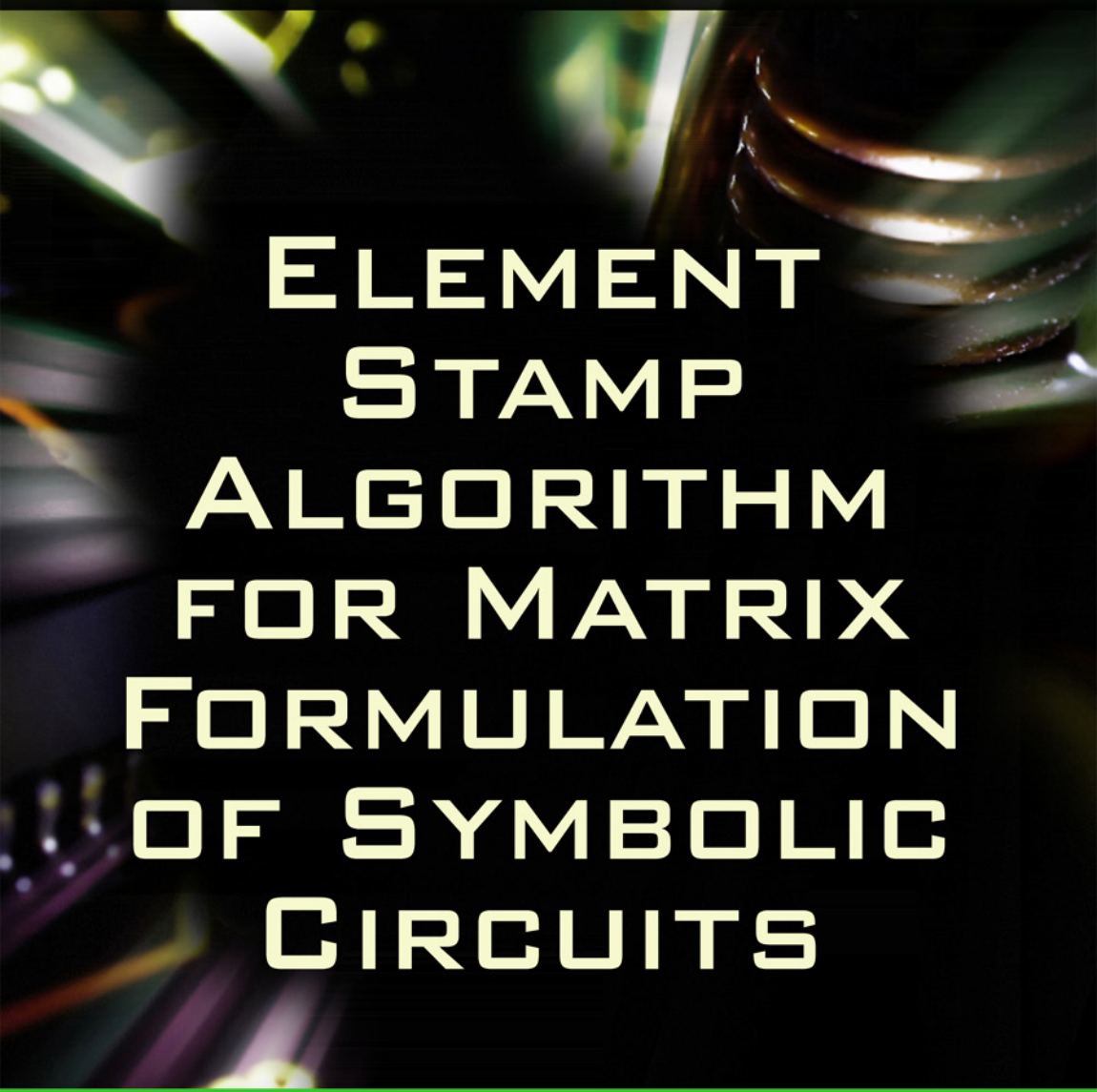
Fawzi M. Al-Naima
Bessam Z. Al-Jewad



Element Stamp Algorithm for Matrix Formulation of Symbolic Circuits

COMPUTER NETWORKS

Novinka

ELEMENT STAMP ALGORITHM FOR MATRIX FORMULATION OF SYMBOLIC CIRCUITS

COMPUTER NETWORKS

Additional books in this series can be found on Nova's website under the Series tab.

Additional E-books in this series can be found on Nova's website under the E-book tab.

MATHEMATICS RESEARCH DEVELOPMENTS

Additional books in this series can be found on Nova's website under the Series tab.

Additional E-books in this series can be found on Nova's website under the E-book tab.

COMPUTER NETWORKS

ELEMENT STAMP ALGORITHM FOR MATRIX FORMULATION OF SYMBOLIC CIRCUITS

FAWZI M. AL-NAIMA

AND

BESSAM Z. AL-JEWAD

Nova Science Publishers, Inc.

New York

LIBRARY OF CONGRESS CATALOGING-IN-PUBLICATION DATA

Al-Naima, Fawzi M.
Element stamp algorithm for matrix formulation of symbolic circuits /
Fawzi M. Al-Naima and Bessam Z. Al-Jewad.
p. cm.
Includes bibliographical references and index.
ISBN 978-1-61761-243-5 (eBook)
1. Symbolic circuit analysis. 2. Electric networks--Mathematical models.
3. Matrices. I. Al-Jewad, Bessam Z. II. Title.
TK7867.A45 2010
621.3815--dc22
2010026980

Published by Nova Science Publishers, Inc. † New York

CONTENTS

PREFACE

The need to analyze a linear network is a recurring requirement in computer-aided network analysis. Not only a majority of the network problems to be solved is posed as linear problems; nonlinear resistive and dynamic networks are usually solved by the analysis of a sequence of "linearized" networks. The analysis of such networks can commonly be viewed as a two-stage process:

1. equation formulation and
2. linear solution.

In this book, an attractive formulation procedure will be uncovered that will not only change the solution strategy but also our view to matrix reduction techniques.

In the integrated symbolic circuit analysis, one might attempt substituting every component of the circuit by its model. Such an analysis is based on a strictly network-modeling point of view and it appears highly descriptive of the circuit behavior. However, a strictly symbolic network analysis is not actually of any interest to engineers. This is mainly due to two inherent drawbacks in such an analysis:

1. As this analysis introduces many additional elements and variables to the original circuit, the resultant symbolic expression will be extremely large and beyond any human interpretation or comprehension even for the smallest circuits (not to mention the size limitation imposed on the circuit to be analyzed). Such a result will be completely useless especially for the designer. The reason

behind this is simple: To design a circuit, it is easier to have five or six variables to control within some simplified constraints than to have a hundred variables most of which having complicated constraints and do not affect the circuit characteristics by any considerable amount.

2. A strictly symbolic network analysis requires extremely high processing power and storage even for small circuits. Furthermore, the system matrix suffers often from ill-conditioning and singular values, which imposes an additional analysis difficulty against producing a result that can be produced much more efficiently by other methods of circuit analysis. Ill-conditioning of symbolic system matrices is further complicated by the fact that numerical values do not exit to check for such a problem during the formulation process

The compacted modified nodal analysis CMNA method (or the element-stamp method) is a very nice and easy way to illustrate the impact of each element on the matrix since it constitutes going through each branch of the circuit and adding its contribution to the system matrix in the appropriate positions. It represents an automatic technique to construct the nodal admittance matrix. This method consists of programming a lookup table for every element type in the network. This table has link-lists that will test which variables of the element are actually needed in the final compacted matrix and introduce the element in a way so as to eliminate the redundant variables.

Chapter 1

COMPUTER REPRESENTATION OF SYMBOLIC MATRICES

Before introducing the CMNA method, a bit of background is needed on how to represent a symbolic system of equations in a computer program.

In general, a symbolic variable or expression must be represented as a string data type or a pointer to a string data type that is stored in memory [1]. The first thing to do is to set a list, which will include the names of the symbolic variables that will be used later. The rule is that the instance of any symbolic variable that is not included in the list will cause the addition of the variable name to the list. If an identifier has not been assigned a value, then it stands for itself (or its name). In other words, each variable in our list is a pointer and it will be handled accordingly. If it has no value to point to then it points to its name. Therefore, it is a symbol.

Example: Consider the following assignment operation

$$p = x^2 + 4x + 4$$

Here the identifier *p* has been assigned the formula x^2+4x+4. The identifier x has not been assigned a value; it is just a symbol, an unknown. The identifier p has been assigned a value. It is now like a programming variable. The value of p is

x^2+4x+4

The value of *x* on the other hand is "*x*".

Because a variable can be assigned a value, which contains symbols, the issue of evaluation immediately arises. For example, consider the assignment

x=3

it should result in

p=25

So the other rule of our list is that if A is pointing to B and B is pointing to C then A should use C in its evaluation.

In general, handling the string by itself should be avoided as much as possible. The string should always stay in the memory and only pointers to it should be manipulated. This is very important since we might have many functions in our routine and certainly passing string parameters to such functions is quite a waste of time and memory.

Mathematical formulae, e.g. things like $\sin(x+\pi/2)$, and x^3y^2-2/3 are called expressions. They are made up of symbols, numbers, arithmetic operators and functions. Symbols are things like sin, x, y, Pi etc. Numbers include 12, 2/3, 2.1, etc. The arithmetic operators are + (addition), – (subtraction), × (multiplication), / (division), and ^ (exponentiation). Added to these are the brackets, which are used to separate the terms in the expression, to reset the priority of performing the operations, and to pass parameters to functions like in $\sin(x)$. Strictly speaking, formulae in the routine should be represented as expression trees or DAGs (Directed Acyclic Graphs) in computer jargon [1], [2]. When the routine is programmed to manipulate formulae, it is basically manipulating expression trees.

Having all these rules stated, the routine for any of the basic operations is now easier to deal with. The arguments and cases that should be tested before executing any of the operations are

1. The identity element I (e.g. the zero for addition)
2. The negation element a^* (e.g. negative arguments for addition)
3. The invalid element O (e.g. the zero divisor for division)
4. The null element (e.g. the zero for multiplication)
5. The association and commutation between the basic operations and bracket handling
6. The separation and evaluation of numerical values as they arise during the operation

The above list of if-conditions are not hard to program but they require a considerable amount of time to execute. The result is to make the routine much slower than what is required. The best way to deal with that is to compile the basic routines and use them as object-code functions whenever needed. Furthermore, some of the above tests can be removed (for example, if no division by zero is expected then the invalid element test can be removed).

Following the programming of the basic symbolic operations, a data structure for symbolic polynomials needs to be defined. The data structure representing those polynomials would need to support addition, subtraction and multiplication.

Finally a symbolic matrix representation needs to be included in the program. A symbolic matrix $\boldsymbol{A}$ can be described by the multiplication of a row operator $\boldsymbol{P}$ and a column operator $\boldsymbol{Q}$ with a diagonal matrix of symbols

$$\boldsymbol{A} = \boldsymbol{PYQ} \tag{1}$$

$\boldsymbol{P}$ and $\boldsymbol{Q}$ are the matrix operators (or topological matrices) that indicate location of the matrix element value in the symbolic matrix $\boldsymbol{A}$. Thus, if a matrix with $n \times n$ dimensions has b symbolic elements, then $\boldsymbol{P}$ and $\boldsymbol{Q}$ matrices are $n \times b$ and $b \times n$ operator matrices.

Example: Consider the following fully symbolic matrix

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

This can be represented by:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} . \text{diag} \left(\begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} \right) . \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where diag(.) refers to diagonal matrix with diagonal elements being those of the inside array while every other element is zero.

One can immediately notice that the $\boldsymbol{Y}$ matrix is a diagonal matrix that can be represented by a simple array while both $\boldsymbol{P}$ and $\boldsymbol{Q}$ are sparse numerical matrices.

One benefit of this representation is that linear operations on rows can be simply implemented on $\boldsymbol{P}$ while linear operations on columns can be implements on $\boldsymbol{Q}$ without altering the diagonal matrix. For example if one would like to add the first row of the matrix in the previous example with double the second row and put the result in place of the second row then these operations can be performed as follows

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{11} + 2a_{21} & a_{12} + 2a_{22} & a_{13} + 2a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} . \text{diag} \left(\begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} \right) . \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For most circuit analysis applications, system equations are rarely fully dense or fully symbolic. In that respect, some matrix elements may contain not only a single symbolic element value but constant values as well. These constants do not affect the structures of matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$. The numerical value of the element can be moved to either $\boldsymbol{P}$ or $\boldsymbol{Q}$ and the number 1 can be added to the diagonal matrix. In fact even if the element values are complete polynomials the representation is still not altered. However, semi-symbolic sparse system matrices offer room for reduction in $\boldsymbol{P}$ and $\boldsymbol{Q}$.

Example: Consider the following semi-symbolic matrix

$$\boldsymbol{A} = \begin{bmatrix} 7 & 8 \\ c & d \end{bmatrix}$$

This can be represented as

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix} \begin{bmatrix} 7 & 0 \\ 0 & 8 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

It can also be reduced by reducing the 8 (element in **row 1 column 2** on the original matrix). Simply delete **column 2** from the $\boldsymbol{P}$ matrix and replace **row 1** from the $\boldsymbol{Q}$ matrix by the addition of rows 1 and 2 then delete row 2

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & d \end{bmatrix} \begin{bmatrix} 7 & 8 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Reduction algorithms can be advised to help in reducing the size of $\boldsymbol{P}$ and/or $\boldsymbol{Q}$ and thereby save the memory space considerably [3].

In the same way symbolic matrix equations of the form

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{2}$$

Where $\boldsymbol{x}$ is the column vector of unknowns and $\boldsymbol{b}$ is a symbolic column vector, can be represented as

$$\boldsymbol{PYQx} = \boldsymbol{IBq} \tag{3}$$

Where $\boldsymbol{I}$ is initially the identity matrix and $\boldsymbol{q}$ is a column vector. Hence only arrays of symbolic polynomial data structures need to be stored to represent diagonal matrices $\boldsymbol{Y}$ and $\boldsymbol{B}$ while numerical matrices $\boldsymbol{P}$, $\boldsymbol{Q}$, $\boldsymbol{I}$ and vector $\boldsymbol{q}$ can be manipulated to solve this system.

While this representation of matrix equations manifests itself directly towards the algebraic solution, it does not represent the only approach to solve or represent the system of equations. An alternative topological approach based on graphical representation of the equations exists as will be shown later on.

