paragraph for precise definition) is accepted, the simplified function has the following form:

$$Z_{o(10\%)} = \frac{g_{m1}(g_{m2} + g_1)(g_2 + sC_B)}{g_{o1}g_{\pi 1}[g_2(g_{m2} + g_1) + sC_B(g_1 + g_{\pi 2})]} \quad (20)$$

If a 25% error is allowed, the output impedance is given by

$$Z_{o(25\%)} = \frac{g_{m1}g_{m2}(g_2 + sC_B)}{g_{o1}g_{\pi 1}(g_2g_{m2} + sC_Bg_1)} \quad (21)$$

The impedance levels and pole and zero estimations can be easily retrieved from Eq. (21).

$$\begin{aligned} Z_o(\text{low } f) &\cong \frac{g_{m1}}{g_{\pi 1}g_{o1}} \\ z &\cong -\frac{g_2}{C_B} \\ p &\cong -\frac{g_{m2}g_2}{g_1 C_B} \\ Z_o(\text{high } f) &\cong \frac{g_{m1}g_{m2}}{g_{\pi 1}g_{o1}g_1} = \frac{g_{m2}}{g_1}(\text{low } f) \end{aligned} \quad (22)$$

The general form of impedance magnitude as well as the influence of different parameters is summarized in Fig. 13.

The principle of approximation (or simplification) involves finding an expression that is as simple as possible, which, for a given range of parameter values, reproduces the behavior of the original function with an error below a specified level.

The approximation amplitude and phase errors (ϵ_A and ϵ_p) can be expressed as follows:

$$\begin{aligned} \|H(s, x) - |H^*(s, x)\| &\leq \epsilon_A |H(s, x)| \\ |\angle H(s, x) - \angle H^*(s, x)| &\leq \epsilon_p \end{aligned} \quad (23)$$

for $s = j\omega$, $\omega \in (\omega_1, \omega_2)$, and $x_i \in (x_{i1}, x_{i2})$, where $H(s, x)$ is the exact circuit function and $H^*(s, x)$ is the approximating function.

The majority of the approximation methods proposed to date use a simplified criterion, where the error is measured only for a given set of circuit parameters x^0 , called the nominal design point.

Network characteristic functions are expressed as rational functions in the complex frequency s , with coefficients being the sums of products of symbolic parameters. The most natural way of approximation seems to use only the most significant terms present in the exact expression. Although it is easy to define an error of a real coefficient by eliminating some composing terms, it is more difficult to derive univocal conclusions concerning the general approximation criteria. In fact, the function characteristics depend strongly on the position of poles and zeros, and this error should be closely monitored during the simplification process.

Existing approximation strategies can be classified as follows:

- simplification after generation (SAG) and
- simplification during generation (SDG).

The SAG [known also as approximation after computation (AAC)] approach consists of generating the exact symbolic expression and then eliminating the less significant terms. An obvious disadvantage of such an approach lays in the necessity of generating a large number of unnecessary terms that not only lengthen the analysis but require a storage space to save a huge amount of terms for further comparison and elimination.

The SDG [known also as *approximation during computation* (ADC)] methods were developed to avoid these inconveniences. The generation of symbolic terms in a decreasing order of magnitude is the basis of these methods. The generation is stopped when the error reaches the specified level. Proposed methods use the Gabow (12) algorithm of ordered spanning trees generation. In the case of a frequency-dependent circuit, this procedure should be applied separately to different powers of s . Reference 13 shows that mathematical formalism of matroids is well suited to describe problems of SDG. The approximation techniques were developed by Wambacq et al. (14) and Yu and Sechen (15). More details on this subject can be found in Ref. 16.

Another category of approximation methods, when simplification starts before the generation process, is called *simplification before generation* (SBG). A method, proposed by Yu and Sechen (15) and called *sensitivity-based two-graph simplification*, involves eliminating some graph branches weakly contributing to the final results. In the approach presented by Sommer et al. (17), the simplification starts on the equation level. Algorithms are proposed, allowing us to eliminate some terms in the system of equations and in the corresponding matrix. Because of this simplification, less significant terms are not generated. The SBG methods can be defined as an

equation (or graph)-based approach, whereas SAG and SDG are considered as *solution-based* methods.

