On minimal cost-reliability ratio spanning trees and related problems

Yung-Cheng Chang \( ^a, \) Lih-Hsing Hsu \( ^b,^* \)

\( ^a \) Institute of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan 30050, ROC

\( ^b \) Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 30050, ROC

Received 1 July 1994; revised 1 October 1995

Abstract

The minimal cost-reliability ratio spanning tree problem is to find a spanning tree such that the cost-reliability ratio is minimized. This problem can also be treated as a specific version of a more generalized problem discussed by Hassin and Tamir. By Hassin and Tamir's approach, the minimal cost-reliability ratio spanning tree problem can be solved in \( O(q^4) \) where \( q \) is the number of edges in the graph. In this paper, we reduce the complexity of the algorithm proposed by Hassin and Tamir to \( O(q^3) \). Furthermore using our approach, related algorithms proposed by Hassin and Tamir can also be improved by a factor of \( O(q) \).

Keywords: Combinatorial algorithms; Complexity; Spanning trees

1. Introduction and notation

The minimal cost-reliability ratio spanning tree problem, or MCRRST problem for abbreviation, was first discussed by Chandrasekaran et al. [2, 3]. A non-polynomial algorithm for the MCRRST problem was proposed in [3]. Then a polynomial but not strongly polynomial algorithm was introduced by Chandrasekaran and Tamir in [4]. Their algorithm is based on the fact presented in [4] that a query of the form "Is \( a^d \geq c^2? \)" can be solved in time which is polynomial in the binary encoding of the numbers \( a, b, c, \) and \( d \). Later, Hassin and Tamir [5] developed a different, unified approach that yields a strongly polynomial algorithm for classes of optimal spanning tree problems which include the MCRRST problem. By Hassin and Tamir's approach, the MCRRST problem can be solved in \( O(q^4) \) where \( q \) is the number of edges in the graph. In this paper, we reduce the complexity of the algorithm proposed by Hassin and Tamir to \( O(q^3) \).

Now we formally introduce the MCRRST problem. Most of the graph definitions used in this paper are standard (see, e.g., [1]). Let \( G = (V,E) \) be a graph. We associate with each edge \( e_j \in E \) an ordered pair of rational numbers \((a_i, b_i)\), namely a non-negative cost \( a_i \) and a positive probability \( b_i \). For a spanning tree \( T \), the cost of \( T \), \( c(T) \), is defined as \( \sum_{e_i \in T} a_i \) and the reliability of \( T \), \( r(T) \), is defined as \( \prod_{e_i \in T} b_i \). Naturally the cost-reliability ratio of \( T \), \( w(T) \), is defined...
as $c(T)/r(T)$. The MCRRST problem is to find a spanning tree $T$ in $G$ such that $w(T) < w(T')$ for every spanning tree $T'$ in $G$. The MCRRST problem is a specific version of the following generalized problem discussed in Hassin and Tamir [5]. Let $G = (V,E)$ be a graph. In each edge $e_i \in E$ is associated with an ordered pair of rational numbers $(a_i, b_i)$. For a subset $E'$ of $E$, we define $A(E') = \sum_{e_i \in E'} a_i$ and $B(E') = \sum_{e_i \in E'} b_i$. Let $g$ be a real-valued strictly convex function defined in $\mathbb{R}^2$. The problem is to find a spanning tree $T$ which maximizes $g(A(T), B(T))$ over all spanning trees $T$ of $G$. In [5], various $g(A(T), B(T))$, such as $\prod_{e_i \in T} b_i / \sum_{e_i \in T} a_i$, are studied. In the MCRST problem, we need to find a spanning tree $T$ that minimizes $\sum_{e_i \in T} a_i$. This is equivalent to finding a spanning $T$ that minimizes $\prod_{e_i \in T} b_i / \sum_{e_i \in T} a_i$. Since the log function is strictly increasing, it is also equivalent to finding a $T$ that maximizes $\sum_{e_i \in T} \log b_i - \log(\sum_{e_i \in T} a_i)$. Thus, the MCRST problem can be modeled in terms of maximizing $g(A(T), B'(T)) = B'(T) - \log A(T)$ with $B'(T) = \sum_{e_i \in T} \log b_i$ and can be solved in $O(q^4)$ using Hassin and Tamir’s algorithm. We propose a modification of their approach which can reduce the time complexity to solve the MCRST problem to $O(q^3)$ and moreover, improve related algorithms reported in [5] by a factor of $O(q)$.