Solving equations similar in form to equation (2) can be done in many ways. Among which, parameter reduction method is one alternative when there is a need to find the transfer characteristics of a system. With this method, there is no need to solve the whole linear system just to find the ratio of two system variables. The system matrix is reduced successively before any attempt of evaluation is made. It should be clear that such a reduction does not reduce the

amount of the required operations. But the key point here is that the reduction helps in reducing the matrix size for the purpose of storing and thus reducing the number of memory excess times. It also increases the speed of passing parameters to functions by reducing the sizes of their arguments. In this sense, the reduction can be carried out during the formulation of the system matrix and in this way the memory requirement is highly reduced. This provides a way to handle very large system matrices.

Due to its importance in later on discussion, a reduction method will now be demonstrated (namely Kron's reduction) of a row (or set of rows) and its associated column (or their associated columns) that are collectively called a *matrix axis* [4].

Consider the following matrix equation:

$$\begin{bmatrix} \boldsymbol{A_1} & \boldsymbol{A_2} \\ \boldsymbol{A_3} & \boldsymbol{A_4} \end{bmatrix} \times \begin{bmatrix} \boldsymbol{X_1} \\ \boldsymbol{X_2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{B_1} \\ \boldsymbol{B_2} \end{bmatrix} \tag{4}$$

In which $\boldsymbol{A}_1$, $\boldsymbol{A}_2$, $\boldsymbol{A}_3$ and $\boldsymbol{A}_4$ can be thought of as matrices or single coefficients and $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, $\boldsymbol{B}_1$, and $\boldsymbol{B}_2$ are vectors or single variables respectively. Equation (4) in expanded form is

$$\boldsymbol{A_1X_1 + A_2X_2 = B_1} \tag{5}$$

$$\boldsymbol{A_3X_1 + A_4X_2 = B_2} \tag{6}$$

Rewriting equation (6) gives

$$\boldsymbol{A_4X_2 = B_2 - A_3X_1}$$

Pre-multiplication by $\boldsymbol{A_4^{-1}}$ yields

$$\boldsymbol{X_2 = A_4^{-1}B_2 - A_4^{-1}A_3X_1} \tag{7}$$

On substitution of $\boldsymbol{X}_2$ into equation (5) we get

$$\boldsymbol{A_1X_1 + A_2(A_4^{-1}B_2 - A_4^{-1}A_3X_1) = B_1}$$

Rearranging and collecting terms gives

$$[\boldsymbol{A_1} - \boldsymbol{A_2}\boldsymbol{A_4^{-1}}\boldsymbol{A_3}][\boldsymbol{X_1}] = [\boldsymbol{B_1} - \boldsymbol{A_2}\boldsymbol{A_4^{-1}}\boldsymbol{B_2}] \tag{8}$$

or

$$[\boldsymbol{A_1^{'}}][\boldsymbol{X_1}] = [\boldsymbol{B_1^{'}}] \tag{9}$$

From which $\boldsymbol{X}_2$ can still be found from equation (7). Hence, the dimension of matrix that needs to be inverted is reduced by eliminating the unknown $\boldsymbol{X}_2$. Thus, the axes corresponding to $\boldsymbol{A}_2$ and $\boldsymbol{A}_3$ are eliminated. The sub-matrix $[\boldsymbol{A_1^{'}}]$ was modified to reflect the system solution corresponding to the unknown $[\boldsymbol{X_1}]$. This is valid as long as $[\boldsymbol{A_4}]$ is ***not singular***. The above reduction is very useful in large network analysis.

The Kron's reduction is greatly simplified when a single axis is being eliminated (i.e. $\boldsymbol{A}_4$ has a single element). Repeating the elimination of a single axis is superior to the elimination of many axes in a single reduction as is verified in [5]. In the case of a single axis reduction, $[\boldsymbol{A_4^{-1}}]$ becomes $1/a_{44}$. The modification of the elements not in the axis being eliminated can be carried out element-by-element rather than by application of equation (8) (hence no matrix multiplication is required). All elements are modified in-place and no additional computer memory is required for the storage of the minor $\boldsymbol{A_2}\boldsymbol{A_4^{-1}}\boldsymbol{A_3}$. It can be easily verified that the elimination of axis *k* is simply [4]

$$a_{ij}^{'} = a_{ij} - a_{ik}a_{kj}/a_{kk}, \quad i,j \neq k \tag{10}$$

The reduction can be carried out recursively in a very efficient way. Further memory efficiency can be gained if the reduction is done in place without preserving additional space for sub-matrices. The algorithm would go something like that:

CalcReducedMatrix (input Matrix "e" and requested output matrix size n, output Matrix)
{
Check for invalid conditions (not square, zero pivotal element etc)
If number of rows = 1 then return only element, e[0][0]
Initialize a pivotal element
Initialize return value to empty matrix
Initialize current matrix size m

```
While m is smaller than requested output size
{
For each row i (i smaller than m)
And for each column (j smaller than m)
{Replace e[i][j] with e[i][j]-e[i][m]*e[m][j]/e[m][m]
}
Return CalcReducedMatrix of matrix e[m-1][m-1]

}
}
```

Notice this algorithm will work regardless of whether the matrix is composed of numbers or symbolic polynomials. For it to function with polynomials the data structure representing them would need to support the basic mathematical operations as said before. Solving the linear system may incorporate additional steps for the right-hand side vector and for pivoting (in case there is a zero pivot element) for which a separate evaluation function needs to be written. In practice the evaluation does not need to be complicated. Since only a zero pivotal element is forbidden there is no need to build a detailed simplification algorithm. An evaluation of the pivot element can be done by giving random arbitrary values to the symbols so if there are some cancellations in the pivotal element expression resulting in setting its value to zero, it will immediately show up with the random numerical values given to the symbols. In such a case swapping the rows of the matrix can generally solve the problem easily. Luckily if the matrix is in the form of equation (3), the row swapping can be implemented easily on the ***P*** and ***I*** matrices without disturbing the symbolic string vectors.

One drawback to the algorithm is the need to successively divide by a matrix element which is not easily implemented in the form of equation (3). To overcome that, scaling of the unknown variables can be used to convert the divisions into multiplications as shown in the example below.

Example: In the following matrix equation, we would like to find the ratio (transfer characteristic) of axis 1 to axis 2. That is, x_1/x_2

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{11}$$

Solution: Partitioning and eliminating the pivotal element a_{33}, using equation (11), we get

$$\begin{bmatrix} a_{11}-a_{13}a_{31}/a_{33} & a_{12}-a_{13}a_{32}/a_{33} & 0 \\ a_{21}-a_{23}a_{31}/a_{33} & a_{22}-a_{23}a_{32}/a_{33} & 0 \\ a_{31}/a_{33} & a_{32}/a_{33} & 1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1-a_{13}b_3/a_{33} \\ b_2-a_{23}b_3/a_{33} \\ b_3/a_{33} \end{bmatrix}$$

However, before eliminating row 3 and column 3 of the matrix it is worth noting the high number of divisions and simplifications needed after just one setp of the solution. This can be overcome simply by scaling the unknow variable in equation (11) by the pivotal element a_{33}

$$\begin{bmatrix} a_{11}a_{33} & a_{12}a_{33} & a_{13} \\ a_{21}a_{33} & a_{22}a_{33} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ a_{33}x_3 \end{bmatrix} = \begin{bmatrix} a_{33}b_1 \\ a_{33}b_2 \\ b_3 \end{bmatrix} \tag{12}$$

Next, the new variable $a_{33}x_3$ can be eliminated by applying equation (10) with unity pivotal element

$$\begin{bmatrix} a_{11}a_{33}-a_{13}a_{31} & a_{12}a_{33}-a_{13}a_{32} & 0 \\ a_{21}a_{33}-a_{23}a_{31} & a_{22}a_{33}-a_{23}a_{32} & 0 \\ a_{31} & a_{32} & 1 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ a_{33}x_3 \end{bmatrix} = \begin{bmatrix} b_1a_{33}-a_{13}b_3 \\ b_2a_{33}-a_{23}b_3 \\ b_3 \end{bmatrix}$$

and finally scaling back the unknown variable x_3 can be done to get

$$\begin{bmatrix} a_{11}a_{33}-a_{13}a_{31} & a_{12}a_{33}-a_{13}a_{32} & 0 \\ a_{21}a_{33}-a_{23}a_{31} & a_{22}a_{33}-a_{23}a_{32} & 0 \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1a_{33}-a_{13}b_3 \\ b_2a_{33}-a_{23}b_3 \\ b_3 \end{bmatrix}$$

Now, this form of the reduction does not require any divisions and does not disturb the original storage arrangement of the symbolic matrix (yet the test for a zero pivotal element is still needed as a zero pivot would turn both sides of the equation to zero). Thus the structure of equation (3) is kept as it is without the need to implement sophisticated division/simplification algorithms. Hence equation (10) can now be rewritten as

$$a_{ij}^{'} = a_{ij}a_{kk} - a_{ik}a_{kj}, \quad i,j \neq k \tag{13}$$

Now it is very easy to shift focus to the 2x2 sub-matrix generated

$$\begin{bmatrix} a_{11}a_{33} - a_{13}a_{31} & a_{12}a_{33} - a_{13}a_{32} \\ a_{21}a_{33} - a_{23}a_{31} & a_{22}a_{33} - a_{23}a_{32} \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1a_{33} - a_{13}b_3 \\ b_2a_{33} - a_{23}b_3 \end{bmatrix}$$

or

$$\begin{bmatrix} a_{11}^{'} & a_{12}^{'} \\ a_{21}^{'} & a_{22}^{'} \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1^{'} \\ b_2^{'} \end{bmatrix}$$

Solving this matrix equation gives

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{a_{11}^{'}a_{22}^{'} - a_{12}^{'}a_{21}^{'}} \begin{bmatrix} a_{22}^{'} & -a_{12}^{'} \\ -a_{21}^{'} & a_{11}^{'} \end{bmatrix} \times \begin{bmatrix} b_1^{'} \\ b_2^{'} \end{bmatrix}$$

The required ratio can be found as

$$\frac{x_1}{x_2} = \frac{a_{22}^{'}b_1^{'} - a_{12}^{'}b_2^{'}}{-a_{21}^{'}b_1^{'} + a_{11}^{'}b_2^{'}}$$

This follows from the well-known butterfly relation [6].

Chapter 2

TOPOLOGICAL (GRAPHICAL) REPRESENTATION OF MATRICES

Topological methods of solving linear systems of equations relate directly to the study of electronic circuits. Study of these methods was initiated by Kirchhoff [7] at the end of the 19th century and intensified in the sixties and seventies [8]-[15], to a large degree due to the development of computer technology and related devices requiring advanced methods of electronic circuit analysis and design.

Concurrently, algebraic methods that represent network topology and can be used for its analysis were developed, most notably by Wang [16]-[19] and Bellert [20]-[24]. The most attractive feature of topological methods that was apparent since the early stage of their development was their ability to obtain transfer functions directly from the circuit *netlist* or from its graph description simply by inspection.

Since then graph based methods were generalized to solve any set of linear equations associated with electrical circuits using signal flow graphs (Coates [25],[26] and Mason's [27], [28] graphs), linear graphs (current-voltage [29], [30] and nullator-norator [14] , [31] graphs), and directed graphs (unistor [32], [33] and dispersor [33], [34] graphs) to describe the matrix structure. Specialized analysis methods were developed for each of these graph representations and recently there has been an effort on unifying graphical methods that would handle these various representations and reuse results from one form of graph representation to another [35].

As indicated above, three major types of graph representations are used to describe the system matrix in circuit topology. These three major types and some of their better known subtypes are as follows:

1. Flow graphs
   a. Coates' graph
   b. Mason's graph
2. Directed graphs
   a. Unistor graph
   b. Dispersor graph
3. Conjugated graphs
   a. Current-voltage graph
   b. Nullator-norator graph

These graphs represent both the interconnection structure of an electronic circuit and its element values. Since topological analysis and diagnosis is performed on a linear system, it is assumed that the circuit elements represented by the graph are linearized around their DC operating points. Thus, in general, graphs represent circuit interconnected structures and associated linearized element values.

Computer representation of the graph is simple. A *directed* graph $G=(V,E)$ consists of a finite, nonempty set of vertices V and a set of edges E that are ordered pairs $e_i=(v_i,v_i')$ of vertices; v_i is called the tail and v_i' is called the head of edge e_i. Each edge $e_i \in E$ has a weight w_i associated with it. If the order of the pairs is irrelevant then the graph is called *undirected*. In addition, the following notation will be adopted: if all the associated weights are unity then the graph will be called *oriented*, while if both the weights and the order are irrelevant then it will be called *associated* [36].

Each edge of a graph may describe one, two, or four elements of the coefficient matrix depending on whether it is in a signal-flow graph, directed graph or pair of conjugated graphs. Flow graphs represent each element of the coefficient matrix as an independent edge of the graph, resulting in relatively complex graphs that contain as many edges as the number of nonzero entries in the coefficient matrix. Directed graphs may represent two elements of the coefficient matrix, either in a single row (dispersor graph) or a single column (unistor graph) as an independent edge of the graph. Finally, the conjugated graphs represent four elements of the coefficient matrix as an independent edge of the graph, resulting in graphs that contain the minimum number of edges. The level of graph complexity reflects directly onto its simplification through Graph Decomposition.

Graph decomposition is used in many applications dealing with large systems like linear programming [37], [38], the shortest path problem [39]-[41], information encoding [37], synthesis of VLSI circuits [42], partition of sparse matrices [6], [43], job shop scheduling [44], gene assembly [45],

software synthesis [46], etc. Thus very well developed algorithms and packages exist to handle graph decomposition. The aim of graph decomposition is to improve the algorithmic performance of problems represented by a system graph. Network decomposition is used in analysis of computer and communication networks [47]-[52]. Decomposition plays an important role in stability analysis of large systems [53]-[55] or layout compaction in very large scale integrated (VLSI) circuits [56]. In circuit analysis one can further distinguish diakoptics [57]-[59], generalized hybrid analysis [60], and topological analysis with nodal decomposition [61], [62].

Detailed discussion of the different graphical representations and their decomposition variations is beyond the scope of this book. However, we will comment on how to generate such graphs with minimum efforts and relate them easily to the algebraic approach during the formulation of the system matrix equation for any circuit.

Chapter 3

BASIC NETWORK ELEMENTS AND EQUATIONS (DISTRIBUTED VS. LUMPED)

The electrical networks of interest to engineers consist of an interconnection of some components. For a two-terminal component, which is called a *branch*, two variables are of primary interest: the voltage across the branch and the current through the branch. Due to the physical construction of the components, there is a definite voltage-current relationship characterizing each branch. These relationships are collectively called the *constitutive equations* (**CE**) [6]. For example, a resistance is characterized by the conventional Ohm's law, while a capacitance and an inductance are characterized by differential equations in the time-domain. The relationship may involve also variables of other branches, leading to what are called "coupled branches". In such a case, it is convenient to use a two-port representation of each one of the two-coupled branches.

