Optimum Branchings*

Jack Edmonds

Institute for Basic Standards, National Bureau of Standards, Washington, D.C. 20234

To Professor Marcel Riesz on His 80th Birthday

(November 16, 1966)

An arborescence $T$ is a tree whose edges are directed so that each is directed toward a different node. Exactly one node of $T$, called the root, has no edge of $T$ directed toward it. Let $G$ be any directed graph with a real numerical weight on each edge. A good algorithm is described for finding in $G$ (if there is one) a spanning arborescence, with prescribed root, whose edges have maximum (or minimum) total weight.

Key Words: Algorithms, arborescences, branchings, combinatorics, graphs, linear programming, traveling salesman, trees.

Section 1

A (directed) graph $G$, for purposes here, is a finite set of nodes and a finite set of edges, where each edge is said to be directed toward one of the nodes, called the front end of the edge, and said to be directed away from a different one of the nodes, called the rear end of the edge. An edge and each of its ends are said to meet. A subgraph of $G$ is a subcollection of its members which, under the same incidence relations, is a graph. A graph is called connected if it is not empty and its members do not partition into two disjoint nonempty subgraphs. A polygon is a connected graph $Q$ such that each node of $Q$ meets exactly two edges of $Q$. An (elementary uniformly directed) circuit is a polygon which contains one edge directed toward, and one edge directed away from, each of its nodes. A forest is a graph which contains no polygon. A tree is a connected forest. A branching is a forest whose edges are directed so that each is directed toward a different node. An arborescence is a connected branching. An (elementary uniformly directed) path $P$ is an arborescence such that each edge in $P$ is directed away from a different node, and such that there is at least one edge in $P$.

We shall occasionally use "obvious" facts about graphs without justifying them.

Clearly, a branching (forest) is the union of a unique family of disjoint arborescences (trees).

Exactly one node in an arborescence $T$, called the root of $T$, has no edge of $T$ directed toward it. A branching (forest) is an arborescence (tree) if and only if it has exactly one less edge than nodes. No branching (forest) has more edges than this.

In a path $P$ there are exactly two nodes, called the ends of $P$, which each meet only one edge in $P$. The rest of the nodes in $P$ each meet exactly two edges in $P$. A path $P$ is said to go from the node which is only a rear end in $P$ (the root of $P$) to the node which is only a front end in $P$. For any arborescence $T$, and any node $v$ in $T$ except the root, there is a unique path in $T$ going from the root to $v$. Any path in $T$ going to $v$ and any path in $T$ going from $v$ have only $v$ in common, and their union is a path. And so on.

Section 2

Let $G$ be any graph with a real numerical weight $c_j$ corresponding to each edge $e_jG$. The problem treated here is to find in $G$ a branching $B$ which has maximum total weight, $\Sigma c_j$, summed over $e_jB$. $B$ is called an optimum branching in $G$.

First we show that certain variations of the problem reduce immediately to it.

A spanning subgraph of $G$ is a subgraph which contains all the nodes of $G$. A branching in $G$ is a spanning arborescence of $G$ if and only if the number of its edges is one less than the number of nodes in $G$. No branching in $G$ can have more edges than this.

An optimum branching in $G$ of course contains no edge with negative weight, and indeed may be empty if all $c_j \leq 0$. Even if all $c_j > 0$ and $G$ contains a spanning arborescence, an optimum branching in $G$ need not be an arborescence.

If there is a spanning arborescence $T$ in $G$, then an optimum one, i.e., one which has maximum total weight, $\Sigma c_j$, $e_jT$, can be found as an optimum branching in $G$ where the edges carry new weights.

*Prepared while the author was a visiting professor at the University of Waterloo, Ontario, Canada. Presented under the title Optimum Arborescences at the International Seminar on Graph Theory and Its Applications, Rome, July 1966.
A spanning arborescence in $G$ which is optimum relative to weights $c_j$, $e_j\in G$, is also optimum relative to weights $c_j + k$, $e_j\in G$, for any constant $k$, since every spanning arborescence has the same number of edges. Constant $h$ is larger than the difference in total weights (relative to weights $c_j$, $e_j\in G$) of any two branchings in $G$. It follows that an optimum branching in $G$, relative to weights $c_j = c_j + h$, will be a branching with a maximum number of edges. In particular, it will be a spanning arborescence if and only if $G$ contains a spanning arborescence.

