Proceedings of the 10th WSEAS International Conference on CIRCUITS, Vouliagmeni, Athens, Greece, July 10-12, 2006 (pp235-240)

Maple Implementation of the Kirchhoff's "Third and Fourth Laws"

NICOLAS RATIER[1], MAYA MARKOVA[2]
[1] Institut FEMTO–ST, département LPMO, CNRS UMR 6174
32 avenue de l'Observatoire, 25044 Besançon Cedex, France
[2] Department of Informatics and Information Technologies
University of Rousse, 7017 Rousse, Bulgaria

Abstract: - This paper proposes an implementation in Maple language of the Kirchhoff's third law and its dual, the so–called Kirchhoff's fourth law. These laws, stated in graph theory language, allows one to compute symbolically any network functions of linear passive circuits.

Key-Words: - Symbolic, Computation, Kirchhoff's Laws, Network, Graph, Maple.

1 Introduction

Symbolic analysis for analog circuits has received quite some research interest over the last ten years. In numerical simulation, the values of all circuit elements are given and the circuit response to an input excitation is calculated in numerical form. In symbolic computation, the circuit elements are represented by symbols and the desired network characteristic is derived in finite terms (or in physicist language: as an analytic expression).

This paper focuses more particularly on the symbolic computation of network functions of linear passive circuits. The computation of network functions in symbolic forms is a starting point for tolerance analysis, fault diagnosis, behavioral model generation, statistical circuit analysis, optimization and the associated calculation of component values. In brief, symbolic computation can be very useful in all applications which involve the repeated evaluation of a circuit's characteristics.

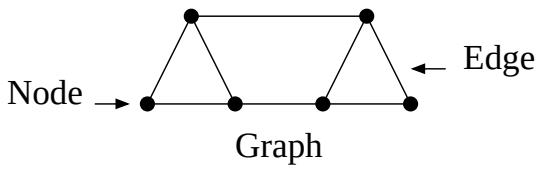
This paper proposes an implementation in Maple language of the computation of any network functions (E/I, E/E, I/E, I/I) of linear passive circuits by means of the so–called Kirchhoff's "third and fourth laws" [5]. These laws allows one to express network functions, in a very concise way, in terms of trees enumerations. The symbolic computation of network functions is closely related to the computation of the inverse of a nodal admittance matrix. Both of problems are efficiently solved by tree enumeration methods [2][1].

Section 2 recalls some definitions of graph theory which are necessary to understand the statement of the Kirchhoff's laws. Sections 3 and 4 give the statement of the third and fourth Kirchhoff's laws. The proofs of these laws can be found in [4](with minor corrections [3]). Two hand calculations of these laws are detailed on sections 5 and 6. Finally, sections 7 and 8 present the algorithm and propose a practical implementation of the Kirchhoff's third and fourth laws.

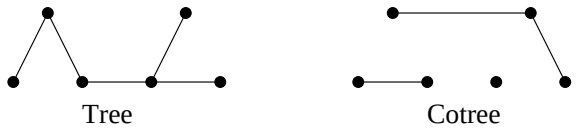
2 Graph Theory

Some definitions of graph theory are recalled in this section. The terminology is standard except for the terms *split–tree* and *co–split–tree*. These new terms are proposed by the authors to give an uniform statement of the third and fourth Kirchhoff's laws.

- *Graph*: A graph (or network) consists of a set of nodes together with a set of edges that connect the nodes.



- *Tree*: A tree is any set of edges that connects all the nodes but does not form any loop.
- *Cotree*: The cotree of a tree is its complement, i.e. the set of all edges of the graph that do not appear in the tree.



- *Loop*: A loop is any set of edges that forms a closed path.
- *Coloop*: The coloop of a loop is its complement, i.e. the set of all edges of the graph that do not appear in the loop.

- *Split–tree*: A splittree (split into trees) is any set of edges whose removal would disconnect a graph into two trees.
- *Co–split–tree*: The cosplittree of a splittree is its complement or the remaining two trees, i.e. the set of all edges of the graph that do not appear in the splittree.

The definition of a *split–tree* must not be confused with a *cutset* which is a set of edges whose removal would disconnect a graph. Similarly, the definition of a *co–split–tree* must not be confused with a *two–tree* which is a set of edges obtained by deleting an edge from a tree.

