On the History of the Minimum Spanning Tree Problem

R. L. GRAHAM AND PAVOL HELL

It is standard practice among authors discussing the minimum spanning tree problem to refer to the work of Kruskal (1956) and Prim (1957) as the sources of the problem and its first efficient solutions, despite the citation by both of Borůvka (1926) as a predecessor. In fact, there are several apparently independent sources and algorithmic solutions of the problem. They have appeared in Czechoslovakia, France, and Poland, going back to the beginning of this century. We shall explore and compare these works and their motivations, and relate them to the most recent advances on the minimum spanning tree problem.

*Categories and Subject Descriptors: G.2.2 [**Discrete Mathematics**]: Graph Theory—trees; E.1 [**Data Structures**]—trees; K.2 [**History of Computing**] —software*
General Terms: Algorithms, Theory
Additional Key Words and Phrases: minimum spanning tree

Introduction

The minimum spanning tree problem (MSTP) is one of the most typical and well-known problems of combinatorial optimization; methods for its solution, though simple, have generated important ideas of modern combinatorics and have played a central role in the design of computer algorithms [AhHoUl74], [ReNiDe77], [Chr75], [HorSah78].[1] The problem is typically stated as follows.

> Given such a weighted graph [a graph G, whose nodes represent cities, whose edges represent possible communication links, and whose edge weights represent the costs of construction, or the lengths of the links] one would then wish to select for construction a set of communication links that would connect all the cities and have minimum total cost or be of minimum total length. In either case the links selected will have to form a tree (assuming all weights are positive). In case this is not so, then the selection of links contains a cycle. Removal of any one of the links on this cycle will result in a link selection of lower cost connecting all cities. We are therefore interested in finding a spanning tree of G with minimum cost. (The cost of a spanning tree is the sum of the costs of the edges in that tree.) [HorSah78]

The importance and popularity of the MSTP stem from several facts. It admits an efficient solution, which makes it practical to solve MSTPs for large graphs. (Graphs with thousands of vertices can be handled [BenFri76].) It has obvious applications in the design of computer and communication networks, power and leased-line telephone networks, wiring connections, links in a transportation network, piping in a flow network, etc. [ChoKer73], [DRH70], [EsaWil66], [SaBoRu73], [LobWei57], [Pri57], [Bor26a], [Cho38]. It offers a method of solution to

© 1985 by the American Federation of Information Processing Societies, Inc. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the AFIPS copyright notice and the title of the publication and its date appear, and notice is given that the copying is by permission of the American Federation of Information Processing Societies, Inc. To copy otherwise, or to republish, requires specific permission.
Authors' Addresses: R. L. Graham, Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974. P. Hell, Department of Computing Science, Simon Fraser University, Burnaby, British Columbia V5A 1S6.
© 1985 AFIPS 0164-1239/85/010043–057$01.00/00

[1] In this paper, References are shown in the preceding way in the text and are alphabetized by the abbreviations shown in brackets.

other problems to which it applies less directly, such as network reliability, surface homogeneity tests, picture processing, automatic speech recognition, clustering and classification problems, etc. [VanFra71], [VanFra72], [Kal64], [FiKaSh81], [OstLin74], [PynWar72], [DudHar73], [Zah71], [Zah74], [ZGLL74], [JarSib71], [Sib73], [McQ57], [Bre82], [FLPSZ51], [GowRos69], [Sti67], [EdwCav64], [ClaMil]. It often occurs as a subproblem in a solution of another problem, and MSTP alogrithms are used in several exact and approximation algorithms for the traveling salesman problem, the multiterminal flow problem, the matching problem, etc. [Chr76], [Chr70], [HelKar70], [HelKar71], [Rei79], [HanKra74], [GilGom64]. (Some applications take advantage of the fact, observed in [Pri57], that the minimum spanning tree also minimizes any increasing symmetric function of the weights—see [Hu61], [Kal64].) On the theoretical side, the so-called greedy method typical of all the MSTP solutions is an important concept that can be applied to various other problems and is studied in its general form in the theory of matroids [Edm71], [Wel68], [Gal68], [Law76], [KorLov81].

It is standard practice among authors discussing the MSTP to refer to Kruskal [Kru56] and Prim [Pri57] as the sources of the problem and its first efficient solutions, even though both of these papers refer to Borůvka [Bor26]. In fact, there are several apparently independent sources and algorithmic solutions of the problem. They have appeared in Czechoslovakia, France, and Poland, going back almost to the beginning of this century. We shall explore and compare these works and their motivations, and we shall relate them to the most recent advances on the MSTP.

We expect that the reader has seen some standard algorithm for solving the MSTP. Three of the algorithms that have played a central role in the history of the problem are described below, using the terminology of [Pri57]. It should be noted that there are several variants of the algorithms and that important differences do occur in their implementations.

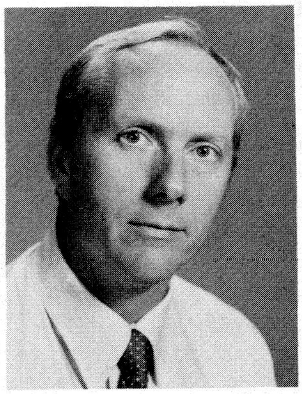
A spanning forest of a graph G (i.e., a subgraph of G containing all vertices and no cycles of G) consists of subtrees of G, each of which is called a fragment of G. Some fragments are just one-vertex trees; these are called trivial fragments. In a weighted graph G, the distance between vertices u and v, or the length of the edge uv, is understood to mean the weight of the edge uv. The distance between two fragments of G is the shortest pairwise distance between a vertex of the first fragment and a vertex of the second fragment. All three algorithms begin with the spanning forest consisting of trivial fragments and selectively add edges to it until it becomes a spanning tree of G. They differ in the criterion used to select the next edge or edges to be added in each iteration.

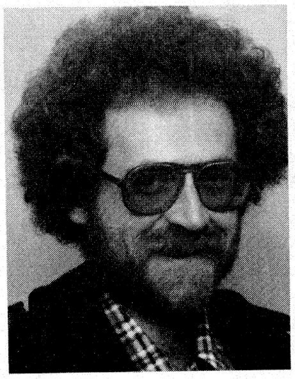
Algorithm 1 (Two Nearest Fragments)
Add a shortest edge which joins different fragments.

Algorithm 2 (Nearest Neighbor)
(A vertex v is arbitrarily chosen.) Add a shortest edge which joins the fragment containing v to another fragment.

Algorithm 3 (All Nearest Fragments)
For every fragment add the shortest edge which joins it to another fragment.

(In Algorithm 3 it is assumed that all edge weights are different, so that a cycle cannot occur.)


Ronald L. Graham received his Ph.D. in mathematics from the University of California at Berkeley in 1962. He has been employed at AT&T Bell Laboratories since that time; he is currently the director of the Mathematical Sciences Research Center. He has also held visiting faculty appointments in mathematics and computer science at the University of California at Los Angeles, Stanford University, and the California Institute of Technology. His research areas include combinatorics, number theory, and analysis of algorithms and their applications.


Pavol Hell is a professor of mathematics and computing science at Simon Fraser University, Vancouver, Canada. He received a Ph.D. in mathematics from the University of Montreal. Previously he was employed at Rutgers, the State University of New Jersey. His interests include graph theory, combinatorial optimization, and computational complexity.

Algorithm 1

Algorithm 1 can also be described as follows. Sort the edges by weight; examine each edge in the order of increasing weight; if the edge examined does not create a cycle with the edges in the current forest, it is added to the forest; otherwise, it is discarded. This is the algorithm proposed by Kruskal [Kru56].

> *Construction A.* Perform the following step as many times as possible: Among the edges of G not yet chosen, choose the shortest edge which does not form any loops [cycles] with those edges already chosen.

Loberman and Weinberger [LobWei57] also discovered Algorithm 1, although they became aware of [Kru56] before their paper was finalized. They included a footnote referencing [Kru56].

> This reference was discovered by the present authors after their procedures had been formulated. It is seen that the procedures presented here and Kruskal's constructions are identical. However, it is felt that the more detailed implementation and general proofs of the procedures justify this paper.

They formulated Algorithm 1 as follows.

> In general, the rules of procedure A for selecting the necessary branches [edges] from the sorted sequence of branches are as follows: Examine the branches sequentially, beginning with the shortest branch. For each branch examine the two nodes [vertices] which constitute its terminals. The first branch is always used, and its two nodes are recorded as belonging to a subtree [nontrivial fragment]. One of four possible conditions are recognized for each succeeding branch which is considered.
>
> *Condition 1.* Neither of the two nodes is present in a subtree. Therefore the branch is made. The two nodes are recorded as constituting another new subtree.
>
> *Condition 2.* Only one of the nodes is present in a subtree. Therefore the branch is made. The new node is added to the subtree containing the other node of this branch.
>
> *Condition 3.* Each of the two nodes is present in a different subtree. Therefore the branch is made. The two different subtrees, each containing one of the nodes, are combined into a single subtree.
>
> *Condition 4.* Both nodes are present in the same subtree. Therefore the branch is *not* made.

To implement Algorithm 1 efficiently, one usually takes advantage of efficient algorithms to sort (or partially sort) the edges by weight, to find the fragment containing a given vertex (determining whether the two ends of the current edge lie in the same fragment), and to merge two fragments into one (if the current edge does join two different fragments). Suitable data structures for such algorithms were developed in [HopUll73]; their application to the MSTP is attributed to M. D. McIlroy [HopUll73]. The best implementations of Algorithm 1 find a minimum spanning tree in time $O(e \log v)$; nearly $O(e)$ time can be achieved if the edge weights are small integers and radix sorting can be used or if the edges are given in sorted order [CheTar76]. (Here and in subsequent discussion of complexities, v is the number of vertices and e the number of edges of the graph in which the minimum spanning tree is sought; we say that an algorithm is nearly time $O(e)$, or nearly linear in e, if its time complexity is $O(e \cdot a(e, v))$, where a is the inverse of Ackermann's function; see [Tar75].)