A spanning arborescence $T$ in $G$ which has minimum total weight, $\Sigma c_j, e_j\in T$, is the same as one which has maximum total weight $\Sigma c_j', e_j\in T$, relative to weights $c_j' = -c_j$.

It will be evident that the efficiency of the method for treating optimum branchings is not seriously effected by a large change $h$ (say of the form $10^n$) in all the weights. In fact the method is easily modified to treat optimum spanning arborescences directly.

If there is a spanning arborescence in $G$ which is rooted at a prescribed node, say $r$, then an optimum one can be found by finding an optimum spanning arborescence in the graph $G'$ obtained from $G$ by adjoining a new edge $e_0$ (carrying arbitrary weight $c_0$) which is directed toward $r$ and directed from a new node having no other incident edges. Clearly, $T$ is a spanning arborescence in $G$ which is rooted at $r$ if and only if $T$ together with $e_0$ is a spanning arborescence of $G'$.

If the edges in graph $G$ represent the links for possible direct communication from one node to another, if each $c_j$ is the cost of direct communication from the rear end of $e_j$ to the front end of $e_j$, and if cost is additive, then a minimum-total-weight spanning arborescence rooted at prescribed node $r$ represents the least costly way to have a message communicated from $r$ to all other nodes of $G$.

Another application is where it is desired to arrange an institution into an optimum hierarchy (branchocracy).

**Section 3**

Our main result is

**Theorem 1.** There exists a good algorithm for finding, in any graph $G$ with a numerical weight corresponding to each edge, an optimum branching.

We say an algorithm is good if there is a polynomial function $f(n)$ which, for every positive-integer valued $n$, is an upper bound on the "work" the algorithm does for any input of "size" $n$. The concept is easy to formalize—relative, say, to a Turing machine, or relative to any typical digital computer with an unlimited supply of tape.

For optimum branching, the largest number of significant digits in an edge weight, as well as the number of edges of $G$, must be figured somehow into the measure $n$ of input "size." One might for example take $n$ to be the maximum of these two numbers or to be the vector consisting of both numbers.

The proof of Theorem 1 is constructive. The theorem is proved by displaying one particular algorithm for optimum branching which is obviously good.

If we remove from the optimum-spanning-arborescence problem the condition that each member of the set $T$ of edges being optimized must have a different front end, then we get the optimum-spanning-tree problem. That is to find, if there is one, in any graph $G$ with a numerical weight on each edge, a spanning tree which has maximum (or minimum) total weight.

**Section 4**

Here is the algorithm for finding a maximum-total weight branching in any (directed) graph $G$ with a numerical weight $c_j$ on each edge $e_j\in G$. Recall that a branching is a forest such that each edge is directed toward a different node.

Begin the algorithm by applying instruction (I1) where $G'$ is $G''=G$ and where $D^i$ and $E^i$ are empty buckets, $D^0$ and $E^0$.

(I1) Choose a node $v$ in $G'$ and not in $D^i$. Put $v$ into bucket $D^i$. If there is in $G'$ a positively weighted edge

---

1 Figures in brackets indicate the literature references at the end of this paper.
directed toward \( v \), put one of them having maximum weight into bucket \( E' \).

Repeat (1) until (a) \( E' \) no longer comprises the edges of a branching in \( G' \), or until (b) every node of \( G' \) is in \( D' \), and \( E' \) does comprise the edges of a branching. When case (a) occurs, apply (I 2).

For convenience assume that every branching which we consider in graph \( G' \) contains all the nodes of \( G' \). We say that a set of edges in \( G' \) forms the unique subgraph of \( G' \) consisting of those edges and all nodes in \( G' \).

Each edge \( e \) put into \( E' \) according to (I 1) is directed toward a node \( v \) which is the root of a connected component of the branching, say \( B \), formed by the edges in \( E' \) before \( e \) is put into \( E' \). If the rear end \( v_6 \) of \( e \) is in a different component of \( B \) than \( v \), then \( B \cup e \) is a branching, and so when \( e \) is put into \( E' \), (a) does not hold.

If \( v_6 \) is in the same component of \( B \) as \( v \), then \( B \) contains a unique path \( P \) going from \( v \) to \( v_6 \). In this case, \( Q' = P \cup e \) is a circuit contained in \( B \cup e \), so as soon as \( e \) is put into \( E' \), (a) does hold.

