Generation of Sequential Symbolic Network Functions for Large-Scale Networks by Circuit Reduction to a Two-Port

Marian Pierzcha»a and **Benedykt Rodanski**, *Member, IEEE*

***Abstract*— The major stumbling block in symbolic analysis of large-scale circuits is the exponential growth of expression complexity with the circuit size. Sequential techniques, introduced more than a decade ago, reduced that growth to quasi-linear. The fundamental assumption in all sequential methods developed so far was that the circuit must be decomposed in order to reduce the complexity of the final expression. In this paper we will show conclusively that this is not the case. We describe a new algebraic approach to symbolic analysis of large-scale networks, based on the reduction of the compacted modified node admittance matrix to a two-port matrix. No circuit partitioning is required. Internal variables are suppressed one by one using Gaussian elimination. To minimise the number of symbolic operations we employ a locally optimal pivoting strategy. Formula complexity is estimated to grow quasi-linearly with circuit size. The technique is conceptually very simple and produces sequential formulae of significantly lesser complexity than any exact method published to date.**

I. INTRODUCTION

Symbolic network functions can be used as an effective tool in the analysis and design of electronic circuits. When large linear circuits are concerned, there exist two different aims of symbolic analysis:

- To get insight into circuit behaviour.
- To generate the shortest sequence of expressions for use in repetitive numerical calculations.

The first aim is realised by generation of symbolic network function in the cancellation-free expanded format as the ratio of two polynomials and/or by producing the symbolic expressions for the poles and zeros. Due to the exponential growth of the number of product terms in the symbolic functions when the circuit size grows, symbolic function must be approximated during or after computation [1, 2]. However, in the case of repetitive numerical calculations, we need an exact symbolic expression with as few arithmetic operations as possible. In this field important developments have been achieved with the introduction of *hierarchical decomposition* [3] and the concept of the *sequence of expressions* [4]. Both concepts were effectively used in [5] to produce expressions of lower complexity.

Until recently, decomposition (of graphs, circuits or matrices) was thought of as the only hope for exact symbolic analysis of large circuits. Most of the circuits, however, do not possess the loose coupling structure and high regularity desired in the hierarchical decomposition approach. In addition, partitioning imposes restrictions on the order in which the circuit variables are eliminated. This usually leads to longer expressions. Moreover, the graph decomposition methods and related formulae are complex and require an advanced knowledge of the graph theory.

Matrix decomposition approach, on the other hand, is conceptually simpler, but requires sophisticated (and time-consuming) discrete optimisation techniques to optimally partition circuit matrices [6, 7]. In some special cases, where the circuit consists of cascaded blocks with no feedback, matrix decomposition could yield results better than other methods.

Recently two similar techniques were proposed simultaneously and independently for direct symbolic analysis of large circuits. Both methods do not require partitioning. One method uses the Coates graph representation and node exploding technique to construct symbolic expressions [8]. The other one, suggested by the authors, is based on symbolic Gaussian elimination with locally optimal pivoting [9]. The method is exceedingly simple and does not require circuit decomposition. Our approach is based on the reduction of the *compacted modified node admittance matrix* (CMNAM) to a 2×2 matrix. It could be viewed as a variation of the method described in [5], with the whole circuit treated as a single terminal block. The major difference is in a pivoting scheme employed to locally minimise the number of symbolic operations. As a result, our technique generates sequences of expressions significantly better (in terms of the execution time) than any exact method published to date. As in other sequential approaches, complexity of the formulae generated by the new method grows quasi-linearly with circuit size [5]. Simple modification to our method allows it to create fractionless sequences, similar to those obtained in [8].