Algorithm 2

Kruskal [Kru56] viewed his construction A as a special case of a more general construction.

> *Construction B.* Let V be an arbitrary but fixed (nonempty) subset of the vertices of G. Then perform the following step as many times as possible: Among the edges of G which are not yet chosen but which are connected either to a vertex of V or to an edge already chosen, pick the shortest edge which does not form any loops with the edges already chosen.

Kruskal remarks that when V is the set of all vertices of G, construction B reduces to construction A. On the other hand, when V consists of a single vertex v, construction B reduces to Algorithm 2. Indeed, Algorithm 2 may also be described as follows. Sort the edges by weight. Given a fragment F containing v, examine the unused edges in order of increasing weight until an edge is found joining a vertex in F to a vertex outside F. Add that edge of F. At the same time, edges that are found to join two vertices of F may be discarded. The same algorithm was proposed in [LobWei57].

> In general the rules of procedure B for selecting the necessary branches from the sorted sequence of branches are as follows: The branches are examined sequentially unless otherwise specified. Beginning with the shortest branch, the two nodes which constitute its terminals are examined. The first branch is always made and its two nodes are recorded as belonging to the single subtree of nodes of procedure B. One of three possible conditions are recognized for each succeeding branch which is considered:
>
> *Condition 1.* Neither of the two nodes is present in the subtree. Therefore, the branch is *not* made at this time. The succeeding branches are then examined in sequence. After the first subsequent branch is made which fulfills the requirements of condition 2 (which follows), the examination reverts back to the first branch which has not yet been made because of condition 1.
>
> *Condition 2.* Only one of the two nodes is present in the subtree. Therefore the branch is made and the other node is added to the subtree. Next, it is determined

whether there are any branches in previous positions in the sequence which met condition 1 (above) and have not yet been made. If so, the examination of the branches reverts back to the first of such branches. If not, the examination of the branches proceeds in normal sequence.

Condition 3. Both nodes are present in the subtree. Therefore the branch is *not* made.

The reader may be aware that Algorithm 2 is usually attributed to Prim [Pri57]. Prim formulated a general class of algorithms.

Principle 1. Any isolated terminal [trivial fragment] can be connected to a nearest neighbor.

Principle 2. Any isolated [nontrivial] fragment can be connected to a nearest neighbor by a shortest available link.

Principles 1 and 2 by themselves are sufficiently flexible to incorporate all three of the preceding algorithms. Prim preferred Algorithm 2 because of computational considerations.

When it is desired to determine a shortest connection network directly from the distance table representation—either manually, or by machine computation—one of the numerous practical algorithms obtainable by restricting the freedom of choice allowed by principles 1 and 2 is distinctly superior to other alternatives. This variant is the one in which principle 1 is used but once to produce a single isolated fragment, which is then extended by repeated applications of principle 2. The successive steps of an efficient computational procedure [are illustrated in an example. The illustration operates on so-called F tables defined as follows:] The entries in the top rows of successive F tables are the distances from *the* connected [i.e., nontrivial] fragment to the unconnected terminals [i.e., trivial fragments] at each stage of fragment growth. The entries in parentheses in the second rows of these tables indicate the nearest neighbor *in the fragment* of the external terminal in question. The computation is started by entering the first row of the distance table into the F table (to start the growing fragment from Terminal 1). A smallest entry in the F table is then selected [and the corresponding edge is deleted from the F table and entered in the solution]. The remaining entries in the first stage F table are then compared with the corresponding entries in the [appropriate] row of the distance table. If any entry of this "added terminal" distance table is smaller than the corresponding F table entry, it is substituted for it, with a corresponding change in the parenthesized index. This process is repeated to yield the list of successive nearest neighbors to the growing fragment.

Computational considerations also prompted Dijkstra to formulate Algorithm 2 as follows [Dij59].

In the course of the construction that we present here, the branches are subdivided into three sets:

I. the branches definitely assigned to the tree under construction (they will form a subtree);
II. the branches from which the next branch to be added to set I, will be selected;
III. the remaining branches (rejected or not yet considered).

The nodes are subdivided into two sets:

A. the nodes connected by the branches of set I,
B. the remaining nodes (one and only one branch of set II will lead to each of these nodes).

We start the construction by choosing an arbitary node as the only member of the set A and placing all branches that end in this node in set II. To start with, set I is empty. From then onwards we perform the following two steps repeatedly.

Step 1. The shortest branch of set II is removed from this set and added to set I. As a result one node is transferred from set B to set A.

Step 2. Consider the branches leading from the node that has just been transferred to set A to the nodes that are still in set B. If the branch under consideration is longer than the corresponding branch in set II, it is rejected; if it is shorter, it replaces the corresponding branch in set II, and the latter is rejected. We then return to step 1 and repeat the process until the sets II and B are empty. The branches in the set I form the tree required.

The approaches to Algorithm 2 taken by Dijkstra and Prim are similar. (Prim's paper appeared earlier, but Dijkstra was apparently unaware of it.) Both Prim and Dijkstra were primarily concerned with storage requirements.

The F and "added terminal" distance tables grow shorter as the number of unconnected terminals is decreased. This computational procedure . . . never requires access to more than two rows of distance data at a time—no matter how large the problem. [Pri57]

The solution given here is to be preferred to the solution given by J. B. Kruskal [Kru56] and those given by H. Loberman and A. Weinberger [LobWei57]. In their solution all the—possibly $\frac{1}{2}n(n-1)$—branches are first of all sorted according to length. Even if the length of the branches is a computable function of the node coordinates, their methods demand that data for all branches are stored simultaneously. Our method only requires the simultaneous storing of the data for at most n branches, viz. the branches in sets I and II and the branch under consideration in step 2. [Dij59]

It turns out that Algorithm 2 was formulated quite clearly over a quarter century before [Pri57], [Kru56], [LobWei57], and [Dij59], by V. Jarník [Jar30].

Definice Množství J. Jest

$$J = [a_1, a_2], [a_3, a_4], \ldots, [a_{2n-3}, a_{2n-2}],$$

kde $a_1, a_2, \ldots$ jsou definována takto:

1. *Krok.* Za a_1 zvolme kterýkoliv z prvků $1, 2, \ldots, n$;

a_2 budiž definováno vztahem

$$r(a_1, a_2) = \min r(a_1, l) \qquad (l = 1, 2, \ldots, n; 1 \neq a_1).$$

k-tý Krok. Je-li již definováno

$$a_1, a_2, \ldots, a_{2k-2} \qquad (2 \leqslant k < n), \qquad (5)$$

definujme a_{2k-1}, a_{2k} vztahem

$$r(a_{2k-1}, a_{2k}) = \min r(i, j),$$

kde i probíhá všechna čísla $a_1, a_2, \ldots, a_{2k-2}$; j všechna ostatní z čísel $1, 2, \ldots, n$. Při tom budiž a_{2k-1} jedno z čísel (5), takže a_{2k} není obsaženo mezi čísly (5).

Je patrno že při tomto postupu je mezi čísly (5) právě k čísel různých, takže pro $k < n$ lze k-tý krok provésti.

Translation

Definition of the set J. Let

$$J = [a_1, a_2], [a_3, a_4], \ldots, [a_{2n-3}, a_{2n-2}],$$

where $a_1, a_2, \ldots$ are defined as follows:

First Step. Choose as a_1 any of the elements $1, 2, \ldots, n$; let a_2 be defined by the relation

$$r(a_1, a_2) = \min r(a_1, l) \qquad (l = 1, 2, \ldots, n; l \neq a_1).$$

kth Step. Having defined

$$a_1, a_2, \ldots, a_{2k-2} \qquad (2 \leqslant k < n), \qquad (5)$$

we define a_{2k-1}, a_{2k} by the relation

$$r(a_{2k-1}, a_{2k}) = \min r(i, j),$$

where i ranges over the numbers $a_1, a_2, \ldots, a_{2k-2}$; j ranges over the other numbers among $1, 2, \ldots, n$. When doing this, we let a_{2k-1} be one of the numbers in (5) so that a_{2k} is not among the numbers in (5).

It is evident that in this process exactly k of the numbers in (5) are different, so that for $k < n$ the kth step can be performed. [J is the set of edges of a minimum spanning tree; $r(i, j)$ is the weight of the edge joining i and j.]

The first implementations of Algorithm 2 were implicit in [Pri57], [Dij59]. They run in time $O(v^2)$. It is possible to implement the algorithm to run in time $O(e \log v)$ [KerVan72], [Joh75]. Using general heaps (to store vertices adjacent to the fragment F), as in [Joh75], Algorithm 2 can be implemented in time $O(e \log v)$, where the log is to the base $\max(2, e/v)$ [Tar82] . In dense graphs, Johnson's implementation runs in time $O(e)$ [Joh75]. (Dense graphs satisfy $e \geqslant Cv^p$ for some constants $C > 0$ and $p > 1$.)

Algorithm 3

Algorithm 3 is historically the most interesting. It dates back in its first documented full formulation to two 1926 papers by Otakar Borůvka [Bor26], [Bor26a]; it has been independently rediscovered by a number of other authors; and it seems to lead to the most efficient implementations. In the next section we will follow the work of Borůvka and relate Algorithm 3 to some recent progress on the MSTP. Here we trace the other early formulations of Algorithm 3.

In 1938 Gustave Choquet published a note (communicated to the *Comptes Rendus* by Elie Cartan) in which he analyzed the MSTP; while he also described the algorithm in the case of a general metric space (and considered further generalizations), the main statement is given in terms of n cities and their euclidean distances [Cho38].

Construction du Réseau Minimum. On joint par un segment chaque ville à la ville la plus voisine. Si l'ensemble de tous ces segments forme un continu, celui-ci est le réseau cherché. Sinon, on joint chacun des continus formés au continu le plus voisin par un segment joignant les deux villes les plus voisines de ces deux continus. Si ensemble ainsi formé est un continu, celui-ci est la réseau cherché. Sinon, on recommence de la même façon. Le réseau cherché sera tracé après $2n$ opérations élémentaires au plus, en appelant opération élémentaire la recherche du continu le plus voisin d'un continu donné.

