

Finding Maximum-Success Disjoint Paths

Aaron Neugebauer 

Department of Computer Science, University of Würzburg, Germany

Image Processing Department, Fraunhofer ITWM-Institute for Industrial Mathematics,
Kaiserslautern, Germany

Marie Schmidt 

Department of Computer Science, University of Würzburg, Germany

Abstract

Motivated by challenges occurring in disaster relief operations, we study the problem to find two edge-disjoint paths in a graph whose arcs are labeled with traversability probabilities that maximize the probability that (at least) one of the paths is traversable.

We show that this problem is NP-hard. To solve the problem, we propose to model it as a bi-objective problem, taking traversability probabilities of the two individual paths as objective functions. We show that an optimal solution to our problem is an extreme-supported solution of the so defined bi-objective problem and exploit this property to find optimal solutions. Furthermore, we present two approximation approaches, and compare the approaches experimentally with respect to runtime and solution quality.

2012 ACM Subject Classification Applied computing → Operations research; Mathematics of computing → Graph theory; Mathematics of computing → Graph algorithms; Mathematics of computing → Quasiconvex programming and unimodality; Mathematics of computing → Integer programming; Theory of computation → Shortest paths; Mathematics of computing → Approximation algorithms

Keywords and phrases Route finding under uncertainty, Edge-disjoint paths, Optimization, Exact algorithms, Approximation algorithms, Vectorization

Digital Object Identifier 10.4230/OASICS.ATMOS.2026.8

Supplementary Material *Software (Source Code)*: <https://github.com/PantomInach/uncertain-road-network-traversal-strategies.git> [29]

archived at `swb:1:dir:08084beb228c3c09cb4a6456e342fc5f853b3874`

1 Introduction

Research on humanitarian logistics, and particularly on the distribution of emergency supplies following disasters is gaining increasing attention in recent years, see, e.g., [1, 27, 3]. Providing disaster relief is often complicated by road damage. To get an overview on the current state of the road infrastructure, air footage can be gathered by drones, planes and helicopters. However, the analysis of this material by human experts is time-consuming and constitutes a bottleneck. Image recognition can help to automate this process, but is prone to uncertainties, i.e. the respective applications cannot provide definite answers but only traversability *probabilities* for the roads in question. [37, 21]

Motivated by the sketched situation, in this paper we investigate the following optimization problem: given a graph with a dedicated source and a dedicated destination and edge labels that denote the probability that an edge is traversable, find two edge-disjoint source-destination paths that maximize the probability to reach the destination on either path. We describe the underlying road network by an (directed) undirected graph $G = (V, E)$ with vertices V and edges E . If the road network contains roads that cannot be traversed from both sides, we model the road network using directed edges, otherwise, the edges are undirected. To incorporate the traversability of the road segments, each edge $e \in E$ also gets a probability $p_e \in (0, 1]$, which are independent from each other. A probability of 1 denotes



© Aaron Neugebauer and Marie Schmidt;

licensed under Creative Commons License CC-BY 4.0

26th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2026).

Editors: Valentina Cacchiani and Stefan Funke; Article No. 8; pp. 8:1–8:18

OpenAccess Series in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

that the corresponding road segment is always traversable. A probability of 0 would indicate that the road segment is impassable, but we exclude such edges from our problem. In total the tuple (V, E, p) gives an *uncertain graph*.

We call $P = ((v_1, v_2), (v_2, v_3), \dots, (v_{k-2}, v_{k-1}), (v_{k-1}, v_k))$ a *path* in an undirected graph $G = (V, E^{\leftrightarrow})$ if $\{v_i, v_{i+1}\} \in E^{\leftrightarrow}$ for $i = 1, \dots, k-1$ or in a directed graph $G = (V, E^{\rightarrow})$ if $(v_i, v_{i+1}) \in E^{\rightarrow}$ for $i = 1, \dots, k-1$. An s - t path starts with the vertex s and ends in the vertex t . The *probability* $\pi(P)$ of a path P is given by $\pi(P) = \prod_{(u,v) \in P} p_{\{u,v\}}$ in an undirected graph. For a directed graph, let $\pi(P) = \prod_{e \in P} p_e$. If not further restrictions are given, the edge set E of a graph can be either directed or undirected.

► **Definition 1** (MAXIMUMSUCCESSDISJOINTPATHSPROBLEM (MSDPP)). *Given (G, p, s, t) with $G = (V, E)$, where (V, E, p) is an (directed) uncertain graph and $s, t \in V$, find two (directed) edge-disjoint s - t paths P_1, P_2 in G that maximize the success probability*

$$\tau(P_1, P_2) = \pi(P_1) + (1 - \pi(P_1))\pi(P_2) = \pi(P_1) + \pi(P_2) - \pi(P_1) \cdot \pi(P_2)$$

of traversing from s to t over the main path P_1 and else the alternative path P_2 .

We show NP-hardness of MSDPP for directed and undirected graphs in Section 3.

Our solution approaches for the MSDPP are based on a key observation: any optimal solution to the MSDPP is an extreme-supported efficient solution to an appropriately defined bi-objective optimization problem. We prove this property (for a more general problem class) in Section 4.1. We propose two exact solution approaches: Sections 4.2 and 4.3. The algorithm proposed in Section 4.2 iterates the most-certain paths and computes subsequently a sufficient subset of efficient solutions, while the second approach, presented in Section 4.3, enumerates extreme supported solutions by ILP-based dichotomic search.

Furthermore, we propose an approximation algorithm and a heuristic that compute specific extreme supported solutions. The approximation algorithm presented in Section 4.4 has a $1/2$ approximation ratio, however, the algorithm has exponential time complexity. The heuristic presented in Section 4.5 is based on Suurballe's algorithm to compute edge-disjoint paths, but no approximation guarantee in the input can be given. We can, however, bound the quality of the solution found by properties of an optimal solution. We compare the runtime of the proposed algorithms and the performance in terms of solution quality for the approximation algorithms in Section 5.

2 Related Work

Shortest paths under uncertainty

There are various uncertain variants of the shortest path problem, spanning stochastic, robust, and online optimization perspectives. While many robust and stochastic optimization approaches concentrate on finding a *single* path that is good under uncertain edge lengths (see, e.g., [28, 32, 36]), authors have also proposed two-stage approaches, which allow a single adjustment of the initially planned path (see, e.g., [8, 18]), and online approaches that gain knowledge while crossing the network and subsequently update the proposed path “on the go” (see, e.g., [4, 42, 35]). Particularly related to our problem is the problem to find a single *most-certain* path in an (undirected or directed) uncertain graph (V, E, p) . As explained by Christofides [10], this problem can be solved by applying Dijkstra's algorithm [11] on the graph where edge probabilities p_e are replaced by edge weights $-\ln p_e$.

Multi-objective optimization and vectorization

Multi-objective optimization aims to find solutions to optimization problems with multiple objective functions. Instead of finding a single solution that optimizes one objective function, in multi-objective optimization one aims to find a representative set of all solutions whose performance (with respect to all considered objective functions) is not *dominated* by any other solution. Concretely, a solution x is called *efficient* if there is no solution x' such that x' is at least as good as x with respect to each objective function and better with respect to at least one. Often, multi-objective optimization problems are solved by applying *scalarization* techniques that transform a multi-objective problem to a single-objective one. One example of such a technique is *dichotomic search* which varies the weights in a *weighted sum* of the different objective functions in a way that allows one to find a subset of efficient solutions, namely the ones characterized as belonging to the efficient solutions on extreme points of the convex hull of solution vectors in $\mathbb{R}^m$. See Ehrgott [12] for an introduction to multi-objective optimization and the concepts and method described here.

Vectorization relates to the approach to transform an originally single-objective optimization problem into a multi-objective problem such that an optimal solution to the single-objective problem can be found by solving the multi-objective problem. This relationship can be exploited in robust optimization with a finite scenario set $\mathcal{U}$, where it is easy to show that there is always an optimal solution that is efficient with respect to the $|\mathcal{U}|$ objective functions obtained from evaluating the solution in specific scenarios, though not every robust optimal solution is efficient in this sense; see Inacu and Trichakis [23]. It has also been exploited for several robust and stochastic variants of the shortest path problem; see [36, 9, 33, 34]. However, we are not aware of a general description of this approach in the literature, hence we provide one and prove its correctness in Section 4.1 and then apply it to our problem.