Common symbolic analysis literature focuses on such simple networks. Yet, current needs for symbolic analysis have widened to include fault diagnosis, high frequency network analysis and very fast switching VLSI. Many network elements in such networks can be defined depending on the type of the network and the type of analysis. Generally speaking, different parts of those networks can be lumped, distributed, or mixed. When a number of branches are connected together, we have a *lumped network*. A lumped network is characterized by its small dimensions compared to the wavelengths of the signals of interest. On the other hand, when the element dimensions become comparable to the wavelengths of the electrical signals involved in the network we will have a *distributed network*. The latter is distinguished by the traveling waves propagating through its components.

Any lumped network obeys three basic laws: Kirchhoff's current law (**KCL**), Kirchhoff's voltage law (**KVL**), and a set of constitutive equations (**CE**) defined for each branch. The constitutive equations are intrinsic physical properties, and are independent of how the branches are connected together. On the other hand, **KCL** and **KVL** are linear algebraic constraints on branch voltages and currents, arising from the interconnection of the branches, and are independent of the branch characteristics. Hence they are topological constraints. Even though many of the implementation techniques proposed here are imposed with *linear networks* in mind, it should be pointed out that **KCL** and **KVL** are valid for any lumped network, whether the network is linear or nonlinear, time-invariant or time varying.

The basic elements in a lumped network are linear, lumped and time-invariant resistors, capacitors as shown in Figure 1. No matter how basic the consecutive equations of these elements may seem, they impose constraints that have to be satisfied by the solution variables. In addition, **KCL** and **KVL** impose topological constraints that also have to be satisfied simultaneously by the same solution variables. Fortunately, transforming all the **CE** into the Laplace domain will yield a set of linear (or linearized) equations that can be solved easily.

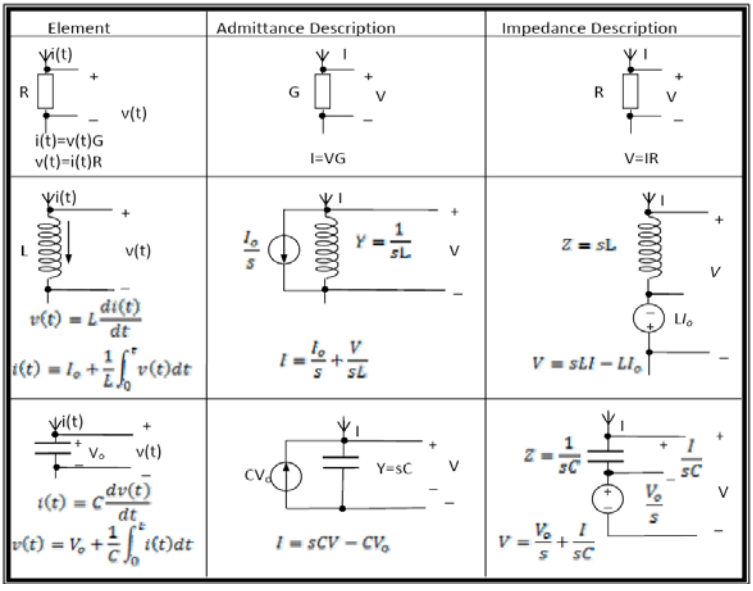


Figure 1. Lumped network elements in time and Laplace domains; Io and Vo represent the initial inductor current and capacitor voltage respectively.

Speaking strictly, in any electrical network involving time-varying signals (whether lumped or distributed), all the variables must satisfy the famous Maxwell's equations (**ME**). The set of Maxwell's equations form the basis of our topological constraints. In addition, the solution variables have to fulfill some boundary conditions (**BC**) that are specific to each element. The equations constitute a complicated partial differential (or integral) set that has a geometry-dependent form. The solution of this set cannot be carried out symbolically for any set of elements and sound numerical techniques seem to be the only possible way in most of the problems. However, it is not possible to solve the electromagnetic integral or differential equations either conveniently or rigorously in regions containing or bounded by geometrically complicated metal or dielectric structures, and the basic analytical discipline of electrical engineering curricula has from the beginning been that of lumped-element circuit analysis. This employs the previously mentioned idealized concepts of two-terminal resistances, inductances and capacitances to represent the localized functions of energy dissipation, magnetic field energy storage, and electric field energy storage, respectively. Voltages and currents, which are related by integral or differential expressions to electric and magnetic fields, are the primary electrical variables.

Such an approximation is an adequate substitute for the electromagnetic theory when the occurrences of the three functions mentioned above can be separately identified. This happens when the dimensions of a circuit are sufficiently small that no appreciable change will occur in the voltage or current at any point during the time electromagnetic waves would require propagating through the entire circuit. The size criterion is obviously a function of frequency. At the power line frequency of 50 Hertz, the methods of lumped element circuit analysis are applicable with high accuracy to circuits several kilometers long, while at microwave gigahertz frequencies the same methods may be useless for analyzing a circuit less than a few centimeters across (e.g. monolithic VLSI). In such a case, the set of **ME** together with the **BC** have to be solved numerically. The reduction of these equations into simple **KCL** and **KVL** relations is not so simple anymore. More importantly, one will not be dealing with currents and voltages any further, but rather with fields. This in general poses a problem against the high frequency symbolic circuit analysis both from the modeling and from the analysis side of views. As an example, the high frequency models of a simple thin-film resistor used in microwave circuit applications are shown in Figure 2 where the complicated frequency-conditioned situations are apparent.

In addition to the techniques of electromagnetic theory and of lumped element circuit analysis, electrical engineers make use of a third analytical procedure for electrical problems, which combines features that is separately characteristic of each of the other two methods. It extends the application of the concepts of the lumped-constant theory to circuits which can be indefinitely long in one dimension, but which must be restricted and uniform in the other dimensions throughout their length. This analysis discloses propagating waves of the voltage and current variables, analogous to the waves of electric and magnetic fields that are the solutions of Maxwell's equations. The method is known as *distributed circuit analysis*. In this method, each device is considered as being the interconnection of an infinite number of lumped elements. The final answer in this case will be the limiting solution as the number of lumped elements grows indefinitely. This method represents the only reasonable means of analyzing high frequency and microwave circuits symbolically and it can be used at different levels of accuracy [63].

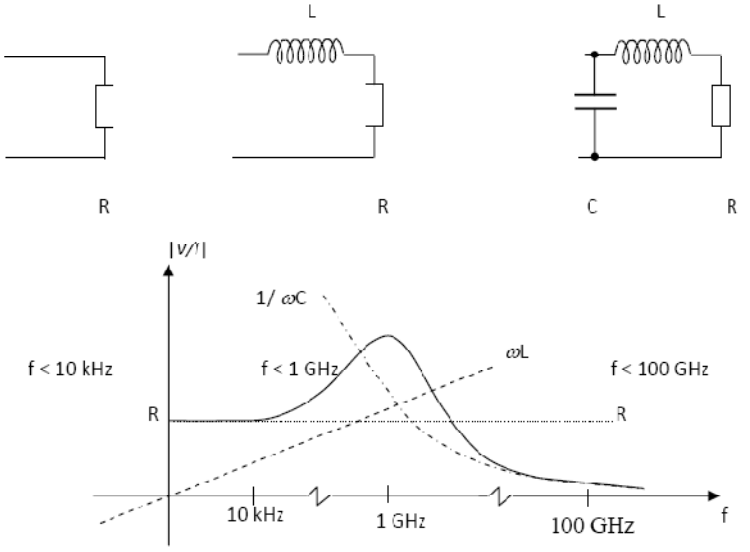


Figure 2. High frequency models of a simple thin - film resistor.

In symbolic circuit analysis (whether distributed or lumped), one might attempt substituting every component of the circuit by its model. In the distributed circuit case, every piece of a wire is considered as a transmission line and every device is represented by a two-port high frequency model. In

the lumped circuit case, every active element (transistor, op-amp, etc) is an interconnected circuit of many elements. Such an analysis is based on a strictly network-modeling point of view and it appears highly descriptive of the circuit behavior. However, a fully detailed symbolic network analysis is not actually of any interest to engineers. This is mainly due to two inherent drawbacks in such an analysis:

1. As this analysis introduces many additional elements and variables to the original circuit, the resultant symbolic expression will be extremely large and beyond any human interpretation or comprehension even for the smallest circuits (not to mention the size limitation imposed on the circuit to be analyzed). Such a result will be completely useless especially for the designer. The reason behind this is simple: To design a circuit, it is easier to have five or six variables to control within some simplified constraints than to have a hundred variables most of which having complicated constraints and do not affect the circuit characteristics by any considerable amount.
2. A fully detailed symbolic network analysis requires extremely high processing power and storage even for small circuits. Furthermore, the system matrix suffers often from ill-conditioning and singular values, which imposes an additional analysis difficulty against producing a result that can be produced much more efficiently by other methods of circuit analysis.

The method of mixed-circuit analysis can be considered as the ultimate solution to these difficulties. It is the best available method that describes the characteristics of a circuit (lumped or distributed) in the simplest form. In this technique, the passive lumped element are left as they are, the active lumped elements are replaced by macro-models (or macro-stamps) while those elements that are known to have non-negligible distributed effects are replaced by their two-port equivalents. In this way, the features of the detailed analysis are preserved within a method that overcomes the drawbacks mentioned above.

Chapter 4

NODAL ANALYSIS TECHNIQUES

For a computer implementation of the mixed-circuit analysis one needs first an algorithm that deals efficiently with lumped circuit elements. This algorithm can then be modified to incorporate the presence of distributed elements in a proper manner by standard known methods of network analysis (namely the s-parameter techniques [64]). The efficiency of the original algorithm will determine the efficiency of all subsequent modifications to it. Therefore, the algorithmic efficiency will be the prime concern during the development of the computer implementation.

The technique of nodal analysis is discussed in most texts on circuit theory and in many versions, but for the present purposes, another version will yet be described that has substantial benefits over existing ones. Despite its popularity, many concepts of the method shall be derived for the sake of completeness as well as to emphasize the important features of the technique.

Nodal analysis is based on the application of Kirchhoff's current law to all the nodes in a circuit; each node (or unique junction of components) gives rise to a current balance equation. The set of nodal equations may be assembled to give a matrix description of the network, known as the nodal admittance matrix, where each matrix element has the dimensions of admittance. Solving all the nodal equations simultaneously yields the voltage at each node. The first problem of implementing the nodal approach in circuit analysis is the formulation of the nodal equations through the nodal admittance matrix in the simplest possible way. The subsequent manipulations, which yields the results of the analysis, then corresponds to nothing more than the solution of a set of simultaneous linear equations using standard matrix techniques.

4.1. FORMULATION OF THE NODAL ADMITTANCE MATRIX

The simplest way to approach the formulation of the nodal equations is to consider the derivation of one such equation at just one node of a passive network [65]. To this end, consider Figure 3, which shows a general passive network with node *j* isolated from the remainder of the *n*-node network.

To maintain generality, it is assumed that node *j* is connected directly to every one of the remaining *n-1* nodes through an admittance Y_{ij}. If there is no such connection in a specific circuit, then clearly the corresponding admittance may be set to zero. It is also assumed that incident on node *j* there is an external current I_j emanating from some external source, which again may be zero in a specific case. Currents i_{j1}, through i_{jn} are also assigned to each admittance respectively. Since all the currents (I_j and i_{j1} to i_{jn}) are at present defined as algebraic quantities, their assumed directions are quite arbitrary, since they may have either positive or negative signs. Finally, it should be noted that, although not shown explicitly in Figure 3, all the other nodes in the network might be considered connected to each other through further admittances, but for the present purpose the precise nature of the admittances involved is immaterial.

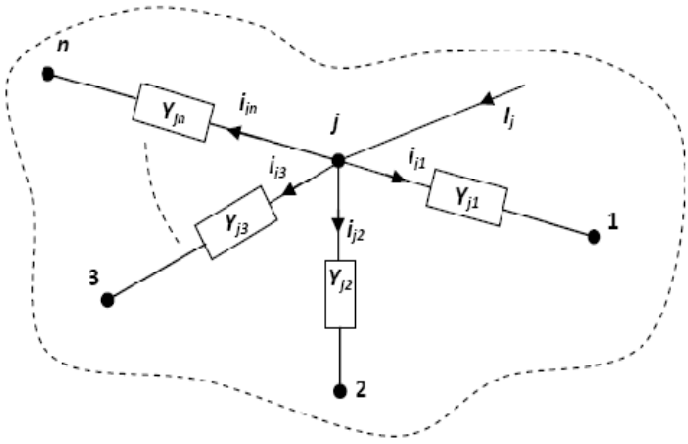


Figure 3. A typical node in a passive n - node network.

Applying Kirchhoff's current law to node *j* yields

$$-I_j + i_{j1} + i_{j2} + i_{j3} + \cdots + i_{jn} = 0 \tag{14}$$

i.e.

$$I_j = \sum_{\substack{k=1 \\ k \neq j}}^{n} i_{jk} \tag{15}$$

Now applying Ohm's law to each of the admittances Y_{jk}, (k=1…n, $k \neq j$) one sees that

$$i_{j1} = (V_j - V_1)Y_{j1}, i_{j2} = (V_j - V_2)Y_{j2}$$

and in general

$$i_{jk} = (V_j - V_k)Y_{jk} \tag{16}$$

Substituting equation (16) into equation (15),

$$I_j = \sum_{\substack{k=1 \\ k \neq j}}^{n} (V_j - V_k)Y_{jk}$$

which may be rewritten as

$$I_j = V_j \sum_{\substack{k=1 \\ k \neq j}}^{n} Y_{jk} - \sum_{\substack{k=1 \\ k \neq j}}^{n} V_k Y_{jk} \tag{17}$$

The quantity $\sum_{\substack{k=1 \\ k \neq j}}^{n} Y_{jk}$ is known as the self-admittance of the node *j* and may be denoted y_{jj}. If the quantities $-Y_{jk}$ are additionally denoted by y_{jk}, equation (17) may be further rewritten as

$$I_j = \sum_{k=1}^{n} y_{jk} V_k \tag{18}$$

By assembling equations derived for all the nodes, the set of nodal equations of the form given in equation (18) can be expressed as a single matrix equation:

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_n \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix} \times \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_n \end{bmatrix} \tag{19a}$$

i.e.

$$\boldsymbol{I = Y_i V} \tag{19b}$$

Where the matrix $\boldsymbol{Y}_i$ is known as the *indefinite nodal admittance matrix* [6], [65].

