Computing a Most Probable Delay Constrained Path: NP-Hardness and Approximation Schemes

Ying Xiao, Krishnaiyan Thulasiraman, Xi Fang, Dejun Yang and Guoliang Xue

Abstract—Delay constrained path selection is concerned with finding a source-to-destination path so that the delay of the path is within a given delay bound. When the network is modeled by a directed graph where the delay of a link is a random variable with a known mean and a known variance, the problem becomes that of computing a most probable delay constrained path. In this paper, we present a comprehensive theoretical study of this problem. First, we prove that the problem is NP-hard. Next, for the case where there exists a source-to-destination path with a delay mean no more than the given delay bound, we present a fully polynomial time approximation scheme. In other words, for any given constant ϵ such that $0 < \epsilon < 1$, our algorithm computes a path whose probability of satisfying the delay constraint is at least $(1 - \epsilon)$ times the probability that the optimal path satisfies the delay constraint, with a time complexity bounded by a polynomial in the number of network nodes and $1/\epsilon$. Finally, for the case where any source-to-destination path has a delay mean larger than the given delay bound, we present a simple approximation algorithm with an approximation ratio bounded by the square root of the hop-count of the optimal path.

Index Terms—Delay constrained path selection, computational complexity, approximation schemes.

1 Introduction

Delay constrained path selection is a fundamental problem in computer networks. Commonly, a connection request between a source node s and a destination node t is associated with a bandwidth requirement and a delay bound. Given a connection request, the network operator needs to find an s–t path such that the bandwidth of the path is no smaller than the specified bandwidth requirement, and the delay of the path is no larger than the specified delay bound. In order to compute such a path, we need to know the network state information (NSI) such as the bandwidth and delay of each network element. When exact NSI is available, one can employ the shortest path algorithm to either compute an s–t path satisfying the bandwidth and delay requirements or confirm the nonexistence of such a path [16].

In practice, exact NSI is not always available [1], [2]. One source of inaccuracy in NSI is infrequent flooding of changes to reduce communication overhead as in OSPF [16]. Guerin and Orda [6] gave a detailed description of the sources of inaccuracy in NSI. We refer the readers to [6] and [20] for sources of inaccuracy in NSI and their impact on QoS routing. The existence of inaccuracy in NSI has led to study the routing problem with uncertain parameters [6], [12], [14]. The objective here is to find a path that is most likely to satisfy the delay requirement. This problem is referred to as the *most probable delay constrained path problem* (MPDCP).

In their pioneering work [6], Guerin and Orda studied several aspects of this problem and related computational issues. They defined the most probable delay constrained path problem, proved that it is NP-hard, and studied several special cases. To simplify this problem, Korkmaz and Krunz [12] used the central limit theorem and made mild assumptions on the probability distribution of link delays which lead to a formulation that requires determining an optimal path with respect to a metric involving the mean of the path delay and the variance of the path delay. They considered two cases of the MPDCP problem. Case-1 is concerned with the scenario where there exists an s–t path whose path delay mean is no more than the given delay requirement. Case-2 is concerned with the scenario where the path delay mean for any s–t path is greater than the given delay requirement. They developed iterative algorithms that converge very fast. For Case-1, numerical results show that their algorithm can quickly find a path within a given specified bound from the optimal solution. However, there is no theoretical bound on the number of iterations required to compute the desired solution.

In [25], we presented a fully polynomial time approximation scheme (FPTAS) for Case-1 of MPDCP. For any given constant ϵ such that $0 < \epsilon < 1$, the FPTAS in [25] can compute a path whose probability of satisfying the delay constraint is at least $(1 - \epsilon)$ times the probability that the optimal path satisfies the delay constraint, in time bounded by a polynomial in $\frac{1}{\epsilon}$ and the input size of the instance. In [17], Nikolova *et al.* presented an exactly algorithm for Case-1 with a running time of $O(n^{\Theta(n)})$, where n is the number of nodes in the network. In [18], Nikolova used a novel approach to design FPTAS for a class of stochastic optimization problems, including Case-1 of MPDCP. The time complexity of the FPTAS in [18] also depends on the input size of the instance.

In [21], Uludag *et al.* studied a variant of MPDCP where the link delay follows the Weibullian distribution, and presented an efficient algorithm to compute an optimal solution. In [22], Uludag *et al.* presented a Laplace transform-based heuristic method for a more

Xiao and Thulasiraman are with the School of Computer Science at the University of Oklahoma. Emails: {yxiao, thulasi}@ou.edu. *Fang, Yang and Xue are with the School of Computing, Informatics and Decision Systems Engineering at Arizona State University. Emails:* {xi.fang, dejun.yang, xue}@asu.edu. *This research was supported in part by ARO grant W911NF-09-1-0467 and NSF grant CCF-0830739. The information reported here does not reflect the position or the policy of the federal government. All correspondences should be addressed to Guoliang Xue.*

general class of MPDCP problems.

In this paper, we present *a comprehensive theoretical study of the MPDCP problem* defined in [12]. Our contributions are three-fold. First, we prove that the problem is NP-hard. Next, we present an FPTAS for Case-1. In other words, for any given constant ϵ such that $0 < \epsilon < 1$, our algorithm computes a path whose probability of satisfying the delay constraint is at least $(1 - \epsilon)$ times the probability that the optimal path satisfies the delay constraint, with a time complexity bounded by a polynomial in the number of network nodes and $1/\epsilon$. Finally, we present an efficient approximation algorithm for Case-2.

We now differentiate our work from closely related existing works. Guerin and Orda [6] proved the NP-hardness of a more general version of the MPDCP problem. Since the hardness of a problem does not imply the hardness of a restricted version of the problem, we are the first to present a hardness result for MPDCP. Guerin and Orda [6] presented an FPTAS for a rate-based most probable delay constrained path problem, which is different from the MPDCP problem studied in [12] and this paper. Also, the techniques used in [6] and those used in this paper are very different. The focus of Korkmaz and Krunz [12] is on the design of *practically effective algorithms* for the MPDCP problem, while that of our paper is on the *theoretical study* of the problem. Our results complement those of Korkmaz and Krunz [12]. The techniques used to develop our FPTAS for Case-1 follow the same principles of scaling and rounding that were used in the works for multi-constrained routing [8], [15], [28], [29], but are different in two major aspects. First, all of these earlier papers are dealing with multiple QoS parameters that are additive, but the square root of the delay variance considered in this paper is not additive. Second, each FPTAS in these earlier papers depends on a pair of tight lower and upper bounds on some function of the optimal path, but the FPTAS in this paper does not compute such a pair of tight lower and upper bounds explicitly. Instead, we search over a set of ordered pairs with polynomial cardinality that contains a pair of tight lower and upper bounds. Part of this paper is based on our preliminary work reported in QShine'2004 [25]. However, the results in this paper are significantly stronger than those in [25]. For example, the running time of the FPTAS in [25] is not strongly polynomial (in the sense that it depends on the *input size* of the instance), but the running time of the FPTAS in this paper is strongly polynomial. The running time of the FPTAS in Nikolova [18] for Case-1 of MPDCP also depends on the input size of the instance. The approximation algorithm for Case-2 in this paper is new. The techniques that we used to design the approximation algorithm may find applications in other multi-constrained routing problems.