Translation

Construction of the Minimum Network. One joins by a segment each city with the city nearest to it. If the set of all these segments forms a continuum [is connected], it is the desired network. If not, one joins each of the continua [connected components] with the continuum nearest to it by means of a segment joining the closest two cities of the two continua. If the set formed in this fashion is a continuum, it is the desired network. If not, one continues the same way. The desired network will be found after at most $2n$ elementary operations, where an elementary operation is the search for the continuum nearest to a given continuum.

(Note Choquet's concern about the number of operations made by Algorithm 3.)

In 1961, George Sollin in Paris prepared a manuscript entitled "Problème de l'Arbre Minimum" and presented the material in it at a seminar organized by C. Berge in Paris on February 8, 1961. Sollin describes his MSTP algorithm as follows ([Sol 61], cf. also [Sol62]).

On procèdera par étapes en joignant un sommet quelconque à son voisin le plus proche, de façon à obtenir des arbres partiels de premier ordre.

Puis, on considèrera ces arbres partiels de premier ordre comme des sommets et on réitèrera l'algorithme jusqu'au moment où l'arbre partiel de kè ordre sera unique.

Translation

One proceeds in steps by joining any vertex to its nearest neighbor, thereby obtaining partial trees of the first order.

Then, one considers these partial trees of the first order as vertices and iterates the algorithm until one obtains a unique tree of the kth order.

Sollin's manuscript was never published, although it was carefully referenced in the book of Berge and Ghoula-Houri [BerGho65].

Another independent discovery of Algorithm 3 was reported in 1950 by a research group based in Wroclaw, Poland [FLPSZ51].

Travail collectif, rédigé par J. Łukaszewicz et contenant les résultats obtenus par le Groupe Général des Applications de l'Institut Mathématique de l'Etat: K. Florek, J. Łukaszewicz, J. Perkal, H. Steinhaus et S. Zubrzycki (Wroclaw).

. . . Voici notre méthode pour former une dendrite avec les points de l'ensemble Z. Unissons, par un segment, chacun d'eux au point le plus proche; les segments ainsi obtenus seront appelés *Liens du Premier Ordre*. Ils forment une ou plusieures lignes polygonales connexes qui sont des liaisons de points de certains sous-ensembles disjoints de Z. Nous appelerons ces sous-ensembles *Groupements du Premier Ordre*. Liens chacun d'eux avec le groupement le plus proche (par *Distance Entre Deux Groupements* on comprend évidemment la plus petite distance entre leurs points deux-à-deux) à l'aide d'un segment qui réalisera la distance entre eux et que nous appelerons maintenant *Lien du Second Ordre*. On procède ainsi, en employant des liens d'ordre de plus en plus élevé, jusqu'à ce qu'on obtienne une ligne polygonal connexe liant tous les points de l'ensemble Z.

Translation

A joint work, directed by J. Łukaszewicz, and containing results obtained by the General Applications Group of the State Mathematical Institute: K. Florek, J. Łukaszewicz, J. Perkal, H. Steinhaus, and S. Zubrzycki (Wroclaw).

. . . Here is our method of constructing a dendrite [spanning tree] with the points of a [given] set Z. Let us join, by a segment, each point to the point nearest to it; these segments will be called *Connections of the First Order*. They form one or more connected polygonal lines [subtrees] which are the connections of the points of certain disjoint subsets of Z. These subsets will be called *Groups of the First Order*. Let us join each such group with the group nearest to it (by distance between groups one understands, of course, the smallest pairwise distance between their points), by a segment realizing their distance, which we shall call a *Connection of the Second Order*. We proceed this way, using connections of higher and higher order, until we obtain a connected polygonal line joining all the points of the set Z.

Łukaszewicz et al. motivated their study of minimum spanning trees with applications in anthropology, biology, linguistics, etc. In essence it was to be a tool in taxonomy ([FLPSZ51, p. 319]).

Séance du 3 Novembre 1950.
Group Général des Applications de l'Institut Mathématique de l'Etat, Une Methode Taxonomique et Ses Applications aux Sciences Naturelles (présenté par J. Perkal).

Les auteurs proposent une nouvelle méthode taxonomique qui tire parti de l'idee connue depuis 1909 et due a J. Czekanowski, à savoir de la notion de distance entre les individus. Le tableau $\{a_{ik}\}$ des distances P_iP_k, ou P_i $(i = 1, 2, \ldots, n)$ désigne le i'ème objet, est point de départ. Or, au lieu de ranger les objets en une suite, ce qui était le but de la méthode ancienne, on les considère comme des points d'un plan, et on les joint part $n - 1$ segments afin d'obtenir une dendrite la méthode en question fournit *la plus courte de toutes dendrites possibles*. Les auteurs ont appliqué leur méthode aux 14 villes principales de Pologne, aux 22 cranes de Ngandong et autres 12 cranes décrits par W. Steslicka, et aux 42 trouvailles préhisoriques décrites par J. Czekanowski.

Translation

Session of November 3, 1950.
General Applications Group of the State Mathematical Institute, A Method in Taxonomy and Its Applications to Natural Sciences (presented by J. Perkal).

The authors propose a new method in taxonomy which is based on an idea known since 1909 and due to J. Czekanowski, namely, the notion of distance between individuals. One starts with the table $\{a_{ik}\}$ of distances P_iP_k, where P_i $(i = 1, 2, \ldots, n)$ denotes the ith object. Then, instead of ordering the objects in a sequence, which was the goal of the old method, we consider them points in a plane, and join them with $n - 1$ segments to obtain a dendrite; our method provides *the shortest among all possible dendrites*. The authors have applied their method to the 14 principal cities in Poland, to the 22 skulls of Ngandong and 12 other skulls described by W. Steslicka, and to the 42 prehistoric finds described by J. Czekanowski.

It is tempting to speculate how much the above acknowledgment of the work of J. Czekanowski was intended to apply to the mathematical basis of their method. Did Czekanowski himself come close to formulating Algorithm 3 in 1909? The same authors (coining the term *Wroclaw taxonomy* to describe their method) wrote in [FLPSZ51a]:

From the English summary

Since 1909 there has been known in anthropology J. Czekanowski's method of classification and division into groups a collection of individuals characterized by metrical features. We propose a new method of classification and division into groups which is in some way an extension and improvement of Czekanowski's method. The starting point for both these methods is Czekanowski's table of distances of individuals.

From the text

Po wyrachowaniu tej tablicy wyrysowuje się diagram (zmodyfikowany) Czekanowskiego, który powstaje z tablicy przez zastąpienie liczb zakreskowaniem. Liczby najmniejsze zastępuje się całkowitym zaczernieniem kratki, odległości coraz większe—coraz rzadszym zakreskowaniem, odległości większe od jakiejś ustalonej liczby zastępuje się białymi kratkami. Następnie zmienia się uporządkowanie badanych indywiduów i dąży się do tego, żeby ciemne pola diagramu utworzyły kwadraty blisko głównej przekątnej diagramu. Uzyskuje się w ten sposób pewne liniowe uporządkowanie indywiduów, oraz wyróżnia się grupy indywiduów wzajemne bliskich. Jak stąd widać . . . metoda polega na odpowiednim uporządkowaniu tej tablicy.

Translation

After the computation of the table, one draws the (modified) Czekanowski diagram, which arises from the table by replacing the numbers with shades [of black]. The smallest numbers are replaced by completely black cells, distances greater and greater by shading the cells less and less darkly, and distances exceeding a fixed value are replaced by white cells. Next one changes the order of the individuals with the aim of having the dark cells form squares near the main diagonal of the diagram. One obtains in this way a certain linear order of individuals, and one can distinguish groups of individuals that are relatively close to each other. As can be seen from this . . . the method depends on a suitable ordering of the table.

Czekanowski described it as follows [Cze09]:

Ich nehme ein Quadratennetz, in diesem Falle mit 13 Quadraten Seitenlange, und ordne jedem Quadrate in der Reihenfolge der obigen Tabelle die Werte der durchschnittlichen Differenzen zu. Hierauf bedecke ich die einzelnen Quadrate mit bestimmten Farben, bzw. Stricharten und zwar in folgender Weise: Die drei kleinsten Werte jeder senkrechten Kolonne erhalten eine schwartze Farbung, der nachstfolgende Wert wird dick senkrecht, der weiterfolgende mitteldick, der drittfolgende dunn gestrichen. Damit sind die sechs niedersten Werte der Kolonne zur Darstellung gebracht. . . . Die Felder, die den relativ hoheren Werten entsprechen, bleiben weiss. Man sieht infolgendessen in der graphischen Darstellung deutlich, dass die Schadel zwei gruppen bilden.

Translation

I take a square grid, in this case 13 squares on each side, and assign to each cell [the value of the corresponding] average difference. Then I equip each cell with a certain color—that is, a kind of shading—in the following manner: The three smallest values of each column will receive the black color, the next value will be shaded by thick vertical lines, the one after by semithick lines, and the third one by thin lines. This causes the six smallest

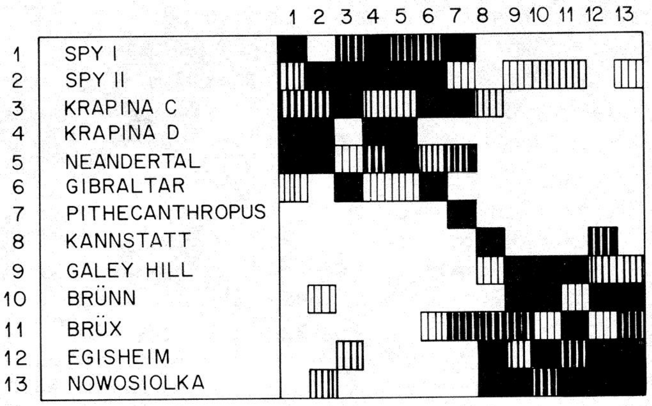


Figure 1

differences to stand out. . . . The cells corresponding to relatively higher values remain white. This graphic depiction illustrates clearly that the skulls form two groups. (See Figure 1.)