(12) Store \( Q' \) and a specification of one of the edges, say \( e'_0 \), of \( Q' \) which has minimum weight in \( Q' \) relative to the edge-weights for \( G' \). Obtain a new graph \( G'_{i+1} \) from \( G' \) by "shrinking" to a single new node, \( v_{i+1} \), the circuit \( Q' \) and every edge of \( G' \) which has both ends in \( Q' \). The edges (denoted as \( e'_{j+1} \)) of \( G'_{i+1} \) are those edges (denoted as \( e'_j \)) of \( G' \) which have at most one end in \( Q' \). Every edge of \( G' \) which has one end in \( Q' \) will in \( G'_{i+1} \) have \( v_{i+1} \) at that end. All other edge-ends are in \( G'_{i+1} \) as in \( G' \). The nodes of \( Q' \) are not in \( G'_{i+1} \).

Every edge, say \( e'_{j+1} \), which as \( e'_3 \) in \( G' \) is directed toward a node, say \( v'_3 \), in \( Q' \) and directed away from a node not in \( Q' \), gets a possibly different weight for \( G'_{i+1} \):

\[
c'_{j+1} = c'_j + c'_0 - c'_4
\]

where \( c'_j \) is the weight of \( e'_j \) for \( G' \); where \( c'_0 \) is the minimum weight for \( e'_j \) of an edge, say \( e'_j \), in \( Q' \); and where \( c'_4 \) is the weight for \( G' \) of the unique edge, say \( e'_4 \), which is in \( Q' \) and directed toward \( v'_3 \). All other edges in \( G'_{i+1} \) keep the same weight as for \( G' \).

In justifying the algorithm we shall make use of the following relations:

\[
\begin{align*}
(2) & \quad c'_0 \geq 0, & \quad (3) & \quad c'_4 \geq c'_0, & \quad (4) & \quad c'_4 \geq c'_3.
\end{align*}
\]

Put into bucket \( D'_{i+1} \) the nodes which are in both \( G'_{i+1} \) and bucket \( D' \). (Do not at this point put \( v_{i+1} \) into \( D'_{i+1} \).) Put into bucket \( E'_{i+1} \) the edges which are in both \( G'_{i+1} \) and bucket \( E' \), i.e., put into bucket \( E'_{i+1} \) the final contents of bucket \( E' \) minus the edges of circuit \( Q' \). It is easy to see that the edges in bucket \( E'_{i+1} \) form a branching in \( G'_{i+1} \). Continue the algorithm by applying (I 1) where \( i \) is one greater.

Eventually, after a small number of applications of (I 1) and (I 2), case (b) must occur.

As soon as (b) occurs, for say \( i = k \), (I 1) and (I 2) are never applied again. Instead, (I 3) is applied successively for \( i + 1 = k, k - 1, \ldots, 1 \), until the graph \( G' \) obtained is the original \( G \). At that point, the branching \( B' = B'' \) is a maximum-total-weight branching of \( G \).

The final contents of bucket \( E''_k \) form a branching in graph \( G'' \) which we call \( B'' \).

(13) It is not difficult to see that since \( B'_{i+1} \) is a forest in \( G''_{i+1} \) and since \( G''_{i+1} \) is obtained from \( G' \) by shrinking the circuit \( Q' \) in \( G' \) (and all edges of \( G' \) with both ends in \( Q' \)) to the node \( v_{i+1} \) of \( G''_{i+1} \), the subgraph \( H''_i \) of \( G'' \), formed by the edges in \( B''_{i+1} \) and the edges in \( Q' \) contains only one polygon, namely \( Q' \).

In the case where \( v_{i+1} \) is not a root of (a connected component of) branching \( B''_{i+1} \) in \( G''_{i+1} \), there is a unique edge, say \( e_{i+1} \), of \( B''_{i+1} \) is directed toward \( v_{i+1} \). In \( G'' \), \( e_{i+1} \) is directed toward a node, say \( v'_2 \), of \( Q' \). Since \( Q' \) is a circuit, there is a unique edge, say \( e'_2 \) of \( Q' \) which is directed toward \( v'_2 \). Clearly, \( e'_1 \) and \( e'_2 \) are the only two edges of \( H''_i \) which are directed toward the same node. Thus, since \( Q''_i \) is in the only polygon of \( H''_i \), deleting \( e'_2 \) from \( H''_i \) yields a branching in \( G'' \), which is called \( B''_i \).