II. SYMBOLIC REDUCTION OF CMNAM TO A TWO-PORT ADMITTANCE MATRIX

Consider a circuit in Fig. 1. We assume that the circuit is lumped, linear and time-invariant, and that it can be described by a compacted modified node admittance matrix, $\boldsymbol{Y}$ (references [10, 11] provide detailed description of the process of formulating the CMNAM). The circuit can be described by the set of symbolic equations:

$$\boldsymbol{Y}\boldsymbol{v} = \left[i_1 \;\; i_2 \;\; 0 \;\; \ldots \;\; 0\right]^T, \tag{1}$$

where matrix $\boldsymbol{Y}$ has symbolic entries of general form: $y_{ij} = \pm G \pm sC$, and the variables v_i may represent node potentials as well as currents in certain circuit elements.

A. Elimination of internal variables

Usually one is only interested in the relationship between the input and output variables (v_1, i_1 and v_2, i_2 in our case) and all other variables can be suppressed. Suppose that we wish to suppress the variable v_p. To achieve this we can use any equation from the set (1), except the first two equations, that has a nonzero coefficient at v_p. Let us choose it to be equation $q > 2$ (the method of selecting p and q is called *pivoting strategy*). The qth equation can be written in the expanded form:

$$y_{q1}v_1 + y_{q2}v_2 + \ldots + y_{qp}v_p + \ldots + y_{qn}v_n = 0. \tag{2}$$

Provided that $|y_{qp}| \neq 0$, we can calculate v_p from (2) as

$$v_p = -\frac{y_{q1}}{y_{qp}}v_1 - \frac{y_{q2}}{y_{qp}}v_2 - \ldots - \frac{y_{qn}}{y_{qp}}v_n. \tag{3}$$

Substituting (3) into (1) will eliminate the variable v_p and equation q from the set. During the elimination, each element y_{ij} of $\boldsymbol{Y}$ undergoes the transformation:

$$y_{ij} \leftarrow y_{ij} - \frac{y_{qj}y_{ip}}{y_{qp}} \; ; \qquad \begin{matrix} i,j = 1,2,\ldots,n \\ i \neq q, j \neq p. \end{matrix} \tag{4}$$

This process of suppression of a variable is the very well known Gaussian elimination. The only difference from the usual appearance of the elimination formula (4) in the literature is the fact that the pivot, y_{qp}, may be off-diagonal. In practice, the transformation (4) is only applied to the matrix elements y_{ij} for which $|y_{qj}||y_{ip}| \neq 0$. When y_{ij} is initially zero, a new nonzero element, a *fill-in*, is created. We will denote the total number of fill-ins at an elimination step by N_{fill}. All internal variables are successively eliminated using identical procedure. At the end of this process we are left with a set of two equations:

$$\begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} I_S - Y_S v_1 \\ -Y_L v_2 \end{bmatrix}, \tag{5}$$

where elements y_{ij} may of course be different from the corresponding elements in the initial equation (1). Any network function can be easily obtained from (5).

B. Alternative form of elimination

Using (4) to suppress internal variables results in very compact sequence of expressions, but containing divisions. In some cases it might be advantageous to obtain the transfer function explicitly in rational form, $N(s)/D(s)$, with no divisions in both numerator and denominator. Instead of using (4) in the elimination process, we can use the alternative expression:

$$y_{ij} \leftarrow y_{ij}y_{qp} - y_{qj}y_{ip} \; ; \quad i,j = 1,2,\ldots,n; \; |y_{ip}| \neq 0; \; i \neq q, j \neq p. \tag{6}$$

Applying (6) repetitively, all internal variables can be suppressed until the system of equations takes the form:

$$\begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} i_1 PP_1 \\ i_2 PP_2 \end{bmatrix}, \tag{7}$$

where PP_1 and PP_2 represent products of pivots for which $|y_{1p}| \neq 0$ and $|y_{2p}| \neq 0$, respectively. Terms y_{ij} in (7) do not contain fractions. Fractionless sequences may be less efficient, due to larger number of arithmetic operations.

