Delay-Constrained Minimum Shortest Path Trees and Related Problems

Junran Lichen[1], Lijian Cai[2], Jianping Li[2](✉), Suding Liu[2], Pengxiang Pan[2], and Wencheng Wang[2]

[1] Institute of Applied Mathematics, Academy of Mathematics and Systems Science, No. 55, Zhongguancun East Road, Beijing 100190, People's Republic of China
[2] Department of Mathematics, Yunnan University, East Outer Ring South Road, University Town, Kunming 650504, People's Republic of China
`jianping@ynu.edu.cn`

Abstract. Motivated by applications in communication networks of the diameter-constrained minimum spanning tree problem, we consider the delay-constrained minimum shortest path tree (DcMSPT) problem. Specifically, given a weighted graph $G = (V, E; w, c)$ and a constant d_0, where length function $w : E \rightarrow R^+$ and cost function $c : E \rightarrow R^+$, we are asked to find a minimum cost shortest path tree among all shortest path trees (in G) whose delays are no more than d_0, where the delay of a shortest path tree is the maximum distance (depending on $w(\cdot)$) from its source to every other leaves in that tree, and the cost of a shortest path tree is the sum of costs of all edges (depending on $c(\cdot)$) in that tree. Particularly, when a constant d_0 is exactly the radius of G, we refer to this version of the DcMSPT problem as the minimum radius minimum shortest path tree (MRMSPT) problem. Similarly, the maximum delay minimum shortest path tree (MDMSPT) problem is asked to find a minimum cost shortest path tree among all shortest path trees (in G) whose delays are exactly the diameter of G.

We obtain the following two main results. (1) We design an exact algorithm in time $O(n^3)$ to solve the DcMSPT problem, and we provide the similar algorithm to solve the MRMSPT problem; (2) We present an exact algorithm in time $O(n^3)$ to solve the MDMSPT problem.

Keywords: Combinatorial optimization · Distances · Delay-constrained shortest path trees · Exact algorithms · Complexity

This paper is supported by the National Natural Science Foundation of China [Nos. 11861075, 12101593], Project for Innovation Team (Cultivation) of Yunnan Province [No. 202005AE160006], Key Project of Yunnan Provincial Science and Technology Department and Yunnan University [No. 2018FY001014] and Program for Innovative Research Team (in Science and Technology) in Universities of Yunnan Province [C176240111009]. Jianping Li is also supported by Project of Yunling Scholars Training of Yunnan Province.

1 Introduction and Problem Description

Many graph optimization problems are motivated from applications in our reality life, for example, the minimum spanning tree problem, the shortest path problem and the minimum Steiner tree problem [17]. In addition, there exist several optimization problems which can be regarded as combinations of some classic graph optimization problems, for example, the single-source shortest path tree problem, briefly as the shortest path tree (SPT) problem, which was first raised in 1957 by Dantzig [6], can be regarded as a combination of the shortest path problem [7] and the minimum arborescence problem [5,9]. In the past five decades, these problems mentioned-above have been deeply studied in the literature and have many wide applications in our reality life, and there exist many polynomial-time exact or approximation algorithms to solve these problems [17].

Spanning trees related problems in weighted graphs have been well studied in theory and widely applied in our reality life [4,17]. So has the diameter problem in which the diameter is originally measured in terms of the number of edges, instead of the total weight of a spanning tree [1,13,17]. In recent decades, the diameter of a weighted graph $G = (V, E; w)$, however, is defined as the longest of the shortest paths among all pairs of distinct vertices in G, *i.e.*, $diam(G) = \max\{\sum_{e \in P_{st}} w(e) \mid P_{st}$ is a shortest path connecting every pair s, t of distinct vertices in $G\}$. In particular, since the path to connect every pair of distinct vertices in a weighted tree $T = (V, E; w)$ is unique, the diameter of T is the maximum weight of a path connecting any two leaves of T. What motivates this investigation is that we want to find a communication network among n vertices, where the communication delay is measured in terms of the total weight of a shortest path between them. A desirable communication network is naturally one tree that has a minimum diameter. Different from studying the minimum spanning tree problem, Ho et al. [13] in 1991 considered the minimum diameter spanning tree (MDST) problem which is formally defined as follows.