2. Previous work

To make this paper self-contained, we first outline the basic strategy of Hassin and Tamir’s approach. Let $g$ be a real-valued strictly convex function defined in $\mathbb{R}^2$. Without loss of generality, we assume that $(a_i, b_i) \neq (a_j, b_j)$ if $e_i \neq e_j$. Let the value of a spanning tree $T$, $g(T)$, be defined as $g(A(T), B(T))$. A spanning tree $T^*$ in $G$ is called an optimum spanning tree (with respect to $g$) if $g(T^*) \geq g(T')$ for all spanning trees $T'$ in $G$. The maximum value $\gamma = g(T^*)$ is called a candidate solution. Hassin and Tamir divided the $(A, B)$ plane into a number of cells and showed that each cell produces at most one local optimal spanning tree. The optimum spanning tree $T^*$ is one of these local optimal spanning trees. $T^*$ will be contributed by the unique cell containing $(A(T^*), B(T^*))$.

More precisely, let $(A, B)$ be a point in $\mathbb{R}^2$. Define a directed graph $D_{A,B}(G)$ with the vertex set being the edge set $E$ of $G$. Let $e_i, e_j$ be distinct elements in $E$. If $e_i, e_j$ is an arc in $D_{A,B}$ if and only if $g(A - a_i + a_j, B - b_i + b_j) > g(A, B)$. An equivalence relation in $\mathbb{R}^2$ can be defined by $(A, B) \sim (C, D)$ if and only if $D_{A,B}(G) = D_{C,D}(G)$. We use $W$ to denote the set of equivalence classes induced by “$\sim$". For any $c \in W$, we use $D_c$ to denote the directed graph $D_{A,B}(G)$ with $(A, B) \in c$.

Let $E(D_c)$ be the arc set of $D_c$. Let $T_1$ and $T_2$ be two distinct spanning trees of $G$. We say that $T_2$ is a $D_c$-improvement of $T_1$ if there exist $e_i \in T_1$ and $e_j \notin T_1$ such that $[e_i, e_j] \in E(D_c)$ and $T_2 = T_1 - \{e_i\} + \{e_j\}$, i.e., $T_2$ is obtained from $T_1$ by a single edge swap. A spanning tree $T$ of $G$ is $D_c$-optimal if there exists no spanning tree $T'$ of $G$ which is a $D_c$-improvement of $T$. In [5], the following theorem is presented.

**Theorem 1.** There is at most one $D_c$-optimal spanning tree of $G$ for every $(A, B)$ in $\mathbb{R}^2$.

We use $T_c$ to denote the $D_c$-optimal spanning tree if it exists. Let $\Gamma(D_c)(e_i)$ be the set $\{e_j | [e_i, e_j] \in E(D_c)\}$ for $e_i \in E$. Also, let $X_c$ be the set $\{e_i |$ the two endpoints of $e_i$ in $G$ are on different connected components of the graph $H=(V, \Gamma(D_c)(e_i))\}$. The following theorem is also from [5].

**Theorem 2.** If $D_c$-optimal spanning tree $T_c$ exists, then the edge set of $T_c$ is exactly $X_c$.

This theorem states that a necessary condition for the existence of the $D_c$-optimal spanning tree is that $X_c$ forms a spanning tree of $G$. If $X_c$ forms a spanning tree, it is a candidate solution. It is suggested in [5] that we do not have to verify that the candidate solution is $D_c$-optimal. To reduce the computational complexity, it will suffice simply to find the candidate solution. The optimum spanning tree is a candidate solution that has maximum value. The following algorithm proposed in [5], Algorithm 1, finds $X_c$ and then tests whether it forms a candidate solution in a $D_c$.

**Algorithm 1**

1. Compute $\Gamma(D_c)(e_c)$ for all $e_c \in E$. 
Step 2. Set \( X_c = \emptyset \).

Step 3. For each \( e_x \in E \), do the following:

If the two endpoints of \( e_x \) are disconnected in \( H = (V, \Gamma(D_c(e_x))) \), set \( X_c = X_c \cup \{ e_x \} \).