Edge-disjoint paths

The problem to decide whether an undirected graph contains disjoint paths between k vertex-tuples (s_i, t_i) is known as the *k-disjoint paths problem* (*k-DPP*) [19]. In directed graphs, it is called *k-directed disjoint paths problem* (*k-dDPP*) [19]. With the further restriction that the respective s_i-t_i paths should be shortest s_i-t_i paths in a graph with edge lengths $\ell: E \rightarrow \mathbb{R}$, this gives the disjoint shortest path problem (*k-DSPP* and *k-dDSPP*). The computational complexity of these problems depends on the assumptions made on edge lengths (being harder in some cases if edge lengths are allowed to be 0), the number of paths k that need to be found, and whether the underlying graph is directed or not. In Table 1 we restate an overview table on the complexity results for these decision problems from Gottschau et al. [19]. For symmetric directed graphs – for each directed edge exists an opposite edge – *k-dDPP* is solvable in polynomial time [24]. Note that similar results hold for the problems to find vertex-disjoint paths, since the respective decision problems can be transformed into each other.

A further problem variant, in which two edge-disjoint s_1-t_1 and s_2-t_2 paths in an undirected graph should be computed, such that the total length is minimum, can be solved in polynomial time [7]. For the case $s_1 = s_2$ and $t_1 = t_2$ Suurballe’s algorithm [39, 6] can be used. Eilam-Tzoref [13] prove that another problem variant, the *two disjoint one shortest path problem* (2D1SPP), is NP-hard both in directed and in undirected graphs. Here, the question is to decide whether there exist two edge-disjoint paths P_1 from s_1 to t_1 and P_2 from s_2 to t_2 , where P_1 should be a shortest s_1-t_1 -path. It is also NP-hard if paths should be vertex-disjoint instead of edge-disjoint.

■ **Table 1** Results on the complexity of the disjoint-path problem from [19].

k	No edge weights $\ell \equiv 0$		Non-negative edge weights ℓ	
	k -DPP	k -dDPP	k -DSPP	k -dDSPP
arbitrary	NP-hard [14]	NP-hard [14]	NP-hard [13]	NP-hard [13]
fixed	P [38]	NP-hard [17]	Open for $\ell > 0$	Open for $\ell > 0$
			Open for $\ell \geq 0$	NP-hard for $\ell \geq 0$ [17]
2	P [38, 41]	NP-hard [17]	P for $\ell > 0$ [13]	P for $\ell > 0$ [5]
			P for $\ell \geq 0$ [19, 26]	NP-hard for $\ell \geq 0$ [17]

3 NP-Hardness

► **Theorem 2.** *MSDPP is NP-hard in both directed and undirected uncertain graphs.*

Proof. We show this statement by reduction to the decision version of our problem from the 2D1SPP which is NP-hard (both in directed and undirected graphs) according to [13]. Since the NP-hardness reduction for 2D1SPP from 3SAT ([25]) in [13] only utilizes integer edge lengths and distinct vertex tuples $(s_1, t_1), (s_2, t_2)$, the 2D1SPP is NP-hard for graphs with integer edge lengths. For the construction we use undirected graph notation, but the construction in a directed graph is analogous.

Consider an instance of the 2D1SPP on an undirected graph $G = (V, E)$ with integer edge lengths ℓ_e and two distinct vertex tuples (s_1, t_1) and (s_2, t_2) . We denote by $\ell(P)$ the length of a path P in G . Denote by L the length of a shortest s_1 - t_1 path in G .

We construct an instance of MSDPP as follows: Let $V' = V \cup \{s, t\}$ and $E' = E \cup \{\{s, s_1\}, \{s, s_2\}, \{t_1, t\}, \{t_2, t\}\}$. Consider the uncertain graph $G' = (V', E', p)$ where, for $e \in E$, $p_e = e^{-\ell_e}$, $p_{\{s, s_1\}} = p_{\{t_1, t\}} = 1$ and $p_{\{s, s_2\}} = p_{\{t_2, t\}} = \varepsilon$ with $0 < \varepsilon < \min\{(\prod_{e \in E} p_e) / 2, \sqrt{(e-1)/(e^{L+1}-1)}\}$. Decide whether there are two edge-disjoint s - t paths with a success probability $\geq e^{-L}$.

Note that for any path P in G , in particular for any s_i - t_i path, P has probability $\pi(P) = \prod_{e \in P} p_e = \prod_{e \in P} e^{-\ell_e} = e^{-\sum_{e \in P} \ell_e} = e^{-\ell(P)}$ in G' .

Assume that there are two edge-disjoint s_1 - t_1 and s_2 - t_2 paths P_1 and P_2 in G such that P_1 is a shortest s_1 - t_1 path ($\ell(P_1) = L$). Define s - t paths P'_1 and P'_2 in G' by appending the edges $\{s, s_i\}$ and $\{t_i, t\}$ to P_i for $i = 1, 2$, respectively. Then

$$\tau(P'_1, P'_2) = e^{-L} + (1 - e^{-L}) \cdot \varepsilon^2 e^{-\ell(P_2)} \geq e^{-L}.$$

On the other hand, consider two edge-disjoint s - t paths in G' with $\tau(P''_1, P''_2) \geq e^{-L}$. Without loss of generality, let P''_1 be the path that goes via s_1 and P''_2 be the path that goes via s_2 . Note that then P''_1 also goes via t_1 , because if it contained $\{t_2, t\}$, we would have

$$\tau(P_1, P_2) \leq \pi(P_1) + \pi(P_2) \leq 2\varepsilon < \prod_{e \in E} p_e \leq e^{-L}.$$

We conclude that P''_i contains edges $\{t_i, t\}$ for $i = 1, 2$.

We now show that the path P'_1 that we obtain from P''_1 by removing edges $\{s, s_1\}$ and $\{t_1, t\}$ corresponds to a path of length L (i.e., of minimum length) in G . Assume, for contradiction, that P'_1 was longer, i.e., $\ell(P'_1) \geq L + 1$. Then, we obtain

$$\begin{aligned}
\tau(P_1'', P_2'') &= \pi(P_2'') + (1 - \pi(P_2''))\pi(P_1'') = \varepsilon^2\pi(P_2') + (1 - \varepsilon^2\pi(P_2'))\pi(P_1') \\
&\leq \varepsilon^2\pi(P_2') + (1 - \varepsilon^2\pi(P_2'))e^{-(L+1)} \\
&= (\varepsilon^2\pi(P_2') \cdot e^{L+1} + 1 - \varepsilon^2\pi(P_2'))e^{-(L+1)} \\
&= (\varepsilon^2\pi(P_2') \cdot (e^{L+1} - 1) + 1)e^{-(L+1)} \\
&\leq (\varepsilon^2 \cdot (e^{L+1} - 1) + 1)e^{-(L+1)} \\
&< \left(\frac{e - 1}{e^{L+1} - 1} \cdot (e^{L+1} - 1) + 1 \right) e^{-1}e^{-L} \\
&\leq ee^{-1}e^{-L} = e^{-L}.
\end{aligned}$$

We can thus conclude that P_1 has length L , and by construction, P_2 (the path corresponding to P_2'') does not share an edge with P_1 . $\blacktriangleleft$

Note that the proof extends to the case where vertex-disjointness of the paths is required since in this case 2D1SPP is also NP-hard according to [13], and to uncertain graphs with edge-probabilities $p: E \rightarrow [0, 1]$.

4 Solution Approaches

In this section, we develop solution approaches for the MSDPP. These are based on an important insight: optimal solutions to the MSDPP are extreme-supported efficient solutions to a suitably defined multi-objective optimization problem. This perspective on the problem, explained in Section 4.1 inspires two exact, an approximation algorithm, and a heuristic approach, which we describe in Sections 4.2–4.5.

4.1 A vectorization approach to the MSDPP

We first explain our vectorization approach to a more general class of problems and prove its correctness in Theorem 3, before we apply it on the MSDPP in Theorem 4.