(A similar description appears in [Cze11], [Cze28].) It appears that Czekanowski's method of classification was originally intended solely as a tool to display graphically the table of differences in order to recognize the natural groupings. It bears a superficial resemblance to Algorithm 3 in its "quasi-greedy" way of selecting related objects. On the whole, however, Czekanowski's contribution, while paving the way for [FLPSZ51], [FLPSZ51a], cannot be linked directly to Algorithm 3, nor to the MSTP.

The clustering technique [FLPSZ51], [FLPSZ51a] that began with Czekanowski's work is similar to methods developed independently in the West just a few years later [Sne57], [McQ57], [SokSne63], [Joh67], [GowRos69], [Zah71]. These methods are usually referred to as single-linkage cluster analysis and form a part of so-called numerical taxonomy [Sor48], [Sne57], [Joh67], [SokSne63], [JarSib71], [RogCar71], [Roh73], etc. To some degree these developments are also related to independent discoveries of MSTP algorithms. Typically one of the Algorithms 1–3 is embedded, without stating the MSTP or claiming optimality of the procedure, in a taxonomic or clustering algorithm. For example, [ArkBra67, p. 110] suggests a clustering algorithm that implicitly defines Algorithm 2; McQuitty [McQ57] applies in his taxonomic method one iteration of Algorithm 3 to identify the clusters (cf. also [Joh67]); in a similar vein [Sne57] is related to Algorithm 1.

Borůvka

In this section we examine the contribution of Otakar Borůvka, whose papers [Bor26], [Bor26a] appear to have the first statement of Algorithm 3, as well as the first explicit formulation of the problem.

Borůvka became aware of the problem during the rural electrification of Southern Moravia. He wrote in his recollections [Bor77]:

> Studium na školách technického směru mně velmi přiblížilo inženýrské vědy a způsobilo, že jsem měl pro technické a jiné aplikace matematiky vždycky plné porozumění. Brzy po skončení I. světové války, na začátku 20. let, prováděly Západomoravské elektrárny v Brně elektrifikaci jižní Moravy. V rámci přátelských styků, které jsem měl s některými jejich pracovníky, jsem byl požádán, abych z hlediska matematického řešil otázku co nejúspornějšího provedení elektrovodní sítě. Podařilo se mně najít konstrukci—dnes bychom to vyjádřili—největšího souvislého podgrafu minimální délky, kterou jsem uveřejnil v r. 1926, tedy v době kdy teorie grafů neexistovala.

Translation

> My studies at polytechnical schools made me feel very close to engineering sciences and made me fully appreciate technical and other applications of mathematics. Soon after the end of World War I, at the beginning of the 1920s, the Electric Power Company of Western Moravia, Brno, was engaged in rural electrification of Southern Moravia. In the framework of my friendly relations with some of their employees, I was asked to solve, from a mathematical standpoint, the question of the most economical construction of an electric power network. I succeeded in finding a construction—as it would be expressed today—of a maximal connected subgraph of minimum length, which I published in 1926 (i.e., at a time when the theory of graphs did not exist).

In a letter to the authors, Borůvka further reminisces about his discovery.

> Není mně známo, že by se někdo přede mnou oním nebo podobným problémem zabýval.
>
> V r. 1926–27 jsem studoval v Paříži u prof. E. Cartana. V té době konal na pařížské universitě přednášky (o algebraické geometrii) prof. Coolidge z U.S.A. Prof. Coolidge vedl take seminář, jehož účastníci přednášeli o svých pracích. Z témat, která jsem prof. Coolidgeovi pro svoji přednášku navrhl, vybral zmíněný minimální problém. Přednášku jsem proslovil někdy na jaře v r. 1927. Vzpomínam si, že této přednášce byl přítomem B. Segre (nyní emer. professor university v Římě), který, také tehdy v Paříži studoval.

Translation

> I am not aware of anyone having worked on this or a similar problem before me.
>
> In the year 1926–1927 I studied in Paris under Professor E. Cartan. At the time, Professor Coolidge from the United States was giving a series of lectures in algebraic geometry. In addition, Professor Coolidge directed a seminar, whose participants gave lectures about their work. Among the topics I suggested for my lecture was the minimal problem (MSTP), which Professor Coolidge chose. I gave the lecture sometime in the spring of 1927. I recall that among those present was Professor B. Segre (now Professor Emeritus at the University of Rome), who was also studying in Paris at the time.

(Apparently Cartan was not present, or did not remember the result when communicating Choquet's paper [Cho38] to the *Comptes Rendus*.)

Borůvka formulated the MSTP as follows [Bor26]:

> Budiž dána matice M čísel $r(x, y)$ $(x, y = 1, 2, \dots, n;\ n \geqslant 2)$, až na podmínku $r(x, x) = 0$, $r(x, y) = r(y, x)$, kladných a vzájemně různých. Jest vybrati z ní skupinu čísel vzájemně a od nuly různých takovou aby 1. bylo možno, jsou-li p_1, p_2 libovolná od sebe různá přirozená čísla $\leqslant n$, vybrati z ní skupinu částečnou tvaru
>
> $$r(p_1, c_2),\ r(c_2, c_3),\ r(c_3, c_4),\ \dots,\ r(c_{q-2}, c_{q-1}),\ r(c_{q-1}, p_2)$$
>
> 2. součet jejích členů byl menší než součet členů kterékoliv jiné skupiny čísel vzájemně a od nuly různých, hovící podmínce 1.

Translation

> Given a matrix M of numbers (x, y) $(x, y = 1, 2, \dots, n;\ n \geq 2)$, all positive and pairwise different, with the exception of $r(x, x) = 0$ and $r(x, y) = r(y, x)$, find a subset of the entries, pairwise different and nonzero, such that 1. for any p_1, p_2, different natural numbers $\leqslant n$, the subset contains some
>
> $$r(p_1, c_2),\ r(c_2, c_3),\ r(c_3, c_4),\ \dots,\ r(c_{q-2}, c_{q-1}),\ r(c_{q-1}, p_2)$$
>
> 2. the sum of its members is smaller than the sum of members of any other set of numbers pairwise different and nonzero satisfying condition 1.

Borůvka's solution appeared in [Bor26], where he gives a complicated description; it nevertheless implicitly describes Algorithm 3 and begins as follows.

> *Řešení.* Budiž f_0 libovolné z čísel x a budiž $[f_0\, f_1]$ nejmenší z čísel $[f_0\, y_0]$ $(y_0 \neq f_0)$. Množství čísel $[f_1\, y_1]$ $(y_1 \neq f_0, f_1)$ jest pak buď prázdné anebo nikoliv. V prvním případě položme $F = [f_0\, f_1]$, v případě druhém jest nejmenší z čísel $[f_1\, y_1]$ buď větší než $[f_0\, f_1]$, anebo menší. Je-li větší, položme $F = [f_0\, f_1]$, je-li menší, budiž $[f_1\, f_2]$ nejmenší z čísel $[f_1\, y_1]$. Množství čísel $[f_2\, y_2]$ $(y_2 \neq f_0, f_1, f_2)$ je pak buď prázdné anebo nikoliv. V případě prvním položme $F = [f_0\, f_1], [f_1\, f_2]$, v případě druhém jest nejmenší z čísel $[f_2\, y_2]$ buď větší než $[f_1\, f_2]$, anebo menší.

Translation

> *Solution.* Let f_0 be any of the numbers x and $[f_0\, f_1]$ the smallest among the numbers $[f_0\, y_0]$ $(y_0 \neq f_0)$. The set of numbers $[f_1\, y_1]$ $(y_1 \neq f_0, f_1)$ is then either empty or not.

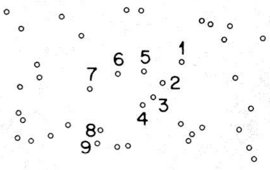


Figure 2

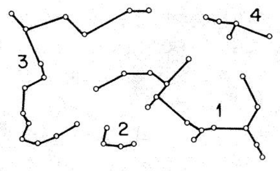


Figure 4

In the former case we let $F = [f_0\, f_1]$; in the latter case the smallest of the numbers $[f_1\, y_1]$ is either greater than $[f_0\, f_1]$ or smaller. If $[f_1\, y_1]$ is greater, we let $f = [f_0\, f_1]$; if $[f_1\, y_1]$ is smaller, let $[f_1\, f_2]$ be the smallest of the numbers $[f_1\, y_1]$. The set of numbers $[f_2\, y_2]$ $(y_2 \neq f_0, f_1, f_2)$ is then either empty or not. In the first case we let $F = [f_0\, f_1], [f_1\, f_2]$; in the second case the smallest of the numbers $[f_2\, y_2]$ is either greater than $[f_1\, f_2]$ or smaller. [Here $[x, y] = r(x, y)$ is the length of the edge xy.]

Instead of giving the full five pages of Borůvka's description, we will briefly summarize it in modern terminology.

Choose a vertex v and the shortest incident edge vw_1. If there exist edges w_1x shorter than vw_1, choose the shortest such edge w_1w_2. Continue in this way, as long as possible, constructing a simple path $vw_1, w_1w_2, \ldots, w_{k-1}w_k$, where each w_iw_{i+1} is the shortest edge incident with w_i and is shorter than $w_{i-1}w_i$. Begin at a new vertex p and construct as above another simple path $pq_1, q_1q_2, \ldots, q_{l-1}q_l$, with l as large as possible under the constraint that $p, q_1, q_2, \ldots, q_{l-1}$ are disjoint from the previous path or paths (as well as the constraint that each q_iq_{i+1} is the shortest edge incident with q_i and shorter than $q_{i-1}q_i$). Repeat until all vertices have been included on some such path. These paths form fragments, and it is easy to see (and is stated explicitly in [Bor26]) that an edge ab is in the resulting forest G if and only if it is the shortest edge at a or at b. Hence the forest G is the same as the one obtained by joining each vertex to its nearest neighbor. The subsequent description in [Bor26] is parallel to our description of Algorithm 3: one forms the distance matrix for the set of fragments of G and repeats the process, producing another forest G_1, then G_2, and so on, until the forest is just one tree G_{u-1}, the solution.

