Generation of the k-trees of a graph

By I. PÁVÓ

Abstract

We present a new procedure that generates all k-trees of a graph in each component of which a vertex is given in advance. Our method makes use of a theorem of Ore [7] concerning finite directed graphs, thus providing an application of this theorem that beyond its theoretical interest can be used in practical analysis of electrical networks.

Introduction

Topological formulas are nowadays playing an ever increasing role in the analysis of electrical networks (see [8]). In the applications of such formulas, however simple they are, the question immediately arises how to generate all the trees, and also the 2-trees satisfying certain requirements, of a given graph.

To overcome this problem several methods were proposed in the last decade. In principle, the most simple way to produce all the trees of a graph G with n vertices would be to scan all the sets containing $n-1$ edges and dispose of those not eligible. Naturally for practical purposes such a procedure would be too lengthy and intricate. A procedure usable also in practice was devised by Hakimi and Green [1] and solves the problem by splitting the graph in two parts the trees of which are assumed to be known. From the trees of these subgraphs the trees of the starting one can be composed and also k-trees satisfying certain requirements can be pinpointed. This procedure is, however, also lengthy and cumbersome; indeed, to carry out the splitting and composition of trees is in itself a complicated algorithm, and it must be repeated also for the subgraphs obtained. A similar procedure was devised by Mayeda [3].

Other procedures were designed by Talbot (a new set of topological formulas) and by Mayeda and Seshu [4]. A common feature of these methods is that they choose an arbitrary tree as a starting one and generate the others from this by edge transformations. (After deleting an edge of the starting tree another edge of the graph is substituted, and then the same procedure is applied to the obtained tree.) These methods seem to have certain advantages but we must note that a recursive formula is used that is, from the viewpoint of computational technique, difficult to handle.

The most feasible method for practical use that has been developed up to now

is perhaps that due to Maxwell and Cline [5]. This method is of algebraic character and, to its advantage, is simple, easy to understand, and adaptable to digital computers relatively easily. A snag is, however, that it can be used only to generate k-trees with $k=1$ or 2 and it uses up a relatively large internal storage capacity.

In the present paper we are going to present a new procedure that generates the k-trees, satisfying certain requirements, of a graph; the procedure in the case $k=1$ generates all the trees. Our method is about as simple as the algebraic method mentioned above is, but it can be applied under more general circumstances; namely, it can be used to handle the case $k \geqq 2$ too. It can also be ascribed to its advantage that, fed into computer, it needs considerably less internal storage room than the algebraic method mentioned. The procedure, as presented here, concerns only the generation of k-trees of graphs without multiple edges, but there is, in principle, no difficulty in extending it so as to apply to graphs with multiple edges. Our considerations are based on a well-known theorem of Ore [7] on finite directed graphs.

1. Basic concepts and definitions

Consider a graph with n vertices $\mathbf{P}_1, \ldots, \mathbf{P}_n$ and select arbitrarily a number k $(1 \leqq k \leqq n)$ from among them, these being denoted by $\mathbf{P}_{i_1}, \ldots, \mathbf{P}_{i_k}$ $(1 \leqq i_1 < \cdots < i_k \leqq n)$.

Definition. A *k-tree* $F^k_{i_1, \ldots, i_k}$ of the graph G is an arbitrary graph satisfying the following stipulations:

1. it is a subgraph of G,
2. it contains all the vertices of G,
3. it consists of exactly k connected components,
4. each component contains exactly one of the selected vertices $\mathbf{P}_{i_1}, \ldots, \mathbf{P}_{i_k}$,
5. each of its components are a tree.

In particular, as seen from the definition, $F^1_{i_1}$ denotes a tree of the graph G; for the sake of simplicity we may sometimes drop the lower index and write only F^1. The purpose of this note is to study the generation of k-trees conforming to the above definition.

Let $\mathbf{M}$ be a matrix of size $n \times n$ with the following properties:

(I) Every element of $\mathbf{M}$ equals either 0 or 1,

(II) each row of $\mathbf{M}$ contains at least one element equal to 1.

Consider also a matrix $\mathbf{M}_{i_1, \ldots, i_k}$, where $1 \leqq i_1 < \cdots < i_k \leqq n$, of size $n \times n$ with the following properties:

(I′) the i_j-th row of $\mathbf{M}_{i_1, \ldots, i_k}$ consists purely of zeros $(j=1, \ldots, k)$,

(II′) in other rows of $\mathbf{M}_{i_1, \ldots, i_k}$ there is exactly one element equal to 1 and otherwise these rows too consist purely of zeros, and finally

(III′) at every place where $\mathbf{M}_{i_1, \ldots, i_k}$ contains a 1 the matrix $\mathbf{M}$ too does so.