Problem 1 (the MDST problem [13]). Given a weighted graph $G = (V, E; w)$ with length function $w : E \rightarrow R^+$, it is asked to find a spanning tree T of G such that the objective is to minimize $\max\{\sum_{e \in P} w(e) \mid P$ is a path connecting any two leaves in $T\}$, *i.e.*, T has a minimum diameter among all spanning trees of G.

Meanwhile, Ho et al. [13] indeed considered the minimum diameter minimum spanning tree (MDMST) problem which is formally defined as follows.

Problem 2 (the MDMST problem [13]). Given a weighted graph $G = (V, E; w)$ with length function $w : E \rightarrow R^+$, it is asked to find a spanning tree $T = (V, E_T)$ of G, the objective is to minimize the total weight $\sum_{e \in E_T} w(e)$ among the all spanning trees that have their diameter values as $diam(G)$ of that graph G, *i.e.*, the all spanning trees considered in this problem have the diameter $diam(G)$ of that graph G.

Ho et al. [13] in 1991 design an exact algorithm in time $O(n^3)$ to find a minimum diameter spanning tree (MDST) of a special graph, called an Euclidean graph, induced by a set of n points in the Euclidean plane, also referred to as a geometric MDST problem. On the other hand, they proved that the MDMST problem is *NP*-complete, using a reduction from the 3SAT problem [10]. Hassin and Tamir [12] in 1995 observed an important fact that the MDST problem is identical to the well studied absolute 1-center problem introduced in 1964 by Hakimi [11], imply that the existing algorithms [8,11], which solves the absolute 1-center problem, also solves the MDST problem on a general graph in time $O(mn + n^2 \log n)$.

At present, we may firmly believe that it would be better to describe the minimum diameter minimum spanning tree (MDMST) problem using the following definition, which involves two different functions.

Problem 3 (the MDMST problem). Given a weighted graph $G = (V, E; w, c)$ with length function $w : E \rightarrow R^+$ and cost function $c : E \rightarrow R^+$, it is asked to find a spanning tree $T = (V, E_T)$ of G, the objective is to minimize the total cost $\sum_{e \in E_T} c(e)$ among the all spanning trees that have their diameter values as $diam(G)$ of that graph G, where the distance between any two vertices depends on computing of length function $w(\cdot)$ and the diameter of G is defined as mentioned-above.

A similar problem, which is called as the diameter-constrained minimum spanning tree (DcMST) problem [10], is formally defined as follows.

Problem 4 (the DcMST problem). Given a weighted connected graph $G = (V, E; w)$ and a positive integer d, where $w : E \rightarrow R^+$, it is asked to seek a spanning tree T on G of minimum weight among all the spanning trees in which no path in T between any two vertices (actually, two leaves) contains more than d edges.

The DcMST problem is sometimes called as the bounded diameter minimum spanning tree problem [14,19,20], and it was shown to be *NP*-complete by Garey and Johnson [10]. In the DcMST problem, the measure of the diameter is in terms of the maximum number of edges in any path of the spanning tree. It is easy to see that the DcMST problem may roughly be treated as a generalization of the MDMST problem (*i.e.*, Problem 2), where no path in spanning tree between any two leaves contains more than d edges for the DcMST problem and the diameter is exactly the longest of a shortest weighted path between any two leaves in spanning tree for the MDMST problem. The DcMST problem arises in various contexts in communication network design, and it has also been given some applications in the area of information retrieval in [2,3]. For the DcMST problem, Kortsarz and Peleg [16] in 1999 showed that, unless $\mathcal{P} = \mathcal{NP}$, no polynomial-time approximation algorithm can be guaranteed to find a solution whose weight is within $\log(n)$ of the optimum. Although Ho et al. [13] in 1991 proved that the MDMST problem is *NP*-complete, using a reduction from the 3SAT problem [10], Seo et al. [18] in 2009 showed that the MDMST problem specialized to Euclidean graphs remains *NP*-complete, using a reduction from the PARTITION problem [10].

