SYMBOLIC ANALYSIS OF LARGE LLS NETWORKS BY MEANS OF UPWARD HIERARCHICAL ANALYSIS

G. Centkowski, J. Starzyk, E.Sliwa
Institute of Electronics Fundamentals
Technical University of Warsaw
Poland

Abstract

A new approach to the topological analysis of large LLS networks is presented in the paper.

A network graph is hierarchically partitioned untill sufficienty small blocks are obtained. Each of these blocks is analysed separately and then analysis of the structure of their interconnections follows. An algorithm of so-called upward analysis, which is developped from downward hierarchical analysis algorithm[1][2], is described. The algorithm allows to symbolically analyse networks with 100-200 nodes and shows a linear dependence of computer time on network size. An efficient algorithm of automatic graph partitioning, which fits the analysis algorithm described, is also proposed. Tests of a computer program based of these algorithms are presented.

1. INTRODUCTION

A new approach to the symbolic analysis of large LLS networks is presented in the paper. Topological methods /both signal flow graph /SFG/ and tree enumeration techniques /which are used for the symbolic analysis of electronic networks, although very convenient, were considered as ineffective in the computer analysis because of rapid increase of analysis time with the growth of the network size. To overcome this difficulty, a direct decomposition method was introduced, but this had not solved the problem. A turning-point has been achieved when a method of analysis by hierarchical decomposition was introduced [1][2]. However the method, called downward hierarchical analysis, has two drawbacks: (i) some parts of the network have to be analysed more than once, (ii) it is difficult to keep more than a few elements in symbolic form through the whole process of analysis.

The method presented in this paper, called upward hierarchical analysis allows to overcome these drawbacks and permits practically fully symbolic analysis of networks having 100-200 nodes.

2. MODELLING

For the purpose of the algorithms, we assume that the model of the network analysed is unistor graph. Such a graph can be obtained directly from indefinite admittance matrix /IAM/ of the network or by connecting unistor models of network elements [3] or, if the network elements have no admittance description, e.g., ideal opamps, by connecting unistor and autonomic formal unistor models /FUM-s/ [2]. Autonomic FUM-s can be derived from the element flow graph by replacing each edge of the flow graph by corresponding set of unistors. Although such elements are convenient in practice because they do not influence the remaining part of the network, they have two disadvantages (i) they introduce additional nodes and edges to the network graph and (ii) they cause that cancelled terms can appear in final formula. To avoid these, so-called nonautonomic FUM-s of elements with no admittance description can be applied. Such models are obtained from modified element SFG-s /obtained form SFG by changing the incidence of some edges from the element neighbourhood [4] /in a similar way as for autonomic FUM-s.
Fig. 1 shows an example of autonomic and nonautonomic FUM-s of the inverter with the opamp.

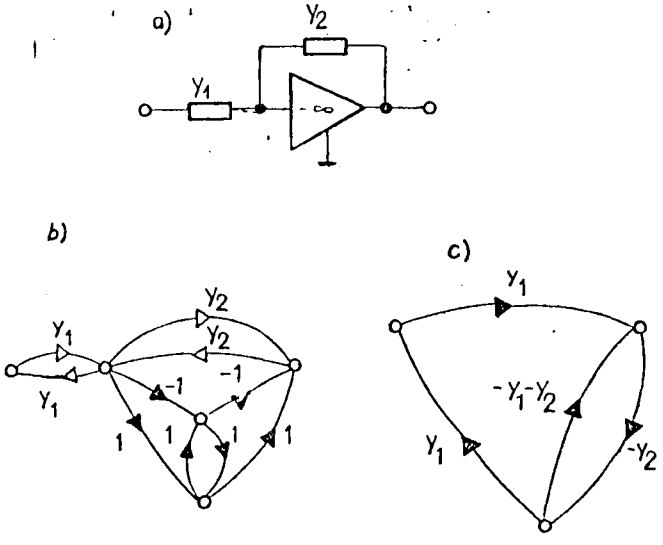

Fig. 1. Inverter /a/ and its autonomic /b/ and nonautonomic /c/ formal unistor models.