Definition. If $\mathbf{M}_{i_1, \ldots, i_k}$ satisfies conditions (I′), (II′) and (III′) then we call it an *$(i_1, \ldots, i_k)$-reduction of* $\mathbf{M}$.

Suppose $\mathbf{M}_{i_1, \ldots, i_k}$ satisfies conditions (I′) and (II′), and write $\mathbf{M}_{i_1, \ldots, i_k} = (a_{ij})$, a_{ij} being the element in the intersection of the ith row and jth column of the matrix in question.

Definition. The function

$$\varphi(x) = \begin{cases} j \text{ if } x \neq i_1, \ldots, i_k \text{ and } a_{xj} = 1, \\ 0 \text{ if } x \text{ coincides with one of} \\ \text{the numbers } i_1, \ldots, i_k \end{cases}$$

is called *the function associated with the matrix* $\mathbf{M}_{i_1, \ldots, i_k}$.

As seen, the domain of $\varphi(x)$ is the set $\{1, \ldots, n\}$ and its range is a subset of $\{0, 1, \ldots, n\}$. Shortly, this function makes correspond to each row index an integer equalling zero unless the row with the considered index contains an element equal to 1, in which latter case this integer coincides with the column index of this unique element. It is clear that the correspondence between the matrices $\mathbf{M}_{i_1, \ldots, i_k}$ and the functions associated with them is one-to-one.

This observation is important since it shows that it is possible to characterize the matrix in question with the aid of the row vector $(\varphi(1), \ldots, \varphi(n))$. This characterization will be called *the row vector representation of the matrix* $\mathbf{M}_{i_1, \ldots, i_k}$.

In what follows, both directed and undirected graphs may turn up. Unless otherwise stated, loops are not, in general, admitted. In addition, undirected graphs are assumed to have no multiple edges. Undirected edges will be viewed as pairs of edges directed in opposite directions that connect the same pair of vertices.

To a graph containing the vertices $\mathbf{P}_1, \ldots, \mathbf{P}_n$ make correspond a matrix $\mu(G)$ of size $n \times n$ that in the intersection of the ith row and the jth column contains a 1 if and only if the vertices $\mathbf{P}_i$ and $\mathbf{P}_j$ in this order are connected by a directed edge, the remaining elements of the matrix being equal to zero.

Definition. The matrix $\mu(G)$ is said to be *the adjacency matrix* of the graph G.

It is easily checked that the correspondance $G \to \mu(G)$ between graphs, containing the vertices $\mathbf{P}_1, \ldots, \mathbf{P}_n$, that have only single edges (loops being allowed too) and matrices of type $n \times n$ containing only 0 or 1 as elements is one-to-one. For graphs without loops the adjacency matrix contains purely zeros in its diagonal. Undirected graphs have symmetrical adjacency matrices.

By a well-known theorem of Ore ([7], [6], [2]), a directed graph, possibly with loops, has the property that at each of its vertices exactly one of the edges is directed outwards if and only if it satisfies the following three conditions:

(a) each component of the graph contains exactly one circuit (this may possibly be a loop),

(b) in this unique circuit the edges are directed cyclically,

(c) in each component the edges that do not belong to its circuit are directed towards this circuit.

Definition. An undirected graph without loops is called *a generalized tree* if each of its connected components contains at most one circuit.

In particular a k-tree is a generalized tree.

To introduce a useful notation, for a directed graph G we shall denote by $v(G)$ the undirected graph obtained by retaining undirected edges between those pairs of vertices that are connected in G in at least one direction.

2. Some properties of the graphs $F^k = v(\mu^{-1}(M_{i_1,\dots,i_k}))$

Consider a simple graph, i.e. an undirected graph without loops and multiple edges, that contains the vertices $\mathbf{P}_1, \dots, \mathbf{P}_n$ ($n \geqq 1$), none of which is isolated, and write $\mathbf{M} = \mu(G)$. Fix the integers $i_1, \dots, i_k$ ($1 \leqq i_1 < \dots < i_k \leqq n$) and let the matrix $\mathbf{M}_{i_1,\dots,i_k}$ run over all the $(i_1, \dots, i_k)$-reductions of $\mathbf{M}$.

Theorem 1. For every graph F^k of form $v(\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k}))$ each of the following five assertions hold:

(A) F^k is a subgraph of G,

(B) F^k is a generalized tree,

(C) the vertices $\mathbf{P}_{i_j}$ belong to mutually distinct components of F^k ($j = 1, \dots, k$),

(D) if a component of F^k contains any of the P_{i_j}'s then this component is a tree (which may possibly degenerate to a single vertex),

(E) those components of F^k not containing any of the vertices $\mathbf{P}_{i_j}$ contain at most one circuit and at least one edge each.