In the case where \( v_{i+1} \) is a root of branching \( B''_{i+1} \) in \( G''_{i+1} \), i.e., where no edge of \( B''_{i+1} \) is directed toward \( v_{i+1} \), no two edges of \( H''_i \) are directed toward the same node. Therefore, deleting any edge of \( Q''_i \) from \( H''_i \) yields a branching in \( G'' \). To obtain the branching \( B''_i \) in \( G'' \), delete from \( H''_i \) one of the edges \( e'_0 \) of \( Q''_i \) which has minimum weight \( c'_0 \).

That completes the description of the algorithm. Evidently it is a good algorithm. Evidently its output is a branching \( B'' \) in graph \( G \). In order to prove Theorem 1, what remains to be done is prove that \( B'' \) has maximum total weight.

Section 5

Theorem 1 and the following geometric theorem are proven together.

Let \( G \) be any graph. (No edge-weights are specified.) Let there be a real variable \( x_j \) for each edge \( e_j \) of \( G \). Let \( P_G \) be the polyhedron of vectors \( x = [x_j] \) which satisfy the system \( L_G \), consisting of inequalities \( L_1, L_2, \) and \( L_3 \).

\( (L_1) \) For every edge \( e_j \) of \( G \), \( x_j \geq 0 \).

\( (L_2) \) For every node \( v \) of \( G \), \( \Sigma x_j = 1 \), summed over all \( j \)'s such that \( e_j \) is directed toward \( v \).

\( (L_3) \) For every set \( S \) of two or more nodes in \( G \),

\[
\Sigma x_j \leq |S| - 1,
\]

summed over all \( j \)'s such that \( e_j \) has both ends in \( S \). \(|S| \) denotes the cardinality of \( S \).)

Any vector \( x = [x_j] \) of zeroes and ones is called the (incidence) vector of the subset of \( e_j \)'s such that \( x_j = 1 \).

**Theorem 2.** The vertices of polyhedron \( P_G \) are precisely the vectors of the subsets of edges in \( G \) which comprise branchings.
A polyhedron (convex polyhedron) $P$ is the set of all the vectors, i.e., points, which satisfy some finite system $L$ of linear inequalities. A vertex (extreme point) of $P$ is a point which, for some linear function, is the unique point in $P$ which maximizes that function.

A basic point $x=x^0$ of a finite system $L$ of linear inequalities is the unique solution of a system, $\Sigma a_{ij}x_j = b_i, \ i \in I$, such that $\Sigma a_{ij}x_j \leq b_i, \ i \in I$, is a subsystem of $L$.

If basic point $x^0$ of $L$ is in the polyhedron $P$ of $L$, then it is a vertex of $P$, because clearly $x^0$ is then the unique point in $P$ which maximizes $\Sigma (i \Sigma a_{ij}x_j)$, $i \in I$.

We shall see without difficulty that any point $x^0$, which is the vector of a branching say $B^0$ in $G$, is a vertex of $P_G$. Vector $x^0$ satisfies $L_1$ since it is all zeroes and ones. Vector $x^0$ satisfies $L_2$ for any node $v \in G$, since, by the definition of branching, at most one of the $x_j$'s in this inequality have value 1 for $x^0$.

The branching $B^0$ is a forest, so any set $S$ of nodes, together with the subset $E^0_S$ of the edges in $B^0$ which have both ends in $S$ forms a forest. The number of edges in a forest is at most the number of nodes in the forest minus 1; in particular, $|E^0_S| \leq |S| - 1$. Therefore, vector $x^0$ satisfies $L_3$ for any subset $S$ of (two or more) nodes in $G$, since $|E^0_S|$ of the $x_j$'s in this inequality have value 1 for $x^0$. Summarizing the conclusion so far, $x^0$ is a point in $P_G$.

Vector $x^0$ is the unique solution of the linear system: $x_j=0$ for every edge $e_j$ not in $B^0$, and $\Sigma x_j=1$ (summed over $e_j$'s directed toward $\nu$) for every node $\nu$ which has some edge of $B^0$ directed toward it. This system can be obtained from certain of the relations of $L_1$ and $L_2$ by replacing their inequality signs. Therefore $x^0$ is a basic point of $L_G$, and hence a vertex of $P_G$.

