Control and Cybernetics

vol. **22** (1993) No. 1/2

Automatic analysis of spatial electrical circuits with graph theory

by

Jacek Wojutyński

Scientific Centre for Terotechnology
Radom,
Poland

The paper deals with calculation of system functions in full symbolic shape as a function of all circuit parameters. Concentrated, linear and stationary electrical circuits are being considered Kudrewicz, Osiowski (1974). The spatial circuits can also contain all types of control sources. The notion of spatial electrical circuit has been introduced in Wojutyński (1988), meaning the circuit where the graph of signal flow is not the s–t–planar graph. For electric circuit modelling the methods of graph theory have been applied. The connections between the analysis of spatial electrical circuits with the task of the flow in nets have also been indicated. The algorithms have been implemented in GCLISP.

1. Basic definitions

Before defining the spatial electrical circuit, the definitions of directed graph, flow signal graph of and s–t–planar graph will be introduced.

DEFINITION 1. (DIRECTED GRAPH) Kudrewicz, Osiowski (1974) *Let X determine the set of nodes, Y — a set of branches. An ordered triple $G = < X, Y, P >$ such, that:*

1) $X \neq \emptyset$,

2) $Y \cap X = \emptyset$,

3) P is a function which for every branch assigns an ordered pair of nodes from the set $X \times X$ i.e.

$$P : Y \to X \times X,$$

is called directed graph.

In memoriam Professor Edmund Lipiński

DEFINITION 2. (THE GRAPH OF SIGNALS FLOW) Kudrewicz, Osiowski (1974) *The graph of signals flow is a directed graph built in the following way:*

1) *variables of the set of linear equations are assigned to the graph nodes,*
2) *transfer functions being the coefficients placed at variables are assigning to the branches,*
3) *variable x_k assigned to the node which is the end of several branches is equal to the sum of products of variables assigned to the begínings of these branches by their transfer function.*

DEFINITION 3. (s–t–PLANAR GRAPH) Hu (1970) *If two nodes s and t in graph are distinguished (in the graph of signals flow we traditionally determine by s — source node and t — receiving node) then the graph is called s–t–planar graph if after addition of the branches between s and t nodes it remains planar.*

DEFINITION 4. (SPATIAL ELECTRICAL CIRCUIT) Wojutyński (1988) *An electrical circuit will be called a spatial electrical circuit if its graph of signals flow is not s–t–planar graph.*

An accepted definition of directed graph assumes existence of more than one branch (parallel branches) between pair of nodes. The directed graph will be defined by a set of ordered triples $(x_j\ y_i\ x_k)$, where $y_i \rightarrow (x_j x_k)$. This method of graph assignment is very comfortable for programs realized in LISP, considering the mutually univocal correspondence of formal notation with LISP notation in the form of a list. In LISP the description of branches can be any s–expression, for example a symbolic notation of transfer function. It is an unquestionable advantage of using LISP for description of eletrical circuits represented by graphs of signal flow. LISP allows to make a simple realizations of algorithms acting on expressions. GRASPE language (University of Houston) has been applied for operations on graphs. The GRASPE language has been implemented by the author in GCLISP.

The definition of spatial electrical circuit indicates to relationships between node variables. The dependences are not easily seen in the case of circuit graph (topology of circuit) application. It is possible that the circuit graph is planar and there are control sources in the circuit and the signals flow graph is not an s–t–planar graph.

Most programmes for computer analysis of electronic circuits using the Mason's rule make use of shut signals flow graph, i.e. the graph in which input and output are joined by additional branch, which allows for a single calculation of graph determinant Chua, Lin (1981).

Eletrical circuit represented by spatial graph usually gives a larger number of loops and it is more complicated for most computer algorithms.

Ford and Fulkerson (after Hu (1970)) noticed that the task of finding the maximum flow in net is, in general case, easier to solve for an s–t–planar net.

Taking into account the above facts the introduced definition of spatial electrical circuit, in author's opinion allows to calculate the complexity of the task connected with determinination of the transfer function of the circuit.

2. Determination of system functions in full symbolic shape

Determination of system functions (e.g. transmitations) has been implemented in LISP, allowing to act, in an easy way, on symbols, ensuring also compatibility of LISP notation with mathematical notation. The methods of graph theory — signals flow graphs (the Mason's graph) and decomposition into components of strong cohesion (new exceptionally effective algorithm) have been used. The basic rules of reduction of flow graphs Kudrewicz, Osiowski (1974) have also been used: the rule of parallel branch reduction, the rule of series branch reduction, the rule of indirect node reduction and the rule of own–loop reduction. The flow graph reduction is realized by reducing, initially, the components of strong cohesion. Then the acyclic graph is obtained (because the so called condensation graph, where nodes are components of strong cohesion, is acyclic), for which only the rule of series and parallel branches reduction is used. By reducing, initially, the strong cohesion, one obtains in a natural way a simplification of formulae for the system function, because the expressions appearing in feedback loops do not propagate over the whole graph.