Conversely, if a graph F^k satisfies conditions (A), (B), (C), (D), and (E) then it can be represented in at least one way in the form $F^k = v(\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k}))$.

Proof. Choose an arbitrary $(i_1, \dots, i_k)$-reduction $\mathbf{M}_{i_1,\dots,i_k}$ of $\mathbf{M}$. It is obvius that $F^k = v(\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k}))$ is a subgraph of G.

Construct a matrix $\mathbf{M}'_{i_1,\dots,i_k}$ from $\mathbf{M}_{i_1,\dots,i_k}$ by writing 1 in the intersection of the i_j-th row and the i_j-th column instead of 0 for every $j = 1, \dots, k$. Then $\mu^{-1}(\mathbf{M}'_{i_1,\dots,i_k})$ is a directed graph such that to each of its vertices there is exactly one edge incident that is directed outwards. Applying Ore's theorem we obtain that every component of $\mu^{-1}(\mathbf{M}'_{i_1,\dots,i_k})$ contains exactly one circuit or loop. By transition from $\mu^{-1}(\mathbf{M}'_{i_1,\dots,i_k})$ to $\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k})$ we must delete exactly k loops; thus the graph $\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k})$ turns indeed out to be a generalized tree. Since the loops were deleted precisely at the vertices $\mathbf{P}_{i_j}$, we obtain (D) too ($j = 1, \dots, k$).

Furthermore, since to each vertex $\mathbf{P}_{i_j}$ in $\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k})$ there is a loop incident, Ore's cited theorem also implies that each component of F^k may contain at most one of the vertices $\mathbf{P}_{i_j}$; therefore (C) is also established.

Finally, consider those components of F^k containing none of the points $\mathbf{P}_{i_j}$, and consider simultaneously also the corresponding componentes in $\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k})$. These latter contain exactly one directed circuit each. Passing back to $\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k})$, it is clear that the component in question of $\mu^{-1}(\mathbf{M}_{i_1,\dots,i_k})$ contains this circuit; moreover, since a directed circuit contains at least two distinct vertices, it is guaranteed that this component of F^k contains at least one edge. This completes the proof of (E).*

For the proof of the converse of the theorem assume in what follows that F^k satisfies conditions (A), (B), (C), (D), and (E). To accomplish the proof we

* It may happen that a component of F^k is, despite the fact that it contains none of the vertices $\mathbf{P}_{i_j}$, a tree. This is the situation if the subgraph in $\mu^{-1}(\mathbf{M}'_{i_1,\dots,i_k})$ corresponding to this component contains a directed circuit consisting only of two vertices; by transition to F^k, the two edges, directed in opposite directions, that are incident to both of these edges reduce to one single edge, and the component of F^k in question does not contain any circuit.

associate with F^k a directed graph $F^{k'}$ for which the one hand $F^{k'} = v(F^{k'})$ holds, and the other hand for which $\mu(F^{k'})$ coincides with one of the graphs $\mathbf{M}_{i_1,\ldots,i_k}$.

Introduce a directing of the edges in each component of F^k by abiding by the following rules:

1. If the component in question contains a vertex $\mathbf{P}_{i_j}$ and this $\mathbf{P}_{i_j}$ is not an isolated point then direct the edges of this component towards $\mathbf{P}_{i_j}$. (C) and (D) provides for the unique possibility of this.

2. If the component in question contains a circuit then directed the edges of this latter cyclically and the other edges towards the circuit. Such a directing is made possible by (B).

3. In the remaining cases, i.e. when the considered component contains neither a vertex $\mathbf{P}_{i_j}$ nor a circuit then it contains at least one edge. Let us choose an edge and replace it by two edges directed in opposite directions and, if there exist any, direct the other edges towards the circuit constructed just now.

In this way we obtain a directed graph $F^{k'}$ for which $F^k = v(F^{k'})$ obviously holds.

Consider now the directed graph $F^{k''}$ obtained by adding loops at each point $\mathbf{P}_{i_j}$ of $F^{k'}$. To this $F^{k''}$ we can apply Ore's theorem. We derive that each row of the matrix $\mu(F^{k''})$ contains exactly one 1. By deleting the loops of $F^{k''}$ we can pass back to $F^{k'}$; in terms of the adjacency matrices this amounts to replacing the ones by zeros in each of the rows i_j of $\mu(F^{k''})$. The obtained matrix $\mu(F^{k'})$ is then easily seen to be an $(i_1, \ldots, i_k)$-reduction of the matrix $\mathbf{M} = \mu(G)$. The proof is complete.