3. UPWARD ANALYSIS OF DECOMPOSED GRAPH

Let us consider directed network graph $G = (V, E)$. Let B be the set of block vertices of G /e.g. the set of terminals of a four-pole/. Let us assume that G has been partitioned to subgraphs $G_i = (V_i, E_i)$ /only vertex decomposition is considered here, i.e. $E_i \cap E_j = \emptyset$ for $i \neq j$ and $\bigcup_i E_i = E$; generalization for edge and mixed decompositions follows immediately/. By block vertices B_i of G_i we denote block vertices of G and cut vertices, which belong to V_i. By substitute graph $G_i^S = (B_i, E_i^S)$ of graph G_i we denote complete directed graph spanned over vertices from B_i, and by substitute graph of decomposition G^S we call union of all G_i^S.

By T_B we denote set of multitrees of G spanned over B, i.e. such multitrees t_V, that

$$\left\{ r_1, v_{11}, \ldots, v_{1m_1}, \ldots, r_k, v_{k1}, \ldots, v_{km_k} \right\} = B \qquad /1/$$

/in t_V,

$$V = \left\{ (r_1, v_{11}, \ldots, v_{1m_1}), \ldots, (r_k, v_{k1}, \ldots, v_{km_k}) \right\}$$

is the set of components, k is the number of components, and r_i, i=1, 2 ... k are reference nodes of components/.
The formula

$$T_B = \bigtimes_{t \in P_B} \prod_{G_i} T^i_{V^t} \qquad /2/$$

follows immediately from Theorem 3 in [1]. P_B is the set of proper multitrees of G^S spanned over B /proper multitree t is such a multitree, that any subgraph $E_i^S \cap t$ does not contain directed paths longer than 1/. T^i_V is the set of k-trees of G_i described by the set of components V; V^t may be obtained for given t and G_i^S as was described in [1]. It is obvious, that $T^i_{V^t} \subset T^i_{B_i}$, where $T^i_{B_i}$ is the set of multitrees of G_i spanned over B_i. In conclusion, /2/ allows us to determine all multitrees of a graph spanned over their sets of block vertices based on all multitrees of the subgraphs and the structure of interconnection of these subgraphs. In addition, T_B is a full description of G considered as n-pole, where $n = \bar{B}$. Of course, if G is a subgraph of the network graph, /2/ may be used to determine description of a graph, whose decomposition resulted in G.

This procedure is opposite to the downward analysis, where, on the basis of k-trees of the substitute graph, sets of multitrees of subgraphs are determined. Thus upward analysis avoids multiple enumeration of the same sets of k-trees. Enumeration of k-trees of all types can be proceeded with the use of a more efficient algorithm than that of generation of normal k-trees.

The algorithm of upward analysis is as follows

Step 0. Decompose hierarchically the network graph /see next paragraph/.

Step 1. Enumerate sets of multitrees spanned over block vertices for proper blocks /i.e. parts of the network graph, which were not further partitioned/

Step 2. Find substitute graph of decomposition such that sets of multitrees $T^i_{B_i}$ spanned over B_i's for all its subgraphs were previously determined. For this graph, determine its set of multitrees T_B with the aid of /2/. Repeat this step untill the set of multitrees T_{B_o} of the network graph spanned over the set of network terminal vertices is obtained.

Network functions can be easily obtained: they can be expressed as ratios of weights of appropriate subsets of T_{B_o} [3].

If we assume more or less uniform partition of the network graph, time of analysis $\bar{\tau}$ can be estimated /for the case of hierarchical bisection/ by

$$\bar{\tau} = n \cdot \bar{\tau}_p + (n-1) \cdot \bar{\tau}_s \cong n \cdot (\bar{\tau}_p + \bar{\tau}_s) \qquad /3/$$

where $\bar{\tau}_p$ and $\bar{\tau}_s$ are average values of time needed for multitree enumeration in proper blocks and substitute graphs, respectively. n is the number of proper blocks. So, /3/ shows a linear dependence of analysis time on the network size. Remember that the corresponding dependence for downward analysis is $\text{const} \cdot N^{\alpha}$, where N is the number of network nodes and $\alpha \cong 2 \div 3$.