Since the MPDCP problem is related to the traditional multi-constrained QoS routing problems, we briefly review some of the related works in this area. Handler and Zang [7] presented a dual algorithm for the constrained shortest path problem, which forms the basis for all of the Lagrangian relaxation based approaches for QoS routing problems [10], [26], [27]. Jaffe [9] presented a provably good approximation algorithm for the problem of finding a path subject to multiple constraints. Chen and Nahrstedt [4] used a scaling and rounding technique to design very effective algorithms for computing a path subject to two additive constraints. Korkmaz and Krunz [11] proposed a randomized heuristic algorithm. Hassin [8] designed an FPTAS for the delay constrained least cost path problem (DCLC), which was later improved by Lorenz and Raz [15], Xue *et al.* [28] and Xue *et al.* [29]. Other recent works on the QoS routing problem can be found in [13], [19], [23], [24].

The rest of this paper is organized as follows. In Section 2, we formally define the problem to be studied, along with notations that will be used in later sections. In Section 3, we prove the NP-hardness of the MPDCP problem. In Section 4, we study Case-1 of the MPDCP problem, and present our FPTAS. In Section 5, we study Case-2 of the MPDCP problem, and present a very simple polynomial time approximation algorithm. We conclude this paper in Section 6.

2 Problem Definitions

We model the network by a directed graph $G(V, E)$, where V is the set of n *nodes* and E is the set of m *links*. For each link $e \in E$, there is a nonnegative *link delay* $d(e) \geq 0$. If $e = (u, v)$, we will use e and (u, v) interchangeably. Therefore $d(e)$ and $d(u, v)$ both denote the delay of link $e = (u, v)$. W.O.L.G, assume that $n > 2$.

Following Korkmaz and Krunz [12], we assume that the link delays are *nonnegative random variables* (RV's) with mean μ and variance σ^2, and that these RV's are mutually independent.

Let p be a path connecting source node s to destination node t. The *delay of path* p is a random variable defined by

$$d(p) \triangleq \sum_{e \in p} d(e), \tag{2.1}$$

with *delay mean*

$$\mu(p) \triangleq \sum_{e \in p} \mu(e), \tag{2.2}$$

and *delay variance*

$$\sigma^2(p) \triangleq \sum_{e \in p} \sigma^2(e). \tag{2.3}$$

Let $\mathbb{D} > 0$ be a given *delay bound*. The probability that the delay of path p is no larger than $\mathbb{D}$ is

$$\pi(\mathbb{D}, p) \triangleq Pr\{d(p) \leq \mathbb{D}\}. \tag{2.4}$$

For a given delay bound $\mathbb{D}$ and a source-destination pair (s, t), we are interested in computing an s–t path p such that $\pi(\mathbb{D}, p)$ is maximized. Under mild assumptions that the link delays are mutually independent, and that the probability distribution function of the link delays is continuous and differentiable, Korkmaz and Krunz [12] have shown that the path delay is approximately normally distributed. As a result, they have shown that

$$\pi(\mathbb{D}, p) \approx \Phi\left(\frac{\mathbb{D} - \mu(p)}{\sigma(p)}\right), \tag{2.5}$$

where $\sigma(p) \triangleq \sqrt{\sigma^2(p)}$ and

$$\Phi(x) \triangleq \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-y^2/2} dy. \tag{2.6}$$

Since $\Phi(x)$ is a monotonically increasing function, the *most probable delay constrained path* problem (MPDCP) can be defined as follows [12].

Definition 2.1 (MPDCP**):** Let $G(V, E)$ be a directed graph, where each link $e \in E$ has a delay with mean $\mu(e) > 0$ and variance $\sigma^2(e) > 0$. Let s be the source node, t be the destination node, and $\mathbb{D} > 0$ be the delay bound. The *most probable delay constrained path* problem (MPDCP) asks for a simple s–t path p^{opt} such that for any simple s–t path p, we have $\chi(\mathbb{D}, p^{opt}) \geq \chi(\mathbb{D}, p)$, where $\chi(\mathbb{D}, p) = \frac{\mathbb{D}-\mu(p)}{\sigma(p)}$. □

Korkmaz and Krunz [12] have concentrated on practical algorithms for MPDCP that can obtain good solutions quickly. In this paper, we present a comprehensive theoretical study of this problem. In particular, we study its computational complexity, and present efficient approximation algorithms.

The following defines some of the notations that will be used in this paper. For each link $e \in E$, we will use $\sigma(e)$ to denote the value $\sqrt{\sigma^2(e)}$. For a path p in G, we will use $\mathcal{H}(p)$ to denote the *hop count* of the path. We will also use $\sigma(p) \triangleq \sqrt{\sigma^2(p)}$ to denote the *σ-length of path p*. Note that $\sigma(p)$ is different from $\sum_{e \in p} \sigma(e)$ in general.

3 Computational Complexity

In this section, we study the computational complexity of the MPDCP problem. We will prove that the MPDCP problem is NP-hard. In other words, *there exists no polynomial time optimal algorithm for the MPDCP problem unless* P = NP.

Theorem 3.1: The MPDCP problem is NP-hard. □

PROOF. We will prove the NP-hardness of the MPDCP problem by a polynomial time reduction from the longest path problem, which is known to be NP-hard [5].

Let an instance $\mathcal{I}_1$ of the longest path problem be defined by an *undirected graph* $G_U(V, E_U)$, a source node $s \in V$, and a destination node $t \in V$. The goal of the longest path problem is to find a longest simple path connecting node s to node t in G_U.

We define an instance $\mathcal{I}_2$ of the MPDCP problem by a directed graph $G_D(V, E_D)$ (whose construction and link weights are explained below), source node s, destination node t, and delay bound $\mathbb{D} = 1$. The directed graph G_D is constructed in the following way: For each undirected edge $(u, v) \in E_U$, E_D contains a pair of directed links (u, v) and (v, u). The link weights of G_D are defined in the following way: For each link $(u, v) \in E_D$, we set $\sigma^2(u, v) = 1$. For each link of the form $(u, v) \in E_D$ where $u = t$ or $v = t$, we set $\mu(u, v) = M + 1$, where $M = (n-1)^2(\sqrt{n/(n-1)} + 1)$, and n is the number of nodes. For all the other links $(u, v) \in E_D$ where $u \neq t$ and $v \neq t$, we set $\mu(u, v) = 1$.

