# Comparison of Matroid Intersection Algorithms for Large Circuit Analysis*

Mariano Galán, Francisco V. Fernández and Angel Rodríguez-Vázquez
Dept. of Analog and Mixed-Signal Integrated Circuit Design, IMSE-CNM
Edif. CICA, Avda. Reina Mercedes s/n, E-41012 Sevilla, SPAIN
Tel.: +34 5 4239923, FAX: +34 5 4231832, E-mail: pacov@cnm.us.es

## Abstract

This paper presents two approaches to symbolic analysis of large analog integrated circuits via simplification during the generation of the symbolic expressions. Both techniques are examined from the point of view of matroid theory. Finally, a new approach which combines the positive features of both approaches is introduced.

## 1. Introduction

Symbolic analyzers are CAD tools which calculate network functions of analog circuits, with the complex frequency $s$ and the circuit parameters kept as symbols. Those network functions are typically given as a cancellation-free sum of products:

$$H(s,\boldsymbol{x}) = \frac{f_0(\boldsymbol{x}) + sf_1(\boldsymbol{x}) + s^2 f_2(\boldsymbol{x}) + \ldots + s^N f_N(\boldsymbol{x})}{g_0(\boldsymbol{x}) + sg_1(\boldsymbol{x}) + s^2 g_2(\boldsymbol{x}) + \ldots + s^M g_M(\boldsymbol{x})} \quad (1)$$

in which $\boldsymbol{x}^T=\{x_1, x_2, \ldots, x_Q\}$ is the vector of symbolic parameters, and $f_i$ and $g_i$ are sums of products. Many potential applications have been reported for symbolic analysers, most of which can be found in [1].

One of the major drawbacks that has prevented symbolic analyzers from being widely accepted by the analog design community has been that the size of the circuits that they can analyze is still much smaller than for numerical simulators. This is due to the exponential growth of the symbolic formula complexity with the circuit size. This not only limits the maximum analysable circuit size but also makes more difficult formula interpretation and its use in design automation applications, where expressions should be the smallest possible for increased evaluation efficiency.

Experience with the application of symbolic analyzers to real analog integrated circuits shows that, usually, a very small part of the symbolic expression contains most relevant information. So the approximation of symbolic expressions, defined as the reduction of formula complexity while maintaining the accuracy as high as possible, has become a must. The general topic of expression simplification will be described in Section 2.

* This work has been performed in the framework of the AMADEUS Project of the ESPRIT IV Program of the CEC.

However, formula approximation after the complete exact expression has been generated only eases interpretation and manipulation but circuit size limitations remain. A major breakthrough has been produced with the introduction of simplification during generation algorithms [2],[3],[4]. Section 3 will explain how the approximation during generation problem can be understood in terms of matroids, a general mathematical theory that guarantees that if the matroid problem has a solution, the particular problem at hand can also be solved, and at least as efficiently as the matroid problem.

## 2. Symbolic expression approximation

Conventionally, reported symbolic analyzers have incorporated the approximation feature by first calculating the exact network function and then simplifying it as a postprocessing step [5],[6],[7]. For this reason we will call it *Simplification After Generation* (SAG). It is based on the elimination of the least significant terms or subexpressions while some error criterion is satisfied. Relative significance is evaluated based on numerical estimates of the symbolic circuit parameters. Assume

$$h_k(\boldsymbol{x}) = h_{k1}(\boldsymbol{x}) + h_{k2}(\boldsymbol{x}) + \ldots + h_{kT}(\boldsymbol{x}) \equiv \sum_{l=1}^{T} h_{kl}(\boldsymbol{x}) \quad (2)$$

represents either $f_k(\boldsymbol{x})$ or $g_k(\boldsymbol{x})$ in (1). The $P$ least significant terms in (2) are eliminated, one by one and beginning with the smallest, while the sum of the eliminated terms keeps below some given threshold,

$$\sum_{l=1}^{P} |h_{kl}(\boldsymbol{x}_o)| < \varepsilon_k \sum_{l=1}^{T} |h_{kl}(\boldsymbol{x}_o)| \quad (3)$$

where $\boldsymbol{x}_o$ represents the design point of the circuit parameters and $\varepsilon_k$ is a discrimination threshold.

