アルゴリズム 12－7
(1989. 11. 20)

# Minimum Delay Networks

Jan-ming Ho
Academia Sinica
R. O. C.

D. T. Lee
Northwestern University
U. S. A.

## Abstract

In this paper, two network optimization problems are studied. In a *minimum delay Steiner tree problem*, a graph $G = (V, E)$, a source set $S$ and a destination set $D$ are given, where $S, D \subseteq V$. The question is to find a Steiner tree containing vertices in both $D$ and $S$ such that the maximum distance between any pair of vertices $s \in S$ and $d \in D$ is minimized. This problem is shown to be $O(|E|m \log m + |V|(|E| + |V| \log |V|))$ time solvable, where $m = |D \cup S|$. In a *network extension problem*, we want to find a minimally connected subgraph $\hat{G}$ of graph $G = (V, E)$ containing a given connected subgraph $G'$ of $G$ such that the diameter of $\hat{G}$ is minimized. An $O(|V||E| \log |V|)$ time algorithm is presented.

## 1 Introduction

In this paper, we propose two problems in which the length of signal paths is used as the objective of optimization. In a communication network, a vertex of a graph is usually used to represent a station in the network and edges represent connections between the stations. An edge can be associated with weight to denote signal delay between two stations. Given an edge $e = (i, j)$ of a tree $T = (V, E)$, $T_i = (V_i, E_i)$ and $T_j = (V_j, E_j)$ denotes the trees resulted from the deletion of $e$ from $T$, where $i \in V_i$ and $j \in V_j$ respectively. The length of a path, defined as the sum of the weight of each edge of the path, is used to represent the total signal delay along the path. The distance between two vertices is defined as the minimum length of the paths between them, and thus is a lower bound of the communication delay. The diameter of a graph is defined as the maximum distance for all pairs of vertices. The distance between two vertices is also called the signal delay between the vertices. The distance between vertices $x$ and $y$ with respect to a graph $G$ is denoted as $dist_G(x, y)$. We have previously studied the problem of minimum diameter spanning tree [4], in which the goal is to generate a spanning tree such that the diameter of the tree is minimized. If the stations in a communication network are regarded as uniform in the sense that each station is allowed to broadcast signals to all the others, then a minimum diameter spanning tree minimizes the maximum signal delay. But the role of a station in a network can be quite different. For example, some stations may be assigned as broadcasting centers while some others are authorized to accept the information. In this case, we formulate the *minimum delay Steiner tree problem* in which broadcasting centers are called the *source vertices* and authorized stations as the *sink vertices*. On the other hand, we also consider the problem of upgrading a network based on the current installations with the objective of minimizing the maximum signal delay, which is called the *network extension problem*.

We now formally introduce the two problems as follows:

**Problem 1 (Minimum delay Steiner tree problem)** *Given a positive weighted graph $G = (V, E)$, a set $S \subseteq V$ of source vertices and a set $D \subseteq V$ of destination vertices, find a Steiner tree*

*$T$ containing all the vertices in $S \cup D$ such that the maximum signal delay of $T$ is minimized, where the maximum signal delay refers to the maximum among the distance between each pair of source and destination vertices.*

**Problem 2 (network extension problem)** *Given a positive weighted graph $G = (V, E)$ and a connected subgraph $G' = (V', E')$ of $G$, find a minimally connected spanning subgraph $G^*$ of $G$ containing $G'$ such that the maximum signal delay on $G^*$ is minimized, where the maximum signal delay refers to the maximum among the distance between each pair of vertices of $G^*$, i.e., the diameter of $G^*$.*

The problem of minimum delay Steiner tree is studied in section 2 and the problem of network extension is studied in section 3.

## 2 Minimum Delay Steiner Trees

In this section, the *minimum delay Steiner tree problem* is studied. We'll show that the domain of searching for a *minimum delay Steiner tree* can be reduced to only a subclass of Steiner trees, called the *canonical Steiner trees*. The definitions of *virtual graph* and *virtual tree* and that of *canonical Steiner tree* are first given as follows.

