Zeitschrift für Operations Research, Band 18, 1974, Seite 59–67. Physica-Verlag, Würzburg.

Cost-Minimal Trees in Directed Acyclic Graphs

By *L. Nastansky*, Saarbrücken [1]), *S. M. Selkow*, Montréal [2]), and *N. F. Stewart*, Montréal [2])

Eingegangen am 3. November 1972

Summary: A number of examples are given where one seeks a minimal-cost tree to span a given subset of the nodes of a connected, directed, acyclic graph (we call such a graph monotonic). Some of these examples require an algorithm to transform the problem into the form of a minimization problem in a monotonic graph; this algorithm is also given. Finally, an implicit enumeration algorithm is presented for finding the cost-minimal tree of the graph, which spans the designated subset of the nodes, and some computational results are given.

Zusammenfassung: Es werden Beispiele aufgezeigt, bei denen ein kostenminimaler Baum gesucht wird, der eine gegebene Untermenge von Knoten in einem verbundenen, gerichteten und azyklischen Graphen aufspannt. Ein solcher Graph wird hier als monotoner Graph bezeichnet. Einige dieser Beispiele erfordern einen Algorithmus, der das gegebene Problem in eine Minimierungsaufgabe in einem monotonen Graphen überträgt. Dieser Algorithmus zur Konstruktion des Graphen wird formuliert. Schließlich wird ein spezialisiertes implizites Enumerationsverfahren vorgestellt, das den kostenminimalen Baum zu der gegebenen Untermenge von Knoten in dem vorliegenden monotonen Graphen konstruiert. Rechenerfahrungen bilden den Abschluß.

1. Introduction

The general Steiner problem in graphs consists of finding a minimal subtree to span a given subset of the nodes. We consider the Steiner problem for directed graphs in which directed cycles may not occur (no directed chain leads from any node to itself) and such that there exists a node in the graph from which a directed chain emanates to every other node. We call this type of graph a monotonic graph.

The most common form of Steiner's problem involves the construction of a minimal length tree to a set of points in Euclidean space where the 2-norm (Euclidean distance) is used to measure the cost of an arc [*Courant* and *Robbins*, 1941; *Gilbert* and *Pollak*, 1967; *Melzak*, 1961; *Chang*, 1972]. The implications of a solution of this general problem are given by *Hakimi* [1971]. For the problem for which distance is measured using the 1-norm (as in the principal case of interest for this article), without the monotonicity requirement, *Hanan* [1966] outlines

[1]) Ass. Prof. Dr. *Ludwig Nastansky*, Fachbereich Wirtschaftswissenschaften, Universität des Saarlandes, 66 Saarbrücken, W-Germany.

[2]) Prof. *Stanley M. Selkow*, Ph. D. and Prof. *Neil F. Stewart*, Ph. D., Département d'informatique, Université de Montréal, C. P. 6128, Montréal 101, P. Q., Canada.

an algorithm to construct a Steiner tree to four given nodes which lie in the plane. For the Steiner problem in graphs, again without the monotonicity requirement, an algorithm has been given by *Dreyfus* and *Wagner* [1970].

2. Applications

Motivation for the problem considered in this article is provided by the search for a minimal tree in d-dimensional space, from a designated node to a given set of goal nodes, such that each branch of the tree is monotonic and is composed of a number of arcs which change in only one dimension. That is, we seek an optimal tree which consists of a number of directed chains (hereafter called paths) in a d-dimensional city connecting the designated node to the set of goal nodes such that the total length of the streets used as paths is minimized, subject to the constraint that each path must be minimal in length.

An example of this type of problem is *snow removal* in a two-dimensional city where paths must be cleared from a number of seekers of services to a center of supply. If the operating costs incurred in traversing the paths many times are higher than the costs of snow removal, then the constraint that the paths be monotonic is appropriate.

An example in three dimensions is the design of an optimal *drainage system* for a large building. Here, the origin is defined by the location of the sewer which is to collect the waste from the building's drainage system, and the monotonicity constraint is imposed by the law of gravity.