Note that a specific procedure should be applied for approximation in cases when matching elements are present in the network. In fact, some terms may either cancel or result in a new term with weight largely different than the original ones. In many situations, better interpretation and understanding of results can be obtained by introducing explicitly the mismatching between device parameters. For instance, instead of two symbolic parameters x_1 and x_2 , which are considered as matching ones, the two new parameters x and Δx can be used, with $x_1 = x$ and $x_2 = x + \Delta x$.

DIFFERENT TYPES OF SYMBOLIC ANALYSIS

Sensitivity Analysis

In the practical design process, it is necessary not only to obtain desired circuit performance at the nominal point but also to control the effect of parameters' deviation on this performance. This is especially important in the integrated circuit design where only limited precision of device parameters is guaranteed and the postfabrication tuning is, in general, not possible. Unnormalized sensitivity of a network characteristic function H with respect to a device parameter x is defined as $\partial H/\partial x$. The normalized sensitivity, denoted as S_x^H , is defined as

$$S_x^H = \frac{\partial H}{\partial x} \frac{x}{H} \quad (24)$$

H is a rational function with numerator and denominator being linear functions with respect to x and can be presented as

$$H = \frac{N}{D} = \frac{N_0 + xN_x}{D_0 + xD_x} \quad (25)$$

where N_0 , N_x , D_0 , D_x are polynomials in s not containing the variable x .

After the derivation of Eq. (25), we obtain

$$S_x^H = x \left(\frac{N_x - D_x}{D} \right) \quad (26)$$

The analytical formula for sensitivity can thus be calculated by adequate sorting of terms in the symbolic expressions for numerator and denominator.

Switched-Capacitor Networks

Linear sampled-data or time-discrete systems can be described and analyzed using a z -transform. The symbolic approach presented earlier can be extended to such systems. Switched-capacitor (SC) networks are a specific case of periodically varying sampled-data systems. These SC networks are composed of capacitors, operational amplifiers, and switches. For simplicity, let us consider two-phase clocking scheme. The generalization to a multiphase case can easily be done.

In the two-phase case, with period $T = 1/f_c$, the SC circuit is observed at times $t_n = nT/2$. Because of varying positions of switches, two different topologies are obtained, one for even and another for odd switching moments. The behavior of such

networks can be described using voltages and charges of capacitors. Using e and o superscripts for even and odd circuit configurations, the following equations in the z domain describe circuit behavior:

$$\begin{aligned} B^e I^e &= 0 & B^o I^o &= 0 & \text{K.Ch.L.} \\ D^e V^e &= 0 & D^o V^o &= 0 & \text{K.V.L.} \\ I^e &= C(V^e - z^{-1}V^o) & I^o &= C(V^o - z^{-1}V^e) \end{aligned} \quad (27)$$

where K.Ch.L. and K.V.L. identify Kirchhoff's charge and voltage laws, I is the element charge vector, V is the element voltage vector, B is the fundamental cut-set matrix, D is the fundamental loop matrix, and C is the diagonal capacitance matrix.

Symbolic analysis of switched-capacitor networks was proposed by Bon and Konczykowska (18). For each SC circuit, an equivalent, time-invariant substitute network can be constructed in the following way:

- Two time-invariant networks N^e and N^o correspond to the SCN in the even and odd time slot, respectively. They are connected by a common reference node. Currents and voltages of the new network correspond to charges and voltages of the initial network (in the even and odd parts, respectively).
- Each switch is replaced by a short-circuit or an open-circuit in N^e and N^o according to the controlling switching pattern.
- Each capacitance C_i is replaced by a conductance C_i in both N^e and N^o .
- In parallel with each conductance C_i of N^e (respectively N^o), a voltage-controlled current source with a controlling coefficient $C_i z^{-1}$ is connected. It is controlled by the voltage across the twin capacitance in N^o (respectively N^e).
- The input and output ports of the initial network are doubled and correspond to ports in each switching phase.

Such a linear, time-invariant network is described by exactly the same set of Eq. (27) as the initial SC network. Any symbolic method of analysis can be applied to such a network. Transfer functions H are the rational functions of z^{-2} , and a multiplicative term z^{-1} is present in even to odd and odd to even transfers. Frequency analysis can be done with $z = e^{sT/2}$.

It is possible to obtain a more general symbolic description taking into account the finite gain of operational amplifiers. The only modification consists in replacing the operational amplifier in both parts of the network by a voltage-controlled voltage source (VCVS) with a symbolic finite gain A .