Most of this paper is directed toward proving:

**Lemma 1:** Every linear function, $\Sigma c_jx_j$ (summed over all edges $e_j \in G$), is maximized in $P_G$ by the vector of some branching in $G$.

From Lemma 1 and from the definition of vertex, it follows immediately that every vertex of $P_G$ is the vector of a branching in $G$. This will conclude the proof of Theorem 2.

A branching $B^0$ in graph $G$ has maximum total weight relative to the vector $c=[c_j]$ of edge-weights if and only if the vector $x^0=[x_j]$ of $B^0$ maximizes $(c, x) = \Sigma c_jx_j$ over all vectors of branchings in $G$. If $x^0$ maximizes $(c, x)$ over $P_G$, then it maximizes $(c, x)$ over the vectors of branchings in $G$, since the latter are in $P_G$.

Our task, therefore, is to show that the vector of the branching $B^0$, produced by the algorithm, maximizes $(c, x)$ over $P_G$. This will prove that the algorithm is valid and will prove Lemma 1.

**Section 6**

The following computations are well-known in linear programming. Suppose that $x=[x_0]$ is any vector which satisfies

$$x_0 \geq 0 \text{ for every } \eta, \text{ and}$$

and that $y=[y_\xi]$ is any vector which satisfies

$$y_\xi \geq 0 \text{ for every } \eta, \text{ and}$$

$$\Sigma a_{\xi\eta}x_\eta \leq b_\xi \text{ for every } \xi.$$
For every $v_h, b_h = 1$. For every $S$, $b_s = |S| - 1$.
Therefore, (8) becomes

for every edge $e_j \in G$,
$y_h + w_j \not\in c_j$, where $v_h$ is the front end of $e_j$, and where $w_j = \Sigma y_s$, summed over all sets $S$ which contain both ends of $e_j$. (17)

Function $(b, y)$ becomes

$$(b, y) = h \Sigma y_h + \Sigma (|S| - 1)y_s,$$

summed over all $v_h$ and over all $S$.

Recall that our task is to show that the vector $x^0$ of the branching $B^0$, produced by the algorithm, maximizes $(c, x)$ over $P_G$.

In view of (14), we do so by constructing a vector $y = [y_h, y_s]$ which satisfies (15), (16), (17), and which satisfies (12) and (13). For the present system, (12) is

for every node $v_h$ such that $y_h \neq 0$,
$\Sigma x^0 = 1$, summed over $j$'s such that $e_j$

is directed toward $v_h$;

and

for every set $S$ such that $y_s \neq 0$,
$\Sigma x^0 = |S| - 1$, summed over $j$'s

such that $e_j$ has both ends in $S$. (18)

In other words (18) says that if $y_h \neq 0$ then an edge of the branching $B^0$ is directed toward $v_h$, and (19) says that if $y_s \neq 0$ then exactly $|S| - 1$ edges of $B^0$ have both ends in $S$.

For the present system (13) is

for every edge $e_j$ in the branching $B^0$,
$y_h + w_j = c_j$, where $v_h$ and $w_j$ are as in (17). (20)

Section 7

For each graph $G^i (i = k, k - 1, \ldots, 0)$ with weight $c^i_j$ on each edge $e_j \in G^i$, and for the branching $B^i$ in $G^i$, we will describe a vector $y^i$ which satisfies (15)–(20), where $G$ and $B^0$ are replaced by $G^i$ and $B^i$ and where vector $y$ is $y^i$.

First we describe a $y^k$, and then, assuming a

$y^{i+1}(i = k - 1, \ldots, 0)$,

we describe a $y^i$. Thus by induction we obtain a $y = y^0$

and the proof of Theorems 1 and 2.

The vector $y^k = [y^k_h, y^k_s]$ is $y^k = 0$ for every set $S$ of two or more nodes in $G^k$, $y^k_s = 0$ for every node $v^k_h$ in $G^k$ which has no edge of $B^k$ directed toward it, and, for every other node $v^k_h$ in $G^k$, $y^k_h = c^k_j$ where edge $e^k_j$ of $B^k$ is directed toward $v^k_h$. Conditions (15)–(20) for $y^k$ can be immediately verified from the fact that for every node $v^k_h \in G^k$ either there is no edge of $B^k$ directed toward $v^k_h$ and there is no positively weighted edge directed toward $v^k_h$, or else, among all the positively weighted edges directed toward $v^k_h$, the one in $B^k$ has maximum weight.