Consider an optimization problem

$$\begin{aligned}
&\text{maximize} && h(x) \\
&\text{subject to} && x \in X
\end{aligned} \tag{1}$$

where $X \subset \mathbb{R}^n$ is compact and h can be written as $f \circ g$ with f and g having the following properties: 1.) $g: X \rightarrow \mathbb{R}^p$ is continuous. 2.) There exists a compact and convex superset $Y' \supseteq Y = g(X)$, in which f is continuous, monotonously increasing in each argument, and quasi-convex, i.e.,

$$f((1 - \lambda)y + \lambda y') \leq \max\{f(y), f(y')\} \quad \forall y, y' \in Y' \quad \forall \lambda \in [0, 1]. \tag{2}$$

For this function g , we now consider the multi-objective optimization problem

$$\begin{aligned}
&\text{maximize} && g(x) = \begin{pmatrix} g_1(x) \\ \vdots \\ g_p(x) \end{pmatrix} \\
&\text{subject to} && x \in X.
\end{aligned} \tag{3}$$

In multi-objective optimization, a solution for x of Problem (3) is called *efficient* if there is no $x' \in X$ with $g_i(x) \leq g_i(x')$, for all $i \in \{1, \dots, p\}$, and $g_j(x) < g_j(x')$ for some $j \in \{1, \dots, p\}$. If, for an efficient solution x , there exists a $\lambda \in \mathbb{R}_{>0}^p = \{\lambda \in \mathbb{R}^p \mid \forall i \in \{1, \dots, p\}: \lambda_i > 0\}$ such that $x \in \operatorname{argmax}_{x' \in X} \sum_{i=1}^p \lambda_i g_i(x')$, then x is *supported efficient* for Problem (3). A supported efficient solution x of Problem (3) is *extreme-supported efficient* if there are no $\mu \in (0, 1)$ and no supported efficient solutions x_1, x_2 with $x \neq x_1, x \neq x_2$ such that $x = \mu x_1 + (1 - \mu)x_2$.

► **Theorem 3.** *Let x^* be an optimal solution for Problem (1) and denote by X_S the set of extreme-supported efficient solutions to the multi-objective optimization Problem (3). Then there exists a solution $x^S \in X_S$ to the Problem (3) with $h(x^*) = h(x^S)$. If f is strictly quasi-convex on Y' , i.e., if the inequality in (2) is strict, every optimal solution to (1) is extreme-supported efficient for Problem (3).*

Proof. Since X is non-empty and compact and g is continuous, for any $\lambda \in \mathbb{R}_{>0}^p$, the set $\operatorname{argmax}_{x' \in X} \sum_{i=1}^p \lambda_i g_i(x')$ is non-empty and compact and thus the set of extreme-supported non-dominated points, $Y_S := g(X_S)$, is also compact. Then $\operatorname{conv}(Y_S) - \mathbb{R}_{\geq}^p \supset Y$, because all supported non-dominated points $\operatorname{conv}(Y_S)$ are in the convex hull of the extreme-supported non-dominated points. That is, there is a finite set $\{y_1, \dots, y_q\} \subseteq Y_S$, non-negative numbers λ_i , where $i = 1, \dots, q$, with $\sum_{i=1}^q \lambda_i = 1$ and non-negative numbers μ_j , where $j = 1, \dots, q$, such that $g(x^*) = \sum_{i=1}^q \lambda_i y_i - \sum_{j=1}^p \mu_j e_j$ with e_j being the j -th unit vector in $\mathbb{R}^p$. We find that

$$\begin{aligned} h(x^*) &= f(g(x^*)) = f\left(\sum_{i=1}^q \lambda_i y_i - \sum_{j=1}^p \mu_j e_j\right) \leq f\left(\sum_{i=1}^q \lambda_i y_i\right) \leq \max_{i=1, \dots, q} f(y_i) \leq \max_{x \in X_S} f(g(x)) \\ &= \max_{x \in X_S} h(x) \end{aligned}$$

where the first inequality holds due to monotonicity of f on Y' (and $y \in Y \subset Y'$) and the second one due to quasi-convexity on Y' .

If f is strictly quasi-convex, then the second inequality is strict, unless x^* is extreme-supported efficient. ◀

► **Theorem 4.** *Denote by $P_{s,t}(G)$ the set of pairs of edge-disjoint s - t paths in G . Any optimal solution (P_1, P_2) to an instance (G, d, p, s, t) of MSDPP is an extreme-supported solution for the multi-objective optimization problem*

$$\begin{aligned} &\text{maximize} && \begin{pmatrix} \pi(P_1) \\ \pi(P_2) \end{pmatrix} \\ &\text{subject to} && (P_1, P_2)^T \in P_{s,t}(G) \end{aligned} \tag{4}$$

as well as for the multi-objective problem

$$\begin{aligned} &\text{maximize} && \begin{pmatrix} \sum_{e \in P_1} \ln p_e \\ \sum_{e \in P_2} \ln p_e \end{pmatrix} \\ &\text{subject to} && (P_1, P_2)^T \in P_{s,t}(G). \end{aligned} \tag{5}$$

Proof. Note that $P_{s,t}(G)$, the feasible set of both Problem (4) and MSDPP, is finite (and thus compact). To show the first statement, we write the objective function of MSDPP as $h = f \circ g$ with $g: P_{s,t}(G) \rightarrow [0, 1]^2$ with $g(P_1, P_2) = (\pi(P_1), \pi(P_2))^T$ and $f: \mathbb{R}^2 \rightarrow \mathbb{R}$ with $f(a, b) = a + b - a \cdot b$. Since $P_{s,t}(G)$ is finite, the continuity of g trivially follows. For the compact and convex set $Y' = [0, 1]^2 \supseteq g(P_{s,t}(G))$ we now show that f is monotonically increasing in each argument and quasi-convex on Y' .

For $d \in [0, 1]$ with $d > b$ it follows that

$$f(a, b) = a + (1 - a)b < a + (1 - a)d = f(a, d),$$

thus f is strictly monotonically increasing in the second argument. Since f is symmetric, it follows that f is strictly monotonically increasing in both arguments.

To show the quasi-convexity of f , we prove that the level sets $L_\alpha = \{(a, b)^T \in [0, 1]^2 \mid f(a, b) \leq \alpha\}$ are convex for any $\alpha \in [0, 1]$. This definition of quasi-convexity is equivalent to the one given above, see, e.g., [16]. Note that the sets L_α are convex if and only if the sets $L_\alpha - (1, 1)^T$ are convex. These can be rewritten as

$$L_\alpha - (1, 1)^T = \{(a-1, b-1)^T \in [0, 1]^2 \mid (1-a)(1-b) \geq 1-\alpha\} = \{(a, b)^T \in [0, 1]^2 \mid ab \geq 1-\alpha\}$$

We now show convexity of these sets: Consider $\lambda \in [0, 1]$ and $(a, b)^T, (c, d)^T \in [0, 1]^T$ with $ab \geq 1 - \alpha$ and $cd \geq 1 - \alpha$. Due to the inequality of arithmetic and geometric means, $(ad + cb)/2 \geq \sqrt{abcd}$ for non-negative real numbers a, b, c, d (see, e.g., Heuser [22]), it follows that

$$\begin{aligned} (\lambda a + (1 - \lambda)c)(\lambda b + (1 - \lambda)d) &= \lambda^2 \underbrace{ab}_{\geq 1-\alpha} + \lambda(1 - \lambda) \underbrace{(ad + cb)}_{\geq 2\sqrt{abcd} \geq 2(1-\alpha)} + (1 - \lambda)^2 \underbrace{cd}_{\geq 1-\alpha} \\ &\geq (1 - \alpha)(\lambda^2 + 2\lambda(1 - \lambda) + (1 - \lambda)^2) = (1 - \alpha)(\lambda + (1 - \lambda))^2 = 1 - \alpha. \end{aligned}$$

This proves that the sets $L_\alpha - (1, 1)^T$ and thus L_α are convex. Thus, all conditions of Theorem 3 are fulfilled and there exists an extreme-supported solution of Problem (4) that is an optimal solution of MSDPP.