A third application, in which the dimension d typically will be greater than three, is the solution of the basic problem in that biological field known as *cladistics* [*Camin* and *Sokal*, 1965; *Hendrickson*, 1968]. In this context, one seeks an evolutionary tree from an ancestor to a set of operational taxonomic units (goal nodes), each node being defined by measurements on d characters of an operational taxonomic unit. The biological assumptions underlying the problem are that characters being measured can be ordered according to evolutionary trends (a character state is "more advanced" than those which are closer to the origin but of the same sign), evolution is irreversible with respect to the characters being measured (monotonicity), and only one character may evolve at a time. A cladogram is an evolutionary tree which satisfies these requirements and is of minimal length, thus requiring the minimal number of evolutionary steps to explain the presence of the given operational taxonomic units.

Arguing that (phenetic) similarity between the operational taxonomic units implies a common evolutionary history, a number of researchers [*Camin* and *Sokal*, 1965; *Hendrickson*, 1967] have proposed heuristics for finding parsimonious evolutionary trees. There is no guarantee that these heuristics, based upon finding clusters of operational taxonomic units, will ever produce a minimal tree. *Hendrickson* [1968] has listed the conditions defining the cladogram to

three given nodes and has indicated that the enumeration of all possible trees is infeasible for more than a few given nodes.

When the original problem is posed as a search for a minimal tree in d-dimensional space, as is the case for the examples we have just given, we must first transform the problem to the Steiner problem for monotonic graphs. The algorithm required to effect this transformation is described in Section 3, and in the Appendix. An algorithm for the solution of the Steiner problem in a monotonic graph is given in Section 4.

It will sometimes be the case, however, that the original problem is posed directly as a Steiner problem in a directed graph. One such example is the search for the cost-minimal *information flow in a firm.* Here the different organizational units within the hierarchial structure of the firm are represented as the nodes of the graph. The connecting arcs show the various possibilities of providing the necessary information between the different organizational units. The values assigned to the arcs indicate the estimated costs of the specific underlying information connection between two organizational units. In such a case, the algorithm of Section 4 can be applied directly, without the use of the algorithm of Section 3.

3. Construction of a Monotonic Graph

A *directed graph* G is a pair (N, A), where N is a set of nodes and A is a set of directed arcs connecting members of N. A *path* from $X \in N$ to $Y \in N$ is a nonempty sequence $(Z^{(0)}, Z^{(1)}), (Z^{(1)}, Z^{(2)}), \ldots, (Z^{(p-1)}, Z^{(p)})$ of arcs, where $Z^{(0)} = X$, $Z^{(p)} = Y$. If there is a path from X to Y, we say that X is a *predecessor* of Y and Y is a *successor* of X. A *monotonic graph* is a graph for which no node is a successor of itself and there exists an origin 0 such that every node in $N - \{0\}$ is a successor of 0. Clearly, 0 must be unique.

With each arc $(X, Y) \in A$ we will associate a cost $C(X, Y) > 0$. Suppose a subset of the nodes, $\Omega \subseteq N$, is given. A *Steiner graph to Ω in G* is that monotonic graph (N', A') such that $\Omega \subseteq N' \subseteq N$, $0 \in N'$, and $A' \subseteq A$, which minimizes

$$\sum_{(X,Y) \in A'} C(X, Y).$$

Clearly, a Steiner graph will be a *Steiner tree.*

In Section 4 we will describe an algorithm which, when presented with a monotonic graph, will construct a Steiner tree. In the remainder of this section, we will describe a procedure for the construction of a reduced monotonic graph which contains a Steiner tree to a set of points in Euclidean d-space when the distance between two points is defined by the 1-norm [3]).

[3]) The 1-norm, which we denote by $||\cdot||$, denotes the sum of the absolute values of the components. Superscripts distinguish between different vectors. Subscripts denote components of a vector.