Borůvka's pioneering contribution to the MSTP has not been completely unrecognized. Kruskal [Kru56] attributes the formulation of the problem to [Bor26] and mentions that the uniqueness of the minimum spanning tree (if all edge lengths are different) is proved by

a not unreasonable method of constructing a spanning tree of minimum length.

Several other authors referring to [Kru56] have also included [Bor26] among their references. Because the description of Algorithm 3 in [Bor26] is quite complicated, the paper has been called unnecessarily elaborate. In the same year, however, Borůvka published another communication [Bor26a] directed at engineers, in which he left no doubt that he had Algorithm 3 clearly in mind. There he gave the example shown in Figure 2, assuming the vertices to be points in the plane and their distances to be euclidean distances.

Každý z daných bodů spojím s bodem nejbližším. Tedy na př. bod 1 s bodem 2, bod 2 s bodem 3, bod 3 s bodem 4 (bod 4 s bodem 3), bod 5 s bodem 2, bod 6 s bodem 5, bod 7 s bodem 6, bod 8 s bodem 9 (bod 9 s bodem 8), atd. Obržím řadu polygonálních tahu 1, 2, . . . , 13 (obr. 3).

Každý z nich spojím nejkratším způsobem s tahem nejbližším. Tedy na př. 1 s tahem 2 (tah 2 s tahem 1), tah 3 s tahem 4 (tah 4 s tahem 3) atd. Obdržím řadu polygonálních tahů 1, 2, . . . , 4 (obr. 4).

Každý z nich spojím nejkratším způsobem s tahem nejbližším. Tedy tah 1 tahem 3, tah 2 s tahem 3 (tah 3 s tahem 1), tah 4 s tahem 1. Obdržím konečně jediný polygonální tah (obr. 5), jenž řeší danou úlohu.

Translation

I will join each of the given points with the point nearest to it. Thus, for example, point 1 with point 2, point 2

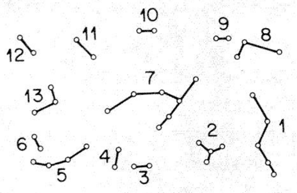


Figure 3

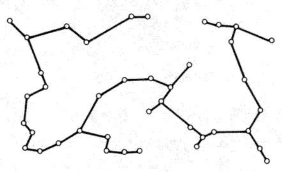


Figure 5

with point 3, point 3 with point 4 (point 4 with point 3), point 5 with point 2, point 6 with point 5, point 7 with point 6, point 8 with point 9 (point 9 with point 8), etc. I will obtain a sequence of polygonal strokes [i.e., fragments] 1, 2, . . . , 13 (Figure 3).

I will join each of these in the shortest possible way with the stroke nearest to it. Thus for example, stroke 1 with stroke 2 (stroke 2 with stroke 1), stroke 3 with stroke 4 (stroke 4 with stroke 3), etc. I will obtain a sequence of polygonal strokes 1, 2, . . . , 4 (Figure 4).

I will join each of these in the shortest possible way with the stroke nearest to it. Thus stroke 1 with stroke 3, stroke 2 with stroke 3 (stroke 3 with stroke 1), stroke 4 with stroke 1. I will finally obtain a single polygonal stroke (Figure 5), which solves the given problem.

In [Bor26], Borůvka also gave a justification of the greedy method. Similar arguments have been used since to justify any of the greedy algorithms discussed here. Borůvka makes the following two statements to prove the correctness of his algorithm.

Věta III. Skupina K′ obsahuje řadu skupin G.

Věta IV. Budiž $u \geqslant 2$, $v \leqslant u - 1$. Obsahuje-li skupina K′ skupiny $G, G_1, \ldots, G_{v-1}$, obsahuje skupinu G_v.

Translation

Theorem III. The set K′ [solution of the MSTP] contains [the fragments] of G.

Theorem IV. Let $u \geqslant 2$, $v \leqslant u - 1$. If the set K′ contains the [forests] $G, G_1, \ldots, G_{v-1}$, then it contains [the forest] G_v.

Theorem III is proved by the usual method: each edge of uv of G is the shortest edge at u or at v. Assume it is the shortest edge at u, and it is not in the solution K′. Then one can insert uv in K′, deleting an appropriate uw, to obtain a tree of lower cost than K′. Theorem IV is deduced from Theorem III by shrinking the fragments of G_{v-1}.

In a somewhat more modern terminology, Jarník justified Algorithm 2 as follows [Jar30].

1. Pomocná Věta. [. . .] každá mkč obsahuje dvojici $[a_1, a_2]$.

2. Pomocná Věta. Budiž S souvislá část; $h_1\ h_2, \ldots, h_s$ bud'te všechny indexy množství S; budiž $s < n$. Bud'te $l_1, l_2, \ldots, l_t$ ona z čísel $1, 2, \ldots, n$, jež nejsou indexy množství S; budiž

$$r(a,b) = \min r(h_i, l_j) \qquad (i = 1, 2, \ldots, s;\ j = 1, 2, \ldots, t).$$

Tvrdím: Každá mkč, jež obsahuje S, obsahuje i dvojici $[a, b]$.

Translation

Lemma 1. [In the formulation cited above] each minimum spanning tree contains the pair $[a_1, a_2]$.

Lemma 2. Let S be a connected [subgraph]; let the vertices of S be $h_1, h_2, \ldots, h_s$; let $s < n$. Let $l_1, l_2, \ldots, l_t$ be those numbers $1, 2, \ldots, n$, which are not vertices of S; let

$$r(a, b) = \min r(h_i, l_j) \qquad (i = 1, 2, \ldots, s;\ j = 1, 2, \ldots, t).$$

Claim: Each minimum spanning tree which contains S also contains the pair $[a, b]$.

The proofs in Jarník's paper are nearly identical to the usual textbook proofs justifying Algorithm 2.

A straightforward implementation of Algorithm 3 would run in time $O(e \log v)$, because each time the rule defining Algorithm 3 is applied, the number of fragments decreases by at least one-half. Yao [Yao75] has discovered an implementation of Algorithm 3 in time $O(e \log \log v)$ using the linear selection algorithm [BFPRT73]. Since then a number of other efficient MSTP algorithms have been given that either directly depend on Algorithm 3 or are similar to it in spirit. These include several $O(e \log \log v)$ algorithms of Cheriton and Tarjan [CheTar76], a linear expected time algorithm of Karp and Tarjan [KarTar81], and a probabilistic MSTP algorithm for coordinate spaces of Rohlf [Roh78]. The algorithms of [CheTar76] are more practical than that of [Yao75] because they do not require a linear selection algorithm; moreover, they run in time $O(e)$ for dense graphs and $O(v)$ for planar graphs. Another potential advantage of Algorithm 3 is that it appears well suited for parallel computation [Ben80], [Tar82].

The recent interest in Algorithm 3 was apparently initiated by its rediscovery by Sollin [Sol61], [Sol62], although it was not generally known that Algorithm 3 is actually first due to Borůvka [Bor26], [Bor26a]. See also [Ben80], [RoFiHo72], [PaSa81], [Sp77], [Cha79], [GaHuSp79].

Other Algorithms

Although we have emphasized the history of Algorithms 1, 2, and 3, several other early algorithms were proposed for the minimum spanning tree problem. For the most part these algorithms have taken the dual approach of greedily excluding long edges (or some combination of the original approach and this dual approach).

The first algorithm of this type, which we call Algorithm 4, appears to be Kruskal's construction A' [Kru56].

Construction A′. This method is in some sense dual to A. Perform the following step as many times as possible. Among the edges not yet chosen, choose the longest edge whose removal will not disconnect [the graph]. Clearly the set of edges *not* eventually chosen forms a spanning tree of G, and in fact it forms a shortest spanning tree.

Algorithm 4 was discovered independently by A. Kotzig [Kot61] (who then found out about [Kru56] from a referee).

Utvorme postupnost' podgrafov grafu G: $G_0, G_1, \ldots, G_m$ a postupnost' hrán z G: $h_1, h_2, \ldots, h_m$ $[m = e - v + 1]$ takto; Hrana h_i je l'ubovol'ne pevne zvolená taká hrana z $\bar{H}(G_{i-1})$ o ktorej platí: žiadna z hrán v $\bar{H}(G_{i-1})$ nemá hodotu väčšiu než $l(h_i)$: graf G_i vznikne z grafu G_{i-1} ked' z neho odstránime hranu h_i a je $G_0 = G$. Podgraf G_m ma vlastnosti (x), (y), a (z)

Translation

Let us form a sequence of graphs G: $G_0, G_1, \ldots, G_m$ and a sequence of edges of G: $h_1, h_2, \ldots, h_m$ $[m = e - v + 1]$ as follows: The edge h_i is any [edge of maximum length among edges which belong to a cycle of G_{i-1}]; the graph G_i arises from the graph G_{i-1} by the removal of the edge h_i, where $G_0 = G$. [Then G_m is a solution.]

Another algorithm with an interesting history, which we call Algorithm 5, has been viewed as a modification of Algorithm 1 [Ros67].

Données: Une liste des arêtes de G dans un ordre quelconque, et pour chaque arête son numéro d'ordre et ses deux extrémités.

Principe: On lit la liste des arêtes. Toute arête lue est "retenue." Quand l'une d'elles constitue un cycle (élémentaire) avec d'autres arêtes "retenues," la plus grande arête de ce cycle est "rejetée" des arêtes "retenues." Les arêtes "retenues" et non "rejetées" constituent V^*.

Translation

Given: A list of edges of G in an arbitrary order, and for each edge its number and its endpoints. [Instead of giving edge lengths [Ros67] considers the edges numbered in the order of increasing lengths.]

Principle: One reads the list of edges. Each edge read is "retained." When one of these forms an (elementary) cycle with other "retained" edges, the longest edge [i.e., one with the greatest number] of this cycle is "rejected" from among the edges "retained." The edges which are "retained" and not "rejected" form V^* [a solution].