4. DECOMPOSITION

Before upward analysis is performed, as step 0 of the algorithm, hierarchical decomposition of the network graph must be made. So far, no efficient algorithms for quasi-optimal graph partition are known. In this paragraph, an hueristic algorithm of hierarchical bisection will be presented. It fits well the needs of the analysis algorithm and ensures efficient analysis for most of network structures. The algorithm has been obtained as an extension of the method presented in [6]. It has the property that whenever possible, the graph is partitioned through block vertices. It minimizes the number of subgraph block vertices and consequently storage space needed for keeping symbolic network functions can be smaller.

The algorithm determines the series of vertex bisections of the graph. Let us consider unigraph $G = (V, E)$ /for this purpose we need only information about structure of the graph, so G can be undirected and unigraph/ with the set of block vertices B. Assume that this graph can be partitioned onto two subgraphs, and $G_a = (V_a, E_a)$ is one of these subgraphs containing vertex N_o, which will be called the vertex generating partition. Let theoutward degree of the vertex $v_a \in V_a$ be the number of vertices incident with v_a with the exception of vertices of V_a and B. i-th bisection of G is determined by attaching to the set of boundary vertices S_i of i-th bisection such block vertices, which are incident with N_o, and with such vertices of S_i, which have zero outward degree. Succesive set of boundary vertices, S_{i+1}, is obtained from S_i in the following way:

(i) as new vertex N_o generating partition S_{i+1} we choose such a vertex from S_i, which has the least, non-zero outward degree.

(ii) we attach N_o to V_a, and include in S_i all vertices of G incident with N_o and not belonging to B and V_a.

Obtained S_{i+1}, witch attached block vertices, determines new bisection of G.

Important problem lies in choosing starting vertex N_o. This vertex determines the direction of moving through the graph, and thus strongly influences the series of bisections. The problem may be solved with the use of the algorithm determining the peripherial vertex and graph diameter [7].

After the series of bisections is determined we can choose the best bisection. The quality of the partition can be estimated; in the case of hierarchical analysis we may assume, that it is worth while to further partition the graph if

(i) the number of block vertices introduced is sufficiently small.

(ii) the cardinality of sets of multitrees of subgraphs and substitute graph is less than the cardinality of the set of multitrees of the graph.

In comparison with the method from [6], the algorithm allows obtaining subgraphs with less number of block vertices and more uniform decomposition. Tests performed showed that time of decomposition linearly depends on the network size.

5. IMPLEMENTATION

A computer program HADEN 2 was elaborated on the basis of algorithms presented. The program deals with the case of hierarchical bisection. This does not cause the loss of generality, because any decomposition may be treated as a hierarchical bisection.

The most important problem in the implementation is the form of data structures, because of necessity of storing all sets of multitrees in the memory. Any multitree weight can be represented as

$$c \cdot \prod_i y_i^{p_i} \qquad /4/$$

where c is product of nonsymbolic branch admittances of the tree, y_i are symbols, and p_i is the exponent of the symbol /we may identify some elements/. One of symbols may be complex variable s. Because p_i's are small integers, they may be stored in a single computer word in packed form.

It is convenient to store the sets at multitrees spanned over set of block vertices as a sorted file of records where record consists of the set of k-trees of the same type /i.e. having the same set of components/. The same can be done with the sets of proper multitrees. However, the proper k-tree t may be considered as a pair /for bisection/ of symbolic addresses pointing to appropriate sets of k-trees $T^1_{v_t}$ of subgraphs G_i, i = 1, 2. These addresses can be evaluated during computations and stored in the place of multitree edges.

The symbolic formula obtained in the analysis may be considered as a sum of products with hierarchical parenthesis 1 :

$$\sum_{T_1^s} \prod_{G_1^s} \left(\ldots \left(\sum_{T_k^s} \prod_{G_k^s} \left(\sum_{t \in T} \prod_{e \in t} y_e \right) \right) \ldots \right) \qquad /5/$$

Computing numerical values from /5/ can be done without expanding the formula, because the partial results are stored, thus saving computer time.