**Definition 1** *Let $e = (i, j)$ be an edge of a positive weighted graph $G = (V, E)$ and $0 \le \theta \le w(e)$, where $w(e) > 0$ denotes the weight of the edge $e$. The graph $(V \cup \{v\}, E \cup \{e_i, e_j\} - \{e\})$ is said to be a virtual graph of $G$ denoted as $G^{(v)}(e, \theta)$, where $v$ is called the virtual vertex and $e_i = (v, i)$ with weight $w(e_i) = \theta$ and $e_j = (v, j)$ with weight $w(e_j) = w(e) - \theta$ are called the virtual edges.*

**Definition 2** *Let $T = (V_T, E_T)$ be a subtree of $G = (V, E)$ containing every vertex in $S \cup D$ and for every leaf $v \in V_T$, $v \in S \cup D$. $T$ is said to be a canonical Steiner tree of $G$ if there exists $e \in E_T$ and $0 \le \theta \le w(e)$, such that $\forall$ vertex $u$ of the virtual tree $T^{(v)}(e, \theta)$, the path from $u$ to the virtual vertex $v$ of $T^{(v)}(e, \theta)$ is a shortest path in $G^{(v)}(e, \theta) A$ canonical Steiner tree $T$ is also denoted as $T^{(c)}(e, \theta)$.*

**Lemma 1** *There is a canonical Steiner tree which is a* minimum delay Steiner tree *of the given graph $G = (V, E)$ with source set $S$ and destination set $D$.*

*Proof:* Let $T^*$ be a minimum delay Steiner tree and $s$ and $t$ be the pair of source and destination vertices defining the maximal signal delay on $T^*$. Let $T'$ be the subtree created by recursively deleting every leaf Steiner vertex from $T$. By identifying the path $P$ between the vertices $s$ and $t$, the weighted center edge $e$ of $P$ is said to be the *c-edge* of $T'$ and thus that of $T^*$. Now, let's define $\theta = (w(e) - dist_{T^*}(d_i, i) + dist_{T^*}(d_j, j))/2$, where vertex $d_i$ in $T_i^*$ and vertex $d_j$ in $T_j^*$ are the extremes of a diameter of $T'$ passing through $e$. Since $e$ is a *c-edge* of $T'$, $0 \le \theta \le w(e)$ is satisfied. Note that $s$ and $t$ must lie on different sides of $e$ unless $\theta = 0$ or $w(e)$. We are going to show that the maximal signal delay of $T = T^{(c)}(e, \theta)$ is no greater than that of $T^*$, the *minimum delay Steiner tree.*

Let $s'$ and $t'$ denote a pair of source and destination vertices defining the maximal signal delay of $T$. Consider the first case in which $s'$ and $t'$ lie on different sides of $e$. Without loss of generality, assume $s'$ be in $T_i$ and $t'$ be in $T_j$. Now, we have

$$\begin{aligned}
&\textit{maximal signal delay of } T \\
&\quad \le dist(s', i) + w(e) + dist(t', j) \\
&\quad = dist(s', i) + w(e_i) + w(e_j) + dist(t', j) \\
&\quad = dist_{G^{(v)}}(s', v) + dist_{G^{(v)}}(t', v) \\
&\quad \le dist_{T^{*(v)}}(s', v) + dist_{T^{*(v)}}(t', v) \\
&\quad \le dist_{T^{*(v)}}(s, v) + dist_{T^{*(v)}}(t, v) \\
&\quad = \textit{maximal signal delay of } T.
\end{aligned}$$