3 Kirchhoff's "Third Law"

Given any connected resistance network G, if a voltage generator E_l is inserted in a branch R_l, then the current I_k through any branch R_k is given [5] by Eq. 1.

$$I_k = \frac{N_{kl}}{D} E_l \quad (1)$$

$$N_{kl} = \sum_{\text{loops}(R_k,R_l)}^{+/-} \prod_{R_m \in \text{coloop}} R_m \quad (2)$$

$$D = \sum_{\text{cotrees}(G)} \prod_{R_m \in \text{cotree}} R_m \quad (3)$$

- **How to read the expression of D:** The summation is carried out over all the cotrees of the network. A term in the summation is the product of all the edges (resistances) of a cotree.

- **How to read the expression of N:** The algebraic summation is carried out over all the loops that include as its edges both R_k and R_l. The associated sign of the term is positive if the directions of I_k and E_l, along the loop, are in the same sense. A term in the summation is the product of all the edges (resistances) of the associated coloop.

4 Kirchhoff's "Fourth Law"

Given any connected conductance network G, if a current source I_l is placed across any branch G_l, then the voltage E_k across any branch G_k is given [5] by Eq. 4.

$$E_k = \frac{N'_{kl}}{D'} I_l \quad (4)$$

$$N'_{kl} = \sum_{\text{splittrees}(G_k,G_l)}^{+/-} \prod_{G_m \in \text{cosplittree}} G_m \quad (5)$$

$$D' = \sum_{\text{trees}(G)} \prod_{G_m \in \text{tree}} G_m \quad (6)$$

- **How to read the expression of D':** The summation is carried out over all the trees of the network. A term in the summation is the product of all the edges (conductances) of a tree.

- **How to read the expression of N':** The algebraic summation is carried out over all the splittrees that include as its edges both G_k and G_l and so that G_k and G_l connect the two trees. The associated sign of a term is positive if the direction of E_k and I_l, connecting the two trees, are in the same sense. A term in the summation is the product of all the edges (conductances) of the associated two trees (cosplittree).

5 Hand calculation of Kirchhoff's third law

The third Kirchhoff's law is illustrated in the network shown in Fig. 1. We want to express the current I_5 through the resistance R_5 in terms of the supply E_0 and the given resistances.

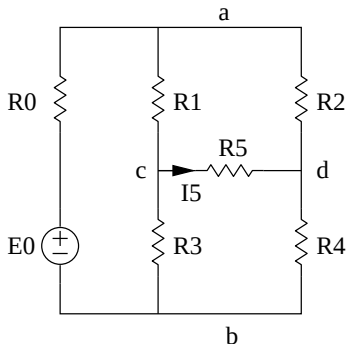


Fig. 1: Bridge resistance example.

All the subgraphs, that contains edges 0 and 5 and possessing one loop, are shown in Fig. 2.

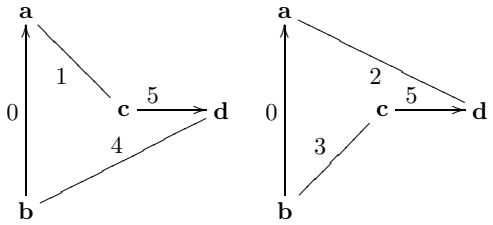


Fig. 2: Loops of network in Fig. 1.

The respective coloops of the previous subgraphs is given on Fig. 3.

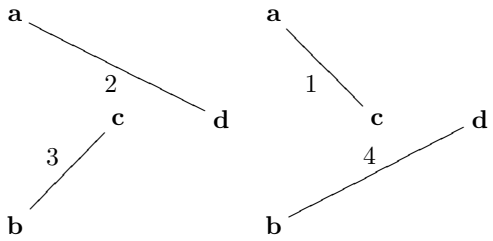


Fig. 3: Coloops of network in Fig. 1.

The trees and their associated cotrees are given in Fig. 7 and 8. By inspection of subgraphs on Fig. 2, 3 and 8, the current I_5 is:

$$I_5 = \frac{+R_2R_3 - R_1R_4}{R_3R_4R_5 + R_2R_3R_5 + \cdots + R_0R_1R_3}E_0 \tag{7}$$

6 Hand calculation of Kirchhoff's fourth law

The fourth Kirchhoff's law is illustrated in the network shown in Fig. 4. We want to express the voltage E_5 across the resistance R_5 in terms of the supply I_0 and the given conductances.

