IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

-

Digital Object Identifier -

Solving the Safest-Shortest Path Problem under Correlated Failures in Backbone Networks with Ant Colony Optimization

ZOLTÁN TASNÁDI[1], BALÁZS VASS[1,2], AND NOÉMI GASKÓ[1]
[1]Faculty of Computer Science, Babeș-Bolyai University, Cluj-Napoca, Romania
[2]Dept. of Telecommunications and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary

Corresponding author: Zoltán Tasnádi (e-mail: zoltan.tasnadi@ubbcluj.ro).

This work was supported by a grant of the Ministry of Research, Innovation and Digitization, CNCS-UEFISCDI, project number PN-IV-P2-2.1-TE-2023-1977.

ABSTRACT

Computing a safest path in the presence of multiple correlated failures is a computationally demanding problem with numerous practical applications. In communication backbone networks, for instance, identifying the safest possible path between two endpoints is essential to meeting the stringent availability requirements of emerging technologies such as autonomous driving, AR/VR applications, and telesurgery. Many of these applications on the other hand, rely on low latencies, that can be ensured by limiting the path lengths between communication endpoints. Whereas previous papers optimized solely for the availability, this paper tackles the naturally arising bi-objective variant of the safest path problem: one of the objectives is to maximize the availability of a path, and the other objective is to minimize its length. After demonstrating the computational hardness of the problem, to address it, we design the Safest-Shortest Path Ant Colony Optimization (S^2P-ACO) algorithm, which is inspired by the Max–Min Ant System framework. A thorough evaluation of the efficiency of the S^2P-ACO algorithm in providing the Pareto-optimal solutions is made via comparing it with selected baseline algorithms. Numerical experiments were conducted on real-world internet backbone network topologies coupled with correlated network element failure probabilities distilled from real seismic hazard data. Overall, out of the 90,520 test cases where the optimal Pareto front is known, S^2P-ACO recovered it in 96.68% against 62.12% for the weighted-sum method and 87.92% for the random_walk algorithm with runtimes rarely exceeding 10 seconds per node pair.

INDEX TERMS Ant Colony Optimization; bi-objective safest-shortest path; correlated failures; safest path problem

I. INTRODUCTION

Several network-related computational problems are of increasing interest in recent years due to their large application possibilities in several fields, for example influence maximization in viral marketing [1] or motif detection in biological networks [2]. Among these, studying the wide family shortest-path-related problems [3] is crucial for optimal resource use, efficiency and speed in real-world networks. A well-studied classical variant of the problem is to find a shortest path between two nodes s and t of a given graph G, with base variants of this problem enabling computationally efficient solutions (e.g., for nonnegative weights, Dijkstra algorithm [4] can be used, or for networks containing negative weights the Bellman-Ford [5] algorithm, for improved theoretical runtime complexities, see e.g., [6]–[8]).

From a computational viewpoint, the shortest paths and safest paths problems are closely interconnected under the critical assumption that the single network element failures are independent of each other [9] – using this assumption, there is a traditional trick (explained in the paper) that enables safest paths problems to be solved as *shortest* path problems. However, in the real world, network element failures are correlated [10]. It is already known that with correlated failures, the problem becomes $\mathcal{NP}$-hard and even inapproximable in polynomial time (unless $\mathcal{P}$=$\mathcal{NP}$) [11]. This computational hardness sets the stage for studying Ant Colony Optimization (ACO) Algorithms for (the single-objective version) of the problem. Notably, a recent paper [12] designed the so-called

SP-ACO (Safest Path ACO) algorithm.