The principal drawback of SAG techniques is that simplification is performed after the exact network function has been generated. So the exact expression, which usually is not used any more after the approximation, limits the analysable circuit size to around 10 transistors.

In order to extend the capabilities of conventional symbolic analyzers, the new concept of *Simplification During Generation* (SDG) has recently been proposed. SDG is based on the following idea: terms are generated in strictly decreasing order of magnitude, starting with

0-7803-3583-X/97 $10.00 ©1997 IEEE

the largest, for each power of $s$, until they represent a given fraction of the total magnitude of the coefficient.

SDG has two main advantages. First, the analysis is faster, as no time is wasted generating terms that would be neglected later on. Second, its smaller memory needs make possible that much larger circuits can be analyzed. This is clearly illustrated in Fig. 1(b), which shows the computation time of approximated expressions with $\varepsilon_k$ = 0.25 versus the number of circuit nodes in Fig. 1(a). It can be seen that the new SDG approach is orders of magnitude faster than the conventional one to get the same approximated expression. Moreover, the maximum analysable circuit size increases considerably.

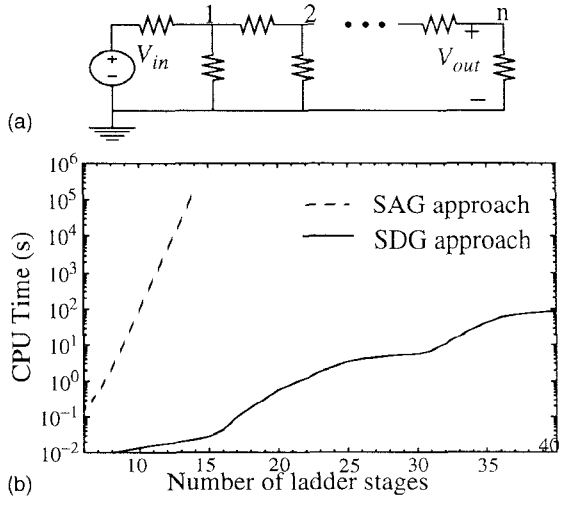


Figure 1: (a) Ladder network; (b) CPU time comparison for the calculation of its voltage gain.

### 2.1. Term generation with the two-graph method

Term generation by the two-graph tree enumeration method has been shown to be the most efficient technique to implement the SDG idea. The two-graph method makes use of a voltage graph $G_V$ and a current graph $G_I$, both easily built from the original network [8]. Fig. 2 illustrates the construction of both graphs for a simple circuit. Each valid term corresponds to the admittance product of every branch in a spanning tree common to both graphs.

Therefore, the term generation problem for each power of $s$ (say the $k$-th power) reduces to the following graph problem: enumerate all spanning trees common to both, the voltage and the current graph, in decreasing order of weight and containing $k$ capacitance branches.

Reported techniques [2],[3],[4] started from the algorithm in [9] for the enumeration of spanning trees of a one-colored graph in decreasing order of weight. As the valid terms are given by common spanning trees to voltage and current graphs, this algorithm is applied to the voltage graph and for each generated spanning tree it is checked if it is also a spanning tree in the current graph. But, each spanning tree must contain exactly $k$ capacitors. Therefore, the original algorithm for one-colored graphs had to be extended to two-colored graphs (containing two branch types: capacitors and conductances).

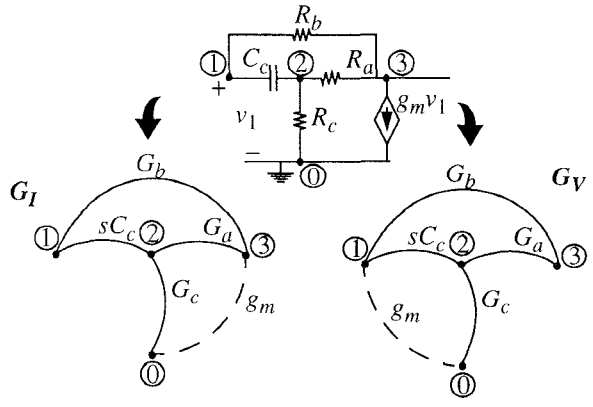


Figure 2: Example circuit and its two-graph representation.