The above development leads to a very simple method for the formulation of the indefinite nodal admittance matrix of passive networks. The element entries for the matrix are found as follows:

- Diagonal elements: y_{ii}=(the sum of all admittances connected to node i), ($i = 1 \ldots n$)
- Off-diagonal elements: y_{ij}=(minus the sum of all admittances connected between node i and node j), ($i,j = 1 \ldots n$, $i \neq j$)

The formulation of the indefinite nodal admittance matrix is thus a trivial task, given some description of the network. Several properties of this matrix can be exploited to provide some efficiency. For example, the symmetric nature of the passive nodal admittance matrix is clearly seen by writing the expressions for y_{ij} and y_{ji} explicitly. This property may be exploited in both the formulation and possibly the storage of the matrix within the computer.

The above development has served to introduce a simple, easily programmed approach to the formulation of the nodal equations for linear passive networks incorporating two-terminal elements. Continuing the simple approach to the formulation problem, one may make the important observation that most practically occurring active devices may be characterized by some two-port description, and where they exist, they can

be provided as data in an extended form of branch node listing (see for example [6] for tables of two-port admittance matrices.

In some cases, the two-port description of an element may exist as a circuit model rather than a set of numerical parameters. The precise nature of such circuit models for common active devices is left until later in this book. However, such models can be analyzed once and their nodal admittance matrix added wherever they appear as part of a big system matrix as will be shown. Such an addition, of course, should be made to the proper elements of the system matrix corresponding to the position of the active element and this offers a big time saving when analyzing very large circuits. Finally, before proceeding with the active formulation, it can be noted that many passive networks are also characterized by two-port short circuit admittance parameters, and the treatment described below is equally applicable to such networks.

With these points in mind, the formulation can proceed by considering that an active network may be represented as a passive network into which (active) two-port devices are embedded. Therefore, attention is turned to the general two-port network shown in Figure 4.

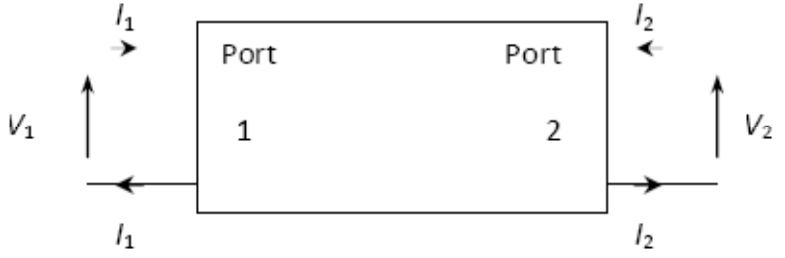


Figure 4. A general two‐port network.

The two-port admittance relationship is defined as

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} \times \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \tag{20}$$

where the upper case (e.g. Y_{12}) is used to avoid confusion with the elements of the nodal admittance matrix. Now equation (20) relates the *port* variables I_1, I_2, V_1 and V_2. If one now considers the two-port to be defined with terminal *node* variables of voltage and current as shown in Figure 5, then it can be seen, with reference to Figure 4:

$$\left.\begin{array}{ll} V_1 = V_a - V_c & V_2 = V_b - V_d \\ I'_a = I_1 & I'_b = I_2 \\ I'_c = -I_1 & I'_d = -I_2 \end{array}\right\} \tag{21}$$

where *a*, *b*, *c*, *d* represent the node numbers to which the two-port is connected in the network.

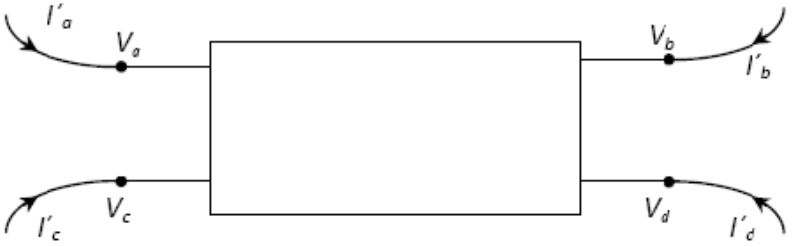


Figure 5. Definition of terminal node variables of a two‑port network.

If equation (20) is now combined with equations (21), it is readily shown that

$$\begin{bmatrix} I'_a \\ I'_b \\ I'_c \\ I'_d \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} & -Y_{11} & -Y_{12} \\ Y_{21} & Y_{22} & -Y_{21} & -Y_{22} \\ -Y_{11} & -Y_{12} & Y_{11} & Y_{12} \\ -Y_{21} & -Y_{22} & Y_{21} & Y_{22} \end{bmatrix} \times \begin{bmatrix} V_a \\ V_b \\ V_c \\ V_d \end{bmatrix} \tag{22}$$

Equation (22) represents the four-terminal indefinite admittance matrix relationship for the network of Figure 5. Implicit in this equation are the constraints $I'_c = -I'_a$ and $I'_d = -I'_b$. These constraints are imposed by the two-port convention for the current flow shown in Figure 4. In some cases (for example the transistor), where two of the terminals are common, the two-port can be represented as shown in Figure 6(a), which has the terminal equivalent shown in Figure 6(b). In this case, the terminal node constraint $V_c = V_d$ reduces equation (22) to

$$\begin{bmatrix} I'_a \\ I'_b \\ I'_c \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} & -(Y_{11}+Y_{12}) \\ Y_{21} & Y_{22} & -(Y_{21}+Y_{22}) \\ -(Y_{11}+Y_{12}) & -(Y_{21}+Y_{22}) & (Y_{11}+Y_{12}+Y_{21}+Y_{22}) \end{bmatrix} \times \begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} \quad (23)$$

At this point, an indefinite admittance matrix description of the two-port has been obtained, and this must now be incorporated into the nodal equations describing the network. As was remarked earlier, the nodal equations for the part of the network containing passive two-terminal components may be obtained first, and these must now be modified to include the effects of the two-port.

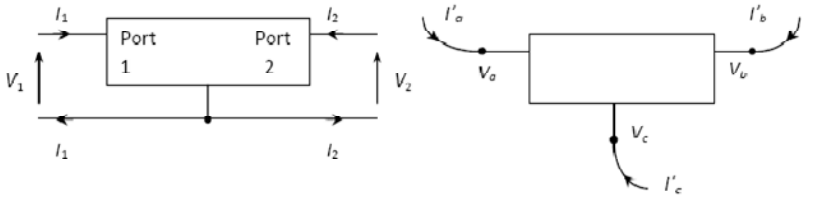


Figure 6. A general three‐terminal network.

Thus, referring to equation (18) the nodal equation pertaining to node *a* prior to the inclusion of the two-port under consideration may be written

$$I_a = \sum_{k=1}^{n} y_{ak} V_k \quad (24a)$$

Embedding the two-port within the remainder of the network will upset this current balance, and noting the relevant equation in equation (23), it is seen that a current I'_a must be added to the right hand side of equation (24a) to express Kirchhoff's current law at this node. Thus

$$I_a = [\sum_{k=1}^{n} y_{ak} V_k] + Y_{11} V_a + Y_{12} V_b - Y_{11} V_c - Y_{12} V_d \quad (24b)$$

Equation (24) shows that the entries y_{aa}, y_{ab}, y_{ac} and y_{ad} in the original indefinite admittance matrix will be modified according to

$$\begin{aligned} y_{aa} &\rightarrow y_{aa} + Y_{11} \\ y_{ab} &\rightarrow y_{ab} + Y_{12} \\ y_{ac} &\rightarrow y_{ac} - Y_{11} \\ y_{ad} &\rightarrow y_{ad} - Y_{12} \end{aligned}$$

and consideration of the remainder of the node currents indicates that similar modifications must be made to all the other elements y_{ij} (i,j = a, b, c, d). Now, if equation (22) and equation (23) are rewritten as

$$\begin{bmatrix} I'_a \\ I'_b \\ I'_c \\ I'_d \end{bmatrix} = \begin{bmatrix} y'_{aa} & y'_{ab} & y'_{ac} & y'_{ad} \\ y'_{ba} & y'_{bb} & y'_{bc} & y'_{bd} \\ y'_{ca} & y'_{cb} & y'_{cc} & y'_{cd} \\ y'_{da} & y'_{db} & y'_{dc} & y'_{dd} \end{bmatrix} \times \begin{bmatrix} V_a \\ V_b \\ V_c \\ V_d \end{bmatrix} \tag{25}$$

and

$$\begin{bmatrix} I'_a \\ I'_b \\ I'_c \end{bmatrix} = \begin{bmatrix} y'_{aa} & y'_{ab} & y'_{ac} \\ y'_{ba} & y'_{bb} & y'_{bc} \\ y'_{ca} & y'_{cb} & y'_{cc} \end{bmatrix} \times \begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} \tag{26}$$

then the embedding of the two-port into the formulation for the remainder of the network may be summarized as the element by element addition of the elements y'_{ij} of the matrix in equation (25) or (26) to the corresponding elements y_{ij} in the indefinite nodal admittance matrix, and this process would be repeated for all two-ports within the network.

It is possible now to formulate the indefinite nodal admittance matrix for a large range of practically occurring networks. The process starts normally for the passive elements as was done before. Then, the four-terminal nodal admittance matrix for each active device can be incorporated by adding the elements in the corresponding positions in the passive matrix. A word of warning is appropriate here. It was noted earlier that the development of the four-terminal indefinite admittance matrix from the two-port equations carried the implicit constraints $I'_c = -I'_a$ and $I'_d = -I'_b$. The simple formulation technique that has been discussed will not work, however, when the circuit

being incorporated into the matrix description does not satisfy these constraints, and care should be taken to ensure that this is not the case. Luckily, such cases occur infrequently. Furthermore, a little consideration shows that when two of the two-port terminals are common, as in Figure 6, the case does not occur at all in practical circuits [65]. With this in mind, larger models for bigger elements can be added to the system matrix provided they have two-port (or multi-port) admittance matrices forming macro-models which again need to satisfy some constraints to maintain the equations balance.

By trying some practical examples, it will be noted that the symmetry of the nodal admittance matrix, which is exhibited by passive networks, has been destroyed by the inclusion of the active device. Two other important observations may be made with regard to the indefinite nodal admittance matrix of all networks. Firstly, all the elements in any particular row sum to zero. This relates to the practical observation that if all the node voltages in a network were made equal, no currents would flow in the circuit, and this may be confirmed by setting all the voltages equal in each of the equations represented by equation (19). Secondly, all the elements in each column also sum to zero. This result is a direct consequence of Kirchhoff's current law, since all the currents entering a network must algebraically sum to zero. Again, reference to Figure 3 shows that if one sums all the currents I_1 to I_n by adding all the equations represented therein, the stated result is obtained.

An alternative way to express the influence of Kirchhoff's current law on the network is to say that the set of nodal equations based on all nodes in the network will be linearly dependent. This implies that equation (19b) may not be solved for the voltage vector $\boldsymbol{V}$. since the matrix $\boldsymbol{Y_i}$ will be singular. This surprising result may be rectified by noting that in the initial considerations, the elements of the node voltages vector $\boldsymbol{V}$ were not defined with respect to any specific internal reference. Practically, node voltages would be measured between each node and a reference node usually designated 'earth' or 'ground'. This ground node is usually assigned a potential of zero volts. Thus, if an *n*-node network is considered, and node *n* is designated as ground, equation (19a) reduces to

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_n \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n-1} \\ y_{21} & y_{22} & \cdots & y_{2n-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn-1} \end{bmatrix} \times \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_{n-1} \end{bmatrix} \tag{27}$$

Equation (27) represents *n* current equations in *n*–1 unknown voltages. The problem is thus over specified, and one current equation may be discarded. It

is most convenient to select the equation corresponding to node *n*, the ground node in this case, and so, finally, equation (27) may be reduced to

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_{n-1} \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n-1} \\ y_{21} & y_{22} & \cdots & y_{2n-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n-1,1} & y_{n-2,2} & \cdots & y_{n-1,n-1} \end{bmatrix} \times \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_{n-1} \end{bmatrix} \tag{28}$$

i.e.

$$\boldsymbol{I = YV} \tag{29}$$

where $\boldsymbol{Y}$ is known as the definite nodal admittance matrix. It should be noted that the vectors $\boldsymbol{I}$ and $\boldsymbol{V}$ in equation (29) are now reduced from *n*-vectors to (*n*−1)-vectors as a result of the above remarks.

The *definite admittance matrix* $\boldsymbol{Y}$ of equation (28) may be simply obtained from $\boldsymbol{Y_i}$ by deleting the row and column corresponding to the ground node, and, as will be shown, subsequent manipulation through equation (29) will then yield the unknown vector $\boldsymbol{V}$ and complete the analysis. One difficulty of the aforementioned analysis procedure is that of incorporating voltage sources. This is a constraint on unknown vector that has to be satisfied during the formulation. The most straightforward way of dealing with voltage sources is by using Norton's theorem to convert the voltage source to a current source. In Figure 7(a), an ideal voltage source is shown. In Figure 7(b), two resistors are introduced in series with the source. These resistors are of opposite signs but equal in magnitude so that they do not alter the operation of the circuit. Finally, in Figure 7(c) Norton's theorem has been used to transfer the circuit to a current source with shunt resistance. This equivalent circuit can be incorporated into the network and the difficulty of formulating the equations vanishes. An extra node has also been added but it is a reasonable price to pay for the ease of analysis. The resistance *R* may be chosen arbitrarily to satisfy any computational requirements so it is common to set its value to 1. Thus voltage sources of all kinds, even though they depend on another branch variable in a *nonlinear* fashion, may be coped with using the nodal approach.

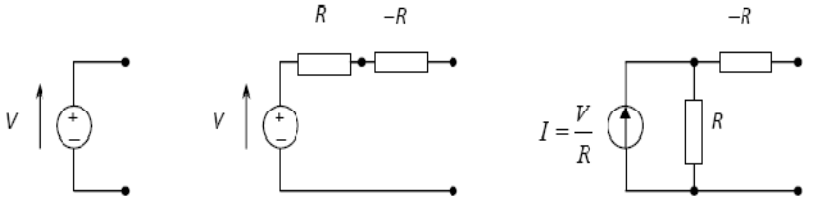


Figure 7. Application of Norton's theorem to a general voltage source.

4.2. TABLEAU AND COMPACTED MODIFIED NODAL ANALYSIS (CMNA) METHODS

The formulation methods introduced in the previous section are quite efficient and have been used successfully in many applications, but they cannot handle all ideal elements (especially those which have no two-port short circuit admittance parameters). In addition nodal analysis will only provide the nodal voltages. Thus if currents or transfer functions are required, back substitution is needed to get the required currents or transfer functions. Finally the generated system matrix may be ill conditioned while there is a trivial solution to the system. Such cases occur when the circuit involves loops of voltage sources and/or ideal current sources in series. Thus a modification to the method is needed to incorporate additional variables in the vector of unknowns that will eliminate the need for back substitution and provide sufficient conditions in the system matrix to avoid having it turn ill-conditioned when there is a trivial solution. To avoid the restrictions, general formulation methods are introduced in this section.