C. Pivoting strategy

For practical circuits the CMNAM has a large number of zero elements. The elimination formula (4) is applied only to matrix elements for which both $|y_{qj}|$ and $|y_{ip}|$ are not zero. Our goal is to produce a sequential formula with the lowest number of arithmetic operations. Although, even for relatively small circuits, it is not possible to find the global optimum in polynomial time (the problem can be shown to be NP-complete), a 'reasonable' local optimum can be found with little effort. This can be achieved by careful pivot selection.

If column p and row q have n_p and n_q nonzeroes, respectively, the number of elements to be updated using (4) is equal to $(n_p\text{-}1)(n_q\text{-}1)$. Every update creates one equation in the resulting sequence of expressions. More importantly, each update adds one multiplication and one division and, when it is not a fill-in, one addition/subtraction. In order to minimise the number of symbolic operations, at each elimination step a pivot should be chosen to minimise the following cost function: $C_1 = N_{mult} = n_p \cdot n_q$. (Almost any nonzero element of the CMNAM

can be chosen as a pivot. Only the rows and columns corresponding to the input and output terminals are excluded.) This criterion was first proposed in [12] and is often referred to as the *minimum multiplication criterion*. In fractionless elimination the number of multiplications is equal to $(n_p\text{-}1)(n_q\text{-}1) - N_{fill}$ plus the total number of nonzeroes in the rows for which $|y_{ip}| \neq 0$.

When several pivot candidates give the same minimum value of C_1, additional criteria may be used. For example, we may wish to minimise the number of fill-ins, N_{fill}. Furthermore, the total number of floating point operations, or *flops*, to suppress a variable can be minimised; we denote this number by N_{flop}. (Calculation of flops in operations involving complex numbers and their relationship to computational efficiency are described in detail in [13] .)

Whatever our primary and secondary objectives are, the cost function has the general form:

$$C = \sum_i w_i C_i \ , \tag{8}$$

where C_i represent the subobjectives and w_i are their relative weights. A pivot is chosen from all eligible matrix elements as to minimise (8).

D. Further reduction of formula complexity

Introduction of intermediate expressions: Further reduction of the formula complexity can be achieved by replacing repetitive operations with a new symbol. Consider the application of (4) along a given row *i*. If the number of updates in the row is greater than one, the ratio y_{ip}/y_{qp} can be replaced by a new symbol, d_i, and the elimination sequence generated by a formula:

$$d_i = \frac{y_{ip}}{y_{qp}}, \quad y_{ij} = y_{ij} - y_{qj} d_i; \qquad \begin{matrix} i,j = 1,2,\ldots,n \\ i \neq q, j \neq p. \end{matrix} \tag{9}$$

The sequence obtained from (9) will have less arithmetic operations than the one generated by (4).

Special case of $i_2 = 0$: Generally, pivots should not be selected from the rows and columns associated with input and output. This restriction can be relaxed, however, if the output current is of no interest, namely for calculation of voltage ratio, v_2/v_1, voltage gain, v_2/V_S, input admittance, i_1/v_1, and transimpedance, v_2/i_1. In those cases we can incorporate the load admittance into the CMNAM: $y_{22} = y_{22} + Y_L$ and set $i_2 = 0$. Pivots can now be selected also from the second equation. This increases the number of degrees of freedom in pivot selection thus improving the chances of reducing the complexity of the final formula.

Heuristic row/column operations: As suggested in [14], selected pairs of rows and/or columns can be added/subtracted to reduce the number of symbols in the matrix thus leading to simpler expressions by avoiding possible term cancellations. Care must be taken, however, not to affect the elements of restricted columns and rows.