From the above proof it becomes clear that to a graph F^k there correspond, in general, several $(i_1, \ldots, i_k)$-reductions. The ambiguity of the construction of these reductions lies in steps 2 and 3 above, where for the directing of the edges there are, in general, several possibilities.

We mention two more interesting properties of the graphs F^k featuring in Theorem 1:

F^k contains at least k components and at most $n-k$ edges.

The remark on the minimal number of components is an easy consequence of (D). That on the maximal number of edges follows from the fact that $F^{k'}$ contains exactly $n-k$ directed edges. Therefore the properties of the correspondance v imply that F^k cannot contain more than $n-k$ edges.

To require that the graph G contains no isolated points is necessary for the existence of an $(i_1, \ldots, i_k)$-reduction for any selection of $i_1, \ldots, i_k$. Still, the assumption on isolated points can be eased if we extend the notion $(i_1, \ldots, i_k)$-reduction for matrices $\mathbf{M}$ that possibly contain rows consisting purely of zeros. In any case, the maximal number of such rows must be limited to k and all such rows must be covered by those of incides $i_1, \ldots, i_k$.

As we pointed out above, the generalized tree F^k in Theorem 1 cannot be represented unambiguously in form $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$. Nevertheless, in case this generalized tree is a k-tree of G, this representation is unambiguous. More precisely, we have the following.

Theorem 2. Assume G is a simple graph without isolated points and with vertices $\mathbf{P}_1, \ldots, \mathbf{P}_n$, and select fixed vertices $\mathbf{P}_{i_1}, \ldots, \mathbf{P}_{i_k}$, where $1 \leqq i_1 < \cdots < i_k \leqq n$. Then the k-tree $F^k_{i_1,\ldots,i_k}$ can be represented unambiguously in form $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$, where $\mathbf{M}_{i_1,\ldots,i_k}$ is a suitable $(i_1, \ldots, i_k)$-reduction of $\mu(G)$.

Proof. That there exists at least one representation of the desired kind follows from the converse Theorem 1 since $F^k_{i_1,\dots,i_k}$ obviously satisfies conditions (A), (B), (C), (D) and (E) of Theorem 1.

In the proof that there exists no more than one representation we may assume $k<n$. Contrary to what we want to prove, suppose that $F^k_{i_1,\dots,i_k}=v(\mu^{-1}(\mathbf{M}^1_{i_1,\dots,i_k}))=v(\mu^{-1}(\mathbf{M}^2_{i_1,\dots,i_k}))$, where $\mathbf{M}^1_{i_1,\dots,i_k}$ and $\mathbf{M}^2_{i_1,\dots,i_k}$ are two different $(i_1,\dots,i_k)$-reductions of $\mu(G)$. We are going exhibit a contradiction.

On account of the assumption that the two matrices are distinct, there exists a $j\neq i_1,\dots,i_k$ $(1\leqq j\leqq n)$ such that in the jth row of $\mathbf{M}^1_{i_1,\dots,i_k}$ the l_1th element is one whereas in the same row of $\mathbf{M}^2_{i_1,\dots,i_k}$ the l_2th element is one, l_1 and l_2 being unequal and both being different from j $(l_1, l_2=1,\dots,n)$. This means that $\mathbf{P}_j$, $\mathbf{P}_{l_1}$, and $\mathbf{P}_{l_2}$ all belong to the same component of $F^k_{i_1,\dots,i_k}$. Assume that $\mathbf{P}_{i_m}$ is the unique one of the selected vertices that also belongs this component $(m=1,\dots,k)$. Then in $\mu^{-1}(\mathbf{M}^1_{i_1,\dots,i_k})$ there exists a directed path leading from $\mathbf{P}_j$ to $\mathbf{P}_{i_m}$ through $\mathbf{P}_{l_1}$, and in $\mu^{-1}(\mathbf{M}^2_{i_1,\dots,i_k})$ there is one leading from $\mathbf{P}_j$ to $\mathbf{P}_{i_m}$ through $\mathbf{P}_{l_2}$. This means that there exist two different paths in some component of $F^k_{i_1,\dots,i_k}$ that connect $\mathbf{P}_j$ with $\mathbf{P}_{i_m}$, contradicting the assumption that $F^k_{i_1,\dots,i_k}$ is a k-tree of G. The proof is complete.

Similarly to what was said after the proof of Theorem 1 the stipulations imposed on G can be eased also here: it is enough to assume that G contains at most k isolated points and these are among the selected ones.