The time consumed for system function determination by means of existing methods is of the order of several hundred seconds, even for electrical circuits with several nodes. The method presented allows to reduce the calculation time to several seconds and it makes this method useful in conversational systems and in computer aided didactics.

It has been proved in Wojutyński (1988) that the time of transfer function determination using the method of flow graph reduction together with decomposition into strong cohesion components is limited multinominally and for full graph the time equals $O(n^3)$, where n — number of nodes (for RC ladder, whose flow graph is s–t–planar graph, the time of calculation equals $O(n)$).

The language of eletrical circuit topology description suggested in this paper allows to analyse the circuits which consist of the following elements: RLC, operational amplifiers, voltage and current generators (also controllable). Parameter values can be given as a number, symbol, expression, for example a formula for the transfer function.

Using macro–generator implemented by the author, the libraries of new elements built from basic elements or elements determined by user can be formed in a simple way.

3. Algorithm of graph decomposition into strong cohesion components

The graph is decomposed into strong cohesion components. The algorithm has been presented in the form of a LISP programme. A considerable effectiveness of the algorithm results from the application of LISP and GRASPE languages for operations on graphs.

The gamma function realizes Berge's operator Γ and Γ^{-1} , sin and son function of GRASPE language (set of input/output node). Gamma function has the following parameters:

s — set of graph nodes,
fa — name of sin or son function,
g — graph's name

Then

$$\Gamma\, s \equiv (\text{gamma } s\,'son\,'g)$$

$$\Gamma^{-1}\, s \equiv (\text{gamma } s\,'sin\,'g)$$

The form of the gamma function is presented in the following way:

```
(defun gamma (s fa g)
        (makeset  (apply    #'append
                  (mapcar   #'(lambda (p) (funcall fa p g))
                            s) )) )
```

The transit function (function tranzyt — in Polish) realizes transitive shut (iterative aplication of operator Γ or Γ^{-1} for the set s). This function characterizes the graph g nodes using identifier *ind*, which are attainable from the set of s nodes. The set of nodes which we do not pass is determined by zb. The fa function is sin or son function.

When iterating : Γ to fa := son and *ind* := 'v+.
When iterating : Γ^{-1} to fa := sin and *ind* := 'v-.

Transit function (function tranzyt — in Polish) has the following form:

```
(defun tranzyt (s ind fa g zb)
    (prog ()
       a (setq s (remove-if '(lambda (p) (get p ind))
                   (setdif s zb) ))
         (cond ((null s) (return nil)))
         (mpc #'(lambda (p) (putprop p ind t))
            s)
         (setq s (gamma s fa g))
         (go a)    ))
```

The s–s–cohesion function (function s–s–spójnosści in Polish) gives a strong cohesion component generated by v node in g graph. The other parameters are

za — set of nodes in which we look for a component,
zb — set of nodes which we do not pass.

The application of za and zb sets makes transit and s–s–cohesion functions universal.

The s–s–cohesion function is as follows:

```
(defun s-s-spojnosci (v g za zb)
    (mpc #'(lambda (p) (putprop p 'v+ nil)) za)
    (mpc #'(lambda (p) (putprop p 'v- nil)) za)
    (tranzyt (list v) 'v+ 'son g zb)
    (tranzyt (list v) 'v- 'sin g zb)
    (remove-if '(lambda (p) (not (get p 'v-)))
          (remove-if '(lambda (q) (not(get q 'v+)))
                za)) )
```

The s–s–cohesion function returns a set of strong cohesion components, where:

g — graph's name,
st — set including the source and receiving nodes.

```
(defun skladowe-s-spojnosci (g st)
    (prog (za zb sk s)
        (setq za (setdif (nodeset g) st))
        (setq zb st)
        (setq sk nil)
       a (cond ((null za) (return sk)))
        (setq s (s-s-spojnosci (car za) g za zb))
        (setq sk (cons s sk))
```

(setq *za* (setdif *za* *s*))
(setq *zb* (union *zb* *s*))
(go a)))

The comparison of the time of determining the strong cohesion components of the presented algorithm with Tarjan's algorithm proved the superiority of author's algorithm. It is quite obvious considering the adaptation of the algorithm to LISP and data format of modified GRASPE. These facts confirm the well known "equation":

programme = algorithm + data format + programming language

and point to strong correlation between the right members of this "equation".