Although its results are promising, [12] left open the question of designing efficient algorithms for providing paths in between a given node pair s and t that are Pareto-optimal in terms of i) availability and ii) length. We argue that the design and study of this problem setting is of high importance since, e.g., in the realm of telecommunications, many mission-critical services rely on uninterrupted network connectivity, whose robustness is often quantified by the Quality of Resilience (QoR) [13]. Examples include telesurgery and high-frequency trading systems, where extremely high availabilities and low latencies are required. However, most operational networks are designed to tolerate only single link or node failures [14], and occasionally dual failures [15], which is often insufficient for highly critical applications. To address these requirements, communication services are typically governed by Service Level Agreements (SLAs) that specify Quality of Service (QoS) parameters such as service availability and delay [16]–[18]. Violations of the agreed availability may incur substantial financial penalties for the communication service provider (CSP) [19]. Consequently, CSPs must carefully estimate and improve the availability of their services, especially for the primary end-to-end paths carrying customer traffic. This requires accounting for both the availability and length of end-to-end paths. Note that additional length translates to additional delay: unnecessary detours not only add to the financial and ecological footprint of data transfer, but they also increase transfer times. Although light in optical fiber is extremely fast, one has to bear this in mind, since in the era of ultra-low latency communications, many applications are underpinned by latencies kept carefully low in the backbone [20].

Based on the above, we identify the need for a bi-objective version of the SP-ACO algorithm proposed by [12], where along the availability of the paths, we treat the length of the paths as an additional objective on board. In line with this, the main goal of the paper is two-fold. First, we formally define the bi-objective safest-shortest path problem, where the first objective is the 'traditional' maximization problem of the availability of an st-path in the presence of correlated failures, while the other objective is the path length, which needs to be minimized. This newly introduced problem can be more realistic in the real world: it is straightforward to take into account not only the quality of the paths but also their length. As a second goal, we design a bi-objective Ant Colony Optimization Algorithm to tackle the problem at hand and demonstrate its effectiveness on real-world problem inputs.

The main contributions of the paper are as follows:

- we formalize and prove the $\mathcal{NP}$-hardness and inapproximability of the safest-shortest path problem in presence of correlated failures, while as a second objective, the path length is considered;
- we propose the Safest-Shortest Path Ant Colony Optimization algorithm (S^2P-ACO) tackling the problem;
- for comparisons we select three baseline algorithms: an exact algorithm (having scalability limitations), a naive algorithm based on Dijkstra algorithm, and a random walk algorithm, where ants randomly choose the next node;
- through extensive simulations on real-world inputs, we show the effectiveness of the resulting S^2P-ACO algorithm in determining the Pareto-optimal paths.

The rest of the paper is organized in the following way: in Sec. II related work is outlined, while Sec. III formally defines the bi-objective variant of the problem and presents results on its computational hardness. In Sec. IV, the proposed Safest-Shortest Path Ant Colony Optimization algorithm (S^2P-ACO) is described. Numerical experiments conducted on real-world problems are presented in Sec. V. Sec. VI concludes the paper.

II. RELATED WORKS

In this Section, we briefly survey the related works of the two most important sub-fields this paper relates to. Namely, Sec. II-A discusses previous works related to telecom network resiliency, that is the practical application we are concentrating on; and Sec. II-B surveys the related Ant Colony Optimization algorithms that, in a sense, offer the solution to our practical problem.

A. RELIABLE PATHS WITH LOW LATENCY IN BACKBONE NETWORKS

Computing maximally reliable (or 'safest') paths and quantifying end-to-end availability between two backbone network nodes under the assumption of independent single-component failures has been studied for decades; see, for instance, classical contributions on backbone protection and survivability planning [21]–[26]. Within the communication-network setting, the treatment of multiple, potentially correlated failures is traditionally grounded in the concept of Shared Risk Groups (SRGs) (e.g., [21]–[24], [27]–[29]). An SRG captures a collection of network elements that are likely to fail simultaneously, for example, several links routed along the same bridge or conduit. In numerous practical scenarios, the failure of a node v is equivalent to the simultaneous failure of all links incident to v. In such cases, it suffices to model only Shared Risk Link Groups (SRLGs), each consisting exclusively of links. Probabilistic generalizations of the SRG framework have also been explored [30]–[34]. In particular, [34] introduced a unified and streamlined terminology for Probabilistic SRLGs (PSRLGs), which we adopt in the present work.

