Bulletin of the *Transilvania* University of Braşov • Vol 13(62), No. 1 - 2020
Series III: Mathematics, Informatics, Physics, 343-346
https://doi.org/10.31926/but.mif.2020.13.62.1.25

INCREMENTAL MINIMUM SPANNING TREE ALGORITHMS

Laura CIUPALĂ[1], Adrian DEACONU[2] and Delia SPRIDON[3]

Abstract

There are several types of problems that can be modeled and solved as minimum spanning tree problems. Sometimes the weighted graph in which we need to determine a minimum spanning tree differs from another weighted graph, in which a minimum spanning tree is already established, only by an edge weight (which is reduced or augmented by a units). We will describe algorithms that determine minimum spanning trees in the new weighted graphs starting from a minimum spanning tree in the original weighted graph.

2000 *Mathematics Subject Classification:* 90B10, 90C90.
Key words: graph algorithms, minimum spanning tree, incremental algorithms.

1 Introduction

The literature on graph algorithms is extensive([1], [2], [3]) and one of the reasons is the widespread and diverse applications of graphs. From the late 1920s through the 1960s, researchers designed independently several minimum spanning tree algorithms. For instance, the broadly and frequently used greedy algorithm, that determines a minimum spanning tree by growing it starting from a single node, is known as Prim's algorithm. It is accurate that Robert Prim developed it in 1957, but this algorithm was earlier described by Vojtech Jarnik, back in 1930, and also later, in 1959, by Edsger Dijkstra. This is not an isolated fact, this happened also to another minimum spanning tree algorithm that was independently developed by Otakar Boruvka in 1926 and by George Sollin in 1965.

[1]Department of Mathematics and Computer Science, Faculty of Mathematics and Computer Science, *Transilvania* University of Braşov, Romania, e-mail: laura_ciupala@yahoo.com
[2]Department of Mathematics and Computer Science, Faculty of Mathematics and Computer Science, *Transilvania* University of Braşov, Romania, e-mail: a.deaconu@unitbv.ro
[3]Faculty of Mathematics and Computer Science, *Transilvania* University of Braşov, Romania, e-mail: delia_spridon@yahoo.com

The minimum spanning tree problem is studied because it arises both when modeling and solving a real world problem and, also, as a sub-problem in more complex optimization problems.

Let $G = (N, A)$ be an undirected graph, defined by a set N of n nodes and a set A of m edges. To each edge $[x, y] \in A$ a weight $w[x, y]$, that can represent length, cost, time, penalty etc, is associated. Consequently, $G = (N, A, w)$ is a weighted undirected graph.

Let $G = (N, A, w)$ be a connected weighted undirected graph. This implies that there is at least one spanning graph of G, denoted by $G' = (N, A')$, with $A' \subset A$, such that $G' = (N, A')$ is a tree. We refer to $G' = (N, A')$ as a *spanning tree* of $G = (N, A, w)$. The weight of G' is defined as $w(G') = \sum_{[x,y]\in A'} w[x, y]$. The edges in A' are named tree edges, while the edges in $A \backslash A'$ are named non-tree edges.

In the connected weighted undirected graph $G = (N, A, w)$, the minimum spanning tree problem is to determine a spanning tree $G' = (N, A')$, whose weight is minimized. Such a spanning tree is called a *minimum spanning tree* of G.

The known minimum spanning tree algorithms are based on the following optimality conditions: cut optimality conditions and path optimality conditions.

Let $G = (N, A, w)$ be a connected weighted undirected graph and let $G' = (N, A')$ be a spanning tree of G. By removing an arbitrary tree edge $[x, y]$ from the spanning tree $G' = (N, A')$ a disconnected graph, $G" = (N, A' \backslash \{[x, y]\})$, is obtained. Let X and Y be the node sets that induce the two connected components of $G"$. The edges of graph G that have one endpoint in X and the other endpoint in Y form a cut, named the *cut formed by deleting the tree edge* $[x, y]$.

Theorem 1. *[1](Cut Optimality Conditions) A spanning tree $G' = (N, A')$ of the connected weighted undirected graph $G = (N, A, w)$ is a minimum spanning tree if and only if it satisfies the following cut optimality conditions:*

For every tree edge $[x, y]$, $w[x, y] \leq w[u, z]$, for every edge $[u, z]$ contained in the cut formed by deleting the tree edge $[x, y]$.

Let $G = (N, A, w)$ be a connected weighted undirected graph and let $G' = (N, A')$ be a spanning tree of G. For every non-tree edge $[x, y]$ there is a unique path $P_{x,y}$ between x and y in the spanning tree $G' = (N, A')$.

Theorem 2. *[1](Path Optimality Conditions) A spanning tree $G' = (N, A')$ of the connected weighted undirected graph $G = (N, A, w)$ is a minimum spanning tree if and only if it satisfies the following path optimality conditions:*

For every non-tree edge $[x, y]$, $w[x, y] \geq w[u, z]$, for every edge $[u, z]$ contained in the unique path $P_{x,y}$ between x and y in the spanning tree $G' = (N, A')$.

2 Incremental algorithms

Data may vary in time in the real life problems that can be modeled and solved as minimum spanning tree problems. For instance, a usual variation in real

world problems might imply an increase or a decrease in the weight of an edge in the corresponding weighted graph. Suppose that a minimum spanning tree has been already determined in the original weighted graph. In this case, instead of applying to the modified graph a known minimum spanning tree algorithm, starting from scratch, one can use the already established minimum spanning tree in the original graph as a starting point. We focus on the second approach because it is more efficient than the first one.