We use the symbol $\ll$ to signify the partial order on the points of R^d, where $X \ll Y$ if, for $i = 1,\ldots,d$, $X_i \cdot Y_i \geq 0$ and $|X_i| \leq |Y_i|$. The cost of arc (X,Y) is defined by $C(X,Y) = \|X - Y\|$. The graph $G^* = (N^*, A^*)$, where $N^* = R^d$ and $A^* = \{(X,Y) \mid X, Y \in N^* \text{ and } X \ll Y\}$, is a monotonic graph. For a finite set $\Omega \subseteq N^*$, we seek a Steiner tree to Ω. As a first step, we reduce the size of G^*, while assuring that the new monotonic graph still contains a Steiner tree to Ω in G^*.

In the new graph, $G = (N,A)$, clearly N must contain $\{0\}$ and Ω. In fact, N will be obtained by noting that a furcation in the Steiner tree in R^d will occur at a point only if that point is the greatest lower bound (with respect to the relation $\ll$) of some subset of Ω. A proof of this appears in the Appendix. We may thus restrict our attention to the set

$$N = \{0\} \cup \{X \mid X \text{ is the } glb \text{ of } S \subseteq \Omega\},$$

which may be enumerated by examining only those subsets of Ω which have cardinality not greater than d.

In defining A, we note that an arc (X,Y) will never be in a Steiner tree if there is a $Z \in \Omega$ such that $Z \ll Y$ and $C(Z,Y) < C(X,Y)$. Thus, in defining the arcs of A which are incomers to any $Y \in N$, we need only consider the predecessors of Y in R^d which are no further than the nearest predecessor of Y in Ω. For each $Y \in N$, this distance is defined by

$$\delta(Y) = \min_{\substack{X \in \Omega \\ X \ll Y}} \{C(X,Y)\}.$$

We use $\Pi(Y)$ to denote those members of N which are predecessors of Y and no further than $\delta(Y)$. For all $X \in \Pi(Y)$, the arc (X,Y) will be in A if X has no successor in $\Pi(Y)$. A proof that G contains a Steiner tree in G^* is shown in the Appendix.

We have implemented a further reduction of G, which we now describe without proof. Application of this reduction is worthwhile when the components of elements of Ω are restricted to a small set (say, the integers less than 100 in modulus), and d is small.

Suppose that $X \in N - \Omega - \{0\}$. For $j = 1,\ldots,d$, let $Y^{(j)}$ be a successor of X in the graph G such that $Y^{(j)} \in \Omega$ and $Y_k^{(j)} = X_k$ for $k \neq j$, if such a successor exists. If such a $Y^{(j)}$ exists for no more than one value of j, then X can be dropped from N. When this procedure has been applied to each node in N, the arcs to be included in A can be recomputed according to the rules given above.

The algorithm which extracts a Steiner tree from G will be discussed fully in the next section. The variables h, i, j and k will be used hereafter to range over the set N. For each $(i,j) \in A$ we introduce a variable x_{ij} and a cost c_{ij}, and we introduce restrictions of the following type. There is no restriction corresponding to the origin. Otherwise, if $i \in \Omega$, then $\sum_{h \in P_i} x_{hi} \geq 1$, where P_i is the set of immediate

predecessors of i. In intuitive terms, we insist that there be a flow of at least one into every member of Ω. If $i \in N - \Omega$, then $\sum_{j \in S_i} x_{ij} - \sum_{h \in P_i} |S_i| x_{hi} \leq 0$, where $S_i(P_i)$ is the set of immediate successors (predecessors) of i, and $|S_i|$ is its cardinality. In intuitive terms, there can not be a flow out of an element of $N - \Omega$ unless there is a flow in. Since the constraint matrix is sparse, we store only the non-zero coefficients.

A table T containing the entries corresponding to the restrictions described above can be created in the following way. We scan inwards in such a way that when we discover a $j \in N$, we are sure that we have already processed all of its successors. If $j \in \Omega$ we find the n arcs ($n \leq d$) in A of the form (i,j), assign each of them a variable number, and store the words defining the restriction in an available part of T. Also, for each (i,j) for which $i \in N - \Omega$, we make an entry in the table which specifies i and the variable number corresponding to (i,j), and a flag to indicate that the entry is only temporary.