In a centralized communication network in which there is a vertex as source, Ho et al. [13] in 1991 defined the minimum radius spanning tree (MRST) problem in a similar manner using the radius instead of the diameter of that weighted graph, the objective is to minimize the maximum communication delay from a source vertex to other vertices in that weighted graph. The minimum radius minimum spanning tree (MRMST) problem is similarly defined as the MDMST problem. The same authors [13] in 1991 proved that the MRMST problem is *NP*-complete.

We have known the fact that, given a weighted connected graph $G = (V, E; w)$ with length function $w : E \rightarrow R^+$, messages are transmitted along a shortest path from vertex to other vertex in G, and shortest path trees will play an important role in such a transmitting system in G. Then we have the following two facts. (1) When messages are transmitted from a source vertex to any other vertices by using shortest path trees, we may consider the delay of messages along a shortest path tree from its source vertex to any other vertices not beyond the expected time d_0, particularly not beyond the radius of that graph G; (2) When we implement this mechanism in (1), we should consider minimum cost to construct such a communication network from the original weighted graph G.

Motivated by the problems and related mechanisms mentioned-above to transmit messages in communication networks, we should consider the following problem and its related variations.

Problem 5 (the DcMSPT problem). Given a weighted graph $G = (V, E; w, c)$ and a constant d_0, where length function $w : E \rightarrow R^+$ and cost function $c : E \rightarrow R^+$, it is asked to find a minimum cost shortest path tree among all shortest path trees (in G) whose delays are no more than d_0, where the delay of a shortest path tree is the maximum distance (depending on $w(\cdot)$) from its source to every other leaves in that tree, and the cost of a shortest path tree is the sum of costs of all edges (depending on $c(\cdot)$) in that tree.

For convenience, we refer to Problem 5 as the delay-constrained minimum shortest path tree (DcMSPT) problem. Particularly, when a constant d_0 is exactly the radius of a weighted graph G, we refer to this version of the DcM-SPT problem as the minimum radius minimum shortest path tree (MRMSPT) problem. Similarly, the maximum delay minimum shortest path tree (MDMSPT) problem is asked to find a minimum cost shortest path tree among all shortest path trees (in G) whose delays are exactly the diameter of G.

So far as what we have known, although the DcMSPT problem and its related variations have many important applications implied in our reality life, where messages are transmitted along shortest paths from a source vertex to every other vertices, these optimization problems have not been studied deeply both in theory and in practice, and there are no exact or approximation algorithms in polynomial time to solve them. For the DcMST problem (*i.e.*, Problem 4), Kortsarz and Peleg [16] in 1999 showed that, unless $\mathcal{P} = \mathcal{NP}$, no polynomial-time approximation algorithm can be guaranteed to find a solution whose weight is within $\log(n)$ of the optimum. However, we hope to design some exact algorithms in polynomial time to optimally solve the DcMSPT problem and its related variations, respectively.

This paper is organized as follows. In Sect. 2, we present some terminologies for easily describing our algorithms and provide key lemmas to ensure the correctness of algorithms. In Sect. 3, we design an exact algorithm to solve the DcMSPT problem, and the similar algorithm solves the MRMSPT problem, where this algorithm runs in time $O(n^3)$; In Sect. 4, we present an exact algorithm to solve the MDMSPT problem, where that algorithm runs in time $O(n^3)$; In Sect. 5, we provide our conclusion and further work.

2 Terminologies and Key Lemmas

In this section, we present some notations and terminologies to solve the delay-constrained minimum shortest path tree (DcMSPT) problem and its related variations, respectively, and other terminologies and notations not defined can be found in those references [1,15,17].