For the second statement, we write the objective function of MSDPP as $h = (f \circ \psi) \circ (\phi \circ g)$ with f, g as defined above, $\psi: [\ln \pi_{\min}, 0]^2 \rightarrow [0, 1]^2$ with $\psi(a, b) \mapsto (e^a, e^b)^T$, and $\phi: [0, 1]^2 \rightarrow [-\infty, 0]^2$ with $\phi(a, b) \mapsto (\ln a, \ln b)^T$ where $\pi_{\min} > 0$ is the probability of the most-uncertain s - t path in G . Thus, $Y' = [\ln \pi_{\min}, 0]^2 \supset (\phi \circ g)(P_{s,t}(G))$ is a convex and compact superset. Since ψ_1, ψ_2 are strictly convex and monotonically increasing functions in Y' and f is strictly quasi-convex in $[0, 1]^2 \supseteq Y'$ and monotonically increasing in each argument, $f \circ \psi$ is monotonically increasing in each argument and strictly quasi-convex in Y' , compare Agrawal and Boyd [2] where this is shown for (not necessarily strict) quasi-convexity, the proof for strict quasi-convexity is analogous. Applying Theorem 3, it now follows that an extreme-supported solution of Problem (5) is an optimal solution to MSDPP. ◀

4.2 Path Iteration

In this section, we propose an algorithm that iterates through the simple s - t paths of a graph $G = (V, E)$ with given $s, t \in V$ in order of decreasing path probability. For each path P_1 , the algorithm computes a second most-certain s - t path P_2 that is edge-disjoint from P_1 , evaluating each solution found with respect to the objective function of MSDPP and keeping track of the best MSDPP solution found. It stops when the probability of the second path found exceeds the probability of the first. We show in Lemma 5 that in this way, the solutions (P_1, P_2) considered by the algorithm constitute a representative set of the efficient solutions from Problem (4) with $\pi(P_1) \geq \pi(P_2)$. Correctness then follows from Theorem 4 since the set of efficient solutions contains the extreme-supported efficient solutions.

The iterator for the simple s - t paths that are ordered by the probability of the paths used in Algorithm 1 is Yen's algorithm [43]. The second path can be found by computing the most-certain path in the graph $(V, E \setminus E(P))$ where P is the path of the current iteration. If

these two paths provide a solution to MSDPP with a higher success probability than the current best solution, the two paths are set as the new current best solution. The pseudocode for our approach is given in Algorithm 1. To compute the second path, we use Fanding's shortest paths faster algorithm [15] (SPFA) with an expected runtime of $\mathcal{O}(|E|)$.

■ **Algorithm 1** Path Iteration Algorithm.

Data: (Directed or undirected) uncertain graph $G = (V, E, p)$ start $s \in V$ and destination $t \in V$

Result: Two edge-disjoint s - t paths that maximize the success probability

Let $P_1^{\text{opt}}, P_2^{\text{opt}}$ be empty paths;

$\tau_{\text{opt}} = 0$;

foreach simple s - t -path P_1 in G ordered by $\pi(P_1)$ **do**

Compute most-certain s - t -path P_2 in $(V, E \setminus E(P_1))$ with SPFA;

if P_2 does not exist **then**

continue;

end

if $\pi(P_1) < \pi(P_2)$ **then**

return $(P_1^{\text{opt}}, P_2^{\text{opt}})$;

end

if $\tau(P_1, P_2) > \tau_{\text{opt}}$ **then**

$\tau_{\text{opt}} \leftarrow \tau(P_1, P_2)$;

$P_1^{\text{opt}}, P_2^{\text{opt}} \leftarrow P_1, P_2$;

end

end

Return $(P_1^{\text{opt}}, P_2^{\text{opt}})$;

► **Lemma 5.** Let (P_1, P_2) be an efficient solution for Problem (4) with $\pi(P_1) \geq \pi(P_2)$. Algorithm 1 considers a solution (P'_1, P'_2) with $\tau(P_1, P_2) = \tau(P'_1, P'_2)$.

Proof. Assume that the algorithm terminates on a solution $(\bar{P}_1, \bar{P}_2)$ before considering the solution (P_1, P_2) . Then $\pi(\bar{P}_2) > \pi(\bar{P}_1) \geq \pi(P_1) \geq \pi(P_2)$. Thus $(\bar{P}_1, \bar{P}_2)$ dominates (P_1, P_2) , which contradicts the assumption that (P_1, P_2) is efficient. ◀

4.3 Integer-Linear-Program-based Dichotomic Search

In our second solution approach, we solve MSDPP by finding a representative set of supported efficient solutions of Problem (5) and comparing their objective values. To this end, we write Problem (5) as a biobjective integer linear program and apply dichotomic search, where in each step we solve a weighted-sum-scalarization of both objective functions. Correctness of this approach is ensured by Theorem 4.

For a directed uncertain graph $G = (V, E^{\rightarrow})$, we use the integer linear programming formulation Problem (6) (with $\lambda_1 \geq \lambda_2 > 0$) for the weighted sum scalarization. Note that since the problem is symmetric, weights with $\lambda_2 > \lambda_1$ do not need to be considered. We denote by $\delta^+(v) = \{(u, w) \in E^{\rightarrow} \mid u = v\}$ the set of edges originating from $v \in V$ and by $\delta^-(v) = \{(u, w) \in E^{\rightarrow} \mid w = v\}$ the set of edges terminating in $v \in V$.

$$\underset{x}{\text{maximize}} \quad \lambda_1 \sum_{e \in E^{\rightarrow}} \ln p_e x_e^1 + \lambda_2 \sum_{e \in E^{\rightarrow}} \ln p_e x_e^2 \quad (6a)$$

subject to

$$\sum_{e \in \delta^+(v)} x_e^i - \sum_{e \in \delta^-(v)} x_e^i = \begin{cases} 1 & \text{if } v = s \\ -1 & \text{if } v = t \\ 0 & \text{else} \end{cases} \quad \forall v \in V \forall i \in \{1, 2\}, \quad (6b)$$

$$\begin{aligned} x_e^1 + x_e^2 &\leq 1 & \forall e \in E^\rightarrow, \\ x_e^i &\in \{0, 1\} & \forall e \in E^\rightarrow \forall i \in \{1, 2\} \end{aligned} \quad (6c)$$

Each path is encoded by the binary variables x_e^i , where an edge $e \in E^\rightarrow$ belongs to path $i \in \{1, 2\}$ if $x_e^i = 1$. The flow constraints (6b) ensure that the paths originate from s , terminate in t , and are actual paths. The constraints (6c) enforce that an edge in $E^\rightarrow$ is only taken once over both paths and in either direction. So, the constraints of Problem (6) encode $P_{s,t}(G)$. A solution x of Problem (6) then induces two edge-disjoint s - t paths P_1, P_2 in G with

$$\pi(P_1) = \prod_{e \in E^\rightarrow} p_e^{x_e^1} \text{ and } \pi(P_2) = \prod_{e \in E^\rightarrow} p_e^{x_e^2}.$$

If the graph G is undirected, exchange $E^\rightarrow$ with $E^{\leftrightarrow} = \{(u, v), (v, u) \mid \{u, v\} \in E^\rightarrow\}$ - replace the edges in G with two directed edges of opposite direction - and swap (6c) with the constraint

$$x_{(u,v)}^1 + x_{(v,u)}^1 + x_{(u,v)}^2 + x_{(v,u)}^2 \leq 1 \quad \forall \{u, v\} \in E^{\leftrightarrow}.$$

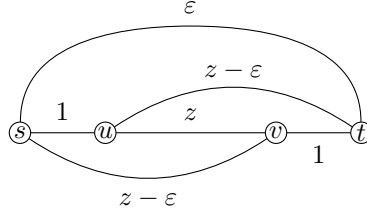
The dichotomic search starts with a lexicographic solution x_1^* and a solution x_2^* of Problem (6) with $\lambda_1 = \lambda_2$. From the paths P_1^1, P_1^2 and P_2^1, P_2^2 induced by x_1^* and x_2^* , compute an optimal solution x^* to Problem (6) with respect to $\lambda_1 = \pi(P_1^2) - \pi(P_2^2)$, $\lambda_2 = \pi(P_2^1) - \pi(P_1^1)$, which describes the slope between $(\pi(P_1^1), \pi(P_2^1))$ and $(\pi(P_1^2), \pi(P_2^2))$. If the objective value of x_1^*, x_2^* for Problem (6) with the new λ_1, λ_2 is smaller than the objective value of x^* for the same problem, then x^* is an extreme supported solution and we repeat this step for (x_1^*, x^*) and (x^*, x_2^*) .