Since $T^*$ is optimum, only equality exists. This completes our proof. ◇

The proof of lemma 1 also indicates that if $T^{(c)}(e, \theta)$ is the minimum delay Steiner tree, then the subpath of $T^{(c)}(e, \theta)$ between a pair of source $s$ and destination $t$ defining the maximum signal delay of $T^{(c)}(e, \theta)$ must passes through at least one of the vertices $i$ and $j$, where $e = (i, j)$. As a consequence, we don't have to generate the tree $T^{(c)}(e, \theta)$ and calculate its maximal signal path in searching for the optimal combination of $e$ and $\theta$. In stead, the criterion

$$z(e, \theta) = \max_{s \in S} dist_{G^v}(s, v) + \max_{t \in D} dist_{G^v}(t, v)$$

is associated with each selection of $e$ and $\theta$, where $v$ denotes the virtual vertex of the virtual graph $G^v(e,\theta)$. Note that this measure over-estimates the maximal signal delay of a non-optimal $T^{(c)}(e,\theta)$, but it equals the maximal signal delay of $T^{(c)}(e,\theta)$ if $T^{(c)}(e,\theta)$ is optimal. That is, the minimizer of $z(e,\theta)$ also minimizes the maximal signal delay of $T^{(c)}(e,\theta)$.

Note that for each edge $e=(i,j)\in E$, $\theta$ assumes any real value in the interval $[0,w(e)]$. Thus, it can be viewed as a continuous optimization problem. But, we are going to show that this problem is discrete in nature. Note that, two values $\theta_1$ and $\theta_2$, $0\le\theta_1,\theta_2\le w(e)$, may give the same tree $T^{(c)}(e,\theta_1)=T^{(c)}(e,\theta_2)$. In this case, $\theta_1$ and $\theta_2$ are said to be related by a relation $\mathcal{R}_e$. The relation $\mathcal{R}_e$ is an equivalence relation and it partitions the interval $[0,w(e)]$ into several equivalent classes. Note that $|dist(u,i)-dist(u,j)|\le w(e)$ for each $u\in V$. For each vertex $u\in V$, the critical value is defined as $\theta_e(u)=((w(e)-dist(u,i)+dist(u,j))/2$, $0\le\theta_e(u)\le w(e)$. A critical value $\theta_e(u)$ divides the interval $[0,w(e)]$ into two subintervals $I_{e,i}(u)=[0,\theta_e(u))$ and $I_{e,j}(u)=(\theta_e(u),w(e)]$. For every $\theta\in I_{e,i}(u)$, $u$ belongs to the subtree $T_i^c$, where $T^c$ denotes the tree $T^{(c)}(e,\theta)$. For every $\theta\in I_{e,j}(u)$, $u$ belongs to the subtree $T_j^c$. In the former case, $dist_{T^c}(u,i)=dist(u,i)$, and in the latter case, $dist_{T^c}(u,j)=dist(u,j)$. The partition of the interval $[0,w(e)]$ by $\mathcal{R}_e$ is determined by the critical values $\theta_e(u), u\in D\cup S$. Furthermore, it is not difficult to show that $z(e,\theta)$ will take its local minimum only at the critical values.

To solve the problem, *algorithm minimum_delay_Steiner_tree* first constructs the distance matrix. Then, for each edge $e=(i,j)$, the function *Calculate_delay_Steiner_tree*$(e)$ computes a local minimum of $z(e,\theta)$ among the critical values $\theta_e(u)$, $u\in D\cup S$. The global minimum $z(\hat{e},\hat{\theta})$ is thus determined. The Canonical Steiner tree $T^{(c)}(\hat{e},\hat{\theta})$ is then reported as the optimum solution.

Detailed implementation of the algorithm is presented in the following:

```
begin
    Calculate the distance matrix;
    $\hat{M}$ = inf;
    for each edge $e=(i,j)\in E$ do
    begin
        $(M,\theta)$ = Calculate_local_minimum(e);
        if $M<\hat{M}$ then
        begin
            $\hat{M}:=M$;
            $\hat{e}:=e$;        $\hat{\theta}:=\theta$;
        end;
    end;
    Report the tree $T^{(c)}(\hat{e},\hat{\theta})$;
end.
```

Calculation of the distance matrix can be done either by Floyd's algorithm [2] in $O(|V|^3)$ time or the shortest path algorithm due to Fredman and Tarjan [3] in $O(|E||V|+|V|^2\log|V|)$ time. The function *Calculate_local_minimum(e)* can be implemented by an $O(m\log m)$-time algorithm as described in the following, where $m=|S\cup D|$.