In case $k=1$ we obtain interesting particular cases of Theorem 1 and 2:

Theorem 3. Assume G is a simple graph with vertices $\mathbf{P}_1,\dots,\mathbf{P}_n$, none of which is isolated. Write $\mathbf{M}=\mu(G)$. Fix an integer i, $1\leqq i\leqq n$, and let the matrix $\mathbf{M}_i$ run over all (i)-reductions of $\mathbf{M}$. Then for each graph F of form $v(\mu^{-1}(\mathbf{M}_i))$ the following four assertions hold:

(A) F is a subgraph of G,

(B) F is a generalized tree,

(C) the component containing $\mathbf{P}_i$ of F is a tree, which may possibly degenerate to an isolated point; and finally,

(D) those components of F not containing $\mathbf{P}_i$ contain at least one edge and at most one circuit each.

Theorem 4. Assume $\mathbf{P}_i$ is an arbitrary but fixed point of the connected simple graph G, where $1\leqq i\leqq n$, and F^1 is a tree in G. Then $\mathbf{M}=\mu(G)$ has precisely one (i)-reduction $\mathbf{M}_i$ such that $F^1=v(\mu^{-1}(\mathbf{M}_i))$ holds.

It is necessary to stipulate in this theorem that G is connected since otherwise it would not contain any tree at all.

3. An algorithm to generate the k-trees

Keeping an eye on Theorem 2, we want to generate all the k-trees $F^k_{i_1,\dots,i_k}$ of G by forming all $(i_1,\dots,i_k)$-reductions of $\mu(G)$. Among these there will turn up those representing the k-trees exactly once. Apart from the k-trees these reduced matrices will represent other generalized trees F^k satisfying conditions (A), (B),

(C), (D), and (E). In the sequel we are going to construct a procedure that selects those producing k-trees $F^k_{i_1,\ldots,i_k}$ from among all the $(i_1, \ldots, i_k)$-reductions of $\mu(G)$.

So our procedure will enable us to sift out from among the mentioned generalized trees F^k the k-trees $F^k_{i_1,\ldots,i_k}$, i.e. those generalized trees satisfying (A), (B), (C), (D), and (E) in Theorem 1 that contain no circuit in any of their components.

To start with the description of our method, consider the sets $\{F^k_{i_1,\ldots,i_k}\}$ and $\{\mathbf{M}_{i_1,\ldots,i_k}\}$, where the elements of the second set denote the $(i_1, \ldots, i_k)$-reductions of the matrix $\mathbf{M}=\mu(G)$. As seen from Theorem 2, all the k-trees $F^k_{i_1,\ldots,i_k}$ occur exactly once among the graphs $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$. Also, we observed earlier in Section 1 that the elements of the second set can be given in row vector representation. So in the way described in Theorem 2, the set $\{F^k_{i_1,\ldots,i_k}\}$ can be mapped in a one-to-one way into the set $\{(\varphi(1), \ldots, \varphi(n))\}$ of row vectors, φ running over the functions associated with any of the $(i_1, \ldots, i_k)$-reductions $\mathbf{M}_{i_1,\ldots,i_k}$. We shall describe a procedure that selects those vectors being in the range of the mapping just described. The selected row vectors $(\varphi(1), \ldots, \varphi(n))$ will be those representing the k-trees $F^k_{i_1,\ldots,i_k}$ of the graph G.

Consider a matrix $\mathbf{M}_{i_1,\ldots,i_k}$ satisfying properties (I'), (II'), and (III') described in Section 1 and let φ be the function associated with this matrix.

Definition. By *a cycle check* performed on the matrix $\mathbf{M}_{i_1,\ldots,i_k}$ starting with the integer x we mean the construction of the sequence

$$x, \varphi(x), \varphi(\varphi(x)), \ldots \qquad (1 \leqq x \leqq n).$$

We say that the outcome of the cycle check is finite if we can construct only a finite sequence, i.e. if somewhere in the sequence a zero turns up, which does not belong to the domain of φ; otherwise we say that the outcome of the cycle check is infinite.

If the outcome cycle check is infinite then, as is easily seen, from a certain point the same segment of the above sequence will occur repeatedly.

Definition. By *a complete cycle check* performed on the matrix $\mathbf{M}_{i_1,\ldots,i_k}$ we mean a bunch of cycle checks starting with the integers $1, \ldots, n$ respectively. The outcome of a complete cycle check is said to be finite if all checks constituting it have finite outcomes; otherwise, the outcome is said be infinite.