Given a weighted graph $G = (V, E; w, c)$ with length function $w : E \rightarrow R^+$ and cost function $c : E \rightarrow R^+$, for any two vertices $s, t \in V$, the distance between s and t, denoted by $dist_G(s, t)$, is the minimum length of a path connecting s and t if such a path exists, and otherwise $dist_G(s, t) = +\infty$. Concretely, $dist_G(s, t) = \min\{\sum_{e\in P_{st}} w(e) \mid P_{st}$ is a path connecting s and t in $G\}$. And if this path P_{st} satisfies $\sum_{e\in P_{st}} w(e) = dist_G(s, t)$, P_{st} is called as a shortest s-t path or a shortest path connecting s and t. For each vertex $s \in V$, we define the eccentricity of s, denoted by $e_G(s)$, is the maximum of all distances from s to other vertices in G, *i.e.*, $e_G(s, V) = \max\{dist_G(s, t) \mid t \in V\}$. In addition, the diameter of a weighted graph G, denoted by $diam(G)$, is the maximum of all distances between pairs of vertices in G, *i.e.*, $diam(G) = \max\{dist_G(s, t) \mid s, t \in V\}$, and the radius of a weighted graph G, denoted by $rad(G)$, is the minimum of eccentricities of vertices in G, *i.e.*, $rad(G) = \min\{e_G(s) \mid s \in V\}$, meanwhile we refer to this vertex s as a center of G if this vertex s satisfies $e_G(s) = rad(G)$. Furthermore, if $T = (V, E_T; w, c)$ is a spanning tree of a weighted graph $G = (V, E; w, c)$, then the diameter of T is the maximum length of a shortest path connecting any two leaves in T, *i.e.*, $diam(T) = \max\{dist_T(s, t) \mid s$ and t are two leaves of $T\}$, and meanwhile the cost of T is defined as $c(T) = \sum_{e\in E_T} c(e)$.

Dantzig [6,17] in 1957 observed the following. Let $D = (V, A; w)$ be a weighted digraph with a fixed source vertex $s \in V$, where length function $w : E \rightarrow R^+$. An arborescence $T = (V', A')$ rooted at s is called a single source shortest path tree (rooted at s) if V' is the set of vertices in D reachable from s and $A' \subseteq A$, such that for each vertex $t \in V'$, the s-t path in T is a shortest s-t path in D, depending on computing of length function $w(\cdot)$. In particular, we call $T = (V, A')$ as a spanning shortest path tree rooted at s, briefly $T = (V, A')$ as a shortest path tree of D if no ambiguity.

With the similar arguments, we define a shortest path tree in a weighted graph as follows. Let $G = (V, E; w)$ be a weighted graph with a fixed source vertex $s \in V$, where length function $w : E \rightarrow R^+$. A spanning tree $T = (V, E')$

is called a single source shortest path tree with the source vertex s of G, if $dist_T(s,t) = dist_G(s,t)$ holds for every other vertex $t \in V$, *i.e.*, the s-t path in T is a shortest s-t path in G, depending on computing of length function $w(\cdot)$. Briefly, we refer to such a tree $T = (V, E')$ as a shortest path tree with the source vertex s of G, simply as a shortest path tree of G if no ambiguity.

Now, we address the minimum shortest path tree (MSPT) problem as follows.

Problem 6 (the MSPT problem). Given a weighted graph $G = (V, E; w, c)$ with a fixed source s, where length function $w : E \rightarrow R^+$ and cost function $c : E \rightarrow R^+$, it is asked to find a shortest path tree $T = (V, E_T; w, c)$ rooted at s, the objective is to minimize the cost $\sum_{e \in E_T} c(e)$ among all shortest path trees $T = (V, E_T; w)$ rooted at s, where the distance from s to every other vertex t in G depends on computing of length function $w(\cdot)$.

We call a shortest path tree T as to be a minimum cost shortest path tree T in G if the cost of T attains the minimum value among all shortest path trees of G, where distance depends on computing of length function $w(\cdot)$. We present some remarks to the MSPT problem. For an instance of the MSPT problem, a weighted graph $G = (V, E; w, c)$ generally involves two different functions, saying $w(\cdot)$ and $c(\cdot)$, which are with no relationships. We have known the fact that the MSPT problem originally appeared in the literature [1,6,17], where the two functions $w(\cdot)$ and $c(\cdot)$ are essentially the same function.