(Rosenstiehl formulated seven algorithms for the MSTP, concentrating on the duality of cycles and cocycles. Among others, he gives an algorithm obtained by a similar modification from Algorithm 4 and an algorithm "dual" to Kruskal's procedure B in the same way A' is due to A. Kruskal wondered about the existence of the latter in [Kru56].) Algorithm 5 appeared earlier [Dij60].

We consider the given branches in an arbitrary order. The first branch is laid down and so is the second one; we continue in this way until a loop is closed. As soon as a loop is closed, we remove the longest branch of the—only!—loop before proceeding to the next branch.

Dijkstra makes the observation that Algorithms 1 and 2 can both be obtained as a special case of Algorithm 5 by choosing the appropriate order on the edges.

A nearly identical algorithm was suggested at about the same time by Kalaba [Kal60].

Select any connecting network with precisely $n - 1$ links. Add another link to this network so that a loop is formed and eliminate from the loop the most costly link. Repeat until no further changes in the connecting network are possible. The resulting network . . . is optimal.

A similar algorithm is also described by Guan in [Gua78], which contains other references to the Chinese literature.

Algorithm 5 depends on the simple proposition that a spanning tree is minimal if and only if any outside edge is at least as long as any edge on the cycle it creates (cf., for example, [ČuDoFi67, theorem 4.1]; similarly, a minimum spanning tree is unique if and only if any outside edge is longer than any other edge on the cycle it creates, cf., [Kot61]). Tarjan has used this idea to develop an algorithm, nearly linear in e, to verify whether a given spanning tree is minimal [Tar79]. Otherwise the algorithms of this section do not seem to allow efficient implementation easily. An $O(e \log v)$ implementation of Algorithm 5 is possible [Tar82] but does not appear to offer any computational advantage over Algorithms 1, 2, or 3.

Conclusions

As far as we can determine, the MSTP was first formulated in 1926 by Otakar Borůvka [Bor26], who was interested in the most economical layout of a power-line network. Borůvka also developed the first solution of the MSTP, using Algorithm 3, and gave the theoretical justification of the greedy method [Bor26], [Bor26a]. Independent discoveries of Algorithm 3 include [Cho38], [FLPSZ51], and [Sol61] (see [BerGho65]).

Before Borůvka, the anthropologist Jan Czekanowski, in his work on various classification schemes [Cze09], [Cze11], [Cze28], fell just short of acknowledging the problem and describing a greedy algorithm for its solution. These steps were taken by Florek et al. [FLPSZ51], who built on the work of Czekanowski. Substantial connections between the MSTP and taxonomy (and clustering) have been independently developed in the West [SokSne63], [Joh67], [GowRos69], [Zah71], [JarSib71], [DudHar73].

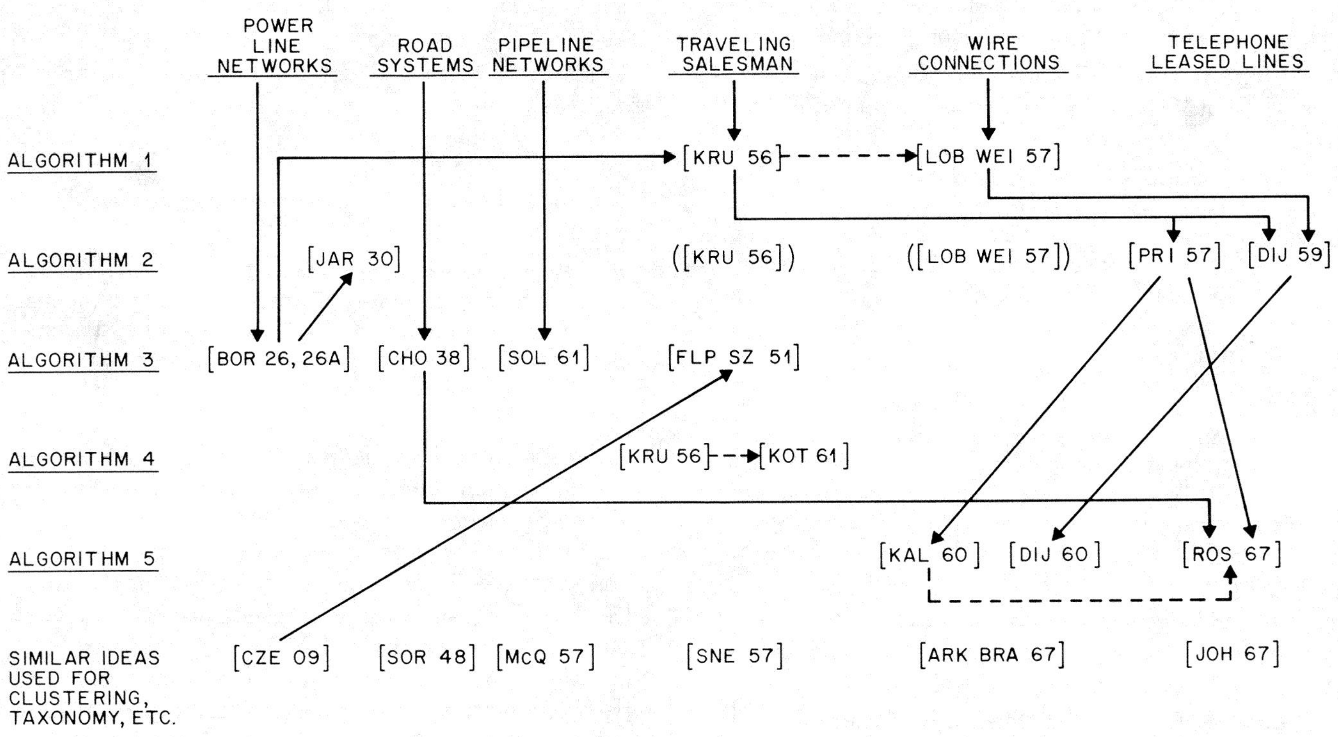


Figure 6

Algorithm 2 appears to have been discovered by Vojtěch Jarník [Jar36]. Jarník viewed his work as a simplification of [Bor26]; in fact, he subtitled his paper, "From a letter to O. Borůvka." Other discoveries of Algorithm 2 appear in [Kru56] (with a suitable interpretation of V) [LobWei57], [Pri57], and [Dij59], the last two also specifying an implementation.

Algorithm 1, perhaps conceptually the simplest greedy algorithm, appears not to have been considered until [Kru56], despite the number of previous papers dealing with the problem. At nearly the same time, it was formulated in [LobWei57].

With the modern implementations, Algorithm 1 is preferred when the edges are presorted or can be sorted by radix sort in linear time. In such a case, Algorithm 1 has a time bound of nearly $O(e)$. In all other cases, Algorithm 2, with the use of heaps, is at least as good as Algorithm 1, and for dense graphs it has a time bound of $O(e)$. (For complete graphs, it is preferable to use the original Prim–Dijkstra implementation without the heaps.) Algorithm 3 has a time bound of $O(e \log\log v)$. There are, in fact, practical algorithms closely related to it with the same worst-case time bounds and with average-case time bounds of $O(e)$. These are preferable in practice to any of the preceding algorithms except when the graph is nearly complete or when the edges are presorted [Tar82]. Algorithm 3 also appears to be well suited for parallel computation.

In Figure 6 we summarize the historical developments and motivations of the MSTP up to about 1965.

After 1965 there was accelerated activity surrounding the MSTP. Much of it was centered around algorithms for MSTP in coordinate spaces [BenFri76], [Yao82], constrained minimum spanning trees [ChaRus72], [ChoKer73], [GloKli75], [Ker74], computer codes for the MSTP [Obr64], [PynWar72], [Roh73a], [Ros69], [Sol60], [Van70], [Whi72], spanning arborescences in directed graphs [ChuLiu65], [Edm67], algorithms for the kth best spanning tree [BurHaf76], [Gab77], [KalbMi81], minimum spanning subgraphs of higher connectivity [Čul60], [Kot61a], [Roy69], updating minimum spanning trees [ChiHou78], [SpiPan75], and such related problems as the Steiner tree problem [DreWag71], [GilPol68], [ChuGra78]. (In the form of the geometric Steiner tree problem, minimum trees had been considered before the development of calculus; cf. [Fer1638], [Tor1646]; cf. also [JarKos34].) We cannot give a complete list of references here; good bibliographies may be found in [Bra75], [Pie75], and [GolMag77].

Acknowledgment

We are grateful to K. Čulík, Ing. L. Hell, J. Nešetřil, R. E. Tarjan, and S. G. Taylor for their assistance in the preparation of this paper.

REFERENCES

[AhHoUl74] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Reading, Mass., Addison-Wesley, 1974.

[ArkBra67] A. G. Arkadev and E. M. Braverman. *Computers and Pattern Recognition*. Washington, D.C., Thompson, 1967.

[Ben80] J. L. Bentley. A parallel algorithm for constructing MST's. *J. Algorithms 1* (1980), 51–59.

[BenFri76] J. L. Bentley and J. H. Friedman. Fast algorithm for constructing minimal spanning trees in coordinate spaces. Tech. Rept. STAN-CS-75-529. Stanford University, January 1976.

[BerGho65] C. Berge and A. Ghouila-Houri. *Programming, Games, and Transportation Networks*. New York, Wiley, 1965. (Algorithm 3, p. 181, is attributed to M. Sollin, Le trace de canalisations, unpublished.)

[BFPRT73] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *J. Comput. Sys. Sci. 7* (1973), 448–461.

[Bor26] O. Borůvka. O jistém problému minimálním. *Práce Mor. Přírodověd. Spol. v Brně (Acta Societ. Scient. Natur. Moravicae) 3* (1926), 37–58.

[Bor26a] O. Borůvka. Příspěvek k řešení otázky ekonomické stavby elektrovodních sítí. *Elektrotechnický obzor 15* (1926), 153–154.

[Bor77] O. Borůvka. Několik vzpomínek na matematický život v Brně. *Pokroky Mat., Fyz., a Astr. 22* (1977), 91–99.