Note that using a similar construction as in the proof of Theorem 2 we can show that solving Problem (6) is NP-hard for weights $\lambda_1 > \lambda_2$ by choosing a sufficiently small $\varepsilon < \min\{\prod_{e \in E} p_e, \sqrt{(1 - \lambda_1 e^{-1})e^{-L}/\lambda_2}\}$, which implies NP-hardness of Problem (6), though the special case $\lambda_1 = \lambda_2$ can be solved by Suurballe's algorithm [39] in polynomial time.

4.4 Approximation by lexicographically optimal solution

In this section and in the following one, we propose two approximation algorithms for the MSDPP. Instead of enumerating a large set of candidate solutions, namely a representative set of (extreme-supported) efficient solutions for Problem (4) or Problem (5), the underlying idea of our approximation algorithms is to find *a specific* extreme-supported efficient solution.

A *lexicographically optimal* solution to a bi-objective optimization problem is one that optimizes the second objective function among all solutions that maximize the first objective function. (In general, the role of "first" and "second" objective function may also be reversed. However, for our optimization problem, this is irrelevant, because the function f mapping the objective vector of the biobjective problem to the objective value of MSDPP is symmetric.) Note, however, that finding a lexicographically optimal solution to Problem (5) corresponds to solving the (2D1SPP) which is NP-hard [13], and this result can be easily transferred to Problem (4).



■ **Figure 1** Example graph that shows the tightness of the approximation guaranteed of Theorem 6.

Despite this theoretical complexity result, we argue that computing *just one* of the potentially exponentially many extreme-supported solutions of Problem (4) is considerably faster than computing a representative set of *all* extreme-supported efficient or even of all efficient solutions, as we do in the approaches proposed in Sections 4.2 and 4.3, respectively.

To compute a lexicographically optimal solution, we can modify the approach from Section 4.2, stopping as soon as the first feasible solution is found or apply integer programming, i.e., the first step of the dichotomic search from Section 4.3. We also refer to computing the lexicographically optimal solution as the *lexicographic* approach.

► **Theorem 6.** *Computing a lexicographically optimal solution of Problem (4) gives a tight approximation guarantee of $1/2$ for MSDPP.*

Proof. Let $(P_1^{\text{opt}}, P_2^{\text{opt}})$ be an optimal solution to MSDPP with $\pi(P_1^{\text{opt}}) \geq \pi(P_2^{\text{opt}})$ and let (P_1, P_2) be a lexicographically optimal solution to Problem (4). Then $\pi(P_1) \geq \pi(P_1^{\text{opt}})$ and $\pi(P_1) \geq \pi(P_2^{\text{opt}})$. It follows that

$$\begin{aligned} \frac{\tau(P_1, P_2)}{\tau(P_1^{\text{opt}}, P_2^{\text{opt}})} &= \frac{\pi(P_1) + (1 - \pi(P_1))\pi(P_2)}{\pi(P_1^{\text{opt}}) + \pi(P_2^{\text{opt}}) - \pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})} \geq \frac{\pi(P_1)}{\pi(P_1^{\text{opt}}) + \pi(P_2^{\text{opt}})} \\ &\geq \frac{\pi(P_1)}{2\pi(P_1)} = \frac{1}{2}. \end{aligned}$$

For the tightness consider the graph in Figure 1. The optimal solution with a success probability of $2(z - \varepsilon) - (z - \varepsilon)^2$ are the paths $s-v-t$ and $s-u-t$. The lexicographically optimal solution with the paths $s-u-v-t$ and $s-t$ has a success probability of $z + \varepsilon - z\varepsilon$. Taking the limit it follows that

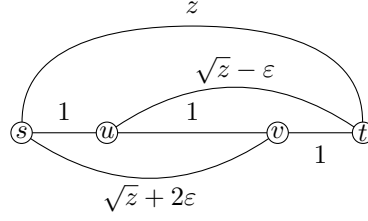
$$\lim_{z \rightarrow 0} \lim_{\varepsilon \rightarrow 0} \frac{z + \varepsilon - z\varepsilon}{2(z - \varepsilon) - (z - \varepsilon)^2} = \frac{1}{2}. \quad \blacktriangleleft$$

4.5 Suurballe's Algorithm based Heuristic

While the approach described in the previous section produces solutions consisting of paths of very different probabilities, we now investigate the worst-case performance of taking the solution (P_1, P_2) that minimizes Problem (6) with equal weights $\lambda_1 = \lambda_2 = 1$. This is *equivalent* to finding two edge-disjoint $s-t$ paths that maximize the product of the paths' probabilities. Analogously to the approach sketched in [10] for the *most-certain path* problem, one can show that this problem is equivalent to finding a solution that minimizes the sum of negative log-probabilities. This problem can be solved in polynomial-time with Suurballe's algorithm [39].

Unlike the approach described in Section 4.4, we cannot give a general bound on the approximation factor of the solution found by this approach. This is illustrated by the instance sketched in Figure 2, where the approximation factor approaches

$$\lim_{z \rightarrow 0} \lim_{\varepsilon \rightarrow 0} 2\sqrt{z} + \varepsilon - z - \varepsilon\sqrt{z} = 0.$$



■ **Figure 2** For $\varepsilon > 0$ and $z \in (0, 1 - 2\varepsilon + \varepsilon^2)$, Suurballe's algorithm gives the paths $s-u-t$ and $s-v-t$ with a success probability of $2\sqrt{z} + \varepsilon - z - \varepsilon\sqrt{z}$ and a multiplied path probability of $z + \varepsilon\sqrt{z}$. The optimal solution with a success probability 1 is given by $s-u-v-t$ and $s-t$ and has a multiplied path probability of z .

Note that in the constructed example, the probabilities of the two paths $(P_1^{\text{opt}}, P_2^{\text{opt}})$ in the optimal solution are very different, $\pi(P_1^{\text{opt}}) = 1$ while $\pi(P_2^{\text{opt}})$ approaches 0. On the other hand, intuitively, Suurballe's algorithm should provide good results for instances where the probabilities of the two paths $(P_1^{\text{opt}}, P_2^{\text{opt}})$ are (almost) equal. In Theorem 7 we see that indeed, the approximation guarantee can be bounded depending on (an upper bound on) the probability of the more certain path and (a lower bound on) the probability of the less certain path.

► **Theorem 7.** *Let $(P_1^{\text{opt}}, P_2^{\text{opt}})$ be an optimal solution to MSDPP with $\pi(P_1^{\text{opt}}) \geq \pi(P_2^{\text{opt}})$ and let $\alpha, \beta \in [0, 1]$ with $\beta \geq \pi(P_1) \geq \pi(P_2) \geq \alpha$. Then, for any solution (P_1, P_2) that is optimal for Problem (5) with $\lambda_1 = \lambda_2 = 1$ we have that*

$$\frac{\tau(P_1, P_2)}{\tau(P_1^{\text{opt}}, P_2^{\text{opt}})} \geq \frac{2\sqrt{\alpha\beta} - \alpha\beta}{\alpha + \beta - \alpha\beta}.$$

Proof. W.l.o.g. assume that $\pi(P_1) \geq \pi(P_2)$ and note that by definition $\pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}}) \leq \pi(P_1)\pi(P_2)$ and $\tau(P_1^{\text{opt}}, P_2^{\text{opt}}) \geq \tau(P_1, P_2)$. This implies that $\pi(P_1^{\text{opt}}) \geq \pi(P_1) \geq \pi(P_2) \geq \pi(P_2^{\text{opt}})$.