Now, suppose that we have a $y^{i+1}_h$ for each node $v^{i+1}_h$ and a $y^{i+1}_s$ for each set $S$ of two or more nodes in $G^{i+1}$, such that (15)–(20) are satisfied (where $B^0$ is replaced by $B^{i+1}$, etc.).

Let $t^{i+1}_h = \Sigma y^{i+1}_s$, summed over the sets $S$ which contain node $v^{i+1}_h$.

To make the induction go through we assume further that in $G^{i+1}$

for every node $v_h$, such that $t_h + y_h > 0$, there exists at least one edge $e_j$ directed toward $v_h$ such that $c_j = t_h + y_h$. (21)

This clearly holds for $G^k$, and we will prove from (15)–(21) for $G^{i+1}$ that (15)–(21) holds for $G^i$.

Obtain the vector $y^i$ as follows:

Where $A$ is the set of nodes in circuit $Q^i$ of $G^i$, where $e^i_2$ is the edge of $Q^i$ not in $B^i$, where $v^i_2$ is the front end of $e^i_2$, where $c^i_0$ is the minimum weight in $Q^i$, and where $v^{i+1}$ is the node in $G^{i+1}$ to which $Q^i$ was shrunk, let

$y^i_2 = y^{i+1}_2 + c^i_2 - c^i_0$. (22)

and

$y^i_1 = c^i_1 - y^i_2 - t^{i+1}_1$. (23)

Where $v^i_1$ is any node in $A$ other than $v^i_2$, and where $e^i_1$ is the edge in $Q^i$ which is directed toward $v^i_2$, let

$y^i_1 = c^i_1 - y^i_2 - t^{i+1}_1$. (24)

Observe that (24) holds also for $v^i_1 = v^i_2$.

Where $v^i_3$ is any node of $G^i$ which is not in $Q^i$, let

$y^i_3 = y^{i+1}_3$. (25)

Where $R$ is a nonempty subset of nodes in $G^{i+1}$ which does not contain $v^{i+1}_1$, where $J = R \cup v^{i+1}_1$, where $K = R \cup A$, and where $L$ is any set of two or more nodes in $G^i$ such that $L \cap A$ is a proper subset of $A$, let

$y^i_K = y^{i+1}_K$, (26)

$y^i_J = y^{i+1}_J$, (27)

and

$y^i_L = 0$. (28)

That completes the description of vector $y^i$. Now we must verify (15)–(21) for it.

For every edge of $G^i$ which is directed toward a node not in $A$, for every node not in $A$, and for every set $S$, except $A$, in $G^i$, conditions (15)–(18), (20), and (21) follow immediately from those same conditions for $y^{i+1}$, (25)–(28), and the local nature of the change from $G^{i+1}$, $B^{i+1}$, and $c^{i+1}$ to $G^i$, $B^i$, and $c^i$. 

237
For every subset of nodes in $G$, which does not contain all of $A$, condition (19) follows immediately as above. For set $A$ and for every set $K$ as in (27), condition (19) follows from (27), condition (19) for set $J$ in $G'$, and the fact that there are exactly

$$|K| - |J| = |A| - 1$$

more edges of $B'$ with both ends in $K$ than there are edges of $B'^+1$ with both ends in $J$, namely the edges of $B' \cap Q'$.

It follows from (24), (27), and (28), that (21) holds for every node $v_i$ in $A$ (in particular where $e_i$ is the $e_i$ of (24)), and that (20) holds for every edge of $B' \cap Q'$, and that (17) holds for $e_i$.

Condition (18) follows immediately for each node of $A$ except $v_i$ since there is an edge of $B' \cap Q'$ directed toward it. If there is an edge $e_i^+$ in $B'^+1$ which is directed toward $v_i^+$, then $e_i$ is an edge of $B'$ which is directed toward $v_i^-$, and so in this case (18) follows for $v_i^-$.