## 3. The enumeration problem in terms of matroids

Matroids are mathematical abstractions which offer a model for many mathematical structures and combinatorial problems. If our problem can be expressed in terms of matroids, then, generally, better algorithms improving the running time (exploiting special structures of those matroids) may be found. Some general concepts about matroids are defined next.

### 3.1. Matroid preliminaries

A *matroid* $M=(E,\mathcal{J})$ is a structure in which $E$ is a finite set of *elements* and $\mathcal{J}$ is a family of subsets of $E$, which satisfy some axioms [10]. A subset $I$ in $\mathcal{J}$ is an *independent set* of the matroid $M=(E,\mathcal{J})$. A maximal independent set is a *base* of the matroid. A matroid $M=(E,\mathcal{J})$ is said to be the *graphic matroid* of the graph G if $E$ is the set of arcs of G and a subset $I \subseteq E$ is in $\mathcal{J}$ if and only if $I$ is a cycle-free subset of arcs. In a connected graph, a base of its graphic matroid corresponds to a spanning tree.

Finally, let $\pi$ be a partition that separates $E$ into $m$ disjoint blocks $B_1, \ldots, B_m$, and let $d_i$, $(i=1, \ldots, m)$ be $m$ non-negative integers. Then, $M=(E,\mathcal{J})$ is a *partition matroid* if $\mathcal{J}$ is the family of subsets $I$ that satisfy $|I \cap B_i| \leq d_i$, $i = 1, \ldots, m$ }.

### 3.2. Application of matroid theory to our problem

The tree enumeration problem can be understood as a 3-matroid intersection problem. These three matroids are:

- A graphic matroid defined on the voltage graph $G_V$, so that each spanning tree in $G_V$ is a base.
- Another graphic matroid defined analogously on $G_I$.
- A partition matroid defined so that each set of branches with $k$ capacitors is a base of the matroid.

The tree enumeration problem corresponds to the enumeration of bases common to the three matroids in decreasing order of weight (grid area in Fig. 3). Unfortunately, the intersection of three matroids is considered to be, in general, a NP-hard problem [11]. However, there are polynomial time algorithms for the weighted intersection of two matroids [12],[11]. Reported symbolic simulators that incorporate SDG have solved the problem by finding first the intersection of two matroids and checking which elements of that intersection are also a base in the third matroid. Two different possibilities arise:

- First intersectioning a graphic matroid and the partition matroid (horizontally-striped area in Fig. 3).
- First intersectioning the two graphic matroids (vertically-striped area in Fig. 3).

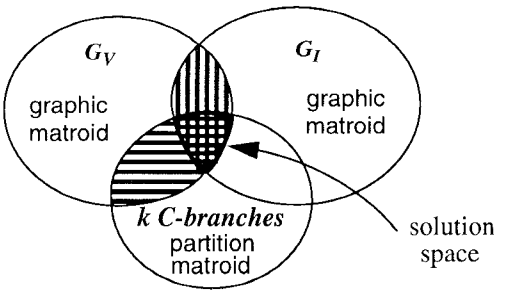


Figure 3: Illustrating the 3-matroid intersection problem.

The first one has been implemented in ADAGIO [2],[3] and RAINIER [4] as it has been described in Section 2.1. This method has the disadvantage that many spanning trees of the voltage graph may be generated that are not spanning trees in the current graph and, hence, do not lead to valid terms. The ratio of generated valid terms over the number of spanning trees in the voltage graph tend to decrease when the circuit size increases, imposing a limit to the maximum analysable circuit size.