Step 4. If \( X_c \) does not form a spanning tree, stop and conclude that the \( D_c \) has no solution. Otherwise, \( X_c \) forms the candidate solution.

Obviously, Step 1 in Algorithm 1 takes \( O(q^2) \) time. Step 3 needs \( q \) tests to see if the endpoints of \( e_x \) are not connected in \( H = (V, F(D_c(e_x))) \). Each test takes \( O(q) \) time. Hence, Step 3 takes \( O(q^2) \) time. Step 4 is completed in \( O(p) \) time. Hence, the complexity of the Algorithm 1 is \( O(q^2) \).

We can also describe the set \( W \) as follows: Let \( e_i, e_j \in E, e_i \neq e_j \). Define the function \( g_{ij}(A, B) \) by

\[
g_{ij}(A, B) = g(A - a_i + a_j, B - b_i + b_j) - g(A, B).
\]

Since \( g \) is strictly convex, every topological component induced on \( \mathbb{R}^2 \) by the set of \( g_{ij}(A, B) \) is an equivalent class induced by \( -\). Define \( R_{ij} = \{(A, B) \mid g(A - a_i + a_j, B - b_i + b_j) > g(A, B)\} \). Let \( f : \mathbb{R}^2 \rightarrow \mathbb{R}^k \) be a mapping of \( \mathbb{R}^2 \) into \( \mathbb{R}^k \). Further, let \( T_{ij}, e_i, e_j \in E, e_i \neq e_j \) be a collection of subsets in \( \mathbb{R}^k \) such that \( (A, B) \in R_{ij} \) if and only if \( f(A, B) \in T_{ij} \) for \( e_i, e_j \in E, i \neq j \). Suppose there exists a polynomial \( h_{ij}(x_1, \ldots, x_k) \) such that \( T_{ij} = \{(x_1, \ldots, x_k) \mid h_{ij}(x_1, \ldots, x_k) > 0\} \). Then the number of elements in \( W \) is bounded by the number of topological components induced on \( \mathbb{R}^k \) by the set of polynomials \( h_{ij} \). If \( d \), the maximum degree of \( h_{ij} \), and \( k \), the dimension of \( \mathbb{R}^k \), are constant and independent of \( q \), then it can be proved that the number of equivalence classes will be a polynomial in \( q \).

Hassin and Tamir suggest that we can pick any point for every topological component induced by the set of \( R_{ij} \) (or corresponding \( T_{ij} \)) and apply Algorithm 1 to obtain a spanning tree as a candidate solution. Then the optimum spanning tree is the candidate solution that has the maximum value.

For the MCRRST problem, \( g(T) = g(A(T), B'(T)) = B'(T) - \log A(T) \), with \( A(T) = \sum_{e \in T} a_i, B'(T) = \sum_{e \in T} b_i, R_{ij} = \{(A, B') \mid |B' - \log b_i + \log b_i| > \log (A - a_i + a_j) > B' - \log A\} = \{(A, B') \mid |\log A + \log b_i > \log b_i + \log (A - a_i - a_j)\} = \{(A, B') \mid b_i - (A - a_i + a_j) > b_i(A - a_i + a_j)\} \). Set \( f(A, B') = A \) and

\[T_{ij} = \left\{ x \mid x > \frac{b_i(a_j - a_i)}{b_j - b_i} \right\}.
\]

Hence, \( (A, B') \in R_{ij} \) if and only if \( f(A, B') \in T_{ij} \) for every pair of distinct edges \( e_i, e_j \in E \). Let

\[d_{ij} = \frac{b_i(a_j - a_i)}{b_j - b_i}, \quad e_i, e_j \in E, e_i \neq e_j.
\]

Let \( W = \{ c \mid c \) is a positive interval induced by the set of 0 and \( d_{ij} \), or a set containing a positive \( d_{ij} \). Then, any \( c \in W \) is either the set of a positive \( d_{ij} \) for some \( i \) and \( j \) or the open (positive) interval defined by two consecutive points in the set of 0 and the sorted sequence of positive \( \{d_{ij}\} \). Assume there are \( s \) elements in \( W \). For each \( c \in W \), we pick any point \( r(c) \in c \) as the representative of \( c \). Let \( S = \{r(c_1), r(c_2), \ldots, r(c_s)\} \), with \( r(c_i) < r(c_j) \) if \( i < j \). The following algorithm proposed in [5], Algorithm 2, solves the MCRRST problem.