Now we are going to study the cycle checks from an aspect that will have some importance for our later purposes. To this end, consider a matrix $\mathbf{M}_{i_1,\ldots,i_k}$ and a cycle check performed on it that starts with the integer x. The construction of the sequence $x, \varphi(x), \varphi(\varphi(x)), \ldots$ can be regarded as starting at a vertex $\mathbf{P}_x$ of the directed graph $\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k})$ and walking through a part of the graph, always proceeding in conformity with the direction of the edges passed along. The sequence obtained by a cycle check coincides with the sequence of vertices passed through during such a walk. In case of a cycle check with finite outcome, after a certain time we arrive at a vertex out of which there does not lead any edge. In case of infinite outcome we get into a directed circuit. during the walk.

To be assured that the outcome of a complete cycle check performed on a given matrix is finite we must perform all the n cycle checks constituting this complete check; we may, however, stop earlier if we happen to find an infinite check among these, because this already implies the infiniteness of the complete check.

Theorem 5. Assume G is a simple graph with vertices $\mathbf{P}_1, \ldots, \mathbf{P}_n$, none of which is isolated, and $\mathbf{M}_{i_1,\ldots,i_k}$ is an $(i_1, \ldots, i_k)$-reduction of $\mu(G)$. Then the complete cycle check performed on the matrix $\mathbf{M}_{i_1,\ldots,i_k}$ is of finite outcome if and only if $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$ is a k-tree $F^k_{i_1,\ldots,i_k}$ of the graph G.

Proof. To verify the "if" part assume that $F^k_{i_1,\ldots,i_k}$ is a k-tree of the graph G. According to Theorem 2 there exists a reduction $\mathbf{M}_{i_1,\ldots,i_k}$ such that $F^k_{i_1,\ldots,i_k} = v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$ holds. Furthermore, in each component of the directed graph $\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k})$ all the edges are directed towards the corresponding vertex $\mathbf{P}_{i_j}$ $(j=1, \ldots, k)$. These imply that the cycle check performed on the matrix $\mathbf{M}_{i_1,\ldots,i_k}$ that starts with an arbitrary $i=1, \ldots, n$ is of finite outcome. Therefore the complete cycle check is also of finite outcome.

To prove the "only if" part assume, that the complete cycle check performed on the matrix $\mathbf{M}_{i_1,\ldots,i_k}$ is of finite outcome. Then by an observation made above on cycle checks we see that none of the components of the directed graph $\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k})$ contains a directed circuit.

This entails that the graph $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$ does not contain any circuit. Indeed, assume the contrary, i.e. that this graph does contain some circuit. Consider the directed subgraph of $\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k})$ corresponding to this circuit. This subgraph cannot be a directed circuit; thus it contains at least one vertex with two edges incident to it that are directed outwards. This contradicts the definition of the matrix $\mathbf{M}_{i_1,\ldots,i_k}$.

Moreover, we can derive that the number of edges of $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$ is $n-k$. In fact, this is obvious for $\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k})$. This latter graph, as was pointed out above, does not contain any directed circuit; so, in particular, it does not contain a directed circuit with two edges. Therefore, the correspondance v does not reduce the number of edges.

We obtained that $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$ is a circuit-free graph with $n-k$ edges, and this means that it is indeed a k-tree. Taking Theorem 2 into account, we see that this k-tree can be represented in form $F^k_{i_1,\ldots,i_k}$. The proof is complete.

From Theorem 2 in Section 1 and Theorem 5 in Section 2 we obtain the following algorithm for the generation k-trees $F^k_{i_1,\ldots,i_k}$ of graph G:

I. Construct the adjacency matrix $\mathbf{M}=\mu(G)$ of the graph G.

II. Form all $(i_1, \ldots, i_k)$-reduction $\mathbf{M}_{i_1,\ldots,i_k}$ of $\mathbf{M}$.

III. Perform a complete cycle on the matrices $\mathbf{M}_{i_1,\ldots,i_k}$. Those leading to finite outcomes give the desired k-trees in form $v(\mu^{-1}(\mathbf{M}_{i_1,\ldots,i_k}))$.

(If the graph G has no k-trees at all this will turn out by performing the algorithm since in this case all the cycles have infinite outcomes.)

The algorithm described can be extended so as to apply to the search of the k-trees of a graph G with multiple edges by keeping in mind the following:

Whenever two points of the graph G is connected with more than one edges we replace these with one single edge and call the number of edges replaced the multiplicity of this single one. By performing the above algorithm on the obtained graph G' we get its k-trees. Each of the k-trees of G' that contains no substituted edges is a k-tree of G too.

Now consider the k-trees of G' that contain also substituted edges. By taking

into consideration the substituted edges with their multiplicities we obtain the k-trees of G. From a k-tree in G' we can derive as many k-trees in G as the product of the multiplicity of its edges.