The strategy to solve the MSPT problem (*i.e.*, Problem 6) is executed as follows. (1) Given a weighted graph $G = (V, E; w, c)$ equipped with a source vertex s for the MSPT problem, depending on computing of length function $w(\cdot)$ in G, we can modify the Dijkstra algorithm [7] to construct an auxiliary acyclic digraph $D_s = (V, A_s; w, c)$ that consists of the union of all shortest s-t paths in G for every other vertex t in $V \setminus \{s\}$. (2) Depending on computing of cost function $c(\cdot)$ in D_s, construct a minimum-cost arborescence at a root s in $D_s = (V, A_s; w, c)$. In fact, we can construct a minimum-cost arborescence at a root s choosing a minimum-cost arc to enter every other vertex in the acyclic digraph $D_s = (V, A_s; w, c)$, without executing an algorithm to solve the minimum arborescence problem specialized to weighted graphs. For convenience, we still denoted such an algorithm in (1) as the Dijkstra algorithm-modified.

Using the Dijkstra algorithm-modified and the strategy mentioned-above, we can obtain the following lemma.

Lemma 1. *There exists a polynomial-time exact algorithm, denoted by the MSPT algorithm, to optimally solve the MSPT problem, and it runs in time $O(n^2)$, where n is the order of weighted graph G.*

3 An Exact Algorithm to Solve the DcMSPT Problem

In this section, we consider the delay-constrained minimum shortest path tree (RcMSPT) problem. Modifying the strategy to solve the MSPT problem (see the Problem 6), we execute the strategy to solve the DcMSPT problem as follows.

(1) For each vertex v in a weighted graph $G = (V, E; w, c)$, depending on computing of length function $w(\cdot)$, use the Dijkstra algorithm-modified [7] to construct an auxiliary acyclic digraph $D_v = (V, A_v; w, c)$ that consists of the union of all shortest paths in G to connect this vertex v to all other vertices v_i in $V \setminus \{v\}$, and in addition, if $dist_{D_v}(v, v_i) \leq d_0$ holds for each vertex v_i in V, depending on cost function $c(\cdot)$, we construct a minimum-cost shortest path tree T_v at the source v in the digraph D_v, otherwise we ignore this vertex v.

(2) Choose a minimum-cost shortest path tree from all shortest path trees constructed in (1).

We describe our algorithm to solve the DcMSPT problem as follows.

```
Algorithm 1: DcMSPT
INPUT: a weighted graph $G = (V, E; w, c)$ and a constant $d_0$;
OUTPUT: a delay-constrained minimum cost shortest path tree, rooted at a
  source $v^* \in V$.
Begin
Step 1. For each vertex $s \in V$, do
    (1.1) Depending on computing of length function $w(\cdot)$, execute the
  Dijkstra algorithm-modified [7] to construct an auxiliary acyclic digraph
  $D_s = (V, A_s; w, c)$, where $A_s$ consists of arcs $(x, y) \in A$ that lies on a shortest
  $(s, v_i)$-path from $s$ to every other vertex $v_i \in V$. For convenience, we may
  assume that all vertices in $D_s = (V, A_s)$ are topologically sorted in the order
  $v_{j_1}$, $v_{j_2}$, ..., $v_{j_n}$, where $v_{j_1} = s$.
    (1.2) If $(e_{D_s}(s) \leq r_0)$ then
        (1.2.1) For each vertex $v_{j_t} \in V$, $t = 2, 3, \ldots, n$, depending on
  computing of cost function $c(\cdot)$, choose a minimum cost arc $e_{i_t} = (v_{i_t}, v_{j_t})$
  in $D_s$ to enter the vertex $v_{j_t}$, where $v_{i_t} \in \{v_{j_1}, v_{j_2}, \ldots, v_{j_n}\}$ (for two distinct
  integers $t, t' \in \{2, 3, \ldots, n\}$, we may have the same vertex $v_{i_t} = v_{i_{t'}}$).
        (1.2.2) Construct a shortest path tree $T_s = (V, A_{T_s})$ with the edge
  set $A_{T_s} = \{e_{i_2}, e_{i_3}, \ldots, e_{i_n}\}$.
Step 2. Choose a minimum-cost shortest path tree $T_{v^*} = (V, A_{T_{v^*}})$ from all
  shortest path trees constructed at Step (1.2.2), i.e., satisfying $c(A_{T_{v^*}}) =$
  $\min\{c(A_{T_s}) \mid T_s = (V, A_{T_s})$ is a shortest path tree in $G$, having $e_{T_s}(s) \leq r_0\}$.
Step 3. Output the minimum-cost shortest path tree $T_{v^*} = (V, A_{T_{v^*}})$ obtained
  at Step 2.
End
```

We can use the DcMSPT algorithm to obtain the following result to optimally solve the DcMSPT problem.