Clearly, the construction process takes $O(|V| + |E_U|)$ time. To finish the proof, it suffices to show that any optimal solution of $\mathcal{I}_2$ must be an optimal solution of $\mathcal{I}_1$. To the contrary, assume that p^{opt} is an optimal solution to $\mathcal{I}_2$, but is not an optimal solution to $\mathcal{I}_1$. Since p^{opt} is not an optimal solution to $\mathcal{I}_1$, we have

$$\sigma^2(p^{opt}) \leq \sigma^2(p^l) - 1, \tag{3.1}$$

where p^l is a longest simple s–t path in $G_U(V, E_U)$. Since there are at most $n-1$ links in a simple s–t path, we have $\sigma^2(p^l) \leq n - 1$. Therefore we have

$$\frac{\sigma^2(p^l)}{\sigma^2(p^{opt})} \geq \frac{\sigma^2(p^l)}{\sigma^2(p^l) - 1} \geq \frac{n-1}{n-2}. \tag{3.2}$$

Since p^{opt} is an optimal solution to instance $\mathcal{I}_2$ and p^l is a feasible solution to instance $\mathcal{I}_2$, we have

$$\frac{1 - \mu(p^{opt})}{\sigma(p^{opt})} \geq \frac{1 - \mu(p^l)}{\sigma(p^l)}. \tag{3.3}$$

Following the definition of the link weights, we have $M + 1 \leq \mu(p^{opt})$ and $M + 1 \leq \mu(p^l) \leq M + n$. Therefore $\mu(p^{opt}) > 1$ and $\mu(p^l) > 1$. Hence equation (3.3) implies

$$\frac{\sigma^2(p^l)}{\sigma^2(p^{opt})} \leq \left(\frac{\mu(p^l) - 1}{\mu(p^{opt}) - 1}\right)^2 \leq \left(\frac{M + n - 1}{M}\right)^2. \tag{3.4}$$

Summarizing the above, we have

$$\frac{n}{n-1} < \frac{n-1}{n-2} \overset{(3.2)}{\leq} \frac{\sigma^2(p^l)}{\sigma^2(p^{opt})} \overset{(3.4)}{\leq} \left(\frac{M + n - 1}{M}\right)^2 = \frac{n}{n-1}.$$

This contradiction proves the theorem. ■

Theorem 3.1 presents a *negative result*. It shows that it is unlikely to design polynomial time optimal algorithms for the MPDCP problem. In the rest of this paper, we will present several important *positive results* on the MPDCP problem. As in [12], we distinguish two disjoint cases of the MPDCP problem.

Case-1: There exists an s–t path p such that $\mu(p) \leq \mathbb{D}$. We denote this case as MPDCP1.

Case-2: For any s–t path p, we have $\mu(p) > \mathbb{D}$. We denote this case as MPDCP2.

The following lemma presents important properties of the two cases, as well as the time required to find out which case an instance of MPDCP belongs to.

Lemma 3.1: For any given instance of the MPDCP problem, we can decide whether it is Case-1 or Case 2 using $O(m + n \log n)$ time. Furthermore,

1) For Case-1, we can compute, in $O(m+n \log n)$ time, an s–t path p such that $\pi(\mathbb{D}, p) \geq 0.5$.
2) For Case-2, there does not exist an s–t path p such that $\pi(\mathbb{D}, p) \geq 0.5$. □

PROOF. We can compute a shortest s–t path p^μ using $\mu(e)$ as the metric on the links. This requires $O(m + n \log n)$ time. We have Case-1 if and only if $\mu(p^\mu) \le \mathbb{D}$.

Assume it is Case-1. We have $\mu(p^\mu) \le \mathbb{D}$. Therefore

$$\pi(\mathbb{D}, p^\mu) \ge \Phi(0) = 0.5. \tag{3.5}$$

Assume it is in Case-2. For any s–t path p, we have $\mu(p) > \mathbb{D}$. Therefore

$$\pi(\mathbb{D}, p) < \Phi(0) = 0.5. \tag{3.6}$$

This completes the proof of the theorem. ■

As a byproduct of this theorem, we know that it is less likely for the network operator to find a path satisfying the delay requirement in Case-2 than in Case-1. Hence the recognition of Case-2 may suggest to the network operator that the delay bound is too small, and should be increased. Therefore Case-1 is the more interesting case of the two.

Our hardness proof of MPDCP is based on the hardness of Case-2, and it does not imply the hardness of Case-1. Since the best-known exact algorithm for Case-1 has a time complexity of $O(n^{\Theta(\log n)})$ [17], we *believe* that Case-1 itself is also NP-hard.

4 CASE-1: A FULLY POLYNOMIAL TIME APPROXIMATION SCHEME

In this section, we concentrate on Case-1 of the MPDCP problem, MPDCP1. We will present an FPTAS for this problem. We will apply scaling and rounding on the link weight $\sigma^2(e)$, and use dynamic programming to solve the corresponding problems where the delay variance of each link is a positive integer (as a result of scaling and rounding).

4.1 Approximating MPDCP1 when a Pair of Tight Lower and Upper Bounds on $\sigma^2(p^{opt})$ Is Known

In this subsection, we present an FPTAS that can compute an arbitrarily good approximation to p^{opt} *when we know a lower bound δ on $\sigma^2(p^{opt})$ and an upper bound Δ on $\sigma^2(p^{opt})$ such that $\delta \le \sigma^2(p^{opt}) \le \Delta \le 2\delta$.* In particular, for any given constant $\varepsilon > 0$, our FPTAS computes an s–t path $p*$ such that $\chi(\mathbb{D}, p^*) \ge \chi(\mathbb{D}, p^{opt})/\sqrt{1+\varepsilon}$, in time $O(\frac{mn}{\varepsilon})$. This FPTAS is presented in Algorithm 1.

Algorithm 1 has three phases. In the first phase (Lines 1-4), we define the scaling parameter θ and perform scaling and rounding on the link weight $\sigma^2(e)$ to obtain a new (positive integer-valued) link weight $\sigma^2_\theta(e)$. We also compute two integers L and U where $L < U$ such that a) $U - L$ is $O(\frac{n}{\epsilon})$, b) for some $c = L, L+1, \ldots, U$, we have $\chi(\mathbb{D}, p_c) \ge \chi(\mathbb{D}, p^{opt})$, where p_c is an s–t path such that $\mu(p_c) = \min\{\mu(p) | \sigma^2_\theta(p) \le c\}$.

In the second phase (Lines 5-15), we compute the entries $\mu[v; c]$ for each node $v \in V$ and each value $c = 0, 1, \ldots, U$ using dynamic programming. Essentially, $\mu[v; c]$ is the minimum μ-length (measured by link weight $\mu(e)$) among all s–v paths whose σ^2_θ-length (measured by link weight $\sigma^2_\theta(e)$) is no more than c.

Algorithm 1 AlgMPDCP1$(G, \mu, \sigma^2, s, t, \mathbb{D}, \varepsilon, \delta, \Delta)$

Input: Graph G with link weights μ and σ^2, source node s, destination node t, delay bound $\mathbb{D}$, precision $\varepsilon > 0$, bounds δ and Δ on the delay variance of the optimal path such that $\delta \le \sigma^2(p^{opt}) \le \Delta \le 2\delta$.

Output: An s–t path $p*$.