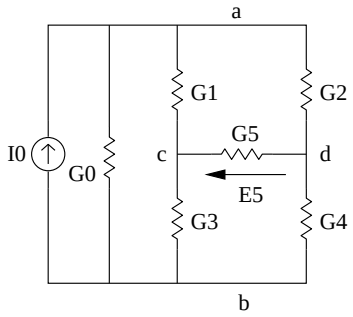


Fig. 4: Bridge conductance example.

All the subgraphs, composed of edges whose removal would split the graph into two trees linked by edges 0 and 5, are shown in Fig. 2.

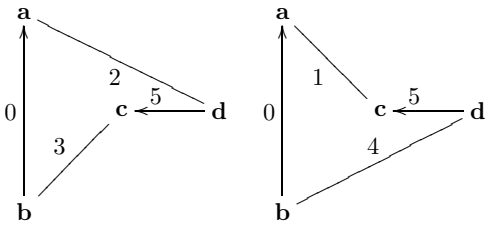


Fig. 5: Split–trees of network in Fig. 4.

The two remaining trees of the previous subgraphs are given in Fig. 6. Edges 0 and 5 are represent in dashed line for clarity.

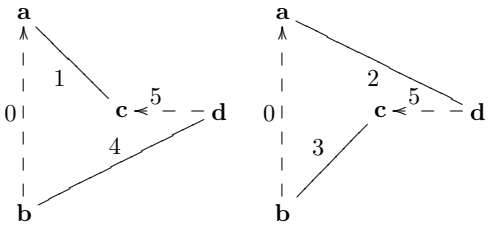


Fig. 6: Co–split–trees of network in Fig. 4.

The trees and their associated cotrees are given in Fig. 7 and 8. By inspection of subgraphs on Fig. 5, 6 and 7, the voltage E_5 is:

$$E_5 = \frac{+G_1G_4 - G_2G_3}{G_0G_1G_2 + G_0G_1G_4 + \cdots + G_2G_4G_5}I_0 \tag{8}$$

7 Algorithm

The theoretical statements of the so–called third and fourth Kirchhoff's laws (Eq. 1 and 4) are intricate and one needs some time to get accustomed to it. At the opposite the algorithms [5] to compute them are straitforward and lead to very concise Maple programs.

Computation of D

1. Compute the determinant of the nodal resistance matrix.
2. The denominator D is the complement of each term.

Computation of N

1. Choose those terms of D that contain R_k and divide each of them by R_k.
2. Choose those terms of D that contain R_l and divide each of them by R_l.
3. The numerator N is the common terms in these two sets. The signs of the terms are determined by inspection of the respective subgraphs.

Computation of D'

1. The denominator D' is the determinant of the nodal conductance matrix.

Computation of N'

1. Choose those terms of D' that contain G_k and divide each of them by G_k.
2. Choose those terms of D' that contain G_l and divide each of them by G_l.
3. The numerator N' is the common terms in these two sets. The signs of the terms are determined by inspection of the respective subgraphs.

8 Implementation

The two Maple programs given in this section are the core of the implementation of the Kirchhoff's third and fourth laws. The code follow almost litteraly the algorithms given in the previous section. The remaining Maple code and numerous examples of use are available by Internet at `http://www.lpmo.edu/~ratier`.

Computation of third law (Eq. 1)

`Imp1` is the resistance of the branch through which the voltage generator is inserted. `Imp2` is the resistance of the branch through which the current is measured. `G` is the graph of the circuit.

```
KirchhoffThirdLaw := proc(Imp1,Imp2,G)
local  Tmp1,Tmp2,Tmp3,Tmp4,Num,Den:
Tmp1 := CoTrees(G);
Tmp2 := map(proc(x,y) if member(y, x) then
        x minus {y}: fi: end, Tmp1, Imp1);
Tmp3 := map(proc(x,y) if member(y, x) then
        x minus {y}: fi: end, Tmp1, Imp2);
Tmp4 := Tmp2 intersect Tmp3;
Den  := convert(map(proc(x)
        convert(x,`*`): end, Tmp1),`+`);
Num  := convert(map(proc(x)
          LoopSign(Imp1,Imp2,x,G)
         *convert(x,`*`): end, Tmp4),`+`);
RETURN(Num/Den):
end:
```