The formulations discussed before can all be derived from a general formulation called the *tableau*. In this formulation, *all* equations describing the network are collected into one large matrix equation, involving the **KVL**, **KCL**, and the constitutive equations, **CE**. All branch currents, all branch voltages and all nodal voltages are retained as unknown variables of the problem. Thus, the formulation is most general (everything is available after the solution) but leads to large system matrices.

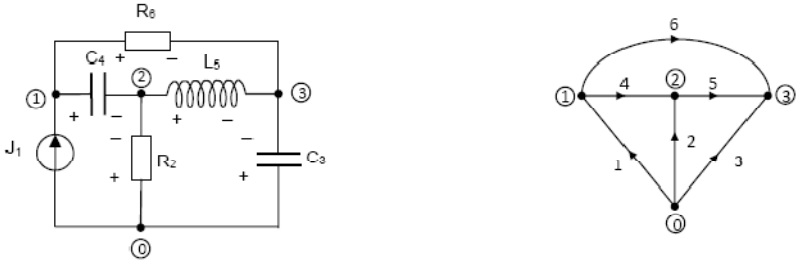


Figure 8. A typical example of network and its oriented graph representation.

We will first comment on the most convenient type of tableau. For initial considerations let the network have *b* branches; *n*+1 nodes; *R*, *G*, *L*, *C* elements; and sources. The topological properties of such a network can be expressed by means of the *incidence matrix* ***A***. This is a matrix representation of the **KCL** for the oriented graph of the network. As discussed in section 2 an *oriented graph* of a network is a directed graph whose weights are all unity. For example, consider the simple network shown in Figure 8 with its oriented graph representation.

Let us write the **KCL** equations with the edge orientations as indicated. A current flowing away from a node will be considered as *positive*. Then

$$\begin{aligned} 1:&\ -i_1 + i_4 + i_6 = 0 \\ 2:&-i_2 - i_4 + i_5 = 0 \\ 3:&-i_3 - i_5 - i_6 = 0 \end{aligned}$$

This can be written in matrix form:

$$\boldsymbol{Ai} = \mathbf{0} \tag{30}$$

where ***A*** is the incidence matrix and, for this example,

$$\boldsymbol{A} = \begin{array}{c} \\ \text{nodes} \\ \downarrow \end{array} \begin{array}{c} \text{edges} \rightarrow \\ \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 1 \\ 0 & -1 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix} \end{array} \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}$$

It has *n* rows and *b* columns, *n* being the number of ungrounded nodes and *b* the number of edges in the graph. It can be shown that equation (30) is always valid [6]. Furthermore, it can be written in a generalized form as

$$\boldsymbol{A I_b = 0} \tag{31}$$

where $\boldsymbol{I_b}$ represents the branch currents. It can also be shown that the branch voltages $\boldsymbol{V_b}$ and the node voltages $\boldsymbol{V_n}$ are related by

$$\boldsymbol{V_b - A^t V_n = 0} \tag{32}$$

This is another form of stating **KVL** in the circuit.

Finally, the general representation describing all possible constitutive equations **CE** has the following form

$$\begin{matrix}\text{currents} \\ \text{voltages}\end{matrix} \begin{bmatrix} \boldsymbol{Y_1} \\ \boldsymbol{K_2} \end{bmatrix} \boldsymbol{V_b} + \begin{bmatrix} \boldsymbol{K_1} \\ \boldsymbol{Z_2} \end{bmatrix} \boldsymbol{I_b} = \begin{bmatrix} \boldsymbol{W_{b1}} \\ \boldsymbol{W_{b2}} \end{bmatrix}$$

where $\boldsymbol{Y}_1$ and $\boldsymbol{Z}_2$ represent admittances and impedances, respectively; $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ contain dimensionless constants; and $\boldsymbol{W_{b1}}$, and $\boldsymbol{W_{b2}}$ include the independent current and voltage sources, as well as the influence of initial conditions on capacitors and inductors. For notational compactness, the following form will be used:

$$\boldsymbol{Y_b V_b + Z_b I_b = W_b} \tag{33}$$

In all subsequent formulations, it is highly advisable to enter capacitors in admittance form and inductors in impedance form to keep the variable *s* in the numerator. Since the Laplace transform variable *s* is equivalent to the differentiation operator, a set of algebraic differential equations will be obtained when performing time domain analysis. Table 1 indicates the choices of Y_b, Z_b, and W_b for various two-terminal elements.

Table 1 The choices of Y_b, Z_b, and W_b for various two-terminal elements

Element	Constitutive Equation (**CE**)	Value of Y_b	Value of Z_b	Value of W_b
Resistor	$V_b-R_bI_b=0$	1	$-R_b$	0
Conductor	$G_bV_b-I_b=0$	G_b	-1	0
Capacitor	$sC_bV_b-I_b=C_bV_0$	sC_b	-1	C_bV_0
Inductor	$V_b-sL_bI_b=-L_bI_0$	1	$-sL_b$	$-L_bI_0$
Voltage source	$V_b=E_b$	1	0	E_b
Current source	$I_b=J_b$	0	1	J_b

Equations (31)-(33) can be collected, for instance, in the following sequence:

$$\boldsymbol{V_b - A^t V_n = 0}$$
$$\boldsymbol{Y_b V_b + Z_b I_b = W_b}$$
$$\boldsymbol{A I_b = 0}$$

and put into one matrix equation

$$\begin{array}{c} \\ b\updownarrow \\ b\updownarrow \\ n\updownarrow \end{array} \begin{array}{c} \begin{array}{ccc} b\leftrightarrow & b\leftrightarrow & n\leftrightarrow \end{array} \\ \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{-A^t} \\ \boldsymbol{Y_b} & \boldsymbol{Z_b} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \end{array} \times \begin{bmatrix} \boldsymbol{V_b} \\ \boldsymbol{I_b} \\ \boldsymbol{V_n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{W_b} \\ \boldsymbol{0} \end{bmatrix} \tag{34}$$

Or, in general,

$$\boldsymbol{TX = W} \tag{35}$$

Where $\boldsymbol{T}$, $\boldsymbol{X}$ and $\boldsymbol{W}$ represent the combined system matrix, unknowns vector and excitation. The arrangement indicated in equation (34) has square sub-matrices on the diagonal. In this tableau formulation, the element numbering can be completely arbitrary. For computer implementation, the SPICE-package input format is usually adopted in numbering the elements.

Until now, we have been discussing the tableau for two-terminal elements only. In order to generalize the tableau to any element the two-port element representations have to be considered. In such a representation, each

port of a two-port network is represented by a constitutive equation and two constitutive equations must therefore be given. A lookup table can thus be programmed to identify each element in the circuit and return its constitutive equation(s). To this end, a summary of the most important ideal elements is given in Table 2.

The tableau formulation discussed so far has mainly theoretical importance. Many ideal two-ports introduce redundant variables. For instance, the input currents of the voltage-controlled voltage source (VCVS) and the voltage-controlled current source (VCCS), or the input branch voltages of the current-controlled voltage source (CCVS) and the current-controlled current source (CCCS) are known to be zero. However, they are kept in this formulation as variables. The tableau formulation has yet another problem: the resulting matrices are quite large and sparse matrix solvers are needed. Unfortunately, the structure of the matrix is such that coding these routines is complicated. A re-structuring algorithm must be implemented before the linear solution techniques can be of any use to us.

A major development step to the tableau method is to eliminate all redundant variables. These can be eliminated by the use of a pivotal condensation procedure (as is explained later) to produce a compacted nodal admittance matrix that can be inverted easily. However, such elimination would take quite a long time especially for symbolic matrices, not to mention the huge storage requirements for the sparse symbolic matrices. A more practical approach is to program a lookup table for every element in the network. This table has conditioned link-lists that will test which variables of the element are actually needed in the final compacted matrix and introduce the element in a way so as to eliminate the redundant variables during the formulation. This method is called the compacted modified nodal analysis CMNA (or the element-stamp method [66]) and it represents an automatic technique to construct the nodal admittance matrix.

The compacted modified nodal analysis method is a very nice and easy way to illustrate the impact of each element on the matrix since it constitutes going through each branch of the circuit and adding its contribution to the system matrix in the appropriate positions. To understand the method, the following elementary examples are introduced. However, programming the method requires a full lookup table with explicit statement of the conditions required to eliminate any of the redundant variables.

Table 2. Constitutive equations of ideal elements for the tableau formulation

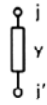
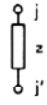
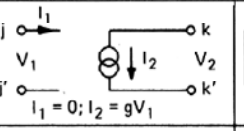
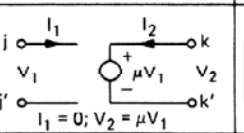
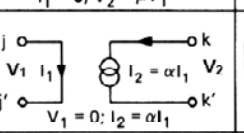
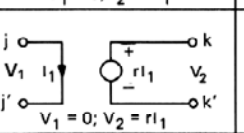
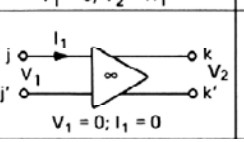
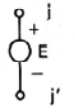
ELEMENT	SYMBOL	CONSTITUTIVE EQUATIONS
ADMITTANCE		$yV - I = 0$
IMPEDANCE		$V - zI = 0$
NULLATOR		$I = 0$ $V = 0$
NORATOR		I, V ARBITRARY (NO CONSTITUTIVE EQUATIONS)
VCT		$\begin{bmatrix}0 & 0\\ g & 0\end{bmatrix}\begin{bmatrix}V_1\\V_2\end{bmatrix}+\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{bmatrix}I_1\\I_2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$
VVT		$\begin{bmatrix}0 & 0\\ \mu & 0\end{bmatrix}\begin{bmatrix}V_1\\V_2\end{bmatrix}+\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}I_1\\I_2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$
CCT		$\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}V_1\\V_2\end{bmatrix}+\begin{bmatrix}0 & 0\\ \alpha & -1\end{bmatrix}\begin{bmatrix}I_1\\I_2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$
CVT		$\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\begin{bmatrix}V_1\\V_2\end{bmatrix}+\begin{bmatrix}0 & 0\\ r & 0\end{bmatrix}\begin{bmatrix}I_1\\I_2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$
OPAMP		$\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}V_1\\V_2\end{bmatrix}+\begin{bmatrix}0 & 0\\ 1 & 0\end{bmatrix}\begin{bmatrix}I_1\\I_2\end{bmatrix}=\begin{bmatrix}0\\0\end{bmatrix}$

Table 2. (Continued).

ELEMENT	SYMBOL	CONSTITUTIVE EQUATIONS
CURRENT SOURCE		I = J
VOLTAGE SOURCE		V = E
OPEN CIRCUIT		I = 0
SHORT CIRCUT		V = 0

Example: Consider the general admittance *y* shown in Figure 9.

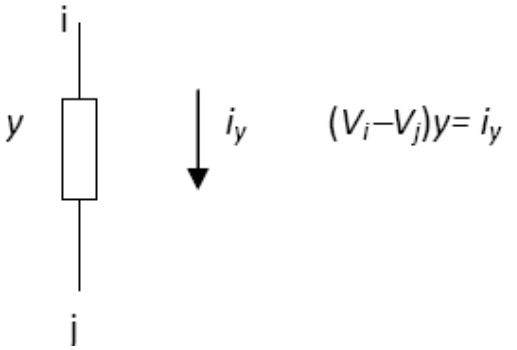


Figure 9. A general admittance.

Assuming that the ***Y*** matrix is already generated for the other branches, the impact of this admittance (following the tableau formulation) on the system matrix is to add an additional row and column corresponding to the new system variable i_y as shown below

$$\begin{array}{c} \\ i \\ j \\ i_y \end{array} \begin{array}{c} \begin{array}{ccc} i & j & i_y \end{array} \\ \begin{bmatrix} y_{ii} & y_{ij} & 1 \\ y_{ji} & y_{jj} & -1 \\ y & -y & -1 \end{bmatrix} \end{array} \times \begin{bmatrix} V_i \\ V_j \\ i_y \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_i \\ w_j \\ 0 \end{bmatrix} \end{array} \tag{36}$$

Now, if i_y is not a solution variable then it has to be eliminated from the system matrix to generate a new matrix that is compacted (condensed) with respect to the axis i_y. Carrying out Kron's reduction described in equation (10) one gets

$$\begin{array}{c} \\ i \\ j \end{array} \begin{array}{c} \begin{array}{cc} i & j \end{array} \\ \begin{bmatrix} y_{ii} + y & y_{ij} - y \\ y_{ji} - y & y_{jj} + y \end{bmatrix} \end{array} \times \begin{bmatrix} V_i \\ V_j \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_i \\ w_j \end{bmatrix} \end{array} \tag{37}$$

Equations (36) and (37) are the conditioned stamps for the admittance shown in Figure 9 and they can be programmed in a lookup table easily. However, although equation (37) is the simplest, it is not the only possible elimination that can be carried out on the system matrix of equation (36). Connecting an admittance between node *i* and node *j* constrains the current through the branch *i*-*j* and thus affecting three system equations at the same time. Thus, if i_y is to be eliminated using an equation other than the i_y equation (say the j^{th} equation) then the elimination can be carried out using Kron's reduction on the following permutation of equation (36)

$$\begin{array}{c} \\ i \\ i_y \\ j \end{array} \begin{array}{c} \begin{array}{ccc} i & j & i_y \end{array} \\ \begin{bmatrix} y_{ii} & y_{ij} & 1 \\ y & -y & -1 \\ y_{ji} & y_{jj} & -1 \end{bmatrix} \end{array} \times \begin{bmatrix} V_i \\ V_j \\ i_y \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_i \\ 0 \\ w_j \end{bmatrix} \end{array} \tag{38}$$

The resulting system matrix is then:

$$\begin{array}{c} \\ i \\ j \end{array} \begin{array}{c} \begin{array}{cc} i & j \end{array} \\ \begin{bmatrix} y_{ii} + y_{ji} & y_{ij} + y_{jj} \\ y_{ji} - y & y_{jj} + y \end{bmatrix} \end{array} \times \begin{bmatrix} V_i \\ V_j \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_i + w_j \\ w_j \end{bmatrix} \end{array} \tag{39}$$

which is just a linear transformation of equation (37). Obviously, the stamp of equation (37) is much easier to implement than the procedure resulting in

equations (39). Yet, this simple example shows how the element stamps and stamping procedures can be derived from Kron's reduction depending on the variables one wants to keep as solution variables without actually performing the reduction.