The function *Calculate_local_minimum(e)* returns two values, $M$, the local minimal $z(e,\theta)$, and $\theta$, the local minimizer of $z(e,\theta)$. The function first sorts the vertices into non-decreasing order using the critical values $\theta_e(u)$, $u\in S\cup D$, as the keys. Denote the sequence of vertices as $v_1,v_2,\ldots,v_n$, where $n=|V|$, and the sequence of critical values without repetition as $\theta_1,\theta_2,\ldots,\theta_a$, where $n=|V|$ and $1\le a\le n$. The algorithm then scans through the sequence for the local minimizer. A vertex $u$ is said to be *unscanned* if $\theta_e(u)$ is greater than the current critical value and is said to be *scanned*, if otherwise. At the $l$th step of the scanning, the farthest source and the farthest destination from the current virtual vertex $v$ are identified so that $z(e,\theta_l)$ can be calculated based on the formula introduced above. To determine the farthest source and destination from $v$, the algorithm maintains for both source and destination vertices the farthest scanned vertex from $v$, $SV_S$ and $SV_D$ respectively, and a sorted list of unscanned vertices, denoted as $UL_S$ and $UL_D$. Note that the unscanned vertices are sorted by their distances from $i$ or equivalently from $v$. These data structures can be updated as follows: let $l$ be the current iteration, i.e., $\theta=\theta_l$. Let $u_l$ be a vertex satisfying $\theta(e,u)=\theta_l$. Then $UL_S$ is updated by the deletion of $u_l$ and $SV_S$ is changed to $u_l$ if $u_l$ is farther from $v$ than is $SV_S$, or remains unchanged otherwise. Obviously, the total updating steps take

$O(m)$ time. It is not difficult to see that the objective function $z(e,\theta)$ can then be performed in $O(m)$ time. Thus, *Calculate_local_minimum(e)* can be implemented in $O(m \log m)$ time. Complexity of this algorithm is $O(|E|m \log m + |V||E| + |V|^2 \log |V|)$.

**Theorem 1** *The minimum delay Steiner tree problem can be solved in time $O(|E|m \log m + |V||E| + |V|^2 \log |V|)$, where $m = |S \cup D|$.*

## 3 Network Extension Problem

In this section, an algorithm is presented to solve the *network extension problem.* In this problem, we are given a connected graph $G = (V, E)$ and a connected subgraph $G' = (V', E')$ of $G$. A subgraph $H$ of $G$ is said to be a *feasible extended network* of $G'$ with respect to $G$ if $H$ is a minimally connected supergraph of $G'$. The goal is to find a feasible extended network $H$ such that the diameter of $H$ is minimized, where the *diameter* of a graph is defined as the maximum among the shortest distances between each pair of vertices. A feasible extended network with minimum diameter is also called an *optimal extended network* of $G'$ with respect to $G$, or an optimal extended network of $G'$ for short.

In the following, we are going to give the definitions of *feasible paths* or *F-paths* for short.

**Definition 3** *A simple path $P(v_1, v_2, \ldots, v_k)$ in $G = (V, E)$ is said to be an F-path with respect to a subgraph $G' = (V', E')$ of $G$ if and only if $\exists i, j, 1 \le i \le j \le k$ such that $v_i, v_{i+1}, \ldots, v_j \in V'$ and $v_1, v_2, \ldots, v_{i-1}, v_{j+1}, v_{j+2}, \ldots, v_k \in V - V'$.*

In other words, the subgraph $G'$ is treated as a supernode in defining an *F-path.* Also note that, every simple path of a feasible extended network $H$ is an *F-path* because $H$ is minimally connected by definition.