1: $\theta := \frac{n}{\varepsilon \cdot \delta}$; $L := \lfloor \frac{n}{\varepsilon} \rfloor$; $U := \lfloor \frac{2n}{\varepsilon} \rfloor + n - 1$;
2: **for** $e \in E$ **do**
3: $\quad \sigma^2_\theta := \lfloor \theta \cdot \sigma^2(e) \rfloor + 1$;
4: **end for**
5: **for** $c := 0$ **to** U **do**
6: $\quad \mu[v; c] := \infty$, $\pi[v; c] :=$ **null**, $\forall\, v \in V$;
7: $\quad \mu[s; c] := 0$;
8: **end for**
9: **for** $c := 0$ **to** U **do**
10: $\quad$ **for** each $(u, v) \in E$ s.t. $b \triangleq c - \sigma^2_\theta(u, v) \ge 0$ **do**
11: $\quad\quad$ **if** $(\mu[v; c] > \mu[u; b] + \mu(u, v))$ **then**
12: $\quad\quad\quad \mu[v; c] := \mu[u; b] + \mu(u, v)$, $\pi[v; c] := [u; b]$.
13: $\quad\quad$ **end if**
14: $\quad$ **end for**
15: **end for**
16: $\chi^* := -\infty$; $p^* :=$ **null**;
17: **for** $c := L$ **to** U **do**
18: $\quad$ **if** $\mu[t; c] \ne \infty$ **then**
19: $\quad\quad$ Starting from $[t; c]$ to trace out an s–t path p_c in reverse order, using the π field.
20: $\quad\quad$ **if** $(\chi^* < (\mathbb{D} - \mu(p_c))/\sigma(p_c))$ **then**
21: $\quad\quad\quad \chi^* := (\mathbb{D} - \mu(p_c))/\sigma(p_c)$; $p^* := p_c$;
22: $\quad\quad$ **end if**
23: $\quad$ **end if**
24: **end for**
25: **return** the path p^*.

Also computed are the $\pi[v; c]$ entries that can be used to trace out the corresponding paths. In particular, for each value of $c = L, L+1, \ldots, U$ such that $\mu[t; c] \ne \infty$, we can trace out an s–t path p_c such that $\sigma^2_\theta(p_c) \le c$ and $\mu(p_c) = \mu[t; c] = \min\{\mu(p) | \sigma^2_\theta(p) = c\}$.

In the third phase (Lines 16-25), we compute an s–t path p^* such that $\chi(\mathbb{D}, p^*) \ge \chi(\mathbb{D}, p_c), \forall L \le c \le U$.

Theorem 4.1: If $0 < \delta \le \Delta \le 2\delta$, then Algorithm 1 has time complexity $O(\frac{mn}{\varepsilon})$. If p^{opt} is an optimal solution to MPDCP1 such that $\delta \le \sigma^2(p^{opt}) \le \Delta \le 2\delta$, then Algorithm 1 finds a solution p^* such that $\chi(\mathbb{D}, p^*) \ge \chi(\mathbb{D}, p^{opt})/\sqrt{1+\varepsilon}$. □

PROOF. The running time of Algorithm 1 is dominated by the loop in lines 9-15, which is bounded by $O(mU) = O(\frac{mn}{\varepsilon})$. We next prove the second part of this theorem.

Let p be any s–t path. Following the definition of $\sigma^2_\theta(e)$, we have

$$\theta \cdot \sum_{e \in p} \sigma^2(e) \le \sum_{e \in p} \sigma^2_\theta(e) \le \mathcal{H}(p) + \theta \cdot \sum_{e \in p} \sigma^2(e). \tag{4.1}$$

Therefore we have

$$\sigma(p) \cdot \sqrt{\theta} \le \sigma_\theta(p) \le \sigma(p) \cdot \sqrt{\theta + \frac{\mathcal{H}(p)}{\sigma^2(p)}}. \tag{4.2}$$

During lines 5-15 of the algorithm, we have computed the entries $\mu[t;c]$ for $c = 0, 1, 2, \ldots, U$ such that for each of the values of c with $\mu[t;c] \neq \infty$, there is an s–t path $p_c = \mathsf{argmin}\{\mu(p)|\sigma_\theta^2(p) = c\}$ with $\mu(p_c) = \mu[t;c]$. Note that $p_c = \mathsf{argmin}\{\mu(p)|\sigma_\theta^2(p) = c\}$ implies that for each $c = 0, 1, 2, \ldots, U$ with $\mu[t;c] \neq \infty$, we have

$$p_c = \mathsf{argmax}\left\{\frac{\mathbb{D} - \mu(p)}{\sigma_\theta(p)}|\sigma_\theta^2(p) = c\right\}. \tag{4.3}$$

Assume that p^{opt} is an optimal solution to MPDCP1 such that $\delta \leq \sigma^2(p^{opt}) \leq \Delta \leq 2\delta$. We claim that

$$L \leq \sigma_\theta^2(p^{opt}) \leq U. \tag{4.4}$$

To prove the right hand side of (4.4), we note that

$$\begin{aligned}
\sigma_\theta^2(p^{opt}) &= \sum_{e\in p^{opt}} \sigma_\theta^2(e) \\
&= \sum_{e\in p^{opt}} (\lfloor\theta\cdot\sigma^2(e)\rfloor + 1) \qquad (4.5)\\
&= \mathcal{H}(p^{opt}) + \sum_{e\in p^{opt}} \lfloor\theta\cdot\sigma^2(e)\rfloor \\
&\leq n - 1 + \sum_{e\in p^{opt}} \lfloor\theta\cdot\sigma^2(e)\rfloor \\
&\leq n - 1 + \lfloor \sum_{e\in p^{opt}} \theta\cdot\sigma^2(e)\rfloor \\
&= n - 1 + \lfloor\theta\cdot\sigma^2(p^{opt})\rfloor \\
&\leq n - 1 + \lfloor\theta\cdot\Delta\rfloor \\
&\leq U, \qquad (4.6)
\end{aligned}$$

where equation (4.5) follows from the definition of $\sigma_\theta^2(e)$, and inequality (4.6) follows from the definitions of θ and U, and the assumption that $\sigma^2(p^{opt}) \leq \Delta \leq 2\delta$.

To prove the left hand side of (4.4), we note that

$$\begin{aligned}
\sigma_\theta^2(p^{opt}) &= \sum_{e\in p^{opt}} (\lfloor\theta\cdot\sigma^2(e)\rfloor + 1) \\
&\geq \sum_{e\in p^{opt}} \theta\cdot\sigma^2(e) \\
&= \theta\cdot\sigma^2(p^{opt}) \\
&\geq \theta\cdot\delta \qquad (4.7)\\
&\geq L, \qquad (4.8)
\end{aligned}$$

where inequality (4.7) follows from the assumption that $\delta \leq \sigma^2(p^{opt})$, and inequality (4.8) follows from the definitions of θ and L.