4. Conclusions

The system has been tested on scores of examples of solutions of electrical engineering problems solution of direct current circuits, substitute impedance determination, transfer functions of transistor amplifiers and active filters with operational amplifiers.

As one of the tasksan example from Fig.65.13 from Mikołajuk, Trzaska (1984) has been solved. It is a system which contains of 4 operational amplifiers, 10 resistors, 2 condensers and 11 nodes. The time of calculation of the transfer function for this system equals 8.41 sec (for PC/AT 10MHz) and series connection of 9 such systems — 1 min. and 50 sec. The flow signal graph is formed by the nethod of node potentials.

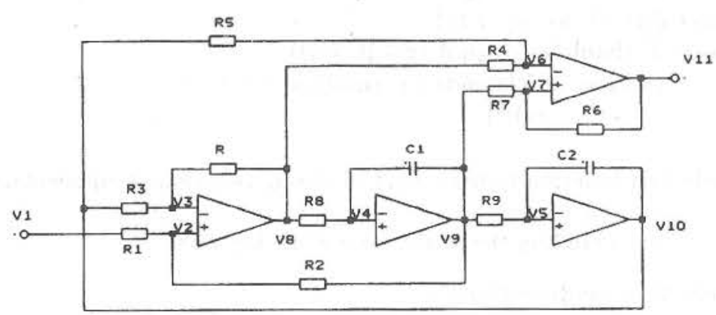

Fig.1. Time calculation of V11/V1 = 8.41 sec .

This paper is a continuation of research work on automation of calculations and computer conversation in natural Polish language. The research was carried out during the seminar "Multiaccessible conversational computer systems" conducted by Professor E. Lipiński at the Technical University in Warsaw. The present article is a part of doctoral thesis Wojutyński (1988).

5. Examples

Example 1
The system is presented in Fig.2. (transistor filter):

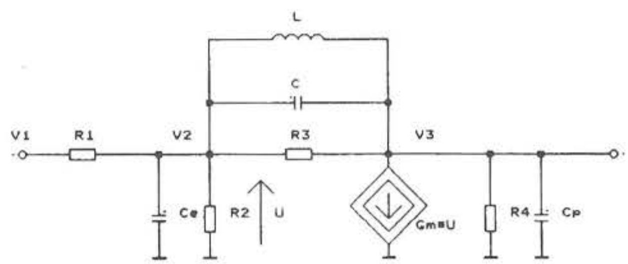

Fig.2. Transistor filter

Description of the system:

```
(POTEN  (  (R v1 v2 R1)
           (Y v2 masa (+ G2 (* s Ce)))
           (Y v2 v3 (+ G3 (* s C) (/ 1 (* s L))))
           (Y v3 masa (+ G4 (* s Cp)))
           (JU masa v3 v2 masa Gm)
           (J  v1 masa J)
           (WE v1)
           (WY v3)              ))
                                masa equiv GND
```

Time of transfer function calculation $v3/v1$ equals 2.41 sec., length of expression printout for transmitation — 17 lines.

Example 2
Definition of macro *RC* ladder with n cells:

```
(MODEL   'RCn  '(Rd Cd n Vin Vout)  '(Vp)
       '((if  (= n 1)
              (prog ()
                  (R Vin Vout Rd)
```

```
        (C Vout 'masa Cd) )
    (prog ()
        (R Vin Vp Rd)
        (C Vp 'masa Cd)
        (RCn Rd Cd (- n 1) Vp Vout) ) )) )
                recurence definition!
```

Call (RCn 'R5 'C6 7 'Vstart 'Vstop).

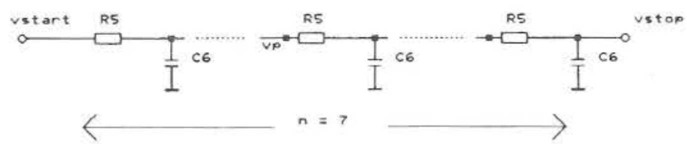

Fig.3. RC ladder

References

CHUA L.O. AND LIN P.M., (1981) Computer Aided Analysis of Electronic Circuits. Algorithms amd Computational Techniques (in Polish), Warsaw WNT.

HU T.C., (1970) Integer Programming and Network Flows, California – London, Addison – Wesley.

KUDREWICZ J. AND OSIOWSKI J., (1974) Selected Problems of Circuit Theory (in Polish), Warsaw, WPW.

MIKOŁAJUK K. AND TRZASKA Z., (1984) Theoretical Electrical Engineering — Analysis and Synthesis of Linear Electrical Circuits (in Polish), Warsaw, PWN.

WOJUTYŃSKI J., (1988) Automatic Analysis of Spatial Electrical Circuits with LISP Language Application, Doctoral Thesis, Technical University in Warsaw, Electrical Department, Warsaw.