Example

In order to illustrate the execution of this program, the circuit depicted in Fig. 1 is defined and analysed by the following Maple code. Vertices and edges are added to the graph using the commands `addvertex`, and `addedge`. Edge names must begin with the letter "e". Edges can be specified as sets or lists of vertices. A list indicates a directed edge, while a set indicates an undirected edge. Current and voltage sources as components though or across which current or voltage are mesured are defined by directed edges.

```
> new(G):
> addvertex({a,b,c,d},G):
> addedge([b,a],names=eR0,G):
> addedge({a,c},names=eR1,G):
> addedge({a,d},names=eR2,G):
> addedge({c,b},names=eR3,G):
> addedge({d,b},names=eR4,G):
> addedge([c,d],names=eR5,G):

> KirchhoffThirdLaw(eR0,eR5,G);

  (-eR1*eR4+eR3*eR2)/(eR0*eR5*eR4
  +eR3*eR0*eR1+eR2*eR4*eR0+eR4*eR3*eR2
  +eR2*eR5*eR1+eR0*eR3*eR2+eR3*eR1*eR4
  +eR3*eR1*eR2+eR3*eR5*eR4+eR0*eR5*eR2
  +eR0*eR3*eR5+eR4*eR1*eR5+eR4*eR1*eR2
  +eR3*eR5*eR2+eR0*eR1*eR5+eR0*eR1*eR4)
```

Computation of fourth law (Eq. 4)

`Adm1` is the conductance of the branch across which the current source is placed. `Adm2` is the conductance of the branch across which the voltage is measured. `G` is the graph of the circuit.

```
KirchhoffFourthLaw := proc(Adm1,Adm2,G)
local  Tmp1,Tmp2,Tmp3,Tmp4,Num,Den:
Tmp1 := Trees(G);
Tmp2 := map(proc(x,y) if member(y, x) then
        x minus {y}: fi: end, Tmp1, Adm1);
Tmp3 := map(proc(x,y) if member(y, x) then
        x minus {y}: fi: end, Tmp1, Adm2);
Tmp4 := Tmp2 intersect Tmp3;
Den  := convert(map(proc(x)
        convert(x,`*`): end, Tmp1),`+`);
Num  := convert(map(proc(x)
          SplitTreeSign(Adm1,Adm2,x,G)
         *convert(x,`*`): end, Tmp4),`+`);
RETURN(Num/Den):
end:
```

Example

The circuit depicted in Fig. 4 is defined and analysed by the following Maple code.

```
> new(G):
> addvertex({a,b,c,d},G):
> addedge([b,a],names=eG0,G):
> addedge({a,c},names=eG1,G):
> addedge({a,d},names=eG2,G):
> addedge({c,b},names=eG3,G):
> addedge({d,b},names=eG4,G):
> addedge([d,c],names=eG5,G):

> KirchhoffFourthLaw(eG0,eG5,G);

  (-eG3*eG2+eG1*eG4)/(eG0*eG1*eG2
  +eG0*eG1*eG4+eG3*eG2*eG4+eG3*eG2*eG5
  +eG3*eG1*eG5+eG0*eG1*eG5+eG0*eG2*eG5
  +eG5*eG2*eG4+eG0*eG4*eG5+eG0*eG3*eG2
  +eG0*eG3*eG4+eG3*eG1*eG2+eG0*eG3*eG5
  +eG2*eG1*eG4+eG3*eG1*eG4+eG5*eG1*eG4)
```

9 Conclusion

The unrecognized Kirchhoff's third and fourth laws have been recalled in this paper. These laws allow one to compute network functions, of linear passive circuits, in straight formulas. The statement of these two laws have been detailed as they are stated in a formalism which is unusual to most scientists from the electronics comunity. A very concise Maple programs to compute them have been proposed.

References:

[1] W.-K. Chen. *Graph Theory and its Engineering Applications*. World Scientific, 1997.

[2] P.M. Lin. *Symbolic Network Analysis*. Elsevier, New York, 1991.

[3] L. Weinberg. Correction to "kirchoff's third and fourth laws". *IRE Transactions on Circuit Theory*, 5(2):139–139, June 1958.

[4] L. Weinberg. Kirchhoff's "third and fourth laws". *IRE Transactions on Circuit Theory*, 5(1):8–30, March 1958.

[5] Louis Weinberg. *Network Analysis and Synthesis*. Krieger Publishing Company, second edition, 1975.