It is not easy to compare the two implementations of this approach: that in [2],[3], on the one hand, and that in [4], on the other. This is mainly due to the different error criteria applied, and the fact that the tool in [4] performs a simplification before generation procedure which significantly reduces the circuit size.

For illustration's sake, the tool in [3] provides a simplified expression of the voltage gain of the folded-cascode opamp in Fig. 4 in 54.7 s. The generation of a symbolic expression for the μA741's transfer function at low frequencies with a magnitude error of 0.1% (110 symbolic terms) requires 38 s. An approximated expression (57 terms) for the μA741 is provided in [4] in 19 s.

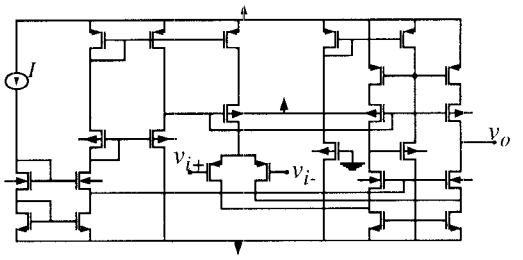


Figure 4: : Folded-cascode opamp.

The second approach, that is, intersectioning the two graphic matroids has been implemented in [13]. There, common spanning trees of the voltage and current graphs, in decreasing order of tree admittance product, are directly generated using the algorithm in [12]. So, no time is wasted in generating spanning trees of one of these graphs which afterwards are not spanning trees in the other graph. But no control can be performed on the branch type (conductance or capacitor), and, hence, capacitor admittances (the branch weight) must be evaluated at a fixed frequency: $\omega C$.

This approach approximates the network function over a frequency range and considers a set of sample frequencies within this range. The algorithm in [12] for generation of common spanning trees (without any constraint in the number of capacitance branches) is applied at each sample frequency until some given error criterion is met. Since there is no constraint in the number of capacitance branches, terms are generated with different powers of $s$. A generated term is kept in the approximated expression if it appears for at least one of the sample frequencies in the frequency range.

But this method has two important drawbacks: on the one hand, most terms are generated at more than one sample frequency, and this means a loss of efficiency. On the other hand, large errors can occur at frequencies different from those of the set of sample frequencies as no control is performed on the expression accuracy between two sample frequencies. Improving the accuracy requires taking more sample frequencies but this is done at the expense of additional computation time, greatly deteriorating the efficiency of the method.

The same expression than [4] for the μA741 has been reported to be obtained in 20 s, practically the same than with the previous technique [13]. However, a more accurate expression (1185 terms) is obtained in 57.7 s, less than half the time needed with previous technique. For this example 4 sample frequencies were used.

### 3.3. A 3-matroid intersection algorithm

The first of the reviewed approaches has in its efficiency for large circuits its main drawback. This is partially overcome in the second approach but at the cost of uncontrolled accuracy of the resulting expressions. The new algorithm introduced in this section combines the positive features of both approaches by directly addressing the intersection of our three particular matroids.

The algorithm in [12] allows the generation of spanning trees common to the voltage and current graphs in decreasing order of weight: from each common spanning tree (the maximum weight common spanning tree is not difficult to find), the next one is obtained by carrying out the branch exchanges indicated by the best *primitive border path* (PBP) of the *border graph* (BG), built from the spanning tree and both the voltage and current graphs. This best primitive border path is more efficiently found in the *condensed border graph* (CBG), which is easily derived from the border graph [12]. This algorithm makes use of the lexicographic Floyd-Warshall algorithm to find the best PBP in either the BG or the CBG. But this way, in both the construction of the CBG and in the Floyd-Warshall algorithm, we have no control on the type of the branches as it only looks at their weights, so it is undetermined the final number of branches of each type (capacitance or conductance), and thus it is not adequate for our needs.