Using the fact that the function $f(a, b) = a + b - a \cdot b$ attains its minimum on $\{(a, b)^T \in [0, 1]^2 : ab \geq \pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})\}$ at $a = b = \sqrt{\pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})}$ (see Lemma 8 in the appendix), we conclude that $\tau(P_1, P_2) \geq \sqrt{\pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})} - \pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})$.

$$\begin{aligned} \frac{\tau(P_1, P_2)}{\tau(P_1^{\text{opt}}, P_2^{\text{opt}})} &\geq \frac{2\sqrt{\pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})} - \pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})}{\pi(P_1^{\text{opt}}) + \pi(P_2^{\text{opt}}) - \pi(P_1^{\text{opt}})\pi(P_2^{\text{opt}})} \\ &\stackrel{\text{Lemma 10}}{\geq} \frac{2\sqrt{\beta\pi(P_2^{\text{opt}})} - \beta\pi(P_2^{\text{opt}})}{\beta + \pi(P_2^{\text{opt}}) - \beta\pi(P_2^{\text{opt}})} \stackrel{\text{Lemma 11}}{\geq} \frac{2\sqrt{\alpha\beta} - \alpha\beta}{\alpha + \beta - \alpha\beta} \end{aligned}$$

This gives the approximation factor. ◀

5 Experimental Results

After we describe the benchmark instances and the implementation of the presented algorithms in Section 5.1, we compare the average runtime of the algorithms in Section 5.2. Lastly, we compare the success probabilities of the approximation algorithms against those of the exact algorithms in Section 5.3.

5.1 Benchmark Instances and Implementation Details

We consider four classes of benchmark graphs: GRID, GRIDLOW, GRIDHIGH, and COASTAL. The base class GRID consists of grid-graphs with 5 columns and $2k$ rows with $k \in \mathbb{N}$ and edge probabilities are uniformly sampled from $[0, 1]$. The classes GRIDLOW and GRIDHIGH are identical to GRID but the edge probabilities are uniformly sampled from $[0.1, 0.35]$ and $[0.75, 1]$, respectively. Graphs from the COASTAL class are metric planar graphs designed to resemble networks in coastal regions. A detailed description of this graph class is provided in [31]. Each graph has $5 \cdot 2k$ nodes and up to $3(5 \cdot 2k)$ edges. The probability of an edge in a COASTAL graph is given by $p \cdot (\ell_{\max} - \ell_e) / (\ell_{\max} - \ell_{\min})$ where p is uniformly sampled from $[0, 1]$ for each edge, ℓ_e is the length of the edge e , and $\ell_{\min}, \ell_{\max}$ are the lengths of the shortest and longest edge of the graph instance, respectively. For each graph class and $k \in \{2, 4, \dots, 100\}$, 10 different graphs were generated. Standalone implementations of the graph generation process are available at [30]. Due to the inefficiency of our COASTALgraph generation process, we limit the number of nodes for these instance to 300. For each graph, we choose a vertex with the lowest x -coordinate, and, if there are several with the same x -coordinate, then take the one with the lowest y -coordinate as start vertex s . To set an end vertex t we perform a breadth-first search in the graph starting from s and take the last vertex found. To ensure that there are two edge-disjoint s - t paths, a direct s - t edge with the smallest probability greater than zero that a 64-bit float value can represent is added.

The algorithms described in the paper are implemented in Rust [40] and are provided in [29]. To improve the runtime and the memory usage of Algorithm 1, only the paths whose probability could improve the current best found solution are kept in the internal heap data structure. For the approximation algorithm described in Section 4.4, we solve Problem (6) with $\lambda_1 = 1$ and a sufficiently small λ_2 since this proved to be considerably faster than applying a modified version of the path iteration algorithm, as proposed in Section 4.4. Problem (6) is solved using Gurobi 13.0.2 [20]. For Suurballe’s algorithm, we used Fanding’s SPFA [15] with an expected $\mathcal{O}(|E|)$ runtime instead of Dijkstra’s algorithm.

For each instance and algorithm, we set a time limit of 60 seconds. The computational experiments were executed on 8 threads of an AMD Epic 9654 with 32GB of RAM on Debian 13.

5.2 Runtime Analysis

■ **Table 2** Number of solved instances by the path iteration algorithms from Section 4.2 per graph class.

	GRID	GRIDHIGH	GRIDLOW	COASTAL
Solved	183 / 500	117 / 500	492 / 500	116 / 150

The path iteration could not solve all considered instance within the time limit expect for the GRIDLOW graphs as we can see in Table 2. Here, we can see that GRIDLOW instances can be solved more often than GRID, GRIDLOW and COASTAL instances. We suspect that comparatively low edge probabilities make it easier for the implemented approach to follow high probability paths.

Furthermore, only for the GRIDLOW instances the path iteration algorithm is faster for graphs with less than 600 nodes than the ILP-based dichotomic search; see Figure 3. For the other graph classes, around 180 nodes the path iteration becomes slower and begins to reach

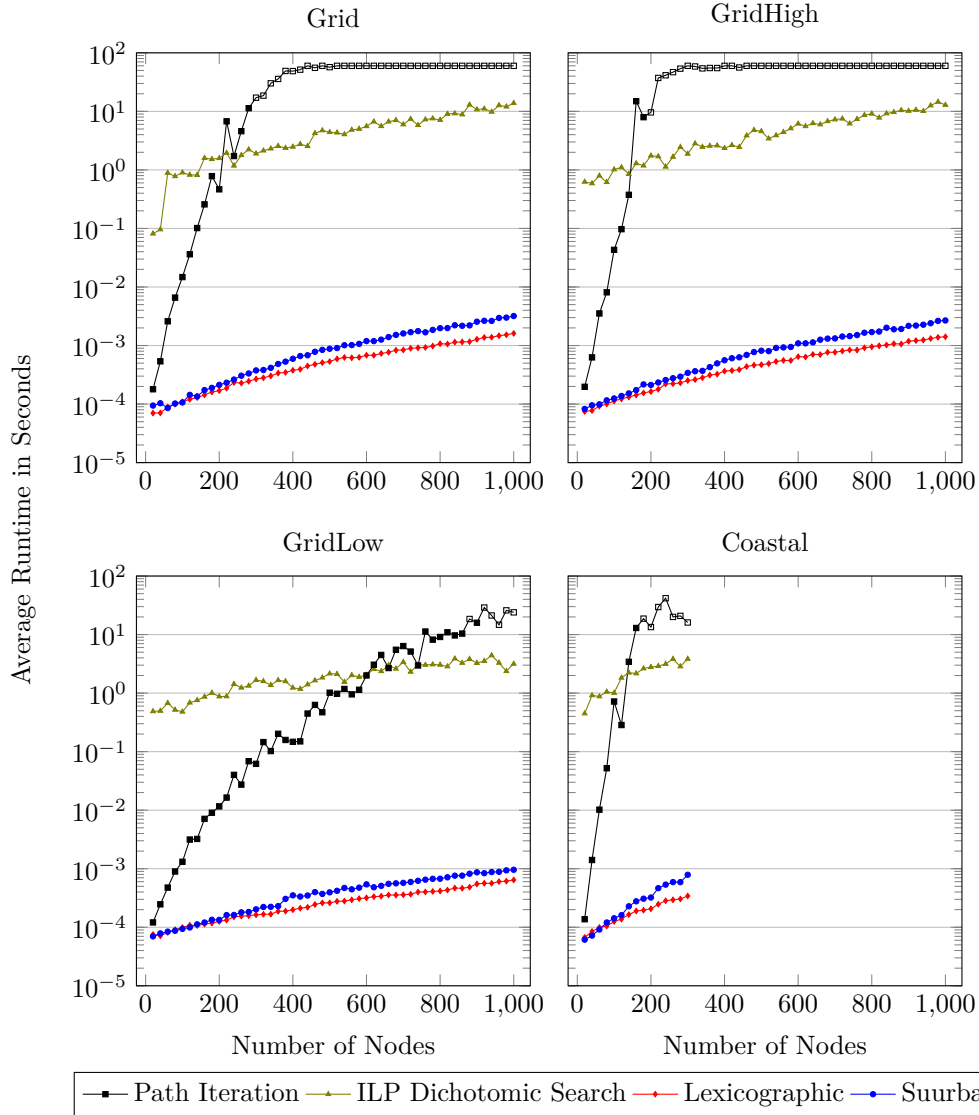


Figure 3 Average runtime of the algorithms over the graph classes. Unfilled markers indicate that not all instances were solved to optimality within the time limit.