**Lemma 2** *Let $H$ be a feasible extended network of $G'$ with respect to $G$, then every path on $H$ is an F-path.*

**Definition 4** *Given two vertices $v_1$ and $v_2 \in V$ of a graph $G = (V, E)$, the F-distance between $v_1$ and $v_2$ is defined as the total length of the shortest F-path between the two vertices denoted as $dist_G^F(v_1, v_2)$.*

We'll show later that the *F-distance* can be calculated using a modified version of Dijkstra's algorithm in $O(|E| + |V| \log |V|)$ time, where $|S|$ denotes the cardinality of a set $S$. The modified Dijkstra's algorithm also generates a spanning tree $T$ of the graph $G$. For each vertex $v \in V$, the path from $v$ to $s$ on the computed spanning tree $T$ is a shortest *F-path.* As a result, we'll call the spanning trees generated by the modified Dijkstra's algorithm the *F-trees.*

**Definition 5** *A spanning tree $T$ of a connected graph $G = (V, E)$ is called an F-tree rooted at vertex $s \in V$ if for each vertex $v \in V$, the path from $v$ to $s$ on the tree $T$ is a shortest F-path.*

Given an edge $e = (i, j) \in E$ and a Let $T$ be a tree containing edge $e = (i, j)$ and $\theta$ be a value satisfying $0 \le \theta \le w(e)$. value $0 \le \theta \le w(e)$, let $T$ be a tree containing the edge $e$, The associated virtual tree $T^v(e, \theta)$ of $T$ and the associated virtual graph $G^v(e, \theta)$ of $G$ are defined by replcing the edge $e$ with the edges $e_i = (i, v)$ and $e_j = (j, v)$, where $v \notin V$ is called a virtual vertex.

**Definition 6** *A spanning tree $T = (V, E_T)$ of graph $G = (V, E)$ is said to be an A-tree of $G$ if there exists an edge $e \in E_T$ and $0 \le \theta \le w(e)$ such that $T^v(e, \theta)$ is the F-tree of $G^v(e, \theta)$ rooted at $v$. In this case, $T$ is also denoted as A-tree$(e, \theta)$. The network constructed by the union of the edges in $G'$ and A-tree$(e, \theta)$ is also denoted as A-network$(e, \theta)$.*

Let's now identify the *c-edge* $e = (i, j)$ of the *optimal extended network* $G^*$ of $G'$ by first identifying a diameter of $G'$, then the weighted center edge of the diameter is the desired *c-edge.* The value $\theta$ is defined as

$$\theta = (w(e) + dist_{G^*}^F(j, v_j) - dist_{G^*}^F(i, v_i))/2,$$

where $v_i$ and $v_j$ are the extreme vertices on the selected diameter. It is not difficult to show that the diameter of the *A-network$(e, \theta)$* is no greater than that of $G^*$. In other words, the optimal *A-network* which gives minimum diameter is also the optimal extended network of $G'$.

By a similar argument as used in the previous section, only $O(n)$ $\theta$ values needs to be examined to find the local optimizer $(e, \theta)$ with respect to an edge $e$ once the distance matrix $dist_F(x, y), \forall x, y \in V$ is given.

The algorithm calculates $M$, which is defined as the minimum of the sum of largest and second largest $dist_G^F(x, v), \forall x \in V$, where $v$ is the virtual vertex of the *A-network*$(e, \theta)$. The procedure *Calculate_local_minimum(e)* is similar to that introduced in the previous section and is also an $O(n \log n)$ time algorithm, where $n$ denotes the number of vertices in $V$.