Stamping procedures have similar properties and some of them were used in the past (e.g. Nathan's method to analyze constrained op-amp networks which is really Kron's reduction in disguise [11]). However, their practical advantage occurs only when they can be programmed easily. Therefore, the art of deriving a stamping procedure lies in its programming suitability. Thus, all the subsequent derivations of stamping procedures will be carried out keeping this in mind.

Example: Consider the independent voltage source E shown in Figure 10.

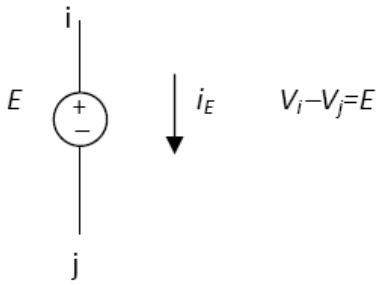


Figure 10. A general independent voltage source.

Again, assuming that the $\boldsymbol{Y}$ matrix is already generated for the other branches, the impact of this voltage source (following the tableau formulation) on the system matrix is to add an additional row and column corresponding to the new system variable i_E as shown below

$$\begin{array}{c} \\ i \\ j \\ i_E \end{array} \begin{array}{c} \begin{array}{ccc} i & j & i_E \end{array} \\ \begin{bmatrix} y_{ii} & y_{ij} & 1 \\ y_{ji} & y_{jj} & -1 \\ 1 & -1 & 0 \end{bmatrix} \end{array} \times \begin{bmatrix} V_i \\ V_j \\ i_E \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_i \\ w_j \\ E \end{bmatrix} \end{array} \tag{40}$$

Now, if i_E is not a solution variable then it has to be eliminated from the system matrix. However, the axis i_E has a zero pivotal element which makes

it impossible to eliminate both the i_E row and column. Carrying out Kron's reduction on the j^{th} equation, the following stamping procedure is obtained:

- Add row j to row i (including the right-hand side element) and put the result in row i,
- Replace row j by the elements of row i_E given in equation (40).

This results in the following system matrix

$$\begin{matrix} & i & j & \text{RHS} \\ i \\ j \end{matrix} \begin{bmatrix} y_{ii}+y_{ji} & y_{ij}+y_{jj} \\ 1 & -1 \end{bmatrix} \times \begin{bmatrix} V_i \\ V_j \end{bmatrix} = \begin{bmatrix} w_i+w_j \\ E \end{bmatrix} \tag{41}$$

Other elimination combinations can also be considered in this respect. Connecting a voltage source between node i and node j constrains the voltage on both nodes so that the voltage of one node will be a linear function of the voltage on the other node. Furthermore, fixing the voltage difference between two nodes constrains the current as well. Therefore, a reduction of one of the node voltages is also possible. Eliminating axis V_j on equation (40) results in the following stamping procedure:

- Add column j to column i and put the result in column i,
- Add column j multiplied by E to the right-hand side,
- Replace column j by the elements of column i_E given in equation (40).

This results in the following system matrix

$$\begin{matrix} & i & i_E & \text{RHS} \\ i \\ j \end{matrix} \begin{bmatrix} y_{ii}+y_{ij} & 1 \\ y_{ji}+y_{jj} & -1 \end{bmatrix} \times \begin{bmatrix} V_i \\ i_E \end{bmatrix} = \begin{bmatrix} w_i+y_{ij}E \\ w_j+y_{jj}E \end{bmatrix} \tag{42}$$

Finally, two axes can be eliminated together resulting in a new stamping procedure. Eliminating both the V_j axis and the i_E axis on equation (40) results in the following stamping procedure:

- Add row j to row i and put the result in row i,
- Add column j to column i and put the result in column i,
- Add column j multiplied by E to the right-hand side,
- Remove the j^{th} row and column.

This results in the following system matrix

$$\begin{matrix} & i & \\ i & \left[y_{ii} + y_{ij} + y_{ji} + y_{jj}\right] \times [V_i] = \left[w_i + w_j + y_{jj}E + y_{ij}E\right] \end{matrix} \quad \text{RHS} \tag{43}$$

It must be emphasized that the system matrix shown in equation (43) is not a 1x1 matrix, but rather a big matrix of which only the (*i*-*i*) element is shown.

It is evident from the example that the number of the stamps and stamping procedures for one element can grow quite large depending on the possible elimination combinations. Obviously, this results in many if-conditions and thus a long processing time. Yet, the practical implementation shows that the time required to eliminate the redundant variables in a symbolic sparse tableau matrix is greater than the time required to go through all the if-conditions by about an order of magnitude. This verifies the efficiency of this approach in generating the compacted system matrix.

Example: Consider the ideal operational amplifier (op-amp) shown in Figure 11.

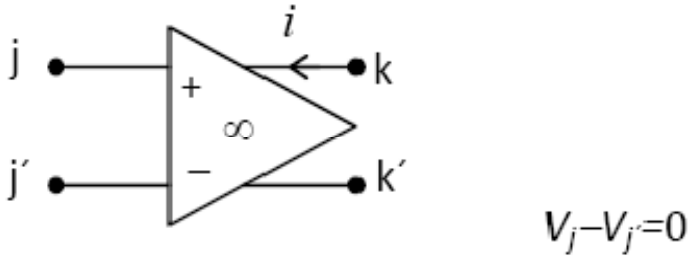


Figure 11. An ideal (infinite gain) operational amplifier.

Again assuming that the ***Y*** matrix is already generated for the other branches, the impact of this device on the system matrix is to add an additional row and column corresponding to the new system variable *i* as shown below

$$\begin{array}{c} \\ j \\ j' \\ k \\ k' \\ i \end{array} \begin{array}{c} \begin{array}{ccccc} j & j' & k & k' & i \end{array} \\ \begin{bmatrix} y_{jj} & y_{jj'} & y_{jk} & y_{jk'} & 0 \\ y_{j'j} & y_{j'j'} & y_{j'k} & y_{j'k'} & 0 \\ y_{kj} & y_{kj'} & y_{kk} & y_{kk'} & 1 \\ y_{k'j} & y_{k'j'} & y_{k'k} & y_{k'k'} & -1 \\ 1 & -1 & 0 & 0 & 0 \end{bmatrix} \end{array} \times \begin{bmatrix} V_j \\ V_{j'} \\ V_k \\ V_{k'} \\ i \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_j \\ w_{j'} \\ w_k \\ w_{k'} \\ 0 \end{bmatrix} \end{array} \tag{44}$$

The elimination combinations and stamping procedures for this device can be summarized as follows:

1. If one of the input voltages (say V_j) is not needed but i is a solution variable then the stamping procedure is:

- Add column j' to column j and put the result in column j,
- Replace the j'^{th} column by the elements of column i given in equation (44) and delete row i.

The resulting system matrix is

$$\begin{array}{c} \\ j \\ j' \\ k \\ k' \end{array} \begin{array}{c} \begin{array}{cccc} j & i & k & k' \end{array} \\ \begin{bmatrix} y_{jj} + y_{jj'} & 0 & y_{jk} & y_{jk'} \\ y_{j'j} + y_{j'j'} & 0 & y_{j'k} & y_{j'k'} \\ y_{kj} + y_{kj'} & 1 & y_{kk} & y_{kk'} \\ y_{k'j} + y_{k'j'} & -1 & y_{k'k} & y_{k'k'} \end{bmatrix} \end{array} \times \begin{bmatrix} V_j \\ i \\ V_k \\ V_{k'} \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_j \\ w_{j'} \\ w_k \\ w_{k'} \end{bmatrix} \end{array} \tag{45}$$

2. If the current i is not a solution variable but both input voltages are, then the stamping procedure is:

- Add row k' to row k and put the result in row k,
- Replace the k'^{th} row by the elements of row i given in equation (44).

The resulting system matrix is:

$$\begin{array}{c} \\ j \\ j' \\ k \\ i \end{array}\begin{array}{c} \begin{array}{cccc} j & j' & k & k' \end{array} \\ \begin{bmatrix} y_{jj} & y_{jj'} & y_{jk} & y_{jk'} \\ y_{j'j} & y_{j'j'} & y_{j'k} & y_{j'k'} \\ y_{kj}+y_{k'j} & y_{kj'}+y_{k'j'} & y_{kk}+y_{k'k} & y_{kk'}+y_{k'k'} \\ 1 & -1 & 0 & 0 \end{bmatrix} \end{array} \times \begin{bmatrix} V_j \\ V_{j'} \\ V_k \\ V_{k'} \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_j \\ w_{j'} \\ w_k+w_{k'} \\ 0 \end{bmatrix} \end{array} \quad (46)$$

3.If neither the current i nor one of the input voltage (e.g. $V_{j'}$) are solution variables then the stamping procedure is:

- Add row k' to row k, put the result in row k and delete row k',
- Add column j' to column j, put the result in column j and delete column j',

The resulting system matrix in this case is:

$$\begin{array}{c} \\ j \\ j' \\ k \end{array}\begin{array}{c} \begin{array}{ccc} j & k & k' \end{array} \\ \begin{bmatrix} y_{jj}+y_{jj'} & y_{jk} & y_{jk'} \\ y_{j'j}+y_{j'j'} & y_{j'k} & y_{j'k'} \\ y_{kj}+y_{k'j}+y_{kj'}+y_{k'j'} & y_{kk}+y_{k'k} & y_{kk'}+y_{k'k'} \end{bmatrix} \end{array} \times \begin{bmatrix} V_j \\ V_k \\ V_{k'} \end{bmatrix} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} w_j \\ w_{j'} \\ w_k+w_{k'} \end{bmatrix} \end{array} \quad (47)$$

Obviously, the op-amp constrains the input nodes but not the output nodes so that only one of the input node voltages can be eliminated. Therefore, the procedures leading to equations (45) through (47) are the basic three stamping procedures of the op-amp. The forms of equations (45)-(47) might be different depending on which of the input node voltages (V_j or $V_{j'}$) is to be eliminated but the general steps are the same.

4.3. PRACTICAL APPLICATION OF THE CMNA METHOD

The above three examples served to introduce the concept of stamping in generating a compacted system matrix using three elementary devices. Other elements like the general two-port network and the controlled sources have more complicated stamps with many conditions. A comprehensive list of all the conditions is beyond the scope of this book but a summary of the most common stamps is shown in Table 3.

Table 3. Stamp models of linear circuit elements

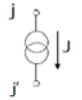
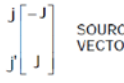
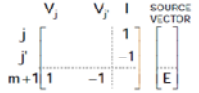
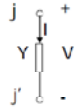
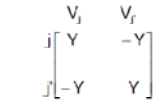
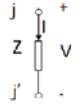
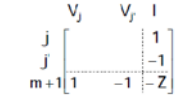
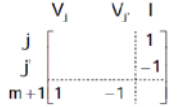
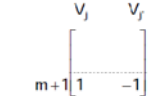
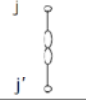
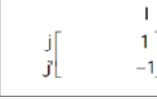
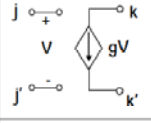
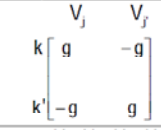
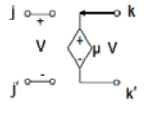
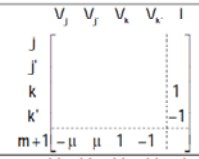
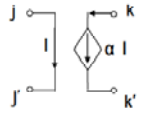
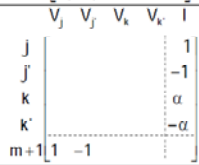
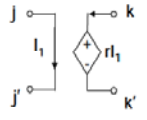
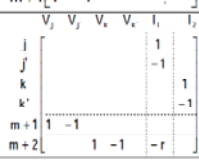
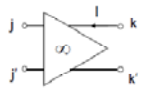
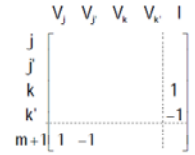
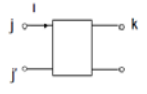
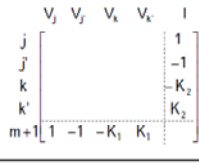
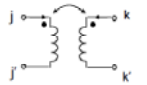
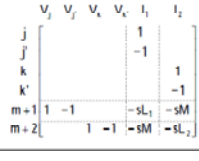
Element	Symbol	Stamp	Equations
Current Source			$I_j = J$ $I_{j'} = -J$
Voltage Source			$V_j - V_{j'} = E$ $I_j = I$ $I_{j'} = -I$
Admittance			$I_j = Y(V_j - V_{j'})$ $I_{j'} = -Y(V_j - V_{j'})$
Impedance			$V_j - V_{j'} - ZI = 0$ $I_j = -I_j' = I$
Open Circuit			$V = V_j - V_{j'}$
Short Circuit			$V_j - V_{j'} = 0$ $I_j = I$ $I_{j'} = -I$
Nullator			$V_j - V_{j'} = 0$ $I_j = I_j' = 0$
Norator			V and I are arbitrary

Table 3. (Continued).

Element	Symbol	Stamp	Equations
VCT			$I_j = 0$ $I_{j'} = 0$ $I_k = g(V_j - V_{j'})$ $I_{k'} = -g(V_j - V_{j'})$
VVT			$-\mu V_j + \mu V_{j'} + V_k - V_{k'} = 0$ $I_k = I$ $I_{k'} = -I$
CCT			$V_j - V_{j'} = 0$ $I_j = -I_{j'} = I$ $I_k = -I_{k'} = \alpha I$
CVT			$V_j - V_{j'} = 0$ $V_k - V_{k'} - rI_1 = 0$ $I_j = -I_{j'} = I_1$ $I_k = -I_{k'} = I_2$
Operational Amplifier			$V_j - V_{j'} = 0$ $I_k = -I_{k'} = I$
Converter			$V_j - V_{j'} - K_1 V_k + K_1 V_{k'} = 0$ $I_j = -I_{j'} = I$ $I_k = -I_{k'} = -K_2 I$ For ideal transformer $K_1 = K_2 = n$
Transformer			$V_j - V_{j'} - sL_1 I_1 - sMI_2 = 0$ $V_k - V_{k'} - sMI_1 - sL_2 I_2 = 0$ $I_j = -I_{j'} = I_1$ $I_k = -I_{k'} = I_2$

In practice the actual linear operation described by the stamping procedure must be postponed until all the elements have been added. However if the system is described using the simplified representation of equations (1) and (3), the linear operations can be carried out on the ***P*** matrix

as it would not affect the element values. Yet the actual row eliminations must be postponed until all the elements are added to avoid eliminating an element whose effect on the system matrix is yet to be added. Even when all the conditions set forth are met, consistency must be maintained and verified in choosing the solution variables as an inconsistent set of solution variables may lead to conflicting stamping procedures that could not be carried out.