In the case when $j \in N - \Omega$ we find the n arcs in A of the form (i,j), assign each of them a variable number, search for the flagged entries in the table representing an arc (j,k), and store the words defining the restriction in an available part of T. The storage used by the flagged entries may now be released.

4. Construction of a Steiner Tree by Implicit Enumeration

We will now describe a linear zero-one programming problem which will accept as input the monotonic graph G. Later, we will show how this program can be solved by a specialized implicit enumeration scheme [*Balas*, 1965; *Geoffrion*, 1967].

Formulation of the Zero-One Program:

A linear zero-one programming problem involves the minimization of a linear boolean objective function subject to a number of linear boolean constraints. Associating a zero-one variable x_{ij} with each arc (i,j), we say that the arc belongs to a tree if $x_{ij} = 1$, and that it does not belong if $x_{ij} = 0$.

The objective function to be minimized, namely, the cost of the tree, is defined as

$$\sum_{(i,j) \in A} c_{ij} x_{ij} \rightarrow \text{Min}\,, \tag{4.1}$$

where c_{ij} is the cost of arc (i,j). The minimization is performed over all subgraphs of G. The set of constraints which will presently be described guarantees that all sets of $x_{ij}[(i,j) \in A]$ defining a tree are feasible solutions, that is, that the resultant tree will indeed be a montonic graph which includes Ω. There is no restriction corresponding to the origin $0 \in N$.

The first type of constraint stipulates that the tree must reach every member of Ω:

$$\sum_{h \in P_i} x_{hi} \geq 1 \quad [i \in \Omega, i \neq 0]\,. \tag{4.2}$$

The inequality constraint could be replaced by the constraint $\sum_{h \in P_i} x_{hi} = 1$, since a graph with two arcs leading to an $i \in \Omega$ could never satisfy the minimality condition.

In order to assure that the solution will consist of one tree, and not a forest of non-connected trees, we introduce the constraints

$$\sum_{h \in P_i} |S_i| x_{hi} - \sum_{j \in S_i} x_{ij} \geq 0 \quad [i \in N - \Omega, i \neq 0] \tag{4.3}$$

where $|S_i|$ denotes the cardinality of S_i. With (4.3) we stipulate that a tree may proceed from a node $i \in N - \Omega$ only if the node is reached by some arc $x_{hi} = 1$. The constant coefficient $|S_i|$ allows all outgoing arcs x_{ij} to be used for constructing subsequent branches if any ingoing arc x_{hi} is used. By minimality, at most one ingoing arc to any node will be one. The existence of a feasible solution may be demonstrated by fixing all x_{ij} $[(i,j) \in A]$ to one.

The Implicit Enumeration Scheme

In our implementation, we solve the linear zero-one program (4.1)–(4.3) with a specialized implicit enumeration scheme which eliminates the enumeration of all solutions which are not trees to Ω. This is due to a *special order* which may be imposed upon the nodes of G. We will only describe those specializations added to the usual implicit enumeration procedure.

Because G is finite and acyclic, N may be ordered such that if i appears before j in the order, then it is not the case that i is a successor of j in G. Because of the natural one-to-one correspondence between nodes and constraints, the same order will be imposed upon the constraints. Also, associating with each node the set of its ingoing arcs, these subsets are ordered to correspond with the order of N. Within each subset of A, the arcs are ordered according to increasing cost.

The algorithm alternates between expansion-phases and backtracking-phases. Due to the order introduced the necessary feasibility tests during the expansion-phase are structured. This is because fixing a variable to one has the following effects on the *feasibility*:

- The associated arc is ingoing to a node $i \in \Omega$ or $i \in N - \Omega$, and from constraints (4.2) and (4.3) respectively, we see that new infeasibility can not be created.
- Either the associated arc is outgoing from a node $i \in \Omega$, in which case no constraint is involved, or the arc is outgoing from a node $j \in N - \Omega$, in which case a one is subtracted from the left-hand side of constraint j in set (4.3). Only in the latter case can infeasibility be created.