In order to solve the problem of the control on the branch types, we perform extensive modifications in the construction of the CBG, leading to the *Extended Condensed Border Graph* (ECBG). In the ECBG, each branch can have up to two weights, instead of one weight as it occurs in the conventional CBG. These two weights correspond to the capacitance and conductance nodes of the BG that satisfy the condition needed to have a corresponding branch in the CBG, with the highest weight of all the capacitance/conductance nodes of the BG. It reflects the fact that a conductance/capacitance branch can be exchanged by, either a conductance or a capacitance branch. Each branch weight has an associated branch type, which corresponds to the number of induced capacitance branch exchanges.

To find the best PBP in this ECBG an algorithm, called *Multilevel Shortest Paths* (MSP), has been developed. The MSP algorithm finds all shortest paths between each pair of nodes in the ECBG. These paths correspond to every possible net number of capacitance branch exchanges with respect to the common spanning tree from which the BG was built. This is done considering the combinations of the different branches between nodes, and taking into account also their types. Thus, calculation of the following common spanning tree in the same coefficient in (1) reduces to taking the best PBP with zero net capacitance branch exchanges.

## Conclusions

Two algorithms of approximation during generation for the symbolic analysis of large analog integrated circuits have been reviewed. The first one has in its accuracy its stronger feature, decreasing its efficiency with the circuit size. The second gives priority to the efficiency but with a high risk of large inaccuracies. This has motivated the introduction of a new approach which overcomes previous drawbacks and constitutes the first polynomial time algorithm for the three matroid intersection problem.

## References


[1] A. Rodríguez-Vázquez, F.V. Fernández, J.L. Huertas and G. Gielen, *Symbolic Analysis Techniques and Applications to Analog Design Automation*, IEEE Press, 1996.

[2] F.V. Fernández, P. Wambacq, G. Gielen, A. Rodríguez-Vázquez and W. Sansen: "Symbolic Analysis of Large Analog Integrated Circuits by Approximation During Expression Generation," *Proc. IEEE Int. Symp. on Circuits and Systems*, pp.25-28, 1994.

[3] P. Wambacq, F.V. Fernández, G. Gielen, W. Sansen and A. Rodríguez-Vázquez, "Efficient Symbolic Computation of Approximated Small-Signal Characteristics of Analog Integrated Circuits," *IEEE JSSC*, Vol. 30, No. 3, pp. 327-330, March 1995.

[4] Q. Yu and C. Sechen, "Approximate Symbolic Analysis of Large Analog Integrated Circuits," *Proc. IEEE Int. Conf. on Computer-Aided Design*, pp. 664-671, 1994.

[5] F.V. Fernández, A. Rodríguez-Vázquez and J.L. Huertas, "Interactive AC Modeling and Characterization of Analog Circuits via Symbolic Analysis," *Analog Integrated Circuit and Signal Processing*, Vol. 1, pp. 183-208, Kluwer, Nov. 1991.

[6] G. Gielen, H. Walscharts and W. Sansen, "ISAAC: A Symbolic Simulator for Analog Integrated Circuits," *IEEE Journal of Solid State Circuits*, Vol. 24, pp. 1587-1597, Dec. 1989.

[7] G. M. Wierzba et al., "Sspice - A Symbolic SPICE Program for Linear Active Circuits," *Proc. 32nd Midwest Symp. on Circuits and Systems*, pp. 1197-1201, 1989.

[8] P.M. Lin, *Symbolic Network Analysis*. Elsevier, 1991.

[9] H.N. Gabow, "Two Algorithms for Generating Weighted Spanning Trees in Order", *SIAM J. of Computing*, Vol. 6, No. 1, pp. 139-150, March 1977.

[10] E.L. Lawler, *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston, 1976.

[11] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Englewood Cliffs, New Jersey. Prentice-Hall Inc., 1982.

[12] P.M. Camerini and H.W. Hamacher, "Intersection of Two Matroids: (Condensed) Border Graph and Ranking", *SIAM J. Disc. Math.*, Vol. 2, pp. 16-27, Feb. 1989.

[13] Q. Yu and C. Sechen, "Efficient Approximation of Symbolic Network Functions Using Matroid Intersection Algorithms," *Proc. IEEE ISCAS*, pp. 2088-2091, 1995.