Let $p^\theta = \mathsf{argmax}\{\frac{\mathbb{D}-\mu(p)}{\sigma_\theta(p)}|L \leq \sigma_\theta^2(p) \leq U\}$. We have

$$\frac{\mathbb{D} - \mu(p^*)}{\sigma(p^*)} \geq \frac{\mathbb{D} - \mu(p^\theta)}{\sigma(p^\theta)}, \quad \frac{\mathbb{D} - \mu(p^\theta)}{\sigma_\theta(p^\theta)} \geq \frac{\mathbb{D} - \mu(p^{opt})}{\sigma_\theta(p^{opt})}, \tag{4.9}$$

where the first inequality in (4.9) holds because p^* must have been compared with p^θ in lines 20-22 of Algorithm 1; and the second inequality in (4.9) holds due to the definition of p^θ and inequality (4.4). Therefore we have (see explanations below)

$$\begin{aligned}
\chi(\mathbb{D}, p^*) &= \frac{\mathbb{D} - \mu(p^*)}{\sigma(p^*)} \\
&\geq \frac{\mathbb{D} - \mu(p^\theta)}{\sigma(p^\theta)} \qquad (4.10)\\
&\geq \frac{\mathbb{D} - \mu(p^\theta)}{\sigma_\theta(p^\theta)} \cdot \sqrt{\theta} \qquad (4.11)\\
&\geq \frac{\mathbb{D} - \mu(p^{opt})}{\sigma_\theta(p^{opt})} \cdot \sqrt{\theta} \qquad (4.12)\\
&\geq \frac{\mathbb{D} - \mu(p^{opt})}{\sigma(p^{opt})} \cdot \frac{\sqrt{\theta}}{\sqrt{\theta + \frac{\mathcal{H}(p^{opt})}{\sigma^2(p^{opt})}}} \qquad (4.13)\\
&= \frac{\mathbb{D} - \mu(p^{opt})}{\sigma(p^{opt})} / \sqrt{1 + \frac{\varepsilon\cdot\delta\cdot\mathcal{H}(p^{opt})}{n\cdot\sigma^2(p^{opt})}} \\
&\geq \frac{\mathbb{D} - \mu(p^{opt})}{\sigma(p^{opt})} / \sqrt{1+\varepsilon} \qquad (4.14)\\
&= \chi(\mathbb{D}, p^{opt})/\sqrt{1+\varepsilon}, \qquad (4.15)
\end{aligned}$$

where inequality (4.10) follows from the first inequality in (4.9); inequality (4.11) follows from the first inequality in (4.2); inequality (4.12) follows from the second inequality in (4.9); inequality (4.13) follows from the second inequality in (4.2); equality (4.14) follows from the definition of θ; and inequality (4.14) follows from the facts that $\mathcal{H}(p^{opt}) \leq n$ and $\sigma^2(p^{opt}) \geq \delta$. This completes the proof. ■

Remark 4.1: The condition $\delta \leq \sigma^2(p^{opt}) \leq \Delta$ guarantees that Algorithm 1 returns a non-null path p^*. When the above condition is not satisfied, the returned path p^* may be a null path. As long as $\delta \leq \Delta \leq 2\delta$, Algorithm 1 has a time complexity of $O(\frac{mn}{\varepsilon})$.

4.2 A Polynomial Sized Set containing a Pair of Tight Lower and Upper Bounds

Theorem 4.1 requires the knowledge of a pair of lower and upper bounds δ and Δ on $\sigma^2(p^{opt})$. *Instead of attempting to compute such a pair of lower and upper bounds, we will compute a set of ordered pairs that contains the desired pair (δ, Δ) which satisfies the condition of Theorem 4.1.* More importantly, the cardinality of this set is at most $m \log n$,

Lemma 4.1: Let p be any s–t path in G. Then

$$\max_{e\in p} \sigma^2(e) \leq \sigma^2(p) \leq (n-1) \times \max_{e\in p} \sigma^2(e). \tag{4.16}$$

PROOF. Note that p must contain at least one edge e^{max} such that $\sigma^2(e^{max}) = \max_{e\in p} \sigma^2(e)$ and that the number of edges in p is at least 1 and at most $n-1$. Therefore we have inequality (4.16). ■

Lemma 4.2: Let p be any s–t path in G. Then

$$\sigma^2(p) \in \cup_{e\in E}[\sigma^2(e), (n-1)\times\sigma^2(e)]. \tag{4.17}$$

PROOF. This follows from Lemma 4.1. ■

We know $\sigma^2(p^{opt}) \in \cup_{e\in E}[\sigma^2(e), (n-1)\times\sigma^2(e)]$ from Lemma 4.2. However, some of these m closed intervals may overlap with each other. For the efficiency of our

FPTAS (to be presented in Section 4.3), we want to write the union of these m closed intervals as the union of a set of $J \le m$ *disjoint* closed intervals. Intuitively, this can be done by a left to right scanning of the m intervals.

Theorem 4.2: For any given instance of MPDCP1, there exist $J \le m$ disjoint closed intervals $[\alpha_j, \beta_j]$, $j = 1, 2, \ldots, J$ such that

1) For any optimal solution p^{opt} of the instance of MPDCP1, we have $\sigma^2(p^{opt}) \in \cup_{1 \le j \le J}[\alpha_j, \beta_j]$.
2) $0 < \alpha_1 \le \beta_1 < \alpha_2 \le \beta_2 < \cdots < \alpha_J \le \beta_J$.
3) These disjoint intervals can be computed in $O(m \log n)$ time.
4) $\sum_{1 \le j \le J} \log \frac{\beta_j}{\alpha_j} \le m \log(n-1)$. □

PROOF. We first sort (by the left end point) the m intervals on the right hand side of (4.17), then scan them from left to right. If two neighboring intervals are not disjoint, we can replace them by their union (which is another closed interval). Therefore we can compute the set of J disjoint intervals in $O(m \log n)$ time. This proves the first three claims of the theorem (the first claim follows from Lemma 4.2).

For non-disjoint intervals $[a, b]$ and $[c, d]$ with $0 < a < c \le b < d$, we have $\log \frac{d}{a} \le \log \frac{b}{a} + \log \frac{d}{c}$. For each $j = 1, 2, \ldots, J$, let $[\alpha_j, \beta_j]$ be the union of I_j closed intervals on the right hand side of (4.17), i.e.,

$$[\alpha_j, \beta_j] = \cup_{i=1}^{I_j}[\sigma^2(e_i^j), (n-1) \cdot \sigma^2(e_i^j)], \ 1 \le j \le J. \quad (4.18)$$

Then we have

$$\log \frac{\beta_j}{\alpha_j} \le \sum_{i=1}^{I_j} \log \frac{(n-1) \cdot \sigma^2(e_i^j)}{\sigma^2(e_i^j)} = I_j \cdot \log(n-1). \quad (4.19)$$

Since $m = \sum_{1 \le j \le J} I_j$, we have

$$\sum_{1 \le j \le J} \log \frac{\beta_j}{\alpha_j} \le \sum_{1 \le j \le J} I_j \cdot \log(n-1) = m \log(n-1). \quad (4.20)$$

This proves the inequality in 4). ■

Theorem 4.3: For any given instance of MPDCP1, we can compute a list of ordered pairs $(\delta_k, \Delta_k), k = 1, 2, \ldots, K$, where $K \le m \log(n-1)$ such that

1) $\Delta_k = 2\delta_k, k = 1, 2, \ldots, K$.
2) $\Delta_k \le \delta_{k+1}, k = 1, 2, \ldots, K-1$.
3) For any s–t path p in G, we have

$$\sigma^2(p) \in \cup_{1 \le k \le K}[\delta_k, \Delta_k]. \quad (4.21)$$

Furthermore, the list of ordered pairs can be computed in $O(m \log n)$ time. □

PROOF. As in Theorem 4.2, we compute the list of $J \le m$ disjoint closed intervals $[\alpha_j, \beta_j], j = 1, 2, \ldots, J$. This takes $O(m \log n)$ time.