Let $G = (N, A, w)$ be the original connected weighted undirected graph in which a minimum spanning tree $G' = (N, A')$ of weight $w(G')$ is already established. Let $\hat{G} = (N, A, \hat{w})$ be the connected weighted undirected graph that differs from G only by the weight of a given edge $[l, k]$, which could be larger or smaller than the initial weight of $[l, k]$ by a given positive amount a. So, $\hat{w}[x, y] = w[x, y]$, for each $[x, y] \in A \backslash \{[k, l]\}$ and $\hat{w}[k, l] = w[k, l] \pm a$.

There are four cases that might occur:

Case 1: If the weight of a non-tree edge $[l, k]$ increases by a units, then the minimum spanning tree $G' = (N, A')$ of G is, obviously, a minimum spanning tree of the modified graph and it has the same weight as in the original graph.

Case 2: If the weight of a tree edge $[l, k]$ increases by a units, then it is possible that the original graph G and the modified graph $\hat{G}$ have different minimum spanning trees. In order to determine a minimum spanning tree of $\hat{G}$, first, we delete the edge $[l, k]$ from the tree $G' = (N, A')$. This way a cut is obtained. Then a minimum weight edge, $[x, y]$, of this cut is determined. The spanning tree $G" = (N, A")$, where $A" = A' \backslash \{[k, l]\} \cup \{[x, y]\}$ is, according to the cut optimality conditions, a minimum spanning tree of $\hat{G}$. The algorithm that determines a minimum spanning tree in the modified weighted graph $\hat{G}$ starting with a minimum spanning tree $G' = (N, A')$ of the original weighted graph G is the following:

UnderestimatedTreeEdge Algorithm;
Begin
determine the cut obtained by deleting the tree edge $[l, k]$;
determine a minimum weight edge $[x, y]$ in this cut;
$A" = A' \backslash \{[k, l]\} \cup \{[x, y]\}$;
end.

Theorem 3. *The UnderestimatedTreeEdge algorithm determines a minimum spanning tree $G" = (N, A")$ in the connected weighted graph $\hat{G}$ with an underestimated tree edge weight starting with a minimum spanning tree $G' = (N, A')$ in the original network G in $O(m)$ time.*

Proof. The most time consuming operation in the algorithm is to determine the cut obtained by deleting the tree edge $[l, k]$, which in a graph with m edges can be done in $O(m)$ time. □

Remark 1. *Since $w[k, l] \leq \hat{w}[x, y]$, it follows that $\hat{w}(G") \geq w(G')$.*

Case 3: If the weight of a non-tree edge $[l, k]$ decreases by a units, then the minimum spanning tree of the modified graph $\hat{G}$ can be different than from minimum spanning tree $G' = (N, A')$ of G. In order to determine a minimum spanning

tree of $\hat{G}$, first, we determine a maximum weight edge $[x, y]$ in the unique path $P_{k,l}$ between nodes k and l in the spanning tree $G' = (N, A')$.

If $w[x, y] \leq w[k, l] - a$ then $G' = (N, A')$ is also a minimum spanning tree in $\hat{G}$; otherwise we form the spanning tree $G" = (N, A")$, where $A" = A'\backslash\{[x, y]\} \cup \{[k, l]\}$. The path optimality conditions imply that the spanning tree $G"$ will be a minimum spanning tree of $\hat{G}$. Moreover, in this case, the weight of $G"$ will be smaller than the weight of G' by $w[x, y] - w[k, l] + a$, which is a strictly positive value. So, in this case we can use the following algorithm to determine a minimum spanning tree in the modified graph:

OverestimatedNonTreeEdge Algorithm;
Begin
determine the unique path $P_{k,l}$ between nodes k and l in the spanning tree $G' = (N, A')$;
determine a maximum weight edge $[x, y]$ in the unique path $P_{k,l}$;
if $w[x, y] \leq w[k, l] - a$ **then** G' is a minimum spanning tree in $\hat{G}$;
else begin
$A" = A'\backslash\{[x, y]\} \cup \{[k, l]\}$;
$G" = (N, A")$ is a minimum spanning tree in $\hat{G}$;
end;
end.

Theorem 4. *The OverestimatedNonTreeEdge algorithm determines a minimum spanning tree $G" = (N, A")$ in connected weighted graph $\hat{G}$ with an overestimated non-tree edge weight starting with a minimum spanning tree $G' = (N, A')$ in the original network G in $O(n)$ time.*

Proof. The most time consuming operation in the algorithm is to determine the unique path $P_{k,l}$ between the nodes k and l in the spanning tree $G' = (N, A')$, which, in a tree with n nodes, can be done in $O(n)$ time. □

Case 4: If the weight of a tree edge $[l, k]$ decreases by a units, both the original graph G and the modified graph $\hat{G}$ have the same minimum spanning tree $G' = (N, A')$. Obviously, the weight of the spanning tree in the modified graph is smaller by a units than the weight of the same minimum spanning tree in the original graph.

References

[1] Ahuja, R., Magnanti, T., Orlin, J., *Network Flow. Theory, Algorithms and Applications*, Prentice Hall, New Jersey, 1993.

[2] Bang-Jensen, J., Gutin, G., *Digraphs, Theory, Algorithms and Applications*, Springer-Verlag, London, 2001.

[3] Cormen, T., Leiserson, C., Rivest, R., Stein, C., *Introduction to Algorithms*, MIT Press, 2009.