Therefore, if there is no edge of $B'^+1$ directed toward $v_i^+$, then (22) is $y_i^+ = 0$, and so (18) follows for $v_i^-$.

For $e_i^+$, the only edge, if any, which is in $B' - Q'$ and directed toward a node in $A$, we have

$$c_i^+ = c_i + c_0 - c_i$$

which is (20) for $e_i^+$, and $w_i = w_i^+ = c_i$ from (27) and (28).

Combining these we get $y_i^+ + w_i^+ = c_i^+$, which is (20) for $e_i^+$.

Thus conditions (18), (19), (20), and (21) are now completely accounted for. Condition (17) for edges not in $Q'$ but directed toward nodes in $A$, condition (16) for $y_i^+$, and condition (15) for nodes in $A$, remain to be verified.

Let $e_i^-$ be any edge of $G$ which has both ends in $A$, and let $v_i^-$ be its front end. To prove (17) for $e_i^-$, which is

$$y_i^+ + w_i^+ = c_i$$

where $w_i^+ = y_i^+ + t_i^+$, combine (24) and (21) for $e_i^-$.

Let $e_i^+$ be any edge of $G$ which has its front end $v_i^+$ in $A$ and its rear end not in $A$. To prove condition (17) for $e_i^+$, which is

$$y_i^+ + w_i^+ = c_i^+$$

where $w_i^+ = w_i^++$, combine (24), (23), (22), (1), and (17) for $e_i^+$.

To prove (16) for $A$, that is $y_i^+ \geq 0$, we use (21) for $v_i^+$. Assuming $t_i^+ + y_i^+ > 0$, let $e_i^-$ be the $e_i$ of that relation, let $v_i^-$ be the front end of $e_i^-$ in $A$, and let $e_i^+$ be the edge of $Q'$ which is directed toward $v_i^+$. Here (21) is $c_i^+ = t_i^+ + y_i^+$. In this case, obtain $y_i^+ \geq 0$ by combining (23), (22), (21) for $v_i^+$, (1), and (4).

If there is no $e_i^+$ directed toward $v_i^+$ such that $c_i^+ = t_i^+ + y_i^+$, then $t_i^+ + y_i^+ = 0$, and all edges directed toward $v_i^+$ have negative weight in $G^+$, so none of them are in $B'^+1$. Therefore since in this case the $c_i^+$ of (22) was chosen to be $c_i^+$, (22) becomes $y_i^+ = 0$, and (23) becomes $y_i^+ = c_i^+$. By (2), we have $y_i^+ \geq 0$.

Prove (15) for any node $v_i^-$ in $A$ by combining (24), (23), (22), (3), and $y_i^+ \geq 0$.

That completes the proof of Theorems 1 and 2.

Section 8

Notice from the proof that if every weight $c_j$, $e_j \in G$, is an integer, then the vector $y^0$, as well as vector $x^0$, is integer-valued. In particular, where every $c_j = 1$, vector $y^0$ is 0,1-valued and $\max(c, x) = \min(b, y)$ is a simple “König-type” theorem, analogous to the maximum-cardinality-matching duality theorem in [1].

The following two theorems can be proved by the methods used here.

**Theorem 3.** Where $(L_4)$ is $\Sigma x_j = n$, summed over all edges $e_j \in G$, the vertices of the polyhedron given by $(L_1)$, $(L_2)$, $(L_3)$, and $(L_4)$ are precisely the vectors of the n-cardinality subsets of edges in $G$ which comprise branchings. (In particular, where $n$ is one less than the number of nodes in $G$, these branchings are the spanning arborescences of $G$).

The present research began when A. J. Goldman asked for a description of “the convex hull of the spanning trees of a graph.” Theorem 4 is proved in [3].

**Theorem 4.** The vertices of the polyhedron $F_0$ given by $(L_1)$ and $(L_3)$ are precisely the vectors of the subsets of edges in $G$ which comprise forests. The vertices of the intersection of $F_0$ and $(L_4)$ are a subset of the vertices of $F_0$.

Section 9

Figures (A) through (E) illustrate the algorithm for finding an optimum (i.e., maximum total weight) branching in graph (A). Each dashed edge is dashed because it has maximum positive weight among those edges directed toward its front end. As soon as a dashed circuit arises it is shrunk and certain new edge weights are computed, thereby producing the edge-weighted graph of the next figure. The final graph is not drawn since it is simply a node. After the sequence of figures is completed, except for the boldness of the bold edges, then working backwards through the sequence, appropriate dashed edges are made bold. The answer is the branching in (A) formed by the bold edges. It is the only correct answer.