Another generalization involves considering a clocking scheme as symbolic. In this case, instead of replacing switches by a short- or open-circuits, we replace them with the symbolic conductances. A switch considered as symbolic is replaced by the conductances: S^e in the even and S^o in the odd part. From the well-known bilinear property of the transfer functions of linear time-invariant networks, we deduce that every polynomial involved in the transfer function can be written as

$$P(z) = P_{00}(z) + P_{10}(z)S^o + P_{01}(z)S^e + P_{11}(z)S^o S^e \quad (28)$$

where P_{00} , P_{10} , P_{01} , P_{11} are independent of S^e and S^o .

The function

$$P(z) = P_{10}(z)S + P_{01}(z)(1 - S) \quad (29)$$

is a symbolic expression for the polynomial P covering both even and odd switching schemes for the selected switch. S is a binary variable defined as

$$S = \begin{cases} 0 & \text{for an odd switching scheme} \\ 1 & \text{for an even switching scheme} \end{cases} \quad (30)$$

Functions P_{00} and P_{11} are not used in the Eq. (29) and correspond to trivial switching schemes, when the switch is either open (P_{00}) or short-circuited (P_{11}) in both phases.

Other Types of Symbolic Analyses

Different types of circuit analyses can also be realized in a symbolic way.

Noise analysis is most often calculated using a small signal regime. The noise sources must be included in the circuit-equivalent scheme. Transfer function from a particular noise source to the output can be calculated symbolically as any other transfer function. Total output noise density or total equivalent input noise density need the calculation of the output variable using multiple noise sources. This type of analysis leads often to complex expressions, and application of approximation techniques is necessary.

Knowledge of poles and zeros is often useful in analog circuit design. *Symbolic poles and zeros extraction* techniques were proposed using the pole-splitting hypothesis. A symbolic Newton iteration can be then applied to refine the pole/zero location.

Symbolic analysis of *weakly nonlinear circuits* with multiple or harmonic distortion can also be realized in a symbolic way. These analyses are based on the Volterra series method. The functions describing nonlinear elements are expanded into power series truncated after the first few terms. Higher-order responses are calculated as a correction of the lower-order responses.

Symbolic analysis of *nonlinear circuits* in direct current (dc) domain is presented in Ref. 19.

More details on different types of symbolic analysis and a large biography on this subject can be found in Refs. 11, 16, and 20.

APPLICATIONS OF SYMBOLIC ANALYSIS

There are two main types of applications of symbolic methods. For the first type, the speed-up of calculation is sought when multiple analyses for the same topology are needed. The second category includes a large variety of situations. The main interest lays here in the symbolic nature of produced results and the possibilities of symbolic treatment and reasoning on these results. The new design methodologies exploit these features more efficiently.

Calculation Speed-Up for Repetitive Circuit Evaluation

Classical applications of symbolic analysis (e.g., statistical analysis, design centering, and optimization) are the result of the possibility of the calculation speed-up. The availability of

symbolic formulae allows us to reduce a computation effort where multiple calculations for the same topology are required. After the symbolic expressions are generated for a given circuit structure, the calculation of circuit responses involves evaluating formulae for a new set of parameters. Depending on the necessary flexibility, symbolic functions can be either compiled and linked to the software (for fixed topology), or generated and evaluated in an on-line way.

Insight Into Circuit Behavior

Symbolic characteristic functions are expressed as analytical formulae where device parameters are kept as symbols. Contrary to the numerical analysis, where the circuit behavior is calculated for a particular set of parameter values, the impact of elements on the circuit performance can be observed directly. In order to be exploited by a designer, a compact, legible form of results is necessary. This often can be achieved with approximation techniques where the most significant parameters and their role in the circuit are identified. Symbolic approach may be useful for beginning analog designers helping in evaluation of various topologies and for more experimented ones in their work on new architectures. It can replace the cumbersome manual calculations in the circuit design process.

One of traditional applications of symbolic simulation is the support that is offered to the education activities in the domains of circuit theory and circuit design [see Huelsman (21)]. The understanding of circuit behavior and the role played by its elements is crucial in the education process.

Model Generation and Automated Sizing

Working with building-blocks and using libraries of models can speed up the design process of analog and mixed-signal circuits. Symbolic simulation can be helpful in establishing such libraries. The derivation of analytic macromodels, which when realized manually is a tedious, time-consuming task, can be done automatically. An approximated version can be useful for evaluating the functionality and sizing, an exact formulae can serve for the more accurate, final simulation. These analytical formulae can be further transformed by the equation manipulation program. They can be automatically ordered to optimize the efficiency of evaluation or organized in a sequence of subsets with a minimum number of simultaneous variables, easing the task of solving the design equations. Such an automated analog design using compiled symbolic blocks was presented by Gielen (16). Reference 19 presents behavioral model generation based on approximated nonlinear symbolic analysis.