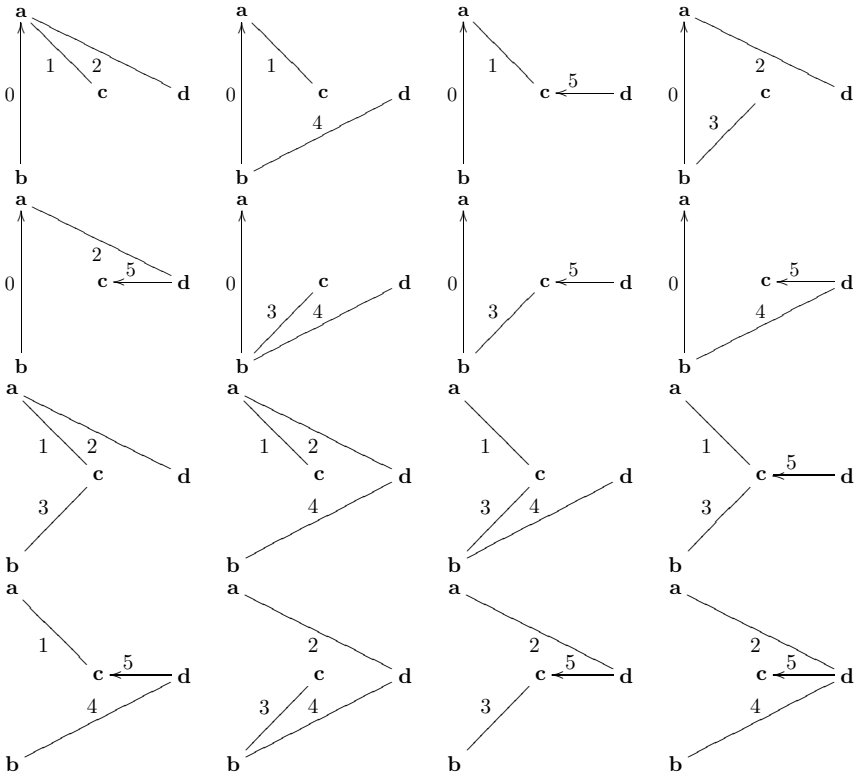


Fig. 7: Trees of network in Fig. 1 or 4.

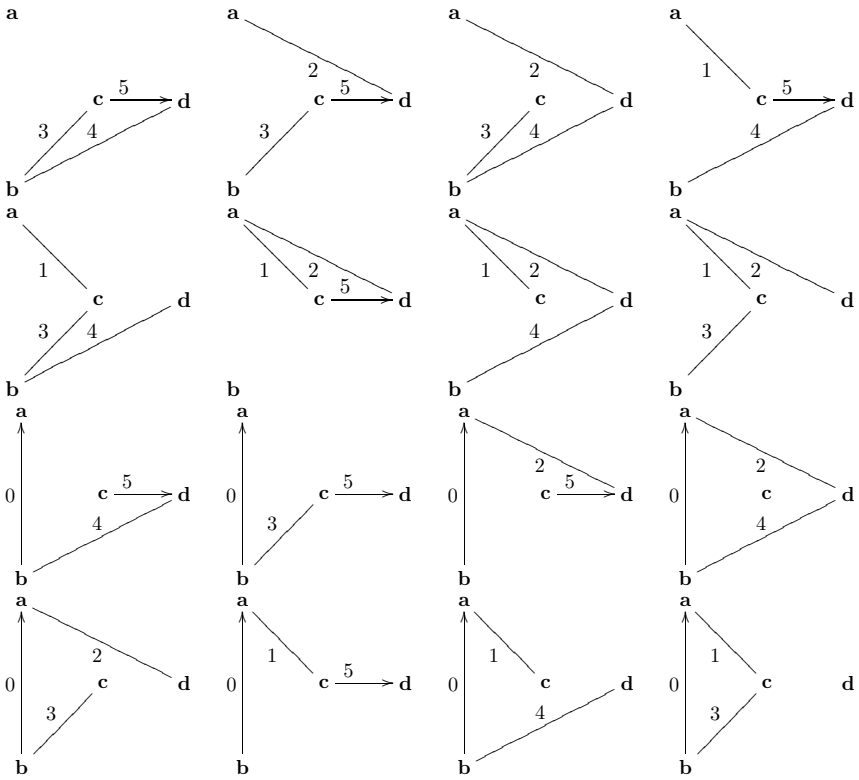


Fig. 8: CoTrees of network in Fig. 1 or 4.