[Bra75] G. H. Bradley. Survey of deterministic networks. *AIEE Transactions 7* (1975), 222–234.

[Bre82] J. J. Brennan. Minimal spanning trees and partial sorting. *Oper. Res. Lett. 1* (1982), 113–116.

[BurHaf76] R. N. Burns and C. E. Haff. A combinatorial ranking problem. *Aeq. Math. 14* (1976), 351–355.

[Cha79] E. J. Chang. Decentralized algorithms in distributed systems. Tech. Rept. CSRG-103, Dept. Comp. Sci., University of Toronto, 1979.

[ChaRus72] K. M. Chandy and R. A. Russel. The design of multipoint linkages in a teleprocessing tree network. *IEEE Trans. Comp. C-21* (1972), 173–182.

[CheTar76] D. Cheriton and R. E. Tarjan. Finding minimum spanning trees. *SIAM J. Comp. 5* (1976), 724–742.

[ChiHou78] F. Chin and D. Houck. Algorithms for updating minimal spanning trees. *J. Comp. Syst. Sci. 16* (1978), 333–344.

[Cho38] G. Choquet. Etude de certains réseaux de routes. *Comptes Rendus Acad. Sci. 206* (1938), 310–313.

[ChoKer73] W. Chou and A. Kershenbaum. A unified algorithm for designing multidrop teleprocessing networks. Data networks—Analysis and design. *Proc. DATACOMM73*, IEEE 3d Data Communications Symp., pp. 148–156.

[Chr70] N. Christofides. The shortest Hamiltonian chain of a graph. *SIAM J. Appl. Math. 19* (1970), 689–696.

[Chr75] N. Christofides. *Graph Theory, An Algorithmic Approach*. New York, Academic Press, 1975.

[Chr76] N. Christofides. Worst case analysis of a new heuristic for the traveling salesman problem. Tech. Rept., Graduate School of Industrial Administration, Carnegie-Mellon University, 1976.

[ChuLiu65] Y. J. Chu and T. H. Liu. On the shortest arborescence of a directed graph. *Scientia Sinica* (Peking) *4* (1965), 1396–1400.

[ChuGra78] F. R. K. Chung and R. L. Graham. Steiner trees for ladders. *Ann. Discrete Math. 2* (1978), 173–200.

[ClaMil] R. Clark and W. F. Miller. Computer based data analysis systems at Argonne. *Methods in Computational Physics 5*.

[Čul60] K. Čulík. K jednomu minimálnímu problému O. Borůvky, *Časopis Pěst. Mat. 85* (1960), 93–94.

[ČuDoFi67] K. Čulík, V. Doležal, and M. Fiedler. Kombinatorická analýza v praxi. *SNTL*, Prague, 1967.

[Cze09] J. Czekanowski. Zur Differentialdiagnose der Neandertalgruppe, Korrespondenz-Blatt der Deutschen Gesellschaft für Anthropologie. *Ethn. u Urg. 40* (1909), 44–47.

[Cze11] J. Czekanowski. Objektive Kriterien in der Ethnologie. *Ebenda 43* (1911), 71–75.

[Cze28] J. Czekanowski. Das Typenfrequenzgesetz. *Anthropologischer Anzeiger 5* (1928), 15–20.

[DRHK70] J. A. Dei Rossi, R. S. Heiser, and N. S. King. A cost analysis of minimum distance TV networking for broadcasting medical information. Memo. RM-6204-NLM, Rand Corporation, Santa Monica, February 1970.

[Dij59] E. W. Dijkstra. A note on two problems in connection with graphs. *Numer. Math. 1* (1959), 269–271.

[Dij60] E. W. Dijkstra. Some theorems on spanning subtrees of a graph. *Indag. Math. 28* (1960), 196–199.

[DreWag71] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks 1* (1971), 195–207.

[DudHar73] R. O. Dude and P. E. Hart. *Pattern Classification and Science Analysis*. New York, Wiley, 1973.

[Edm67] J. Edmonds. Optimum branchings. *J. Res. Nat. Bur. Stand. 71B* (1967), 233–240.

[Edm71] J. Edmonds. Matroids and the greedy algorithm. *Math. Program. 1* (1971), 127–136. (According to [Law76], Edmonds applied the term *greedy algorithm* to Algorithm 1 and announced its extension to arbitrary matroids at least as early as 1967 at the International Symposium on Mathematical Programming in Princeton.)

[EdwCav64] A. W. F. Edwards and L. L. Cavalli-Sforza. Reconstruction of evolutionary trees: Genetic and phylogenetic classification, London, Systematics Association, No. 6, pp. 67–70.

[EsaWil66] L. R. Esau and K. C. Williams. On teleprocessing system design, Pt. II, *IBM Syst. J. 5* (1966), 142–147.

[EvsJas69] V. A. Evstigneev and V. P. Jasakova. A certain algorithm on graphs (in Russian). *Control Systems*, No. 2 (1969), 9–11. (Cf. *Math. Rev. 50*, #12801.)

[Far70] J. S. Farris. Methods for computing Wagner trees. *Syst. Zool. 19* (1970), 83–92.

[Fer1638] P. de Fermat. Maxima et minima, 1638. In P. Tannery and C. Henry, *Oeuvres de Fermat*, Vol. I, Paris, Gauthier-Villars, 1891, pp. 133–174.

[FiKaSh81] J. J. Filliben, K. Kafadar, and D. Shier. Testing for homogeneity of two-dimensional surfaces. Preprint, Clemson University, 1981.

[FLPSZ51] K. Florek, J. Łukaszewicz, J. Perkal, H. Steinhaus, and S. Zubrzycki. Sur la liaison et la division des points d'un ensemble fini. *Colloq. Math. 2* (1951), 282–285; also 319.

[FLPSZ51a] K. Florek, J. Łukaszewicz, J. Perkal, H. Steinhaus, and S. Zubrzycki. Taxonomia Wroclawska. *Przeglad Antropologiczny 17* (1951), 193–211.

[Gab77] H. N. Gabow. Two algorithms for generating weighted trees in order. *SIAM J. Comp. 6* (1977), 139–150.

[GaHuSp79] R. Gallagher, P. Humblet, and P. Spira. A distributed algorithm for minimum weight trees. Tech. Rept. LIOS-P-906A, MIT Lab. for Information and Decision Systems, October 1979.

[Gal68] D. Gale. Optimal assignments in an ordered set: An application of matroid theory. *J. Combinatorial Theory 4* (1968), 176–180.

[GarKum68] R. C. Garg and S. Kumar. The shortest connecting graph through dynamic programming techniques. *Math. Magazine 41* (1968), 170–173.

[Gil65] E. N. Gilbert. Random minimal trees. *SIAM J. Appl. Math. 13* (1965), 376–387.

[GilPol68] E. N. Gilbert and H. O. Pollak. Steiner minimal trees. *SIAM J. Appl. Math. 16* (1968), 1–29.

[GilGom64] P. C. Gilmore and R. E. Gomory. Sequencing a one state-variable machine: A solvable case of the traveling salesman problem. *Oper. Res. 12* (1964), 655–679.

[GloKli75] F. Glover and D. Klingman. Finding minimum spanning trees with a fixed number of links at a node. In *Combinatorial Planning: Methods and Applications* (NATO Advanced Study Inst. Ser., Ser. C), *19* (1975), 191–201; Reidel, Dordrecht.

[GomHu61] R. E. Gomory and T. C. Hu. Multiterminal network flows. *SIAM J. Appl. Math. 9* (1961), 551–571.

[GowRos69] J. C. Gower and G. J. S. Ross. Minimum spanning trees and single linkage cluster analysis. *Appl. Stat. 18* (1969), 54–64.

[GolMag77] B. L. Golden and T. L. Magnanti. Deterministic network optimization: A bibliography. *Networks 7* (1977), 149–183.

[Gua75] Guan Mei-Gu. The method of eliminating cycles for finding minimum spanning trees (in Chinese). *Shuxue de Shijian yu Renshi 4* (1975).

[HanKra74] K. H. Hansen and J. Krarup. Improvements of the Held–Karp algorithm for the symmetric traveling salesman problem. *Math. Program. 7* (1974), 87–96.

[HelKar70] M. Held and R. M. Karp. The traveling salesman problem and minimum spanning trees. *Oper. Res. 18* (1970), 1138–1162.

[HelKar71] M. Held and R. M. Karp. The traveling salesman problem and minimum spanning trees. II. *Math. Program. 1* (1971), 6–25.

[HopUll73] J. E. Hopcroft and J. D. Ullman. Set merging algorithms. *SIAM J. Comp. 2* (1973), 294–303.

[HorSah78] E. Horowitz and S. Sahni. *Fundamentals of Computer Algorithms*. Potomac, Md., Computer Science Press, 1978.

[Hu61] T. C. Hu. The maximum capacity route problems. *Oper. Res. 9* (1961), 898–900.

[JarSib71] N. Jardine and R. Sibson. *Mathematical Taxonomy*, New York, Wiley, 1971.

[Jar30] V. Jarník. O jistém problému minimálním, *Práce Mor. Přírodověd. Spol. v Brně* (*Acta Societ. Scient. Natur. Moravicae*) *6* (1930), 57–63.

[JarKos34] V. Jarník and M. Kössler. O minimálních grafech obsahujících n daných bodu. *Časopis Pěst. Mat. 63* (1934), 223–235.

[Joh67] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika 32* (1967), 241–254.

[Joh75] D. B. Johnson. Priority queues with update and minimum spanning trees. *Inf. Proc. Lett. 4* (1975), 53–57.

[KaIbMi81] N. Katoh, T. Ibaraki, and H. Mine. An algorithm for finding *k* minimum spanning trees. *SIAM J. Comp. 10* (1981), 247–255.

[Kal60] R. Kalaba. On some communication network problems. In *Proc. Symp. Applied Mathematics*, Amer. Math. Soc., 1960, pp. 261–280.

[Kal64] R. Kalaba. Graph theory and automatic control. In E. Beckenbach (ed.), *Applied Combinatorial Mathematics*, New York, Wiley, 1964, Ch. 8.