We compute the list of ordered pairs in the following way. First, we set $\delta_1 := \alpha_1$ and set $k := 1$. For each value of k, we set $\Delta_k := 2 \times \delta_k$. If $\Delta_k \ge \beta_J$, we set $K := k$ and stop. If $\Delta_k < \beta_J$ and $\Delta_k \in [\alpha_j, \beta_j)$ for some $j < J$, we set $\delta_{k+1} := \Delta_k$ and increment k to $k+1$; otherwise, we set δ_{k+1} to the smallest α_j which is greater than Δ_k, and increment k to $k+1$.

We use Fig. 4.2 to illustrate this process. In this example, there are four disjoint closed intervals: $[1, 3]$, $[6, 9]$, $[10, 17]$, and $[19, 22]$. We set $\delta_1 := 1$ and $\Delta_1 := 2 \times \delta_1 = 2$. Since $\Delta_1 \in [1, 3)$, we set $\delta_2 := \Delta_1 = 2$, and set $\Delta_2 := 2 \times \delta_2 = 4$. Since $\Delta_2 < 22$ and $\Delta_2 \notin [\alpha_j, \beta_j)$ for any of the four values of j, we set $\delta_3 := 6$ and set $\Delta_3 := 2 \times \delta_3 = 12$. Since $\Delta_3 \in [10, 17)$, we set $\delta_4 := \Delta_3 = 12$, and set $\Delta_4 := 2 \times \delta_4 = 24$. The process stops here since $\Delta_4 \ge 22$.

Clearly, the list of ordered pairs so computed satisfies all conditions 1)–3) in the theorem. Furthermore, the computation takes $O(m)$ time (a left to right scanning) given the list of disjoint closed intervals. The inequality $K \le m \log(n-1)$ follows from 4) of Theorem 4.2. ■

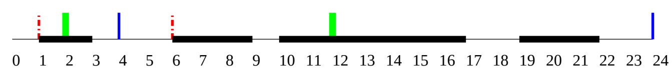

Fig. 1. Illustration of the proof of Theorem 4.3: Black (thick) horizontal line segments denote the disjoint closed intervals; Each red (dashed-dot) vertical line segment denotes a δ_k which is different from Δ_{k-1}; Each blue (solid) vertical line segment denotes a Δ_k which is different from δ_{k+1}; Each green (thick) vertical line segment denotes a Δ_k which is equal to δ_{k+1}.

Note that Theorem 4.3 implies that

$$\delta_k \le \sigma^2(p^{opt}) \le \Delta_k \le 2\delta_k \quad (4.22)$$

is true for some $k \le K$ (which is a condition in Theorem 4.1). We do not know (and *do not need to know*) for which value of k inequality (4.22) holds.

4.3 A Fully Polynomial Time Approximation Scheme

Combining the results obtained in Sections 4.1 and 4.2, we present an FPTAS for MPDCP1, in Algorithm 2.

Algorithm 2 FPTAS-MPDCP1$(G, \mu, \sigma^2, s, t, \mathbb{D}, \epsilon)$

Input: Graph G with link weights μ and σ^2, source node s, destination node t, delay bound $\mathbb{D}$, and precision ϵ with $0 < \epsilon < 1$.

Output: An s–t path p^{apx}.

Compute the $2K$ positive numbers (as in Theorem 4.3)

$$\delta_1 < \Delta_1 \le \delta_2 < \Delta_2 \le \cdots \le \delta_K < \Delta_K$$

such that $\sigma^2(p^{opt}) \in \cup_{1 \le k \le K}[\delta_k, \Delta_k]$.

Set $\chi^{apx} := -\infty$; $p^{apx} :=$ **null**.

for $k := 1$ **to** K **do**

  $p_k :=$ AlgMPDCP1$(G, \mu, \sigma^2, s, t, \mathbb{D}, \epsilon, \delta_k, \Delta_k)$;

  if $\chi^{apx} < (\mathbb{D} - \mu(p_k))/\sigma(p_k)$ **then**

    $\chi^{apx} := (\mathbb{D} - \mu(p_k))/\sigma(p_k)$; $p^{apx} := p_k$;

  end if

end for

return the path p^{apx}.

Algorithm 2 is very simple. First (in Line 1), it computes K disjoint closed intervals $[\delta_k, \Delta_k], k = 1, 2, \ldots, K$ such that $\delta_k \leq \Delta_k \leq 2\delta_k, \ \forall\ 1 \leq k \leq K$ and $\delta_{k'} \leq \sigma^2(p^{opt}) \leq \Delta_{k'} \leq 2\delta_{k'}$ *for some* $k' \in \{1, 2, \ldots, K\}$. It then (in Lines 2-9) applies Algorithm 1 (with $\varepsilon = 2\epsilon$) to each of the K intervals and return the best path.

Theorem 4.4: The time complexity of Algorithm 2 is $O(\frac{m^2 n \log n}{\epsilon})$. It finds a path p^{apx} such that $\chi(\mathbb{D}, p^{apx}) \geq \chi(\mathbb{D}, p^{opt})/(1+\epsilon/2) \geq \chi(\mathbb{D}, p^{opt}) \times (1-\epsilon/2)$, where p^{opt} is an optimal solution for the MPDCP1 problem. □

PROOF. Line 1 takes $O(m \log n)$ time. The for-loop in lines 3-8 takes $O(\frac{mn}{\epsilon} m \log n)$ time, since there are $O(m \log n)$ calls to AlgMPDCP1. This proves the time complexity of the algorithm.

According to Theorem 4.3, there exists an index k' such that $\delta_{k'} \leq \sigma^2(p^{opt}) \leq \Delta_{k'} \leq 2\delta_{k'}$. Therefore, according to Theorem 4.1, we have

$$\frac{\mathbb{D} - \mu(p_{k'})}{\sigma(p_{k'})} \geq \frac{\mathbb{D} - \mu(p^{opt})}{\sigma(p^{opt})} / \sqrt{1+\epsilon}. \tag{4.23}$$

According to lines 6-8, we know that

$$\frac{\mathbb{D} - \mu(p^{apx})}{\sigma(p^{apx})} \geq \frac{\mathbb{D} - \mu(p_{k'})}{\sigma(p_{k'})}. \tag{4.24}$$

One can easily verify that

$$\frac{1}{\sqrt{1+\epsilon}} > \frac{1}{1+\epsilon/2} > 1 - \epsilon/2 \tag{4.25}$$

Inequalities (4.24), (4.23) and (4.25) imply

$$\chi(\mathbb{D}, p^{apx}) \geq \frac{\chi(\mathbb{D}, p^{opt})}{1+\epsilon/2} \geq \chi(\mathbb{D}, p^{opt}) \times (1-\epsilon/2). \tag{4.26}$$