This method provides a way for generating the k-trees of a graph G with multiple edges.

In case $k=1$ our algorithm gives all the trees of a graph G. On account of Theorem 4 we see that the generation of the trees can proceed in n different ways depending on which of the n vertices of G we choose as $\mathbf{P}_i$. We obviously obtain all the trees independently of this choice. This enables us to deliberate as to which of the n ways is the simplest from the angle of computational technique. This problem will be touched upon in the next section.

4. Remarks on adaptation to computers and model examples

We should like to add some computation-technical observations concerning the algorithm outlined in the previous section on the generation of the trees $F^k_{i_1,\ldots,i_k}$ of a graph.

To start with, it seems practical to perform the first two steps of the algorithm immediately on the row vector representation of the considered reduction $\mathbf{M}_{i_1,\ldots,i_k}$. To make this possible we introduce the notion of generating matrix.

Let $\mathbf{M}=\mu(G)=(a_{ij})$ be the adjacency matrix of the graph G.

Definition. The generating matrix $\mathbf{M}_G=(b_{ij})$ associated with the graph G is determined by the formula

$$b_{ij}=\begin{cases}0 & \text{if} \quad a_{ij}=0,\\ j & \text{if} \quad a_{ij}=1.\end{cases}$$

Now the row vector representation of a $(i_1, \ldots, i_k)$-reduction $\mathbf{M}_{i_1,\ldots,i_k}$ of the adjacency matrix $\mathbf{M}=\mu(G)$ of the graph G can be obtained from $\mathbf{M}$ by choosing an element from each row of $\mathbf{M}$ in the following manner:

If the index j of the row considered is different from all i_l $(l=1, \ldots, k)$ then let the choosen element be different from zero, and if $j=i_l$ from some l then choose a 0 (e.g. the element in the diagonal).

Now the cycle check can be performed on the row vector obtained accordingly.

In several cases computational short-cuts can be made in cycle checking. For example, it is easy to see that it is not necessary to start a cycle check at elements which were arrived at in earlier cycle checks. Moreover, if the complete cycle check has infinite outcome we may stop when we stumble upon the first cycle check with infinite outcome.

In principle, it is irrelevant which vertex $\mathbf{P}_i$ we fix when generating the trees of a graph G. In practice, the most clever choice seems to be the one for which the ith row of the generating matrix $\mathbf{M}_G$ contains the largest number of elements different from zero. It can furthermore be observed that the cycle check may have a finite outcome only if the number i occurs in the row vector representation of the matrix $\mathbf{M}_i$.

The remarks made here enable us to sift out those matrices $\mathbf{M}_i$ that are to be

used for cycle checks. Many of the above remarks (e.g. that given in the last sentence of the previous paragraph) can be modified in an obvious way so as to apply also to the case $k \geqq 2$.

Example. Consider the graph G in Fig. 1. We are going to generate all its trees. The generating matrix $\mathbf{M}_G$ of G takes the form

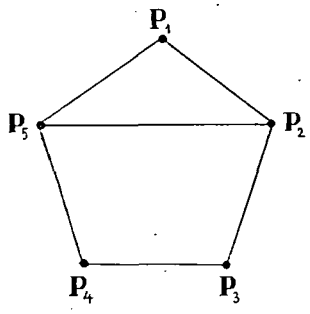

Fig. 1. A simple graph

$$\mathbf{M}_G = \begin{pmatrix} 0 & 2 & 0 & 0 & 5 \\ 1 & 0 & 3 & 0 & 5 \\ 0 & 2 & 0 & 4 & 0 \\ 0 & 0 & 3 & 0 & 5 \\ 1 & 2 & 0 & 4 & 0 \end{pmatrix}.$$

Looking at this matrix we see that it is perhaps the best to choose the vertex $\mathbf{P}_5$ as $\mathbf{P}_i$. If we construct the row vector representations of the $\mathbf{M}_5$ reductions then, according to the remarks made above, we obtain 20 row vectors. Performing cycle checks on these we obtain that 9 of these vectors do not represent any trees. The final result is 11 trees, which are in turn in row vector representation:

(23540), (25230), (25250), (25450),
(51230), (51250), (51430), (51450),
(53450), (55230), (55250).

Observe that from a row vector representation we can easily pass to the actual tree. To do this imagine another row, consisting of the elements $1, 2, \ldots, n$, placed above the row vector representation of $\mathbf{M}_i$; disregarding the one column containing zero, the remaining colums indicate the pairs of vertices that are connected in the tree in question. For example, the tree represented by the row vector (55230) can be seen in Fig. 2.

row vector representation:
(55230)

"completed" row vector:

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 5 & 2 & 3 & 0 \end{bmatrix}$$

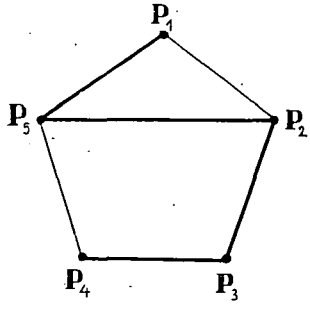

Fig. 2. A tree of the graph of Fig. 1.