Device Modeling and Characterization

Device modeling and characterization is a computationally intensive process that benefits greatly from the use of symbolic analysis. Parameter extraction involves adjusting values of model parameters in order to minimize the difference between measured and simulated device characteristics. Two extraction techniques are generally used: direct extraction and simulation-based extraction. In *direct extraction*, model parameters are determined from linear approximation of experimentally measured data. These data are most often acquired in the form of S parameters and transformed to Z or

Y matrices. On the other hand, analytic expressions for Z or Y matrices can be obtained with the symbolic analysis of a small signal scheme. Approximation of these characteristics (real and imaginary part are used) gives necessary formulae for direct extraction. These two operations (generation and approximation of analytic expressions), when done manually, are cumbersome and error prone and must be repeated for each modification of the model topology and even when ranges of parameter values change (approximate formulae must be verified). Symbolic simulators producing an approximated form of results can be of great help for this purpose. In the *simulation-based iterative extraction*, a circuit simulator is used to provide circuit responses. In this method, repeated, time-consuming simulations are necessary for the optimization procedure. For the linear extraction, instead of repeating the analysis for each iteration, it is possible to derive circuit responses in symbolic form. For each iteration, objective and error functions can be calculated using these expressions with different variable values. In this case, the exact value of circuit function is necessary. As rapidity of evaluation is here an essential goal, a compact, nested form, minimizing the number of operations is sought.

Direct extraction is rapid and simple to execute, but some nonlinear parameters are difficult to extract so that precision may be a problem. Iterative methods are very flexible: different devices realized in various technologies can be modeled with the same tool, and it is easy to include all parasitic, access, and packaging elements in the extraction procedure. On the other hand iterative methods are time-consuming, depend on the starting point, and need a robust global optimization methods, which additionally increases the calculation time. To obtain an efficient characterization methodology, most often a combination of both methods is necessary. More details on realizations of extraction tools and methodologies using symbolic tools, can be found in Ref. 22.

Structural Synthesis and Optimization

Structural synthesis and optimization involve determining the most advantageous circuit topology in a generally specified class of structures to achieve a given design goal. When the optimization objective function is specified in an analytical form, symbolic simulation is necessary for objective evaluation. In the case of a classical, numerical objective, a numerical optimization of each examined topology is included in the procedure. Beside the objective function, which expresses technical performance of the circuit, the cost of each structure expressing its complexity must also be evaluated. In the structural optimization process, a combination of an objective/cost goal is used most often. More details on the goal specification and structural synthesis and optimization strategy can be found in Ref. 16.

A specific class of circuit structure optimization is the systematic exploration of a given class of topologies to find the one best suited for a specified application. Konczykowska and Bon (23) presented an exhaustive evaluation of integrators realized using switched-capacitor technique. The goal was to find integrators most suitable for the high-frequency operation. In design terms, it meant to find integrator structures with reduced influence of the finite amplifier gain. The symbolic simulation was first necessary to identify all integrator structures (objective specified in symbolic form) in the class

of circuits composed of one operational amplifier and three capacitors. Then for all integrators found, an error caused by finite amplifier gain was symbolically calculated and compared. This procedure allows us to identify structures with inherently low sensitivity to the finite amplifiers gain.

Other Applications

Different other applications were developed using the symbolic technique for which the symbolic nature of results is essential. Analog circuit design automation with the use of symbolic programs is presented in Refs. 11, 24, and 25. Fino, Franca, and Steiger-Garçao in Ref. 16 developed a system for analysis and synthesis of discrete-time circuits and in particular switched-capacitor circuits. Aftab and Styblinski in Ref. 16 showed how statistical analysis can be enhanced with the symbolic approach. Analog testing and fault diagnosis using symbolic analysis were studied by Manetti and Liberatore in Ref. 16.

An overview of the symbolic approach can be found in Ref. 20. In Ref. 16, an extensive review of symbolic methods and applications is presented.