This completes the proof of the theorem. ■

Remark 4.2: Algorithm 2 computes an s–t path p^{apx} whose probability of satisfying the delay bound is at least $(1-\epsilon)$ times the probability that p^{opt} satisfies the delay bound, i.e., $\pi(\mathbb{D}, p^{apx}) \geq (1-\epsilon) \times \pi(\mathbb{D}, p^{opt})$. This follows from Theorem 4.4 and Lemma 4.3 with $\bar{x} = \chi(\mathbb{D}, p^{opt})$ and $\tilde{x} = \chi(\mathbb{D}, p^{apx})$. □

Lemma 4.3: Let $\epsilon > 0$, $\bar{x} \geq 0$, and $\tilde{x}$ be given constants such that $\bar{x} \geq \tilde{x} \geq \bar{x}/(1+\epsilon/2)$. Then $\Phi(\tilde{x}) \geq (1-\epsilon)\Phi(\bar{x})$. □

PROOF. One can verify that $\Phi(\bar{x}) \in [0.5, 1]$ and that the maximum value of $xe^{-x^2/2}$ for $0 \leq x < \infty$ is $1/\sqrt{e}$. Hence

$$\begin{aligned} \Phi(\bar{x}) - \Phi(\tilde{x}) &= \int_{\tilde{x}}^{\bar{x}} e^{-y^2/2} dy \leq (\bar{x} - \tilde{x}) e^{-\tilde{x}^2/2} \\ &\leq \frac{\epsilon}{2} \tilde{x} e^{-\tilde{x}^2/2} \leq \frac{\epsilon}{2\sqrt{e}} \leq \frac{\epsilon}{\sqrt{e}} \Phi(\bar{x}) \leq \epsilon \Phi(\bar{x}). \end{aligned}$$

Therefore $\Phi(\tilde{x}) \geq (1-\epsilon)\Phi(\bar{x})$. ■

5 CASE-2: AN EFFICIENT APPROXIMATION

In this section, we concentrate on the design of an efficient approximation algorithm for MPDCP2. Our approximation algorithm is based on the σ-length and the σ_∞-length of a path, defined in the following, and characterized in Lemma 5.1.

Let p be any path in G. For each edge $e \in E$, recall (at the end of Section 2) that $\sigma(e) = \sqrt{\sigma^2(e)}$. We define the σ_∞-length of path p as

$$\sigma_\infty(p) \triangleq \max_{e \in p} \sigma(e). \tag{5.1}$$

Recall that the σ-length of path p is

$$\sigma(p) \triangleq \sqrt{\sum_{e \in p} \sigma^2(e)}. \tag{5.2}$$

Lemma 5.1: For any s–t path p with $\mu(p) > \mathbb{D}$, we have

$$\frac{\mathbb{D} - \mu(p)}{\sqrt{\mathcal{H}(p)} \cdot \sigma_\infty(p)} \geq \frac{\mathbb{D} - \mu(p)}{\sigma(p)} \geq \frac{\mathbb{D} - \mu(p)}{\sigma_\infty(p)}. \tag{5.3}$$

PROOF. Since

$$\begin{aligned} \sigma_\infty(p) &= \max_{e \in p} \sigma(e) && (5.4) \\ &\leq \left(\sum_{e \in p} \sigma^2(e) \right)^{1/2} && (5.5) \\ &= \sigma(p) && (5.6) \\ &\leq \sqrt{\mathcal{H}(p)} \cdot \sigma_\infty(p), && (5.7) \end{aligned}$$

we have

$$\frac{1}{\sqrt{\mathcal{H}(p)} \cdot \sigma_\infty(p)} \leq \frac{1}{\sigma(p)} \leq \frac{1}{\sigma_\infty(p)}. \tag{5.8}$$

This leads to (5.3), noting that $\mu(p) - \mathbb{D} > 0$. ■

The basic idea of our approximation algorithm is to use the σ_∞-length of a path to approximate the σ-length of a path. The algorithm is listed in Algorithm 3.

Algorithm 3 AproxMPDCP2$(G, \mu, \sigma^2, s, t, \mathbb{D})$

Input: Graph G with link weights μ and σ^2, source node s, destination node t, and delay bound $\mathbb{D}$.
Output: An s–t path p^{apx}.

Sort the different values in $\{\sigma(e) | e \in E\}$ into
$$0 < \tau_1 < \tau_2 < \cdots < \tau_K.$$
Set $p_0^{apx} :=$ **null**, $\chi_0^{apx} := -\infty$.
for $k := 1$ **to** K **do**
  $p_k := \mathrm{argmin}\{\mu(p) |\ s\text{-}t \text{ path } p, \sigma(e) \leq \tau_k, \forall e \in p\}$;
  if $p_k \neq$ **null** and $\frac{\mathbb{D} - \mu(p_k)}{\sigma_\infty(p_k)} > \chi_{k-1}^{apx}$ **then**
    $p_k^{apx} := p_k;\ \ \chi_k^{apx} := \frac{\mathbb{D} - \mu(p_k)}{\sigma_\infty(p_k)}$;
  else
    $p_k^{apx} := p_{k-1}^{apx};\ \ \chi_k^{apx} := \chi_{k-1}^{apx}$;
  end if
end for
return the path $p^{apx} \triangleq p_K^{apx}$.

For each $k \leq K$, Algorithm 3 computes an s–t path p_k^{apx} that maximizes $\frac{\mathbb{D} - \mu(p)}{\sigma_\infty(p)}$ among all s–t paths p that do not use any link e with $\sigma(e) > \tau_k$. We prove that $p^{apx} \triangleq p_K^{apx}$ is a good approximate solution to MPDCP2.

Theorem 5.1: The worst-case time complexity of Algorithm 3 is $O(m^2 + mn\log n)$. The algorithm returns a path p^{apx} such that

$$\chi(\mathbb{D}, p^{apx}) \geq \chi(\mathbb{D}, p^{opt}) \cdot \sqrt{\mathcal{H}(p^{opt})}, \tag{5.9}$$

where p^{opt} is an optimal solution to MPDCP2. □

PROOF. The time complexity of the algorithm is dominated by the loop in lines 3-10, where we invoke Dijkstra's shortest path algorithm K times. Since $K \leq m$, the time complexity of the algorithm is $O(m^2+mn\log n)$. Note that for some values of k, the path p_k may not exist. In such a case, the p_k computed in line 4 is set to **null**.