As a further example we shall now generate the trees $F^3_{1,2,4}$ of the graph G given in Fig. 1.

Again, we construct the row vector representations of all the eligibe ones of the matrices $\mathbf{M}_{1,2,4}$:

(00201), (00202), (00204), (00401),

(00402), (00404).

Now performing cycle checks on the matrices corresponding to the rows enumerated we obtain finite outcomes in all six cases.

This means that in this case the number of the 3-trees is 6. Fig. 3 illustrates the 3-tree corresponding to the vector (00404).

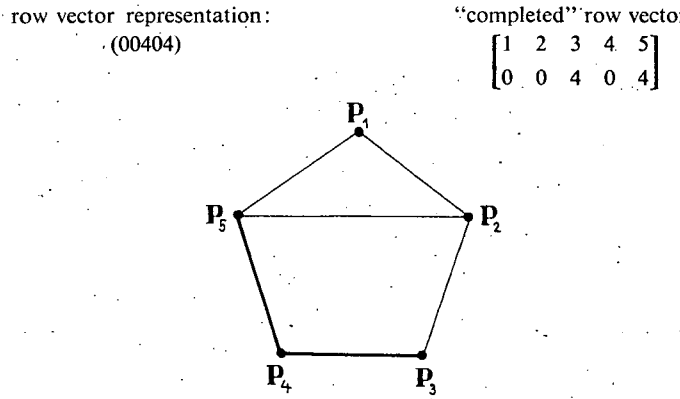

Fig. 3. A 3-tree of the graph of Fig. 1.

The advantages of this method in comparison to others for the generation of *k*-trees of a graph G seems clear if the method is adapted to computer. Namely, the method described in Section 2 is perhaps the most easily fed into computers among the known tree-generation methods. It is also clear that the storage capacity occupied by a programme based on this method is considerably smaller than that needed for the performing of a programme using e.g. the algebraic method [5]. The reason for this is that it is not necessary to store the data representing the tree for further operations, the cycle check decides immediately whether the obtained data (row vector) in fact represent a tree. This is a considerable advantage if we take into account that in pratice the number of the trees can be of magnitude order of several millions [4]. If not programmed clumsily, the complete cycle check does not increase the computing time unfavorably in comparison with time used up by the algebraic method.

RESEARCH GROUP ON MATHEMATICAL LOGIC
AND THEORY OF AUTOMATA OF THE
HUNGARIAN ACADEMY OF SCIENCES,
SOMOGYI BÉLA U. 7, SZEGED, HUNGARY

References

[1] HAKIMI, S. L. & D. G. GREEN, Generation and realization of trees and k-trees, *IEEE. Trans.*, v. CT—11, 1964, pp. 247—255.
[2] HARARY, F., The number of functional digraphs, *Math. Annales*, v. 138, 1959, pp. 203—210.
[3] MAYEDA, W., Reducing computation time in the analysis of networks by digital computer, *IRE Trans.*, v. CT—6, 1959, pp. 136—137.
[4] MAYEDA, W. & S. SESHU, Generation of trees without duplications, *IEEE Trans.*, v. CT—12, 1965, pp. 181—185.
[5] MAXWELL, L. M. & JR. J. M. CLINE, Topological network analysis by algebraic methods, *Proc. IEEE*, v. 113, 1966, pp. 1344—1347.
[6] ORE, O., *Graphs and Correspondences*, Festschrift f. d. 60. Geburtstag von A. Speiser, 1945, pp. 184—191.
[7] ORE, O., *Theory of Graphs*, Am. Math. Soc., Providence, 1962. pp. 68—74.
[8] SESHU, S. & M. B. REED, *Linear graphs and electrical networks*, Addison-Wesley Publ. Co., Massachusette, 1961.

RESUMÉ

Dans cette petite Note nous allons présenter une procedure nouvelle engendrant tous les k-arbres d'une graphe avec un point donné dans chacun de ses composantes connexes. Notre méthode exploite un théorème d'Ore [7] concernant les graphes finites et directées, ainsi rendant une application de ce théorème qui est, hors de son intérêt théorique, aussi utile dans l'analyse des secteurs électriquies pour des buts pratiques.

(Received March 9, 1970)