6. RESULTS

To test the program, the RC-ladder network was choosen because of its regular structure. Fig. 2 shows the dependence of analysis time on the number of nodes of the network /half of elements were kept in symbolic form/. Tests were performed for partitions resulting in proper blocks of Γ (a), Π (b), $\Pi\Pi$ (c) and π - T (d) types. Experimental results confirm strictly linear dependence of time of analysis on the network size. The relation between the computing time and the proper block size shows an existence of an "optimum" and may be useful for evaluating the quality factor for the program of decomposition.

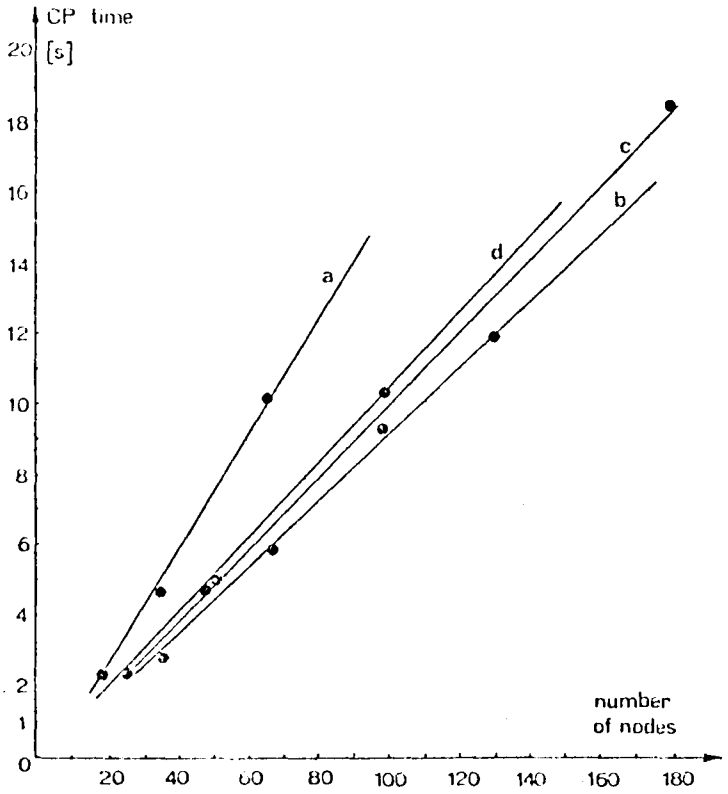

Fig. 2. Dependence the analysis time on the number of network nodes.

7. CONCLUSIONS

The method described can be used for symbolic analysis of large electronic networks, with practicaly unlimited number of elements kept in symbolic form. The authors are convinced that, the method, after user-oriented programs are elaborated, can become a powerful tool for the analysis and design of large LLS networks. Computational effort linearly depends on the network size and this allows the method to compete with the fastest numerical methods /e.g. sparse matrix techniques/.

Works are continued to apply the method for statistical analysis, tolerance analysis and optimization, analysis of large SC networks and hierarchical macromodelling.

7. REFERENCES

[1] Starzyk J and E. Śliwa, "Hierarchic Decomposition Method for the Topological Analysis of Electronic Networks", International Journal of Circuits Theory and Applications, vol. 8, 1980.

[2] Centkowski G, J. Starzyk and E. Śliwa, "Computer Implementation of Topological Methods in the Analysis of Large Networks" ECCTD'80, Warsaw 1980.

[3] Chen W.K., "Applied Graph Theory - Graphs and Electrical Networks", North-Holland P.C. 1976.

[4] Starzyk J., "Topological Analysis of Large Electronic Circuits" /in Polish/, dissertation, Technical University of Warsaw, 1981.

[5] Starzyk J. and E. Śliwa, "Upward topological Analysis of Large Linear Networks", to appear.

[6] Sangiovanni-Vincentelli A., L.K. Chen, and L.O. Chua , "An Efficient Heuristic Cluster Algorithm for Tearing Large-scale Networks" IEEE Trans. on Circuits and Systems, vol. CAS-24 no 12, Dec. 1977.

[7] Centkowski G and J. Starzyk, "An Efficient Hierarchical Decomposition Algorithm", to appear.