Algorithm 2

Step 1. Compute and sort the positive numbers \( \{d_{ij}\} \) and obtain the sorted sequence of \( S, \{r(c_1), r(c_2), \ldots, r(c_s)\} \).

Step 2. Construct \( D_{c_k} \) for each \( c_k \in W \) as follows:

Add arc \( [e_i, e_j] \) if and only if one of the following conditions is satisfied.

(a) \( b_j = b_i \) and \( a_j < a_i \).
(b) \( b_j > b_i \) and \( r(c_k) > d_{ij} \).
(c) \( b_j < b_i \) and \( r(c_k) < d_{ij} \).

Step 3. Use Algorithm 1 to find the candidate solution \( X_{c_k} \) for each \( D_{c_k} \). Then compute the value for each candidate solution.

Step 4. Find an optimal solution for the objective.

Step 1 takes \( O(q^2 \log q) \). Step 2 needs \( O(q^4) \) time since the number of arcs in a \( D_{c_k} \) is \( O(q^2) \) and there are \( O(q^2) \) elements in \( S \). Since Algorithm 1 takes \( O(q^2) \) time, Step 3 takes \( O(q^4) \) time. Obviously, Step 4 takes \( O(q^2) \) time. Hence Algorithm 2 takes \( O(q^4) \) time.

3. Our algorithm

Observe that Steps 2 and 3 of Algorithm 2 are repeated several times. To avoid repeated execution of these steps, we should extract and reuse information from what we have solved. Thus, we need the following observation.

Without loss of generality, we assume that \( b_i \neq b_j \) if \( e_i \neq e_j \) and each \( d_{ij} \) is different. There are \( s \) elements
in $S$ where $s \leq 2q(q - 1) + 1$. Moreover, $\{r(c_k) | k \text{ is even}\}$ is the set of positive $d_{ij}$'s. Following the rule that constructs $D_{c_k}$'s in Step 2 of Algorithm 2, we have the following theorem, Theorem 3.

Theorem 3
(1) Assume that $r(c_k) = d_{ij}$ for some $i$ and $j$. Then, if $b_j < b_i$, $E(D_{c_k}) = E(D_{c_{k-1}}) - \{e_i, e_j\}$. Otherwise, $E(D_{c_k}) = E(D_{c_{k-1}})$.
(2) Assume that $r(c_{k-1}) = d_{ij}$ for some $i$ and $j$. Then, if $b_j > b_i$, $E(D_{c_k}) = E(D_{c_{k-1}}) + \{e_i, e_j\}$. Otherwise, $E(D_{c_k}) = E(D_{c_{k-1}})$.

The following corollary, Corollary 1, follows from Theorem 3.

Corollary 1. Assume that $r(c_k) = d_{ij}$ for some $e_i$ and $e_j$. We have
(1) $X_{c_k} = X_{c_{k-1}} - \{e_i, e_j\} \cup \{e_x | e_x \in \{e_i, e_j\}, \text{ the two endpoints of } e_x \text{ are disconnected in } H = (V, \Gamma(D_{c_k}(e_x)))\}$.
(2) $X_{c_{k+1}} = X_{c_k} - \{e_i, e_j\} \cup \{e_x | e_x \in \{e_i, e_j\}, \text{ the two endpoints of } e_x \text{ are disconnected in } H = (V, \Gamma(D_{c_{k+1}}(e_x)))\}$.

We now propose the following algorithm, Algorithm 3, for the MCRRST problem.