Theorem 1. *The DcMSPT algorithm (i.e., Algorithm 1) is an exact algorithm to solve the DcMSPT problem, and it runs in time $O(n^3)$, where n is the order of weighted graph G.*

Proof. We may assume, without loss of generality, that this weighted graph $G = (V, E; w, c)$ is connected. For each vertex $s \in V$, we shall prove that either $T_s = (V, A_{T_s})$ produced at Step 1 of Algorithm 1 is a minimum-cost shortest path tree at the source vertex s to satisfy $e_{T_s}(v) = e_G(s) \leq r_0$ or this graph G contains no such shortest path trees. For the former, $T_s = (V, A_{T_s})$ is a feasible solution to an instance $G = (V, E; w, c)$ of the DcMSPT problem, and for the latter, there is no feasible solution at the source vertex s to the instance $G = (V, E; w, c)$ of the DcMSPT problem.

Given a fixed source vertex s, since this vertex s is the fixed vertex in G such that each vertex v_r is reachable from s, Step 1.1 at Algorithm 1 executes the Dijkstra algorithm-modified [7] to construct the auxiliary acyclic digraph $D_s = (V, A_s)$ to keep $d_{D_s}(s, v_r) = d_G(s, v_r)$ for every other vertex $v_r \in V$, then this subgraph $D_s = (V, A_s)$ of G contains all shortest path trees at the source vertex s in D_s plus some other edges in G, satisfying $d_{T_s}(s, v_r) = d_{D_s}(s, v_r) = d_G(s, v_r)$. In the view of the choices of arcs at Step 1.2, when $e_{D_s}(s) \leq r_0$, we can indeed prove by induction that the subgraph $T_s = (V, A_{T_s})$ produced at Step 1.2 is a shortest path tree at the source vertex s in D_s, indeed also in G, satisfying $d_{T_s}(s, v_r) = d_{D_s}(s, v_r) = d_G(s, v_r)$ and $dist_{T_s}(s, v_r) \leq r_0$ for every other vertex $v_r \in V$. Thus, this shows that $T_s = (V, A_{T_s})$ produced at Step 1 of Algorithm 1 is a feasible solution to the instance $G = (V, E; w, c)$.

Using greedy technique at Step 1.2 of Algorithm 1 to choose some suitable edges from D_s to be added into T_s, we can indeed prove by induction that $T_s = (V, A_{T_s})$ is a minimum-cost shortest path tree among all shortest path trees in G at the source vertex s, where $A_{T_s} = \{e_{i_2}, e_{i_3}, \ldots, e_{i_n}\}$ and $c(T_s) = \sum_{k=2}^{n} w(e_{i_k})$.

Now, we may assume that $T^*_{s^*} = (V, A_{T^*_{s^*}})$ is a minimum-cost shortest path tree for an instance $G = (V, E; w, c)$ of the the DcMSPT problem, where some vertex $s^* \in V$ satisfies $rad(T^*_{s^*}) \leq r_0$. Since the minimum value outputted at Step 2 is attained by executing the DcMSPT algorithm, enumerating the all minimum-cost shortest path trees at all distinct source vertices in V, we can obtain a minimum-cost shortest path tree $T_{s^*} = (V, A_{T_{s^*}})$ at the source vertex $s^* \in V$ to satisfy $w(T^*_{s^*}) = w(T_{s^*})$ where $rad(T^*_{s^*}) \leq r_0$, implying that $T_{s^*} = (V, A_{T_{s^*}})$ is a minimum-cost shortest path tree and $rad(T^*_{s^*}) \leq r_0$ for an instance $G = (V, E; w, c)$ of the the RMSPT problem.