In the following examples, use will be made of the method in analyzing typical circuits. Through these, a try will be made to outline the general steps of the analysis procedure and to verify its efficiency in generating a small dense set of linear equations.

Example: Consider the following active network shown in Figure 12. The first thing that has to be done is to decide which variables are to be considered as solution variables. Assuming that only the node voltages V_2 and V_4 are needed while all the other variables are to be eliminated, a preliminary $\boldsymbol{Y}$-matrix for the network has to be generated. This matrix has the size of 4x4 since there are 4 ungrounded nodes. It can be generated for the passive elements using the stamp of equation (37) as shown below:

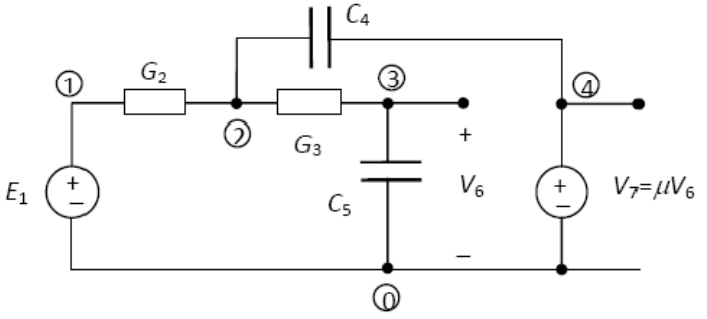


Figure 12. An active network with a VCVS.

$$\boldsymbol{Y} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} G_2 & -G_2 & 0 & 0 \\ -G_2 & G_2+G_3+sC_4 & -G_3 & -sC_4 \\ 0 & -G_3 & G_3+sC_5 & 0 \\ 0 & -sC_4 & 0 & sC_4 \end{bmatrix} \end{array} \tag{48}$$

The next step is to add the influence of the independent voltage source keeping in mind that one of its nodes is grounded and that neither its node voltages nor its current are solution variables. The following stamping procedure was used:

- Subtract column 1 multiplied by E_1 from the right-hand side,
- Remove row and column 1.

The resulting system matrix and the right-hand side after this will be:

$$\boldsymbol{Y} = \begin{matrix} \\ 2 \\ 3 \\ 4 \end{matrix} \begin{matrix} \begin{matrix} 2 & 3 & 4 \end{matrix} \\ \begin{bmatrix} G_2 + G_3 + sC_4 & -G_3 & -sC_4 \\ -G_3 & G_3 + sC_5 & 0 \\ -sC_4 & 0 & sC_4 \end{bmatrix} \end{matrix}, \boldsymbol{W} = \begin{matrix} \text{RHS} \\ \begin{bmatrix} G_2E_1 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \quad (49)$$

Finally, the impact of the voltage-controlled voltage source is added keeping in mind that one of its input nodes is grounded and that neither its input node voltages nor its output current are solution variables. To this end, the following stamping procedure was used:

- Delete row 4,
- Add column 3 multiplied by $1/\mu$ to column 4 and put the result in column 4,
- Delete column 3.

The final system matrix will be

$$\begin{matrix} \\ 2 \\ 3 \end{matrix} \begin{matrix} \begin{matrix} 2 & 4 \end{matrix} \\ \begin{bmatrix} G_2 + G_3 + sC_4 & -sC_4 - G_3/\mu \\ -G_3 & G_3/\mu + sC_5/\mu \end{bmatrix} \end{matrix} \times \begin{bmatrix} V_2 \\ V_4 \end{bmatrix} = \begin{matrix} \text{RHS} \\ \begin{bmatrix} G_2E_1 \\ 0 \end{bmatrix} \end{matrix} \quad (50)$$

To compare various formulations, it is convenient to introduce the matrix density D and the formulation efficiency F, defined respectively as follows [6], [66]:

$$D = \frac{\text{Number of nonzero elements in matrix}}{\text{Total number of elements in the matrix}}, F = \frac{\text{Number of solution variables}}{\text{Number of system variables}} \quad (51)$$

For this example, the matrix has the size 2x2 for two solution variables and there are 4 nonzero entries. Thus the density is *D*=100% and the efficiency is *F*=100%. It can be verified that the tableau formulation produces a matrix of size 18x18 with a density as low as 12% and an efficiency as low as 11% [6].

Example: Consider the following active network shown in Figure 13. Assume the solution variables are only the node voltages V_1 and V_4, while all the other variables are to be eliminated. The preliminary $\boldsymbol{Y}$-matrix has the size of 5x5 since there are 5 ungrounded nodes. It can be generated for the passive elements as given below:

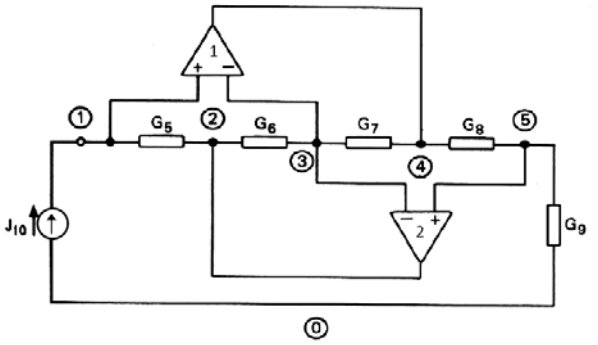


Figure 13. A generalized impedance converter.

$$\boldsymbol{Y} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} G_5 & -G_5 & 0 & 0 & 0 \\ -G_5 & G_5+G_6 & -G_6 & 0 & 0 \\ 0 & -G_6 & G_6+G_7 & -G_7 & 0 \\ 0 & 0 & -G_7 & G_7+G_8 & -G_8 \\ 0 & 0 & 0 & -G_8 & G_8+G_9 \end{bmatrix} \end{array}, \boldsymbol{W} = \begin{array}{c} \text{RHS} \\ \begin{bmatrix} J_{10} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array} \tag{52}$$

Next, the effects of the op-amps are added. The op-amps in this example will be assumed ideal with infinite gains and input impedances. This assumption introduces a virtual ground between the inputs of the op-amps. The stamping procedure used to add the first op-amp is:

- Add column 3 to column 1 and put the result in column 1,
- Delete row 4 and column 3.

On the other hand, the stamping procedure used to add the second op-amp is:

- Add column 5 to column 1 (since column 1 and column 3 are virtually short-circuited) and put the result in column 1,
- Delete row 2 and column 5.

The resulting system matrix equation will be:

$$\begin{matrix} \\ 1 \\ 3 \\ 5 \end{matrix}\begin{matrix} \begin{matrix} 1 & 2 & 4 \end{matrix} \\ \begin{bmatrix} G_5 & -G_5 & 0 \\ G_6+G_7 & -G_6 & -G_7 \\ G_8+G_9 & 0 & -G_8 \end{bmatrix} \end{matrix} \times \begin{bmatrix} V_1 \\ V_2 \\ V_4 \end{bmatrix} = \begin{matrix} \text{RHS} \\ \begin{bmatrix} J_{10} \\ 0 \\ 0 \end{bmatrix} \end{matrix} \tag{53}$$

For this example, the matrix has the size 3x3 for three solution variables and there are 7 nonzero entries. Thus the density is D=77.78% and the efficiency is F=66.67%. It can be verified that the tableau formulation produces a matrix of size 25x25 with a density as low as 9.12% and an efficiency as low as 8% [6].

It is evident from the above example that not all unwanted system variables can be reduced with the stamping procedure. The formulation efficiency is not always 100% unless the proposed method is combined with a matrix reduction technique (like Kron's reduction) to reduce the unconstrained node voltages. In such a case, the reduced system matrix will be obtained. Although the later provides the highest formulation efficiency, it is not the best choice always. The problem lies in the reduction itself. It involves the division of each new term by the pivotal element, which complicates the elements in the new matrix and increases the number of symbolic terms. The complexity increases drastically with each application of the reduction. Therefore, the best utilization of the reduction is to apply it whenever there is a pivotal element of unity. Of course, this means

sacrificing some of the formulation efficiency. However, the final gain surpasses any inefficiency that might be introduced. The benefits of this technique in matrix formulation can be summarized as follows:

1. Ability to use sparse techniques to overcome the possible inefficiency in the system matrix,
2. Reduction of the overall solution time,
3. Reduction of the overall memory requirement, which provides an ability to solve larger systems,
4. Flexibility of the technique to generate system matrices that might be as big as the tableau matrices or as small as the matrices formulated by hand. This, of course, offers solutions that range from transfer functions to voltage and current solutions for each element in the network,
5. Exceptional control over the generated matrix density and efficiency in such a way so as not to disturb the improved time performance of the method.
6. Macro-models of section 4.2 can be easily converted with proper elimination of non-system variables into Macro-stamps with the required complexity level. This provides element stamps for more complicated elements like transistors and non-ideal op-amps

4.4. IMPLEMENTATION CONSIDERATIONS

Practical implementation of the CMNA for symbolic circuit analysis requires further special considerations. Techniques for representing the symbolic matrix need to be applied during the software development. To that end we may assume without loss of generality that, at a given operating point, an electronic circuit is described by the modified nodal equations with the coefficient matrix

$$\boldsymbol{A = PYQ} \tag{54}$$

In the same way used to define equation (1). The coefficient matrix $\boldsymbol{A}$ can be directly obtained from the element equations using the "stamp approach" as described in the previous section [6]. These element models are directly inserted and their symbolic values are added to other element values at the corresponding locations of the modified nodal matrix. As before $\boldsymbol{P}$ and $\boldsymbol{Q}$ are

the topological matrices that indicate location of the element's parameter value (element value) in the modified nodal coefficient matrix $\boldsymbol{A}$. Except for the reference node (ground), typically all element values are placed at most at 4 locations in $\boldsymbol{A}$ using the stamp approach as can be seen in Table 3. More specifically, for each element its symbolic value Y is placed, e.g. equation (37), on the intersection of rows i and j and columns k and l as in the following stamp matrix $\boldsymbol{S_y}$ (all remaining elements of the stamp matrix are zero):

$$\boldsymbol{S_y} = \begin{array}{c} \\ i \\ \\ j \\ \end{array} \begin{array}{c} \begin{array}{cccc} . & k & . & l \end{array} \\ \begin{bmatrix} . & Y & . & -Y \\ . & . & . & . \\ . & -Y & . & Y \end{bmatrix} \end{array} \qquad (55)$$

Each circuit element has a single row and column in topological matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ respectively that represent information about element's interconnections. More specifically, for an element described by the stamp of equation (55), matrix $\boldsymbol{P}$ contains 1 and -1 in rows i and j and matrix $\boldsymbol{Q}$ contains 1 and -1 in columns k and l with all other elements equal to 0. Thus, if a circuit model has b elements, and it is described by nxn modified nodal matrix $\boldsymbol{T}$, then $\boldsymbol{P}$ and $\boldsymbol{Q}$ matrices are nxb *and* bxn matrices respectively.

Some stamps may contain not only the element value represented in equation (54) by Y but constant values as well. These constant values do not affect matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$.

In a circuit model with passive two-terminal circuit components (R, L, and C) only, matrix $\boldsymbol{P}$ is equal to matrix $\boldsymbol{Q}$, and it is known as the incidence matrix.

Having those matrices set up, the method of handling them once the system matrix is obtained depends entirely on the solution algorithm to be adopted. Successive reduction by Kron's equation does not provide the most efficient way for the solution. Practical experience shows that direct algebraic methods guided by topological methods are the most efficient. For that reason it is important to know how to generate basic graphical representation of the circuit without the need to go in the details of graphical analysis.

The three types of graphs (signal-flow graphs, directed graphs, and conjugated graphs) can be easily obtained from the modified nodal stamps, and their major types are clearly established based on how they appear in the

modified nodal matrix. Each edge of a graph may describe one, two, or four elements of the coefficient matrix depending on whether it is in a signal-flow graph, directed graph or pair of conjugated graphs.

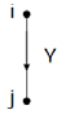
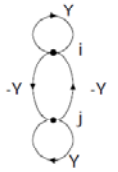
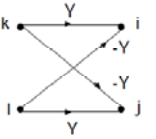
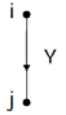
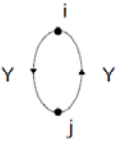
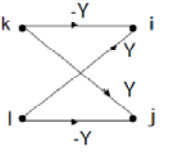
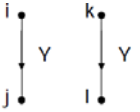
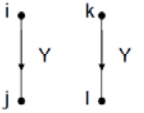
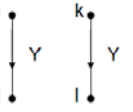
Table 4 contains various types of example graphs of a passive two-terminal element and a voltage controlled current source.

Using the stamp approach based on the modified nodal equations [6] all types of graphs can be directly obtained. A complete set of Coates' graph stamps were presented by Starzyk in [67] using the so called transitor models. Chen [33] presented a subset of unistor graph models for elements with admittance description only. In [68] this description was extended to include all elements with modified nodal equations using formal unistor models. In addition, Starzyk introduced dispersor graphs and used them for topological analysis in [69]. Seshu [30] showed how to obtain current–voltage graphs and Davies [70] presented nullator-norator networks of the controlled sources. Finally, the conjugated norator -nullator graphs and rules to use them in topological analysis were presented by Starzyk in [71]. These stamp based models facilitated topological analysis by automating the graph creation process, a critical step in computer based topological analysis.

Dispersor graphs were introduced in [69] to use topological methods for analysis of electronic circuits with ideal op -amps. Although unistor models of electronic elements were known in the literature, only the introduction of formal unistor models [72] permitted to model ideal op -amps and other active elements for which unistor models did not exist. Other forms of graphical representations based for instance on the tableau equations can be used, however, they lead to larger graphs and, in general, require more effort to analyze.

Solutions using topological methods vary in complexity and efficiency depending on the graph selected. Exact symbolic analysis based on hierarchical decomposition and a graphical representation of symbolic determinants called determinant decision diagrams (DDD) can be used in this respect for a very efficient implementation [73]. DDD's take advantage of the coefficient matrix sparsity leading to exact and canonical symbolic analysis that share symbolic expressions to improve computing efficiency. Efficiency of this method exceeds efficiency of numerical analysis programs like Spice. It is also faster than other symbolic programs such as ISAAC and Maple-V, and uses less computer memory [74].