III. EXPERIMENTAL RESULTS

A. Software implementation

Due to its excellent user interface, ability to display and handle large symbolic arrays and ease of programming (in VBA and/or C/C++), we have chosen Microsoft Excel® as a software platform to test our algorithms. Circuit data is generated automatically from the output file of Orcad PSpice or is entered onto an input spreadsheet in the familiar

Spice-like format. The user then selects the pivoting criteria, the way the output is to be presented and the required network function. We have found that the three pivoting criteria, mentioned earlier, are quite sufficient. The cost function takes the form:

$$C \;=\; w_1 N_{mult} \;+\; w_2 N_{fill} \;+\; w_3 N_{flop}\,, \tag{10}$$

where the weights, w_i, depend on which criterion was chosen as primary, secondary and tertiary. The requested network function can be generated in three different formats: (i) compact sequence of expressions (with divisions), (ii) fractionless sequence of expressions and (iii) ratio of two factorised fractionless expressions. The CMNAM and the calculated expressions are displayed on another spreadsheet. Three text files may also be created. They are: component values (if any), CMNAM elements and the sequence of expressions. These files can be used for further processing (e.g., by MATLAB®, Maple®, Mathematica®, etc.). Our implementation, which is essentially an Excel workbook, is called STAINS - Symbolic Two-port Analysis via Internal Node Suppression. A copy of the software can be obtained via Internet from the authors (e-mail: <benr@eng.uts.edu.au>).

B. Circuit example

Consider the very well known band-pass filter, first analysed symbolically in [3]. Its circuit diagram is presented in Fig. 2a. Fig. 2b shows the compact sequence of expressions generated by STAINS to calculate its voltage ratio. Table I gives efficiency comparison of both STAINS sequences, compact (i) and fractionless (ii), with other published results. All sequences were coded in C++ and compiled with MS Visual C++® 5.0 with either no optimisation or optimised for maximum speed. The timing experiments were conducted on a PC with a Pentium ® II microprocessor, running at 300 MHz (detailed information about the timing experiments can be found in [13]). Table I shows that the STAINS-generated sequences are significantly better than other results published to date for the band-pass filter (7-92% for non-optimised code and 47-470% for speed-optimised code). Fractionless elimination produced the most efficient sequence for this circuit. Unlike MASSAP- and SCAPP-generated sequences, it can be evaluated at $s = 0$ without overflows. Analysis of other circuits confirms the speed-ups achieved with our approach.

IV. CONCLUSION

Since the sequential methods for exact symbolic analysis were introduced over a decade ago, it was always believed that decomposition (of graphs, circuits or matrices) is the only way to reduce the formula complexity. In this paper we present arguments to the contrary. Partitioning, in fact, imposes restrictions on the order in which internal variables are eliminated thus lowering the chances of optimal pivot selection which is crucial in minimising complexity of the expression. Our experiments suggest that unless a circuit contains several identical blocks (in both topology and component symbols) or unless it consists of cascaded subcircuits with no feedback, partitioning could not yield more compact results than elimination with locally optimal pivot selection performed on the entire set of circuit equations.

A new method of generation of symbolic network functions was proposed on the above premise. Our approach is suitable for analysis of large-scale circuits. It is purely algebraic, has the advantages of hierarchical and sequential techniques and requires only simple manipulations of the compacted modified node admittance matrix. No circuit partitioning is performed. In order to minimise the number of symbolic operations in the final sequence of

expressions, a locally optimal pivoting strategy is employed. In other methods, graph/circuit decomposition and combining the results of terminal block analysis into the final formula (middle block analysis) requires substantial effort. By avoiding these steps, our approach is both conceptually and computationally simpler. Most importantly, our compact technique generates sequences of expressions significantly better (in terms of the execution time) than any exact method published to date.

REFERENCES

[1] G. Gielen, W. Sansen, *Symbolic Analysis for Automated Design of Analog Integrated Circuits*. Boston: Kluwer, 1991.

[2] A. Konczykowska, "Symbolic circuit analysis," in: *Wiley Encyclopedia of Electrical and Electronics Engineering*, edited by J.G. Webster. New York: Wiley, 1999.