Finally, we are going to study the complexity of the calculation of the distance matrix $dist_G^F(x, y), \forall x, y \in V$. Before proceeding, we make the assumption that the subgraph $G'$ is given by specifying the adjacency matrix. Which allows $O(1)$ time operation in identifying if an edge $(x, y)$, given by the two adjacent vertices, is an edge in $G'$. In calculating the distance matrix $dist_G^F(x, y)$, we follow Fredman and Tarjan's [3] Fibonacci heap implementation of Dijkstra's shortest path algorithm [1] with the following modifications. Let $P$ denote an *F-path* from $s$ to $u$, and $v$ be the vertex on $P$ preceding $u$. $P$ is said to be of type 1 if every vertex on $P$ is not in the subgraph $G'$. $P$ is said to be of type 2 if the path from $s$ to $v$ is of type 1 or 2 and the edge $(v, u) \in E'$. $P$ is said to be of type 3 if the path from $s$ to $v$ is of type 2 or 3 and the vertex $u \notin V'$. Notice that an *F-path* must be of type either 1, 2 or 3. An extra flag $flag(x)$ is used to designate the type of the tentative shortest path from $s$ to $x$. Also note that in the scanning step, a path of type 1 is given higher priority than a path of type 2, vice versa. The flag of every vertex is initially set to 1 and that of $s$ is then set to 2 if $s \in V'$. In the scanning step, when a vertex $v$ with $d(v)$ minimum is selected, if the path formed by concatenating the edge $(v, u)$ and the tentative shortest path from $s$ to $v$ becomes the new tentative shortest path from $s$ to $u$, the flag of the vertex $u$ is reassigned according to the following formula:

1. if $flag(v) = 1$ and $u \notin V'$, then $flag(u) = 1$;

2. if $flag(v) = 1$ *or* $2$ and $(v, u) \in E$, then $flag(u) = 2$;

3. if $flag(v) = 2$ *or* $3$ and $u \notin V'$, then $flag(u) = 3$.

It is not difficult to show that the given algorithm does the computation of the matrix $dist_G^F(x, y), \forall x, y \in V$ in $O(|E||V| + |V|^2 \log |V|)$ time using Fredman and Tarjan's implementation.

**Lemma 3** *The distance matrix* $dist_G^F(x, y), \forall x, y \in V$, *can be computed in* $O(|E||V| + |V|^2 \log |V|)$ *time.*

Note that the algorithm requires sorting of the critical values. The total time taken by the sorting steps is $O(|E||V| \log |V|)$ which is also the time complexity of the algorithm.

**Lemma 4** *The network extension problem can be solved in* $O(|E||V| \log |V|)$ *time.*

## 4 Summary

In this paper, we propose and study the problems of minimum delay Steiner tree and that of network extension. The problems are solved in $O(|E| m \log m + |E||V| + |V|^2 \log |V|)$ and $O(|E||V| \log |V|)$ time, respectively. It can be shown that these two complexities can be further improve to $O(|V| m \log m + |E||V| + |V|^2 \log |V|)$ and $O(|E||V| + |V|^2 \log |V|)$, respectively, by using the ordering of vertices by their distances to the vertex $i$ when the edge $e = (i, j)$ is chosen (in stead of using the critical values for the ordering). In cases where the vertices are also weighted, our discussions still apply with necessary but simple transformation. If the given graph is directed, the weight of an edge is enhanced by the weight of its destination vertex and the weight of each vertex can then be regarded as zero. In an undirected graph, we simply replace each edge by a pair of anti-parallel edges incident on the same pair of vertices as the original edge. The previous transformation then can be applied. Our formulation of the minimum delay Steiner tree problem reflects the asymmetrical roles that vertices could play in a some applications. It's interesting to consider further generalizations of the asymmetry. In the network extension problem, we made the assumption that the given subgraph is connected. This assumption can be relaxed. We strongly believe that the relaxed problem is still polynomial time solvable.

**Acknowledgement** We want thank Dr. M. T. Ko for valuable discussions.

## References

[1] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, vol. 1:pp. 269–271, 1959.

[2] R. Floyd. Algorithm 97: shortest path. *CACM*, 5(6), 1962.

[3] M. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *JACM*, vol. 34(no. 3):pp. 596–615, July 1987.

[4] J. M. Ho, D. T. Lee, and C. K. Wong. Finding minimum diameter spanning trees. in preparation.