Table 4. Graphs of a two-terminal element and a voltage controlled current source

	Single graph edge	Example stamp	
		Two-terminal element (R, L, or C) $I_i = (V_i - V_j)Y$ $I_j = -(V_i - V_j)Y$	Voltage controlled curent source $I_i = (V_k - V_l)Y$ $I_j = -(V_k - V_l)Y$
Coates' graph	$I_j = V_iY$ $I_i = 0$ 		
Unistor graph	$I_i = V_iY$ $I_j = -V_iY$ 		
Current-voltage graph	$I_i = (V_k - V_l)Y$ $I_j = -(V_k - V_l)Y$ 		

Chapter 5

CONCLUSION

The CMNA algorithm was presented in this book with special considerations adopted to automatically generate the system matrix for symbolic circuit analysis applications. The formulation was presented in a simplified yet generalized form with ability to select solution variables automatically at various levels of complexity. Ultimately a heuristic approach can be adopted in the program instead of a comprehensive list of if-conditions to automatically weigh the complexity of the generated terms versus the reduction gained in the system matrix size.

A note was also made on automatic generation of different graphical representations for the same application much in the same way adopted in the CMNA stamping procedure. This is a crucial step for any topological solver algorithm.

Although not part of the CMNA formulation, constraints imposed by the element stamps were introduced with emphasis on nodal effects under such constraints. Such a view of constrained nodes becomes particularly important in symbolic circuit design where element values are only represented by ranges and spans.

Finally the concept of element model incorporation within the CMNA is briefly introduced through Macro-stamps. Ideally, using this simple description and the derivation methods developed in this book, generalized element stamps can be derived to represent even active element models.

The improved performance of the method was verified by solving a number of examples and comparing the formulation efficiency and the memory requirements with several commercial packages. Indeed, the proposed method can achieve (if applied correctly) significant savings in

efficiency and memory resources of the computer not to mention the potential ability to solve larger circuits.

REFERENCES

[1] M. Monagan. (1997). Programming in maple: the basics. Institute für Wissenschaftliches Rechnen, Zürich, Switzerland. http://amath. colorado.edu/computing/mmm/MapleProgr.pdf

[2] P. Hammerton; D. Stevens. (1999). Introduction to MAPLE. Maths Terminal Room, mth1a31. http://www.uea.ac.uk/~dps/ mth1a31/ maple.pdf.

[3] B. Char et al. *First Leaves: A Tutorial Introduction to Maple V*; Springer-Verlag Pub: New York, 1992.

[4] H. E. Brown. *Solution of Large Networks by Matrix Methods*; John Wiley and Sons: New York, 1985.

[5] W. H. Press; S. A. Teukolsky; W. T. Vetterling; B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*; second edition; Cambridge University Press: New York, 1992.

[6] J. Vlach; K. Singhal. *Computer Methods for Circuit Analysis and Design*; second edition; Van Nostrand Reinhold: New York, 1994.

[7] Kirchhoff G. On the solution of the equations obtained from the investigation of the linear distribution of galvanic currents. *IRE Trans.* 1958, Vol. CT-5, (tlum. pracy z 1947).

[8] Dunn W.R., Jr.; Chan S.P. Topological formulation of active network functions. *IEEE Trans. Circuit Theory.* 1971, Vol. CT -13, pp 554-557.

[9] Jony M.T.; Zobrist G.W. Topological formulas for general linear networks. *IEEE Trans. Circuit Theory.* 1968, Vol. CT-15, pp 251-259,.

[10] Chen W.K. Topological analysis for active networks. *IEEE Trans. Circuit Theory.* 1965, Vol. CT-12.

[11] Nathan A. Topological rules for linear networks. *IEEE Trans. Circuit Theory.* 1965, Vol. CT-12, No 3.

[12] Braun J. Analytical methods in active network theory. *Acta Polytechnica – Prace CVUT v Praze.* 1966, Vol. IV. I.

[13] Braun J. Topological analysis of networks containing nullators and norators. *Electr. Lett.* 1966, pp 427.

[14] Davies A.C. Topological solutions of networks containing nullators and norators, *Electr. Lett.* 1966, pp 90.

[15] Talbot A. Topological analysis for active networks. *IEEE Trans. Circuit Theory.* 1966, Vol. CT-13.

[16] Wang K.T. On a new method for the analysis of electrical networks. National Resources, Institute for Engineering, *Academia Sinica Memoir.* 1934, No 2.

[17] Ting S.L. On the general properties of electrical network determinant. *Chinese Journal of Physics. 1935*, No 1.

[18] Tsai C.T. Shortcut methods for expanding the determinants involved in network problems. *Chinese Journal of Physics.* 1939, No 3.

[19] Duffin R.J. An analysis of Wang algebra of network. *Trans. American Mathematics Society.* 1959.

[20] Bellert S.; Wojciechowski J. Grafy blokowe i liczby strukturalne drugiej kategorii. Prace naukowe P.W., *Seria Elektronika*. 1972, No 1.

[21] Bellert S.; Wozniacki H. Analiza i synteza układów elektrycznych metoda. *liczb Strukturalnych.* WNT: Warszawa, 1968.

[22] Bellert S. Topological analysis and synthesis of linear systems. *Journal of the Franklin Inst.* 1962, Vol. 274.

[23] Bellert S. Topological considerations and synthesis of linear networks by means of the method of structural numbers, *Arch. Elektr.* 1963, t.XII, z.3.

[24] Gandhi B.R.M.; Rao V.P.; Raju G.S. Passive and active circuit analysis by the method of structural numbers. *Int. J. Electr.* 1972, Vol. 32, No 6.

[25] Coates C.L. Flow graph solutions of linear algebraic equations. *IRE Trans. Circuit Theory.* 1959, pp 170-187.

[26] Coates C.L. General topological formulas for linear networks functions. *IRE Trans. Circuit Theory.* 1958, Vol. 5.

[27] Mason S.J. Feedback theory - further properties of signal flow graphs. *Proc. IRE.* 1956, Vol. 44 pp 920-926.

[28] Mason S.J. Feedback theory - some properties of signal flow graphs. *Proc. IRE.* 1953, Vol. 41 pp 1144-1156.

[29] Mayeda W. Topological formulas for active networks. Univ. of Illinois. 1958, *Int. Tech. Rpt.* No 8, U.S. Army Contract No DA-11-022-ORO-1983.

[30] Seshu S.; Reed M.B.; *Linear graphs and electrical networks*; Addison-Wesley P. C., 1961.

[31] Davies A.C. On matrix analysis of networks containing nullators and norators. *Electronic Letters.* 1966, pp 48.

[32] Mason S.J. Topological analysis of linear nonreciprocal networks. *Proc. IRE.* 1957, Vol. 45, pp 829-338.

[33] Chen W.K.; *Applied graph theory - graphs and electrical networks*; North-Holland P. C., 1976.

[34] Chen W.K. On directed trees and directed k -trees of a digraph and their generation. *SIAM J. Appl. Math.* 1966, Vol. 14, No 3.

[35] Janusz A. Starzyk. (2007). Topological Analysis and Diagnosis of Analog Circuits. Wydawnictwo Politechniki Śląskiej Gliwice. http://www.ent.ohiou.edu/~starzyk/network/Research/Papers/Topological_analysis_and_diagnosis.pdf.

[36] L. W. Beineke; J. L. Gross; S. B. Maurer; E. R. Scheinerman; *Graph theory- Handbook of Discrete and Combinatorial Mathematics*; K. H. Rosen; J. G. Michaels, J. L. Gross; J. W. Grossman; D. R. Shier; CRC Press LLC.: New York, 2000.

[37] Csiszar, I.; Korner, J. Graph decomposition: A new key to coding theorems. *IEEE Transactions on Information Theory.* 1981, Vol. 27, No 1.

[38] Ho J.K.; Loute E. An advanced implementation of the Dantzig –Wolfe decomposition algorithm for linear programming. *J. Mathematical Programming.* 1981, Vol. 20, pp 303-326.

[39] Hu T.C.; *Integer programming and network flow Reading*; Addison – Wesley P. C.: MA, 1969.

[40] Goto S.; Sangiovanni-Vincentelli A. A new decomposition algorithm for the shortest path problem. *Proc. IEEE Int. Symp.* on CAS, Tokyo. 1979, pp 653 -656.

[41] Frederickson G.N. Planar graph decomposition and all pairs shortest paths. *Journal of the ACM.* 1991, Vol. 38, No 1, pp 162 -204.

[42] Chu T. Synthesis of Self-timed VLSI Circuits from Graph Theoretic Specifications., Massachusetts Institute of Technology. 1987, *Technical Report*. TR -393.

[43] Pothen A.; Simon H.D.; Liou K-P. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications.* 1990, Vol. 11, No 3, pp 430-452.

[44] Wu S. D.; Byeon E-S.; Storer R. H. A Graph-Theoretic Decomposition of the Job Shop Scheduling Problem to Achieve Scheduling Robustness. *J. Operations Research.* 1999, Vol. 47, No 1, pp 113-124.

[45] Ehrenfeucht A.; Harju T, Rozenberg G. Gene assembly through cyclic graph decomposition. *Theoretical Computer Science. Elsevier.* 2002, Vol. 281, No 1 -2, pp 325-349.

[46] Ko M-Y.; Murthy P.K.; Bhattacharyya1 S.S.; Compact Procedural Implementation in DSP Software Synthesis through Recursive Graph Decomposition. Software and Compilers for Embedded Systems; Springer Verlag, Lecture Notes in Computer Science. 3199; 2004; pp 47 -61.

[47] Engel A.B.; Mlynski D.A. Maximal partitioning of graphs. *Proc. IEEE Int. Symp*. on CAS, Tokyo. 1979, pp 84 -87.

[48] Ferrari D. Improving locality by critical working sets. *Comm. of ACM.* 1974, Vol. 17, No 11, pp 614-620.

[49] Kernighas B.W.; Lin S. An efficient heuristic procedure for partitioning graphs. *Bell System Tech. J.* 1970, Vol. 49, No 2, pp 291 -307.

[50] Luccio F.; Sami M. On the decomposition of networks in minimally interconnected subnetworks. *IEEE Trans. Circuit Theory.* 1969, Vol. CT-16, pp 184-183.

[51] Narayanan H. A Theorem on graphs and its application to network analysis. *Proc. Int. Symp. Circuits and Systems,* Tokyo. 1979, pp 1007 -1011.

[52] Ogbuobiri E.C.; Tinney W.F.; Wal ker J.W. Sparsity-directed decomposition for Gaussian elimination on matrices. *IEEE Trans. Power, Appr. Syst.* 1970, Vol. PAS-89, pp 141-150.

[53] Callier F.M.; Chan W.S.; Desoer C.A. Input-output stability theory of interconnected systems using decomposition techniques. *IEEE Trans. Circuits and Systems.* 1976, Vol. CAS-23, pp 714-729.

[54] Guardabassi G.; Sangiovanni-Vincentelli A. A two levels algorithm for tearing. *IEEE Trans. Circuits and Systems.* 1976, Vol. CAS -23, pp 783-791.

[55] Desai M. P.; Narayanan H.; Patkar S.B. The realization of finite state machines by decomposition and the principal lattice of partitions of a submodular function. *J. Discrete Applied Mathematics.* 2003, Vol. 131, No 2, pp 299-310.

[56] Starzyk J.A. Decomposition Approach to a VLSI Symbolic Layout with Mixed Constraints. *Proc. IEEE Int. Symp. Circuits and Systems, Montreal.* 1984, pp 457-460.

[57] Kron G.; *Diakoptics: the piecewise solution of large scale systems*; Mc Donald: London, 1963.

[58] Nicholson H.; *Structure of interconnected systems*; Peter Peregrines LTD, 1978.

[59] Stenbakken G.N.; Starzyk J.A. Diakoptic and Large Change Sensitivity Analysis. *IEE Proc. Part G, Circuits, Devices and Systems.* 1992, Vol. 139, No 1, pp 114-118.

[60] Chua L.O.; Chen L.K. Diakoptic and generalized hybrid analysis. *IEEE Trans. on CAS.* 1976, Vol. CAS-23, No 12, pp 694-705.

[61] Konczykowska A., Starzyk J. Wyznaczanie liczby strukturalnej grafu zdekomponowanego. *Czesc I i II, Arch. Elektrot*. 1975, z.2.

[62] Konczykowska A. Analiza ukladow elek trycznych z zastosowaniem metod dekompozycji, Praca doktorska, Warszawa, 1976.

[63] D. V. Tosic; A. R. Djordjevic; B. D. Reljin. Symbolic Analysis of Microwave Circuits. *Journal of Applied Electromagnetism.* 1997, Vol. 1, No 1, pp 37-45.

[64] D. V. Tosic; A. R. Djordjevic; B. D. Reljin. Symbolic computation of S-parameters of linear electric networks. *ETF Journal of electrical engineering.* 1996, Vol. 6, No 1, pp 84-98.

[65] J. K. Fidler; C. Nightingale; *Computer Aided Circuit Design*; Thomas Nelson and Sons: Hong Kong, 1978.

[66] F. V. Fernández; A. Rodríguez-Vázquez; J. L. Huertas; G. E. Gielen; *Symbolic Analysis Techniques – Applications to Analog Design Automation*; IEEE Press: New York, 1998.

[67] Starzyk J.A. Analiza topologiczna duzych ukladow elektronicznych, Prace Naukowe PW, *Elektronika.* 1981, z. 55.

[68] Centkowski G.; Starzyk J., Śliwa E. Computer implementation of topological method in the analysis of large networks, *Proc. of the ECCTD Warszawa.* 1980.

[69] Starzyk J.A. The dispersor graphs. Proc. of 3rd Czech-Polish Workshop on CT, Prenet. 1978.

[70] Davies A.C. Nullator-norator equivalent networks for controlled sources. *Proc. IEEE.* 1967, pp 722-723.

[71] Starzyk J.A. Metody topologiczne analizy akladow skupionych liniowych i stacjonarnych z nulatorami i no ratorami, *Prace Naukowe PW, Elektronika.* 1975, No 20.

[72] Centkowski G.; Konczykowska A.; Starzyk J.; Śliwa E. Modernizacja program HADEN zuwzględnieniem analizy hierarchicznej wstę pującej, analizy tolerancjiz możliwością analizy układow z przełączanymi pojemnościami. Problem badań podstawowych, Warszawa. 1980, No 1.11.07.02.

[73] Tan X-D.; Shi C.-J.R. Hierarchical symbolic analysis of analog integrated circuits via determinant decision diagrams. *IEEE Trans. on Computer–Aided Design of Integrated Circuits and Systems.* 2000, Vol. 19, No 4, pp 401 – 412.

[74] Shi C.-J.R.; Tan X-D. Canonical symbolic analysis of large analog circuits with determinant decision diagrams. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*. 2000, Vol. 19, no 1, pp 1 – 18.

INDEX

S

T

V

W