[3] J.A. Starzyk, A. Konczykowska, "Flowgraph analysis of large electronic networks," *IEEE Trans. on Circuits and Systems*, vol. 33, no. 3, pp. 302-315, March 1986.

[4] M.M. Hassoun, K.S. McCarville, "Symbolic analysis of large-scale networks using a hierarchical signal flow-graph approach," *Analog Integrated Circuits and Signal Processing* 3, pp. 31-42. Boston: Kluwer, 1993.

[5] M.M. Hassoun, P.-M. Lin, "A hierarchical network approach to symbolic analysis of large-scale networks," *IEEE Trans. on Circuits and Systems*, vol. 42, no. 4, pp. 201-211, April 1995.

[6] M. Pierzcha»a, B. Rodanski, "Obtaining symbolic network functions for large circuits - an algebraic approach," *Proc. IEEE ISCAS*, pp. 2075-2078, 1995.

[7] M. Pierzcha»a, B. Rodanski, "Obtaining symbolic network functions of large circuits by block decomposition of the node admittance matrix," *Proc. ECCTD*, pp. 71-74, 1995.

[8] J.A. Starzyk, J. Zou, "Direct symbolic analysis of large analog networks," *Proc. MWSCAS'96*, pp. 421-424, 1997.

[9] M. Pierzcha»a, B. Rodanski, "Efficient generation of symbolic network functions for large-scale circuits," *MWSCAS'96*, pp. 425-428, 1997.

[10] P.-M. Lin, *Symbolic Network Analysis*. Amsterdam: Elsevier, 1991.

[11] J. Vlach, K. Singhal, *Computer Methods for Circuit Analysis and Design* (2-nd ed.). New York: Van Nostrand Reinhold, 1994.

[12] H. Markowitz, "The elimination form of the inverse and application to linear programming," *Management Sci.*, vol. 3, pp. 255-269, 1957.

[13] B. Rodanski, "Computational efficiency of symbolic sequential formulae," *Proc. SMACD 2000*, pp. 54-50, Lisbon, Oct. 2000.

[14] J.-J. Hsu, C. Sechen, "DC small signal symbolic analysis of large analog integrated circuits," *IEEE Trans. on Circuits and Systems - I: Fundamental Theory and Applications*, vol. 41, no. 12, pp. 817-828, Dec. 1994.