The *expansion-phase* consists of scanning the constraints for feasibility in their imposed order $h = 1,2,\ldots,|N|$. Whenever an infeasible constraint k is found, we fix the variable associated with the lexicographically first of all ingoing arcs of the node k to one. Infeasibility is either caused by variables previously fixed to one or because a constraint associated with a node from Ω is encountered.

In fixing the variable in row k to one we eliminate the infeasibility of constraint k. On the other hand, the only infeasibility which can be created by this variable currently fixed to one will be associated with a constraint of index higher than k, thus a constraint to be scanned in one of the subsequent steps. The expansion-phase is terminated when the last constraint is encountered, in which case a new complete tree to Ω is constructed, or when the current value of the objective function exceeds (or equals) the current best solution.

Backtracking is performed in the lexicographically reverse order of the constraints, and hence, in the reverse order of the variables also. Backtracking terminates whenever a constraint l is scanned for which a variable has to be reassigned the value zero, but the variable is not the lexicographically last variable associated with all ingoing arcs of node l. In this case the lexicographically next variable is fixed to one and expansion starts from this constraint l. Thus, during backtracking we erase parts of the current tree constructed through nodes $l, l+1, \ldots, |N|$. The current tree for nodes $1, 2, \ldots, l-1$ remains unchanged. As each expansion starting at node l can only yield trees to $\Omega \cap \{l, l+1, \ldots, |N|\}$, during the whole enumeration process only trees to Ω are considered. The graph is completely implicitly enumerated when during backtracking the algorithm tries to return beyond the first constraint.

Because of the sparsity of ones in the matrix defined by (4.1) and (4.2), list processing techniques, which link the arcs used in the current tree, have been found to greatly increase the speed of the algorithm for large, richly connected graphs.

5. Conclusion and Computational Results

The algorithm described above has been implemented on the CDC-6600 at the Université de Montréal. Some of the results obtained are given in the table below. The following remarks may clarify the entries in the table.

1. For $d = 2$, the time required by FINDGRAPH (the algorithm which constructs the graph, described in Section 3) is given as the sum of two numbers. The second number is the time required for the further reduction of G, described at the end of Section 3. The number of nodes $|N|$ refers to the graph obtained after the further reduction in the case $d = 2$.

2. In the column "Time ENUMERATION", two numbers are given, in the format n_1/n_2. The number n_2 gives the total time (in seconds) for which the implicit enumeration program was run; the number n_1 gives the last time when an improvement in the objective function was found. An asterisk in the table indicates that the solution obtained is guaranteed to be optimal, because complete implicit enumeration was possible in the given time. The small number of asterisks is due to the large size of the graphs treated (cf. *Dreyfus* and *Wagner* [1970]).

Table 1. Computational Results

d	$\|\Omega\|$	$\|N\|$	Time FINDGRAPH (sec)	Time ENUMERATION	
				Costs from d-space	Random costs
2	25	74	2 + 2	*62/395	*27/320
2	40	118	6 + 6	*.05/438	
2	50	155	8 + 10	319/440	
2	60	189	12 + 14	229/430	
2	100	320	37 + 41	28/372	
3	15	101	6	*56/253	
3	25	210	28	149/431	77/431
4	10	60	12	70/448	
4	15	156	51	18/405	5/405
5	5	17	6	*.1/.3	
5	10	83	97	47/153	12/153

3. The results given under the heading "Costs from d-space" are those for the graph with costs defined by the algorithm of Section 3. The results under "Random costs" are those obtained for the same graph, with the costs chosen at random from (0,1000). The introduction of such widely varying costs apparently results in the best found solution being found earlier, because the wide variation results in earlier cutting of the solution tree.

4. The members of Ω were chosen at random from the d-dimensional cube $[-32,32]^d$.

Appendix

We will demonstrate that a Steiner tree to Ω in G^* must be in $G = (N,A)$, whose definition was given in Section 3.

Proposition 1:

Any furcation in a Steiner tree takes place at a glb of some subset of Ω.