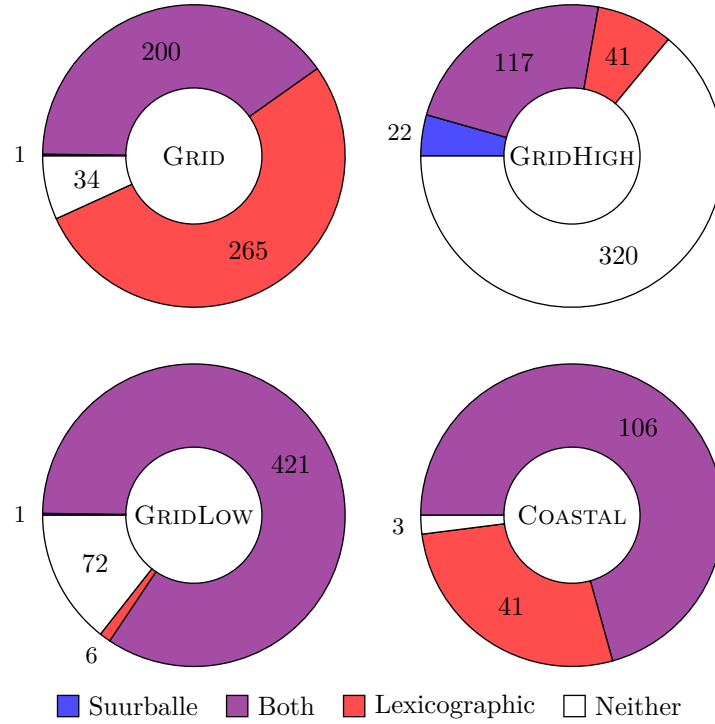
the time limit. Both the approximation algorithms show near identical runtimes. Even if the lexicographic algorithm solves 2D1SPP, which is NP-hard, the low runtime of the path iteration based lexicographic algorithm implies that the first considered most-certain s - t path allows for a second edge-disjoint s - t path. Thus, the lexicographic algorithm solves as many most-certain paths problems as Suurballe's algorithm, which has a higher computational overhead.

5.3 Heuristics' Quality Analysis

For the analysis of the solution quality of the approximation algorithm and the heuristic, consider Table 3 and Figure 4. In the first row of Table 3, we see for how many of the instances the approximation approaches found an optimal solution. In the second to fourth rows we see average, the 5th percentile - the best of the worst 5% -, and worst observed value of the approximation ratio ALG/OPT rounded to two decimal places.

■ **Table 3** Comparison of the solution quality of the Suurballe (Suur.) and lexicographic (Lexico.) approximation algorithms.

	GRID		GRIDHIGH		GRIDLOW		COASTAL	
	Suur.	Lexico.	Suur.	Lexico.	Suur.	Lexico.	Suur.	Lexico.
Found optimum	201	456	139	158	422	427	106	147
Average	0.73	1.00	0.96	0.98	0.98	0.99	0.93	1.00
5th percentile	0.16	1.00	0.84	0.92	0.93	0.93	0.51	1.00
Worst observed	0.02	0.78	0.67	0.87	0.20	0.58	0.06	0.90



■ **Figure 4** Number of optimally solved instances per graph class for the approximation approaches.

The pie charts in Figure 4 show the amount of instances for which each and both approximation approaches found an optimal solution.

For both approaches, we remark that they found the optimal solutions in remarkably many cases. As Figure 4 confirms, for a considerable proportion of the instances, the lexicographic and Suurballe's solution are identical and optimal. This also shows in the average approximation ratios in Table 3, which are almost everywhere close to 1.00.

Overall, the lexicographic algorithm is better than Suurballe's algorithm for all our considered metrics, especially in the 5th percentile and worst observed for GRID and COASTAL.

6 Conclusions and future research

In this paper, we studied the NP-hard problem MSDPP, i.e., the problem to find two edge-disjoint paths between a start and an end node in a graph equipped with edge traversal properties that maximize the success probability. We have shown that the problem is NP-hard and presented four solution algorithms, which are all motivated by the property that optimal

solutions to the MSDPP are extreme supported efficient solutions to a suitably defined bi-objective optimization problem. Regarding exact approaches, we find that the ILP-based dichotomic search approach outperforms the *path iteration* algorithm for larger graphs. When it comes to the approximation algorithm and the heuristic, the lexicographic approximation algorithms proved better than Suurballe's algorithm. It would be interesting to see whether the runtime of the ILP-based dichotomic search approach could be improved by warm-starting it with the result of one of the approximation algorithms. Regarding the hardness results and the proposed algorithms, it would be interesting to investigate to what extent results extend to the case of finding two vertex-disjoint maximum-success paths.

Where this paper focused purely on algorithmic problem aspects, in future research we plan to focus on solving real-world instances, where edge probabilities are derived, e.g. from air surveillance data. It would be interesting to see whether our algorithms perform well on these instances (which are expected to differ considerably from our current test instances) as well, and how the results we obtain compare to the results of other routing algorithms designed for such applications, or, in general routing under uncertainty.

References

- 1 Birce Adsanver, Burcu Balcik, V. Bélanger, and M.-E. Rancourt. Operations research approaches for improving coordination, cooperation, and collaboration in humanitarian relief chains: A framework and literature review. *European Journal of Operational Research*, 319(2):384–398, 2024. doi:10.1016/j.ejor.2023.11.031.
- 2 Akshay Agrawal and Stephen Boyd. Disciplined quasiconvex programming. *Optimization Letters*, 14:1643–1657, 2020. doi:10.1007/s11590-020-01561-8.
- 3 Ana Maria Anaya-Arenas, Jacques Renaud, and Angel Ruiz. Relief distribution networks: a systematic review. *Annals of Operations Research*, 223:53–79, 2014. doi:10.1007/s10479-014-1581-y.
- 4 Amotz Bar-Noy and Baruch Schieber. The Canadian Traveller Problem. In *SODA*, volume 91, pages 261–270, 1991. doi:10.5555/127787.127835.
- 5 Kristóf Bérczi and Yusuke Kobayashi. The Directed Disjoint Shortest Paths Problem. In *25th Annual European Symposium on Algorithms (ESA 2017)*, volume 87 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:13, 2017. doi:10.4230/LIPIcs.ESA.2017.13.
- 6 Ramesh Bhandari. Optimal Physical Diversity Algorithms and Survivable Networks. In *Proceedings of the Second IEEE Symposium on Computers and Communications (ISCC 1997)*, pages 433–441, 1997. doi:10.1109/ISCC.1997.616037.
- 7 Andreas Björklund and Thore Husfeldt. Shortest Two Disjoint Paths in Polynomial Time. In *Automata, Languages, and Programming (ICALP)*, pages 211–222, 2014. doi:10.1007/978-3-662-43948-7_18.
- 8 Christina Büsing. Recoverable robust shortest path problems. *Networks*, 59(1):181–189, 2012. doi:10.1002/net.20487.
- 9 André Chassein, Trivikram Dokka, and Marc Goerigk. Algorithms and uncertainty sets for data-driven robust shortest path problems. *European Journal of Operational Research*, 274(2):671–686, 2019. doi:10.1016/j.ejor.2018.10.006.
- 10 Nicos Christofides. *Graph theory: An algorithmic approach*. Academic Press, Inc., 1975.
- 11 Edsger W. Dijkstra. A Note on Two Problems in Connexion with Graphs. *Numerische Mathematik*, 1:269–271, 1959. doi:10.1007/BF01386390.
- 12 Matthias Ehrgott. *Multicriteria Optimization*. Springer, 2005. doi:10.1007/3-540-27659-9_9.
- 13 Tali Eilam-Tzoref. The disjoint shortest paths problem. *Discrete Applied Mathematics*, 85(2):113–138, 1998. doi:10.1016/S0166-218X(97)00121-2.