Figure (F) illustrates a “dual answer,” $y$, for the same problem. The numbers in the squares on the nodes are the values of the nodal $y$-variables. The numbers in the squares on the closed curves are the nonzero values of $y$-variables corresponding to subsets of nodes. Each closed curve encloses the subset of nodes to which its number corresponds. Observe that the vector $x$, thus represented, satisfies relations (15)–(20), and thus guarantees that the branching is optimum. This $y$ is not the one described in section 8.

The example (A)–(E) was actually obtained by constructing it all except the edge-weights first. Then all the numbers in (F) were chosen so as to yield the structure (A)–(E).

Figure (G), by coincidence, nicely represents three different things.

It illustrates a nonoptimum branching obtained by applying the greedy algorithm. This branching has total weight 128, whereas the branching shown in (A)
has total weight 131. One other branching that might be obtained by applying the greedy algorithm has total weight 127.

Since the optimum branching shown in (A) is a spanning arborescence, it is an optimum spanning arborescence, and thus it is still an optimum spanning arborescence when all of the edge-weights are changed by adding any constant. However, when all the edge-weights are changed by adding the constant, −10, the branching shown in (A) is no longer an optimum branching, even though all the weights in this branching are still positive, and even though this branching is still an optimum spanning arborescence. For these new edge-weights there are two optimum branchings, quite different from each other, and both quite different from the branching shown in (A). Neither one of them is a spanning arborescence. One of them consists of all the bold edges in (G) except for the edge which is weighted −5 (relative to the new weights). This branching has total weight, 53, whereas the branching shown in (A) has total weight, 51.

A direct algorithm for finding, if there is one, an optimum spanning arborescence is obtained from the algorithm for finding an optimum branching simply by deleting the words “positively weighted” from (II). This follows from the fact that the only effect on the resulting algorithm of adding a constant to each number of an input is to add the same constant to every number that arises in the algorithm.

Because of the words “positively weighted” in the optimum branching algorithm, the effect of adding −10 to each edge-weight in (A) is that the resulting application of the optimum branching algorithm does not dash the edge that is weighted −5, and does not dash in (D) the edge that is weighted −1. Thus there is no shrinking in (D); the sequence of graphs stops at (D). In (D), only the edge that is weighted 7 is made bold since it is the only one that gets dashed. (The present computation is not explicitly illustrated.) Unlike in the spanning arborescence problem, we have in (C) no bold edge directed toward a node in the dashed circuit. Therefore, the edges of the dashed circuit in (C), except for one or the other of its minimum-weight edges, are made bold. The choice here is what gives rise to the two correct answers. It is interesting to note that the two edges which are tied in this step of the computation do not have the same edge-weights in (A), and that the two optimum branchings, arising from the two choices, are globally quite different. I recommend carrying through the completion of each.

Figure (G) also illustrates an optimum spanning arborescence having the lower left node prescribed as root. The first phase of the algorithm for obtaining it is the dashing and shrinking and computing of new edge weights just as in (A)–(E). The only difference from (A)–(E) is the way the edges are chosen from among the dashed ones to be made bold. An optimum spanning arborescence rooted at any other prescribed node is obtained from this same first phase of (A)–(E) by appropriately choosing edges from among the dashed ones. The subgraph of (A), formed by the image of (A) of all edges dashed somewhere in (A)–(E), in general contains nonoptimum spanning arborescences with prescribed root as well as optimum ones. Therefore, the choosing does depend on the structure of the sequence (A)–(E). In general, it is a nice feature of the computations for finding in the same edge-weighted graph, when they exist, optimum spanning arborescences with various prescribed roots, that these computations are identical except for the (I3) part.

This paper was to have appeared in the published proceedings of the International Seminar on Graph Theory and Its Applications, Rome, July 1966, sponsored by the International Computation Center. Various international failures of communication during the editorial process precluded it. I am sorry to have lost that opportunity to record my contribution to an outstanding symposium. I wish to acknowledge here my appreciation to the organizers of the symposium for the excellent job they did and for their kindness to me.

References


(Paper 71B4–249)