The complexity of the DcMSPT algorithm (*i.e.*, Algorithm 1) can be determined as follows. (1) For each vertex $s \in V$, the Dijkstra algorithm-modified [7] implies that Step 1.1 needs time $O(n^2)$ to compute the auxiliary acyclic digraph $D_s = (V, A_s)$, and for $e_{D_s}(s) \leq r_0$, Step 1.2 needs time $O(|E|)$ to find such a minimum-cost arborescence T_s at the source vertex s in D_s, showing that the running time at Step 1 is in total $O(n^3)$; (2) Step 2 needs at most time $O(n)$. Hence, the running time of the RMSPT algorithm is in total $O(n^3)$.

This establishes the theorem. ∎

Given a weighted graph $G = (V, E; w, c)$, when we add a step that compute the radius $rad(G)$ of G as the first step in the DcMSPT algorithm, and denoting $d_0 = rad(G)$, we can provide an algorithm to solve the MRMSPT problem. We only present the following conclusion, no description of the MRMSPT algorithm in details to to save a room.

Corollary 1. *The MRMSPT algorithm is an exact algorithm to solve the MRM-SPT problem, and it runs in time $O(n^3)$, where n is the order of weighted graph G.*

4 An Exact Algorithm to Solve the MDMSPT Problem

In this section, we consider the maximum delay minimum shortest path tree (MDMSPT) problem, where maximum delay is exactly the diameter of a weighted graph $G = (V, E; w, c)$.

Using the Dijkstra algorithm [7] for many times and modifying the strategy to solve the DcMSPT problem (seeing the DcMSPT algorithm), we can design our algorithm to solve the MDMSPT problem as follows.

```
Algorithm 2: MDMSPT
INPUT: a weighted graph $G = (V, E; w, c)$;
OUTPUT: a maximum delay minimum shortest path tree, rooted at a source
  $v^* \in V$.
Begin
Step 1. For each vertex $x \in V$, do
    (1.1) Depending on computing of length function $w(\cdot)$, use the Dijk-
  stra algorithm [7] to compute the eccentricity of $x$, i.e., $e_G(x, V) =$
  $\max\{dist_G(x, y) \mid y \in V\}$.
Step 2. Compute the diameter of $G$ as maximum of the eccentricities of all
  vertices in $G$, i.e., $diam(G) = \max\{e_G(x, V) \mid x \in V\}$, and denote the set
  $V_{diam} = \{x \in V \mid e_G(x, V) = diam(G)\}$.
Step 3. For each vertex $s \in V_{diam}$, do
    (3.1) Depending on computing of length function $w(\cdot)$, execute the
  Dijkstra algorithm-modified [7] to construct an auxiliary acyclic digraph
  $D_s = (V, A_s; w, c)$, where $A_s$ consists of arcs $(x, y) \in A$ that lies on a short-
  est $(s, v_i)$-path from $s$ to all other vertices $v_i \in V$. For convenience, we may
  assume that all vertices in $D_s = (V, A_s)$ are topologically sorted in the order
  $v_{j_1}$, $v_{j_2}$, ..., $v_{j_n}$, where $v_{j_1} = s$.
    (3.2) For each vertex $v_{j_t} \in V$, $t = 2, 3, \ldots, n$, depending on computing
  of cost function $c(\cdot)$, choose a minimum cost arc $e_{i_t} = (v_{i_t}, v_{j_t})$ in $D_s$ to
  enter the vertex $v_{j_t}$, where $v_{i_t} \in \{v_{j_1}, v_{j_2}, \ldots, v_{j_n}\}$ (for two distinct integers
  $t, t' \in \{2, 3, \ldots, n\}$, we may have the same vertex $v_{i_t} = v_{i_{t'}}$).
    (3.3) Construct a shortest path tree $T_s = (V, A_{T_s})$ with the edge set
  $A_{T_s} = \{e_{i_2}, e_{i_3}, \ldots, e_{i_n}\}$.
Step 4. Choose a minimum-cost shortest path tree $T_{v^*} = (V, A_{T_{v^*}})$ from all
  shortest path trees constructed at Step (1.2.2), i.e., satisfying $c(A_{T_{v^*}}) =$
  $\min\{c(A_{T_s}) \mid T_s = (V, A_{T_s})$ is a shortest path tree in $G$, having $e_{T_s}(s) \leq r_0\}$.
Step 5. Output the minimum-cost shortest path tree $T_{v^*} = (V, A_{T_{v^*}})$ obtained
  at Step 4.
End
```

Using the MDMSPT algorithm, we can obtain the following result, whose correct proof is similar to the arguments in the proof of Theorem 1, and we omit the details.