- 14 Shimon Even, Alon Itai, and Adi Shamir. On the Complexity of Timetable and Multicommodity Flow Problems. *SIAM Journal on Computing*, 5(4):691–703, 1976. doi:10.1137/0205048.
- 15 Duan Fanding. A faster algorithm for shortest-path-SPFA. *Journal of Southwest Jiaotong University*, 2(9):2, 1994.
- 16 Werner Fenchel. *Convex cones, sets, and functions*. Lecture Notes, Princeton University, Department of Mathematics, 1953.
- 17 Steven Fortune, John Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, 10(2):111–121, 1980. doi:10.1016/0304-3975(80)90009-2.
- 18 Daniel Golovin, Vineet Goyal, Valentin Polishchuk, R. Ravi, and Mikko Sysikaski. Improved approximations for two-stage min-cut and shortest path problems under uncertainty. *Mathematical Programming*, 149(1):167–194, 2015. doi:10.1007/s10107-013-0742-0.
- 19 Marinus Gottschau, Marcus Kaiser, and Clara Waldmann. The undirected two disjoint shortest paths problem. *Operations Research Letters*, 47(1):70–75, 2019. doi:10.1016/J.ORL.2018.11.011.
- 20 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2026. URL: <https://www.gurobi.com>.
- 21 Damjan Hatic, Markus Rauhut, Vladyslav Polushko, Marco Codastefano, Sophia Rosa, Patrick McKay, Francesco Stompanato, Antonio Belezza, and Hans Hagen. Disaster impact analysis in Africa: EDDA-UAV building damage dataset of rural and urban Mozambique. In *Earth Resources and Environmental Remote Sensing/GIS Applications XIV*, volume 12734, pages 153–166, 2023. doi:10.1117/12.2683882.
- 22 Harro Heuser. *Lehrbuch der Analysis*. Springer-Verlag, 2013.
- 23 Dan A. Iancu and Nikolaos Trichakis. Pareto Efficiency in Robust Optimization. *Management Science*, 60(1):130–147, 2013. doi:10.1287/mnsc.2013.1753.
- 24 Aubin Jarry and Stéphane Pérennes. Disjoint paths in symmetric digraphs. *Discrete Applied Mathematics*, 157(1):90–97, 2009. doi:10.1016/j.dam.2008.04.024.
- 25 Richard M Karp. Reducibility Among Combinatorial Problems. *Complexity of Computer Computations*, pages 85–103, 1972. doi:10.1007/978-3-540-68279-0_8.
- 26 Yusuke Kobayashi and Ryo Sako. Two disjoint shortest paths problem with non-negative edge length. *Operations Research Letters*, 47(1):66–69, 2019. doi:10.1016/J.ORL.2018.11.012.
- 27 Tanmoy Kundu, Jiu-Biing Sheu, and Hsin-Tsz Kuo. Emergency logistics management—Review and propositions for future research. *Transportation research part E: logistics and transportation review*, 164:102789, 2022. doi:10.1016/j.tre.2022.102789.
- 28 Roberto Montemanni and Luca Maria Gambardella. The robust shortest path problem with interval data via Benders decomposition. *4OR*, 3(4):315–328, 2005. doi:10.1007/s10288-005-0066-x.
- 29 Aaron Neugebauer. Finding Maximum-Success Disjoint Paths, 2026. Version 1.0.0. URL: <https://github.com/PantomInach/uncertain-road-network-traversal-strategies>.
- 30 Aaron Neugebauer. Gengraph: A Graph Generation Library, 2026. Version 1.2.0. URL: <https://github.com/PantomInach/gengraph>.
- 31 Aaron Neugebauer, Alena Otto, and Marie Schmidt. The relief distribution problem with trucks and drones under incomplete demand information. *arXiv*, 2026. doi:10.48550/arXiv.2603.23135.
- 32 Yu Marco Nie and Xing Wu. Shortest path problem considering on-time arrival probability. *Transportation Research Part B: Methodological*, 43(6):597–613, 2009. doi:10.1016/j.trb.2009.01.008.
- 33 Evdokia Nikolova. High-Performance Heuristics for Optimization in Stochastic Traffic Engineering Problems. In *Large-Scale Scientific Computing*, pages 352–360, 2009. doi:10.1007/978-3-642-12535-5_41.

- 34 Evdokia Nikolova. Approximation Algorithms for Reliable Stochastic Combinatorial Optimization. In *Approximation, Randomization, and Combinatorial Optimization*, pages 338–351. Springer, 2010. doi:10.1007/978-3-642-15369-3_26.
- 35 Evdokia Nikolova and David R. Karger. Route Planning under Uncertainty: The Canadian Traveller Problem. In *AAAI Conferece on Artificial Intelligence*, volume 23, pages 969–974, 2008. doi:10.5555/1620163.1620223.
- 36 Evdokia Nikolova, Jonathan A. Kelner, Matthew Brand, and Michael Mitzenmacher. Stochastic Shortest Paths Via Quasi-convex Maximization. In *Algorithms - ESA*, pages 552–563, 2006. doi:10.1007/11841036_50.
- 37 Vladyslav Polushko, Alexander Jenal, Jens Bongartz, Immanuel Weber, Damjan Hatic, Ronald Rösch, Thomas März, Markus Rauhut, and Andreas Weinmann. Blessemflood21: Advancing Flood Analysis with a High-Resolution Georeferenced Dataset for Humanitarian Aid Support. In *IGARSS 2024-2024 IEEE International Geoscience and Remote Sensing Symposium*, pages 368–373, 2024. doi:10.1109/IGARSS53475.2024.10642128.
- 38 Neil Robertson and Paul D. Seymour. Graph Minors .XIII. The Disjoint Paths Problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995. doi:10.1006/JCTB.1995.1006.
- 39 John W. Suurballe. Disjoint paths in a network. *Networks*, 4(2):125–145, 1974. doi:10.1002/NET.3230040204.
- 40 The Rust Team. The Rust Programming Language, 2020. URL: <https://rust-lang.org>.
- 41 Torsten Tholey. Solving the 2-Disjoint Paths Problem in Nearly Linear Time. *Theory of Computing Systems*, 39(1):51–78, 2006. doi:10.1007/S00224-005-1256-9.
- 42 Stephan Westphal. A note on the k-Canadian Traveller Problem. *Information Processing Letters*, 106(3):87–89, 2008. doi:10.1016/j.ipl.2007.10.004.
- 43 Jin Y. Yen. An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, 27(4):526–530, 1970. doi:10.1090/qam/253822.

A

 Appendix

► **Lemma 8.** Let $M \in (0, 1]$. The function $f(a, b) = a + b - a \cdot b$ takes its minimum on $\{(a, b)^T \in [0, 1]^2 : ab \geq M\}$ at $a = b = \sqrt{M}$.

Proof. Since $f(a, b)$ is strictly monotonously increasing in a and b , the minimum can only occur if $ab = M$. Furthermore, with $\tilde{f}(b) := f(\frac{M}{b}, b)$ we have

$$\frac{\partial \tilde{f}}{\partial b}(b) = 1 - \frac{M}{b^2}.$$

Thus, $b = \sqrt{M}$ is the only relevant critical point and

$$\frac{\partial^2 \tilde{f}}{\partial b^2}(\sqrt{M}) = \frac{2}{\sqrt{M}} > 0,$$

therefore this is indeed a minimum. ◀

► **Lemma 9.** For $1 \geq a \geq b > 0$, it holds that $a + b\sqrt{ab} \geq ab + b$.

Proof. Let $a = bk$ with $k \geq 1$.

$$a + b\sqrt{ab} = bk + b\sqrt{b^2k} = b(\underbrace{k}_{\geq 1} + bk) + b^2(\underbrace{\sqrt{k} - k}_{\leq 0}) > b(bk + 1) = b^2k + b = ab + b. \quad \blacktriangleleft$$

8:18 Finding Maximum-Success Disjoint Paths

► **Lemma 10.** For $b \in (0, 1)$, the function f defined by

$$f(a) = \frac{2\sqrt{ab} - ab}{a + b - ab}$$

is monotonously decreasing on $[0, b]$.

Proof.

$$\frac{\partial}{\partial a} f(a) = \frac{b(b - a + ab - b\sqrt{ab})}{\sqrt{ab}(a + b - ab)^2}$$

The function is monotonously decreasing if $\frac{\partial}{\partial a} f(a) \leq 0$ which is equivalent to $b - a + ab - b\sqrt{ab} \leq 0$. This inequality is true due to Lemma 9. ◀

► **Lemma 11.** For $a \in (0, 1]$, the function f defined by

$$f(a) = \frac{2\sqrt{ab} - ab}{a + b - ab}$$

is monotonously decreasing on $[a, 1]$.

Proof. Analogous to the proof of Lemma 10. ◀