BIBLIOGRAPHY

1. W. K. Chen, *Applied Graph Theory*, Amsterdam: North-Holland, 1976.
2. A. C. Davies, Matrix analysis of networks containing nullators and norators, *Electron. Lett.*, **21**: 48–49, 1966.
3. S. J. Mason, Feedback theory—Further properties of signal flow graphs, *Proc. IRE*, **3**: 920–926, 1956.
4. C. L. Coates, Flow graph solutions of linear algebraic equations, *IRE Trans. Circuit Theory*, **6**: 170–187, 1959.
5. P. M. Lin, *Symbolic Network Analysis*, Amsterdam: Elsevier Science, 1991.
6. K. Singhal and J. Vlach, Generation of imittance functions in symbolic form for lumped distributed active networks, *IEEE Trans. Circuit Theory*, **21**: 57–67, 1974.
7. J. Starzyk and A. Konczykowska, Flowgraph analysis of large electronic networks, *IEEE Trans. Circuits Syst.*, **33**: 302–315, 1986.
8. M. Hassoun and P. M. Lin, A hierarchical network approach to symbolic analysis of large-scale networks, *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, **42**: 201–211, 1995.
9. J. Starzyk and J. Zou, Direct symbolic analysis of large analog networks, *Proc. 39th Midwest Symp. Circuits and Systems*, Ames, IA, 1996, pp. 421–424.
10. M. Pierzchala and B. Rodanski, Efficient generation of symbolic network functions for large-scale circuits, *Proc. 39th Midwest Symp. Circuits Syst.*, Ames, IA, 1996, pp. 425–428.
11. G. Gielen and W. Sansen, *Symbolic Analysis for Automated Design of Analog Integrated Circuits*, Norwell, MA: Kluwer, 1991.
12. H. N. Gabow, Two algorithms for generating weighted spanning trees in order, *SIAM J. Comput.*, **6**: 139–150, 1977.
13. Q. Yu and C. Sechen, Efficient approximation of symbolic network function using matroid intersection algorithms, *Proc. IS-CAS*, Seattle, WA, 1995, pp. 2088–2091.
14. P. Wambacq et al., Efficient symbolic computation of approximated small-signal characteristics of analog integrated circuits, *IEEE J. Solid-State Circuits*, **30**: 327–330, 1995.
15. Q. Yu and C. Sechen, A unified approach to the approximate symbolic analysis of large analog integrated circuits, *IEEE Trans. Circuits Syst. (I)*, **43**: 656–669, 1996.
16. F. Fernandez et al. (eds.), *Symbolic Analysis Techniques. Applications to Analog Design Automation*, Piscataway, NJ: IEEE Press, 1997.
17. R. Sommer et al., Equation-based symbolic approximation by matrix reduction with quantitative error prediction, *Alta Frequenza-Rivista di Elettronica*, Vol. **5**: 29–37, 1993.
18. M. Bon and A. Konczykowska, A topological analysis program for switched-capacitor networks with symbolic capacitors and switching functions, *Proc. ECCTD*, Warsaw, 1980, pp. 159–164.
19. C. Borchers, R. Sommer, and E. Henning, On the symbolic calculation of nonlinear circuits, *Proc. ISCAS*, Atlanta, GA, 1996, pp. 719–722.
20. G. Gielen, P. Wambacq, and W. Sansen, Symbolic analysis methods and applications for analog circuits: A tutorial overview, *Proc. IEEE*, **82**: 287–304, 1994.
21. L. P. Huelsman, Personal computer symbolic analysis programs for undergraduate engineering courses, *Proc. ISCAS*, Portland, OR, 1989, pp. 798–801.
22. A. Konczykowska, Methods and tools for characterisation, modelling and optimisation of electrical devices, *Proc. World Congress—Manufacturing Technology Towards 2000*, Cairns, Australia, 1997, pp. 41–50.
23. A. Konczykowska and M. Bon, Automated design software for switched-capacitor IC's with symbolic simulator SCYMBAL, *Proc. 25th Design Autom. Conf.*, Anaheim, 1988, pp. 363–368.
24. R. Sommer, R. Kamitz, and E. Horneber, Qualitative reasoning in the analog design expert system EASY, *Proc. ECCTD*, Copenhagen, 1991, pp. 387–394.
25. G. Di Domenico et al., BRAINS—A Symbolic Solver for Electronic Circuits, *Proc. SMACD Workshop*, Bagneux, 1991.

AGNIESZKA KONCZYKOWSKA
National Center for
Telecommunications Studies
(CNET)

SYMBOLIC INSPIRED APPROACHES TO KNOWLEDGE REPRESENTATION. See KNOWLEDGE MAN-
AGEMENT.