Theorem 2. *The MDMSPT algorithm (i.e., Algorithm 2) is a polynomial-time exact algorithm to solve the MDMSPT problem, and its complexity is $O(n^3)$, where n is the order of weighted graph G.*

5 Conclusion and Further Research

In this paper, we consider the delay-constrained minimum shortest path tree (DcMSPT) problem and its related variations, respectively, then we obtain two main results

(1) We design an exact algorithm to solve the DcMSPT problem, we provide the similar algorithm to solve the MRMSPT problem, and both algorithms run in time $O(n^3)$.
(2) We present an exact algorithm to solve the MDMSPT problem, and this algorithm runs in time $O(n^3)$.

A challenging task for further research is to design other exact algorithms in lower running times to solve the DcMSPT problem and its related variations.

References

1. Bondy, J.A., Murty, U.S.R.: Graph theory. In: Graduate Texts in Mathematics, vol. 244. Springer, Heidelberg (2008)
2. Bookstein, A., Klein, S.T.: Construction of optimal graphs for bit-vector compression. In: Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 327–342 (1990)
3. Bookstein, A., Klein, S.T.: Compression of correlated bit-vectors. Inf. Syst. **16**, 387–400 (1991)
4. Cheriton, D., Tarjan, R.: Finding minimum spanning trees. SIAM J. Comput. **5**, 724–742 (1976)
5. Chu, Y.J., Liu, Z.H.: On the shortest arborescence of a directed graph. Scientia Sinica **14**, 1396–1400 (1965)
6. Dantzig, G.B.: Discrete-variable extremum problems. Oper. Res. **5**, 266–277 (1957)
7. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1**, 269–271 (1959)
8. Dvir, D., Handler, G.Y.: The absolute center of a network. Networks **43**(2), 109–118 (2004)
9. Edmonds, J.: Optimum branchings. J. Res. Natl. Bureau Stand. Sect. B **71**, 233–240 (1967)
10. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
11. Hakimi, S.L.: Optimal locations of switching centers and medians of a graph. Oper. Res. **12**, 450–459 (1964)

12. Hassin, R., Tamir, A.: On the minimum diameter spanning tree problem. Inf. Process. Lett. **53**, 109–111 (1995)
13. Ho, J.M., Lee, D.T., Chang, C.H., Wong, C.K.: Minimum diameter spanning trees and related problems. SIAM J. Comput. **20**(5), 987–997 (1991)
14. Julstrom, B.A.: Greedy heuristics for the bounded diameter minimum spanning tree problem, ACM J. Exp. Algorithmics **14**, Article No. 1.1 (2009)
15. Korte, B., Vygen, J.: Combinatorial Optimization: Theory and Algorithms, 5th edn. Springer, Berlin (2012)
16. Kortsarz, G., Peleg, D.: Approximating the weight of shallow Steiner trees. Discrete Appl. Math. **93**, 265–285 (1999)
17. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Springer, The Netherlands (2003)
18. Seo, D.Y., Lee, D.T., Lin, T.-C.: Geometric minimum diameter minimum cost spanning tree problem. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) ISAAC 2009. LNCS, vol. 5878, pp. 283–292. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10631-6_30
19. Singh, A., Gupta, A.K.: Improved heuristics for the bounded-diameter minimum spanning tree problem. Soft Comput. **11**, 911–921 (2007)
20. Torkestani, J.A.: An adaptive heuristic to the bounded-diameter minimum spanning tree problem. Soft Comput. **16**, 1977–1988 (2012)