[KarTar81] R. M. Karp and R. E. Tarjan. Linear expected time algorithms for connectivity problems. *J. Algorithms 1* (1981), 374–393.

[KerVan72] A. Kershenbaum and R. Van Slyke. Computing minimum spanning trees efficiently. *Proc. Annual ACM Conf.*, 1972, pp. 518–527.

[Ker74] A. Kershenbaum. Computing capacitated minimal spanning trees efficiently. *Networks 4* (1974), 299–310.

[KorLov81] B. Korte and L. Lovász. Mathematical structures underlying greedy algorithms. In I. F. Gécseg (ed.), *Fundamentals of Computation Theory* (Lecture Notes in Computer Science 114), New York, Springer-Verlag, 1981, pp. 204–209.

[Kot61] A. Kotzig. Súvisle podgrafy s minimálnou hodnotou v konečnom súvislom grafe. *Časopis Pěst. Mat. 86* (1961), 1–6. (Of historical interest is the review of this article by O. Borůvka, *Math. Rev. 26*, #757.)

[Kot61a] A. Kotzig. On spanning graphs of higher order (in Russian). *Časopis Pěst. Mat. 86* (1961), 288–305.

[Kru56] J. B. Kruskal, On the shortest spanning tree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc. 7* (1956), 48–50.

[Law76] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids*. New York, Holt, Rinehart, and Winston, 1976.

[LeSlBl75] R. C. T. Lee, J. R. Slagle, and H. Blum. A triangulation method for the sequential mapping of points from *n*-space to 2-space. *Proc. Conf. Computer Graphics, Pattern Recognition, and Data Structures*, May 1975, p. 374.

[LobWei57] H. Loberman and A. Weinberger. Formal procedures for connecting terminals with a minimum total wire length. *J. ACM 4*, 4 (October 1957), 428–437.

[McQ57] L. L. McQuitty. Elementary linkage analysis for isolating orthogonal and oblique types and typal relevancies. *Educ. Psychol. Meas. 17* (1957), 207–222.

[McQ60] L. L. McQuitty. Hierarchical syndrome analysis. *Educ. Psychol. Meas. 20* (1960), 293–303.

[Obr64] A. Obruca. Algorithm 1 MINTREE. *Computer Bull. 8* (1964), 67.

[Obr68] A. Obruca. Spanning tree manipulation and the traveling salesman problem. *Computer J. 10* (1968), 374–377.

[OstLin74] R. E. Osteen and P. P. Lin. Picture skeletons based on eccentricities of points of minimum spanning trees. *SIAM J. Comp. 3* (1974), 23–40.

[Pie75] A. R. Pierce. Bibliography on algorithms for shortest path, shortest spanning tree, and related circuit routing problems (1956–1974). *Networks 5* (1975), 129–149.

[ParSam81] D. S. Parker and B. Samadi. Distributed minimal spanning tree algorithms. *Proc. Int. Conf. Performance of Data Communication Systems and Their Applications* (INRIA, Paris), September 1981.

[Pri57] R. C. Prim. Shortest connection networks and some generalizations. *Bell Syst. Tech. J. 36* (1957), 1389–1401.

[PynWar72] C. Pynn and J. H. Warren. Improved algorithms for the construction of minimum spanning trees. *Electron. Lett. 8* (1972), 143–144.

[Rei79] E. M. Reingold. A lower bound for the minimum spanning tree heuristic for weighted matching. Preprint, University of Illinois at Urbana-Champaign. 1979.

[ReNiDe77] E. M. Reingold, J. Nievergelt, and N. Deo. *Combinatorial Algorithms: Theory and Practice*. New York, Prentice-Hall, 1977.

[RogCar71] J. H. Roger and R. G. Carpenter. The cumulative construction of minimum spanning trees. *Appl. Stat. 20* (1971), 192–194.

[RoFiHo72] P. Rosenstiehl, J. R. Fiksel, and A. Holliger. Intelligent graphs: Networks of finite automata capable of solving graph problems. In R. C. Read (ed.), *Graph Theory and Computing*, New York, Academic Press, 1972, pp. 219–265.

[Roh73] F. J. Rohlf. Algorithm 76: Hierarchical clustering using the minimum spanning tree. *Comp. J. 16* (1973), 93–95.

[Roh78] F. J. Rohlf. A probabilistic minimum spanning tree algorithm. *Inf. Process. Lett. 7* (1978), 44–48.

[Ros67] P. Rosenstiehl. L'arbre minimum d'un graphe. In *Theory of Graphs* (Int. Symposium, Rome, 1966), New York, Gordon and Breach, 1967, pp. 357–368.

[Ros69] G. J. S. Ross. Algorithm AS 13: Minimum spanning tree. *Appl. Stat. 18* (1969), 103–104.

[Roy69] B. Roy. Graphe partiel s-connexe extremum. *Rev. Roumaine Math. Pures Appl. 14* (1969), 1355–1368.

[SaBoRu73] R. G. Saltman, G. R. Bolotsky, and Z. G. Ruthberg. Heuristic cost optimization of the federal Telpak network. Tech. Note 787, National Bureau of Standards, Washington, D.C., June 1973.

[Sib73] R. Sibson. SLINK: An optimally efficient algorithm for the single link cluster method. *Comp. J. 16* (1973), 30–34.

[ShaElB70] R. L. Sharma and M. T. El-Bardai. Suboptimal communications network synthesis. *Proc. Int. Conf. Communications*, June 1970, pp. 19.11–19.16.

[SmiLie79] J. MacGregor Smith and J. S. Liebman. Steiner trees. Steiner circuits and the interference problem in building design. *Eng. Optimization 4* (1979), 15–36.

[Sne57] P. H. A. Sneath. The application of computers to taxonomy. *J. Gen. Microbiol. 17* (1957), 201–206.

[SokSne63] R. R. Sokal and P. H. A. Sneath. *Principles of Numerical Taxonomy*. San Francisco, Freeman, 1963.

[Sol61] G. Sollin. Problème de l'arbre minimum (unpublished manuscript prepared for C. Berge's Paris seminar, February 8, 1961).

[Sol62] G. Sollin. Problèmes de recherche operationelle. Report C.41, Meeting of Technical Directors, S.E.G. Paris, 1962; in particular, see Chapter III, Le Trace des Canalizations (the design of pipelines), pp. 15–23.

[Sol60] E. W. Solomon. A comprehensive program for network problems. *Comp. J. 3* (1960), 89–97.

[Sor48] T. Sorensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. *Biologiske Skrifter 5* (1948), 3–33.

[Spi77] P. Spira. Communication complexity of distributed minimum spanning tree algorithms. *Proc. 2nd Berkeley Conf. Distributed Data Management and Computer Networks*, June 1977.

[SpiPan75] P. M. Spira and A. Pan. On finding and updating spanning trees and shortest paths. *SIAM J. Comp. 4* (1975), 375–380.

[Sti67] F. H. Stillinger. Physical clusters, surface tension, and critical phenomena. *J. Chem. Phys. 47* (1967), 2513–2533.

[Tar75] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM 22* (1975), 215–225.

[Tar79] R. E. Tarjan. Applications of path compression on balanced trees. *J. ACM 26* (1979), 690–715.

[Tar83] R. E. Tarjan. Data structures and network algorithms, Ch. 6. CBMS Regional Conf., SIAM, Philadelphia, 1983.

[Tor1646] E. Torricelli. De maximis et minimis, 1646. In G. Loria and G. Vassura, *Opere di Evangelista Torricelli*, Vol. I, *Geometria*, pt. II, pp. 79–97.

[Van70] C. J. Van Rijsbergen. Algorithm 52: A fast hierarchical clustering algorithm. *Comp. J. 13* (1970), 324–326.

[VanFra71] R. Van Slyke and H. Frank. Reliability of computer communication networks. *Proc. Fifth Conf. on Simulation*, AFIPS Press, December 1971.

[VanFra72] R. Van Slyke and H. Frank. Network reliability analysis. Pt. I. *Networks 1* (1972), 279–290.

[Wel68] D. J. A. Welsh. Kruskal's theorem for matroids. *Proc. Cambridge Phil. Soc. 64* (1968), 3–4.

[Whi72] V. K. M. Whitney. Algorithm 422: Minimal spanning tree. *Comm. ACM 15* (1972), 273.

[Yao75] A. C. Yao. An $O(e \log \log v)$ algorithm for finding minimum spanning trees. *Inf. Process. Lett. 4* (1975), 21–23.

[Yao82] A. C.-C. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM J. Comp. 11* (1982), 721–736.

[Zah71] C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. Comp. C20* (1971), 68–86.

[Zah74] C. T. Zahn. An algorithm for noisy template matching. *Proc. IFIP Congress 1974*, Vol. 4, New York, American Elsevier, 1974, pp. 698–701.

[ZGLL74] C. T. Zahn, A. Gunther, B. Levratt, and H. Lipps. Using the minimum spanning tree to recognize dotted and dashed curves. Int. Symposium Switzerland, September 1973, North Holland, 1974, pp. 381–387.

Added in Proof

M. L. Fredman and R. E. Tarjan [FreTar84] have very recently discovered an $O(e \log^* v)$ MST algorithm (where $\log^* v$ is defined to be the least integer m such that the m-fold iterated logarithm $\log^{(m)} v$ is less than 1). It is based on an ingenious use of a new data structure they introduce, called a Fibonacci heap. This has subsequently been (slightly!) improved to $O(e \log \log^* v)$ by H. N. Gabow, Z. Galil, and T. H. Spencer [GaGaSp84].

[FreTar84] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses to improve network optimization algorithms. *Proc. 25th Symp. on Foundations of Comp. Sci.* IEEE Comp. Sci. Press (1984), 338–346.

[GaGaSp84] H. N. Gabow, Z. Galil, and T. H. Spencer. Efficient implementation of graph algorithms using contraction. *Proc. 25th Symp. on Foundations of Comp. Sci.* IEEE Comp. Sci. Press (1984), 347–357.