Let k' be the smallest positive integer such that there exists an s–t path p with $\max_{e\in p}\sigma(e) \leq \tau_{k'}$. Then p_k is a **null** path if and only if $k < k'$. Furthermore, we have

$$\frac{\mathbb{D}-\mu(p_i^{apx})}{\sigma_\infty(p_i^{apx})} = \chi_i^{apx} \leq \chi_j^{apx} = \frac{\mathbb{D}-\mu(p_j^{apx})}{\sigma_\infty(p_j^{apx})}, \forall k' \leq i < j \leq K. \tag{5.10}$$

Let k'' be an integer such that $\tau_{k''} = \sigma_\infty(p^{opt})$ for some $k'' \in [k', K]$. Lines 3-11 imply that

$$\frac{\mathbb{D}-\mu(p^{apx})}{\sigma_\infty(p^{apx})} \geq \frac{\mathbb{D}-\mu(p_{k''}^{apx})}{\sigma_\infty(p_{k''}^{apx})} \geq \frac{\mathbb{D}-\mu(p^{opt})}{\sigma_\infty(p^{opt})}. \tag{5.11}$$

The inequalities in (5.3) and (5.11) imply

$$\frac{\mathbb{D}-\mu(p^{apx})}{\sigma(p^{apx})} \overset{(5.3)}{\geq} \frac{\mathbb{D}-\mu(p^{apx})}{\sigma_\infty(p^{apx})} \tag{5.12}$$

$$\overset{(5.11)}{\geq} \frac{\mathbb{D}-\mu(p^{opt})}{\sigma_\infty(p^{opt})} \tag{5.13}$$

$$\overset{(5.3)}{\geq} \sqrt{\mathcal{H}(p^{opt})} \cdot \frac{\mathbb{D}-\mu(p^{opt})}{\sigma(p^{opt})}. \tag{5.14}$$

Therefore $\chi(\mathbb{D}, p^{apx}) \geq \chi(\mathbb{D}, p^{opt}) \cdot \sqrt{\mathcal{H}(p^{opt})}$. ■

6 Conclusions

We have studied the most probable delay constrained path problem (MPDCP) in a computer network where the link delay is a random variable following a normal distribution with a known delay mean and a known delay variance. We have proved the problem to be NP-hard. For the case where there exists a source-to-destination path with a delay mean no more than the delay bound, we have presented a fully polynomial time approximation scheme. For the case where any source-to-destination path has a delay mean larger than the delay bound, we have present a simple approximation algorithm with an approximation ratio bounded by the square root of the hop-count of the optimal path.

References

[1] G. Apostolopoulos, R. Guerin, S. Kamat, and S.K. Tripathi; Quality of service based routing: a performance perspective; *Proceedings of ACM SIGCOMM'1998*, pp. 17–28.

[2] G. Apostolopoulos, R. Guerin, S. Kamat; Implementation and performance measurements of QoS routing extensions to OSPF; *Proceedings of IEEE INFOCOM'1999*, pp. 680–688.

[3] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, R. Orda and T. Przygienda; QoS routing mechanisms and OSPF extensions, *IETF RFC 2676*, 1999.

[4] S. Chen and K. Nahrstedt; On finding multi-constrained paths; *Proceedings of IEEE ICC'1998*; pp. 874–879.

[5] M.R. Garey and D.S. Johnson; *Computers and Intractability: A Guide to the Theory of NP-Completeness*; W.H. Freeman, 1979.

[6] R.A. Guerin and A. Orda; QoS routing in networks with inaccurate information: theory and algorithms; *IEEE/ACM Transactions on Networking*; Vol. 7(1999), pp. 350–364.

[7] G. Handler and I. Zang; A dual algorithm for the constrained shortest path problem; *Networks*; Vol. 10(1980), pp. 293–309.

[8] R. Hassin; Approximation schemes for the restricted shortest path problem; *Mathematics of Operations Research* Vol. 17(1992), pp. 36–42.

[9] J.M. Jaffe; Algorithms for finding paths with multiple constraints; *Networks* Vol. 14(1984), pp. 95–116.

[10] A. Jűttner, B. Szviatovszki, I. Mécs and Z. Rajkó; Lagrange relaxation based method for the QoS routing problem; *Proceedings of IEEE INFOCOM'2001*; pp. 859–868.

[11] T. Korkmaz and M. Krunz; A randomized algorithm for finding a path subject to multiple QoS requirements; *Computer Networks*; Vol. 36(2001), pp. 251-268.

[12] T. Korkmaz and M. Krunz; Bandwidth-delay constrained path selection under inaccurate state information; *IEEE/ACM Transactions on Networking*; Vol. 11(2003), pp. 384–398.

[13] G. Liu and K.G. Ramakrishnan; A^*prune: An algorithm for finding k shortest paths subject to multiple constraints; *Proceedings of IEEE INFOCOM'2001*; pp. 743–749.

[14] D. Lorenz and A. Orda; QoS routing in networks with uncertain parameters; *IEEE/ACM Transactions on Networking*; Vol. 6(1998), pp. 768–778.

[15] D. Lorenz and D. Raz; A simple efficient approximation scheme for the restricted shortest paths problem; *Operations Research Letters*; Vol. 28(2001), pp. 213–219.

[16] J. Moy; OSPF version 2; *IETF RFC 2178*, 1997.

[17] E. Nikolova, J.A. Kelner, M. Brand, and M. Mitzenmacher; Stochastic shortest paths via quasi-convex maximization; *Proceedings of ESA'2006*, pp. 338-351.

[18] E. Nikolova; Approximation algorithms for reliable stochastic combinatorial optimization; *Proceedings of APPROX'2010*, pp. 338-351.

[19] A. Orda and A. Sprintson; Precomputation schemes for QoS routing; *IEEE/ACM Transactions on Networking*; Vol. 11(2003), pp. 578–591.

[20] A. Shaikh, J. Rexford and K.G. Shin; Evaluating the impact of stale link state on quality-of-service routing; *IEEE/ACM Transactions on Networking*; Vol. 9 (2001), pp. 162–176.

[21] S. Uludag, L. Perkovic, A. Kashkanova, and K. Akkaya; Quality-of-service provisioning via stochastic path selection under Weibullian link delays; *Proceedings of QShine'2007*, pp. 11:1–11:7.

[22] S. Uludag, Z.E. Uludag, K. Nahrstedt, K.S. Lui, and F. Baker; A Laplace transform-based method to stochastic path finding; *Proceedings of ICC'2009*, pp. 1–5.

[23] P. Van Mieghem and F. A. Kuipers; Concepts of exact QoS routing algorithms; *IEEE/ACM Transactions on Networking*; Vol. 12(2004), pp. 851–864.

[24] Z. Wang and J. Crowcroft; Quality-of-service routing for supporting multimedia applications; *IEEE Journal on Selected Areas in Communications*; Vol. 14(1996), pp. 1228–1234.

[25] Y. Xiao, K. Thulasiraman and G. Xue; Approximation and heuristic algorithms for delay constrained path selection under inaccurate state information; *Proceedings of IEEE QShine'2004*; pp. 130–137.

[26] Y. Xiao, K. Thulasiraman and G. Xue; QoS routing in communication networks: Approximation algorithms based on the primal simplex method of linear programming; *IEEE Transactions on Computers*; Vol. 55(2006), pp. 815–829.

[27] G. Xue; Minimum-cost QoS multicast and unicast routing in communication networks; *IEEE Transactions on Communications*; Vol. 51(2003), pp. 817–827.

[28] G. Xue, A. Sen, W. Zhang, J. Tang and K. Thulasiraman; Finding a path subject to many additive QoS constraints; *IEEE/ACM Transactions on Networking*; Vol. 15(2007), pp. 201–211.

[29] G. Xue, W. Zhang, J. Tang and K. Thulasiraman; Polynomial time approximation algorithms for multi-constrained QoS routing; *IEEE/ACM Transactions on Networking*; Vol. 16(2008), pp. 656–669.