Algorithm 3
Step 1. Compute and sort the positive numbers $\{d_{ij}\}$ and obtain the sorted sequence of $S$, $\{r(c_1), r(c_2), \ldots, r(c_s)\}$.
Step 2. Set $k = 1$. Construct $D_{c_k}$ as follows:
Add arc $\{e_i, e_j\}$ if and only if one of the following conditions is satisfied.
(a) $b_j = b_i$ and $a_j < a_i$.
(b) $b_j > b_i$ and $r(c_k) > d_{ij}$.
(c) $b_j < b_i$ and $r(c_k) < d_{ij}$.
Compute $\Gamma(D_{c_k}(e_x))$ for all $e_x \in E$.
Step 3. Set $X_{c_1} = \phi$.
Step 4. If $k = s$, go to Step 8.
Step 5. Set $k = k + 1$. Construct $D_{c_k}$ with the rules in Theorem 3.
For $e_i$ and $e_j$ where $r(c_k) = d_{ij}$ or $r(c_{k-1}) = d_{ij}$, compute $\Gamma(D_{c_k}(e_i))$ and $\Gamma(D_{c_k}(e_j))$.
Step 6. Set $X_{c_k} = X_{c_{k-1}} - \{e_i, e_j\}$.
For $e_x = e_i$ and $e_x = e_j$, do the following:
If the two endpoints of $e_x$ are disconnected in $H = (V, \Gamma(D_{c_k}(e_x)))$, set $X_{c_k} = X_{c_k} \cup \{e_x\}$.
Step 8. For $1 \leq k \leq s$, find the $X_{c_k}$ which forms the candidate solution with the optimal objective value then stop.

Step 1 takes $O(q^2 \log q)$. Step 2 needs $O(q^2)$ time. Step 3 computes $X_{c_1}$ and takes $O(q^2)$ time. Steps 5 and 6 can be finished in $O(q)$ time. Since Steps 5 and 6 are executed $O(q^2)$ times, the complexity is $O(q^3)$. Step 8 finds the $X_{c_k}$ that yields the optimal objective value in $O(q^2)$ time. Hence, the complexity of Algorithm 3 is $O(q^3)$.

Algorithm 3 reuses the $D_{c_k}$, $\Gamma(D_{c_k}(e_x))$ for all $e_x \in E$, and $X_{c_k}$ generated from a previously computed adjacent equivalence class. Steps 1–3 actually do the same thing Algorithm 2 does to $D_{c_1}$. Steps 5 and 6 apply Theorem 3 and Corollaries 1 and 2 derived in this section to compute $D_{c_{k+1}}$, $\Gamma(D_{c_{k+1}}(e_x))$ for all $e_x \in E$, $X_{c_{k+1}}$, from $D_{c_k}$, $\Gamma(D_{c_k}(e_x))$ for all $e_x \in E$, and $X_{c_k}$, respectively. Compared with using Algorithm 1 for every $c_k$, Algorithm 3 reduces the time complexity by $O(q)$ for every $c_k$ where $k > 1$. If there are $\{e_i, e_j, e_m, e_n\} \subseteq E$, where $d_{ij}$ and $d_{mn}$ coincide, Steps 5 and 6 are still executed $O(q^2)$ times if we use perturbation on $d_{ij}$. Step 4 verifies that all $c_k$ are computed and finds the optimal candidate solution. It follows that Algorithm 3 correctly solves the MCRRST problem, just as Algorithm 2 does, but with a complexity of $O(q^3)$.

The methodology of our improved algorithm for the MCRRST problem can be used to improve the other algorithms that apply the unified approach proposed in [5]. Applying Theorem 3 and the corollaries, we can obtain $X_{c'}$ for any equivalence class $c$ from $X_{c'}$ in $O(q)$ time, where $c'$ is an adjacent equivalence class of $c$. In [5], since the set of equivalence classes is induced on $\mathbb{R}^2$ by the set of $\{g_{ij}(A,B)\}$, every equivalence
class has adjacent classes. Hence, the related optimal spanning tree algorithms proposed in [5], which maximize 
\[ g(A(T), B(T)) = (\sum_{e \in T} a_i)^2 + (\sum_{e \in T} b_i)^2, \]
\[ \sum_{e \in T} a_i + \prod_{e \in T} b_i \]
or
\[ \sum_{e \in T} a_i + \prod_{e \in T} b_i, \]
can also be improved by a factor of \( O(q) \) when we use the approach in [5] and that proposed in this paper.

References