Figure 1

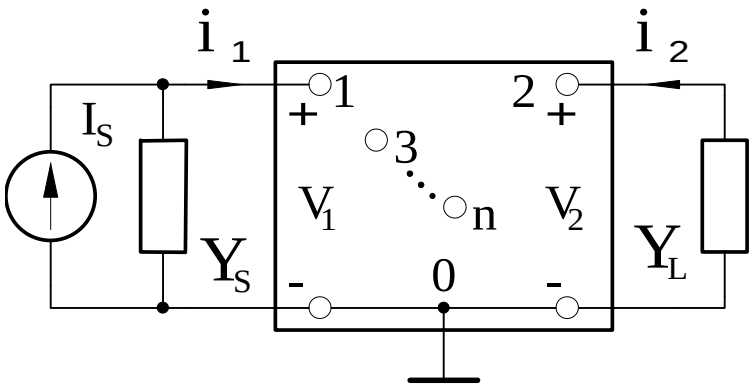
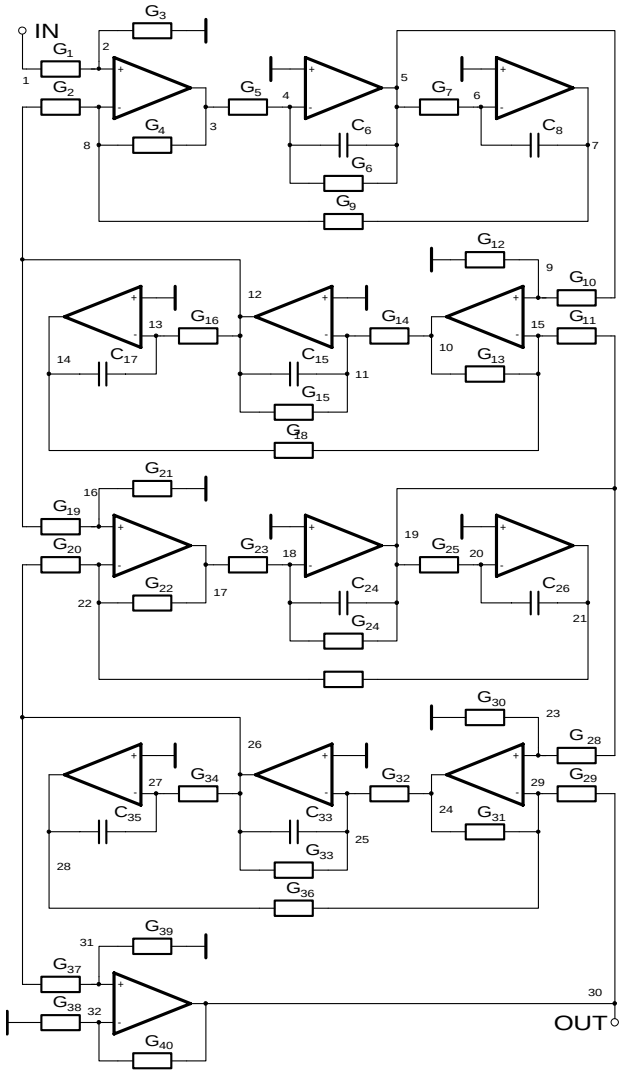


Figure 2



a)

```
% This MATLAB file was generated
  by STAINS 2.3a
% On 14/11/2000 at 12:47:56

% Band-pass filter from [3]

x1 = G37*(G38+G40)/(G37+G39);
x2 = G10*(G11+G13+G18)/(G10+G12);
x3 = G19*(G20+G22+G27)/(G19+G21);
x4 = G28*(G29+G31+G36)/(G28+G30);
x5 = G7*G9/(s*C8);
x6 = x5+(G6+s*C6)*G4/(G5);
x7 = G16*G18/(s*C17);
x8 = x7+(G15+s*C15)*G13/(G14);
x9 = G25*G27/(s*C26);
x10 = x9+(G24+s*C24)*G22/(G23);
x11 = G34*G36/(s*C35);
x12 = x11+(G33+s*C33)*G31/(G32);
d1 = -G1/(G1+G3);
x13 = G1+G1*d1;
x14 = -(G2+G4+G9)*d1;
d2 = -G40/(x1);
x15 = G20*d2;
x16 = -G29-x12*d2;
d3 = x2/(x6);
x17 = -x14*d3;
x18 = x8+G2*d3;
d4 = x3/(x18);
x19 = -x17*d4;
x20 = x10+G11*d4;
x21 = x15-x16*x20/(x4);

% O/C Voltage Ratio:

Tv = -x19/(x21);
```

b)

TABLE I
BAND-PASS FILTER ANALYSIS RESULTS

Programme	**Eqns**	**Compiled code timing [μs]**	
		NoOpt	MaxSpeed
STAINS (i)	26	8.3	2.8
STAINS (ii)	59	8.8	1.7
FLOWUP [3]	25	8.9	2.5
SCAPP [5]	56	9.6	3.6
MASSAP [4]	60	16.0	9.8

Figure Captions

Fig. 1. Terminated two-port configuration for calculating network functions.

Fig. 2. Band-pass filter [3] (a) and the compact sequence of expressions generated by STAINS for its voltage ratio (b).

Footnotes

Manuscript received

M. Pierzcha»a is with Hilmar Ltd., Wroc»aw, Poland.

B. Rodanski is with the Faculty of Engineering, University of Technology, Sydney (UTS), Sydney, NSW, Australia.