Proof:

Assume a Steiner tree to Ω is given which has at least one furcation which is not a glb (reference to some subset of Ω being implicit). Choose such a furcation X (which is, let us say, a q-furcation, $q > 1$) for which each furcation in the subtree rooted at X is a glb. For $1 \leq i \leq q$, let $Y^{(i)}$ be the first glb on the i^{th} branch leading from X. Let Z be the glb of $\{Y^{(1)},\ldots,Y^{(q)}\}$. For each $Y^{(i)}$ there must be a path from X to $Y^{(i)}$ via Z and

$$\|Y^{(i)} - X\| = \|Y^{(i)} - Z\| + \|Z - X\|.$$

Therefore, the cost of the paths joining X to $\{Y^{(1)},\ldots,Y^{(q)}\}$ is $\sum_{i=1}^{q} \|Y^{(i)} - X\| = q\|Z - X\| + \sum_{i=1}^{q} \|Y^{(i)} - Z\|$.

However, if we modify the purported Steiner tree by replacing the paths from X to $\{Y^{(1)},\ldots,Y^{(q)}\}$ with a single path from X to Z and q paths from Z to $\{Y^{(1)},\ldots,Y^{(q)}\}$, the cost of the paths joining X to $\{Y^{(1)},\ldots,Y^{(q)}\}$ is no more than $\|Z-X\|+\sum_{i=1}^{q}\|Y^{(i)}-Z\|$, which is a contradiction.

Having shown that N contains all points of interest, we must now show that A contains enough paths for the definition of a Steiner tree to Ω.

Proposition 2:

A Steiner tree to Ω in G is a Steiner tree to Ω in G^*.

Proof:

Let Φ be a Steiner tree to Ω in G^*. It has previously been shown that all furcations of Φ are in N. It must now be shown that any path between two furcations in Φ has a representative path in A.

Suppose that X, $Y\in N$ and that X is joined to Y in Φ by a path in R^d with no furcations. Suppose further that X has no successor in $\Pi(Y)$. If $X\in\Pi(Y)$, the arc (X,Y) will be in A. If this is not the case, there is a $Z\in\Omega$ such that $Z\in\Pi(Y)$ and $C(Z,Y)<C(X,Y)$. Therefore, Φ can not be a Steiner tree since a cheaper Steiner tree is obtained by replacing the arcs of Φ from X to Y by a sequence of arcs from Z to Y.

Suppose that X has a successor in $\Pi(Y)$. Let Z be the nearest such successor to Y. By the definition of A, $(Z,Y)\in A$. We can replace the sequence of arcs from X to Y in A by an equivalent sequence which passes through Z.

References

Balas, E.: An additive algorithm for solving linear programs with zero-one variables. Operations Res. **13**, 517–546, 1965.

Camin, J. H., and *R. R. Sokal:* A method for deducing branching sequences in phylogeny. Evolution **19**, 311–326, 1965.

Chang, S. K.: The Generation of Minimal Trees with a Steiner Topology. JACM **19**, No. 4, 699–711, 1972.

Courant, R., and *H. Robbins:* What is Mathematics? Oxford University Press, New York 1941.

Dreyfus, S. F., and *R. A. Wagner:* The Steiner Problem in Graphs. Report ORC 70-32, University of California, Berkeley, California September 1970.

Geoffrion, A. M.: Integer programming by implicit enumeration and Balas' method. SIAM Review **9**, No. 2, 178–190, 1967.

Gilbert, E. N., and *H. O. Pollak:* Steiner minimal trees. SIAM J. Appl. Math. **16**, No. 1, 1–28, 1967.

Hakimi, S. L.: Steiner's problem in graphs and its implications. Networks **1**, 113–133, 1971.

Hanan, M.: On Steiner's problem with rectilinear distance. SIAM J. Appl. Math. **14**, No. 2, 255–265, 1966.

Hendrickson, J. A. Jr.: A methodological analysis of numerical cladistics. Ph. D. thesis, University of Kansas, Lawrence, Kansas 1967.

Hendrickson, J. A. Jr.: Clustering in numerical cladistics: a minimum-length directed tree problem. Mathematical Biosciences **3**, 371–381, 1968.

Melzak, Z. A.: On the problem of Steiner. Canad. Math. Bull. **4**, 143–148, 1961.