A modern modeling strategy, followed in this paper as well, is to treat disaster scenarios as explicit input [35], typically derived from prior preprocessing based on historical hazard data or dedicated risk-assessment tools. Earlier works have instead treated disaster modeling as a secondary problem, thus in a more heuristic fashion within network planning tasks. Representative directions included identifying the most vulnerable substructures of a network [20], [36]–[38], estimating expected damage under random disaster events [39]–

[41], adapting routing decisions to mitigate disaster impact [42], [43], and designing or expanding networks with resilience considerations [44]–[47].

More recently, scalable techniques have been developed for efficiently computing and storing correlated link-failure information [34], moving beyond approaches that either restrict attention to a small set of disaster scenarios or assume independence among link failures [48]–[50]. These methods have already been integrated into comprehensive disaster-resilience toolchains [10], [51].

From an algorithmic standpoint, multiple problem variants and solution paradigms have been investigated. For example, [52] introduces two models of correlated link weights (where weights may encode metrics such as delay or failure probability) and studies shortest-path computation via adaptations of Dijkstra's algorithm. In [53], correlated failures are characterized and the k-shortest path problem is addressed under this setting. The work [54] considers single and double link failures and formulates the problem as an Integer Quadratic Program. Overall, explicitly accounting for link correlations within shortest-path computations can substantially influence both the reliability and performance of communication networks.

While often seemingly overlooked, keeping the length of the resulting communication paths in backbone networks under control is also crucial, as discussed in the introduction. We believe the absence of principled studies on the bi-objective optimization problem tackled by this paper is at least partly due to the fact of its relative complexity, and since up until very recently, studies tackled its simpler versions or components.

B. ANT COLONY OPTIMIZATION

Nature-inspired algorithms [55] have been successfully used to solve hard optimization problems (see [56]–[58]). One of the most popular nature-inspired algorithms is the Ant Colony Optimization Algorithm (ACO). First, the Ant Systems (AS) were introduced [59], [60] to solve the Traveling Salesman Problem. The Ant Colony Systems (ACS) [61], [62] came later, which modified the pheromone update and transition rule. There exist several other variants of ACOs, to mention only a few of them: Max-Min Ant Systems (MMAS) [63], Recursive Ant Colony Optimization [64], or Cunning Ant Systems (cAS) [65].

ACOs were first designed to solve discrete network-based problems, but they proved to be useful for continuous problems [66], and for multi-objective optimization problems, as well.

The extension of the different variants of ACO to multi-objective problems was straightforward, due to their ant colony construction methods. In [67], a multi-objective ACO algorithm is designed based on AS for the bi-criteria vehicle routing problem. In [68], multi-objective ACOs are compared for the job shop scheduling problem.

The popularity of the multi-objective ACO algorithm variants is proven by several surveys: the review [69] from 2011 introduces a taxonomy of multi-objective ACOs, and in 2022 [70] describes a refined taxonomy and reviews real-world applications, as well. In this line, [71] is the latest study, surveying the multi-objective ACO literature from 2004 to the beginning of March 2023.

Moving to the natural question of applying ACOs to the shortest path problem, the literature is not yet very rich (a strong indicator of the need for studies like this paper). Most works concentrate on different variants of shortest paths, as follows. Article [72] presents an Ant Colony System for finding the shortest path with preferred edges, using the properties of the links in a map. In [73], the authors investigate ant colony optimization applied to a stochastic shortest path problem in which edge weights are noisy, representing uncertainty or delays. Regarding multi-objective variants of the ACO, in [74] an ACO algorithm is presented to solve the bi-objective shortest path problem.

Dealing with different variants of safest path problems emerged only very recently: both of the works we know about appeared in 2025. In [12], an ACO is proposed to solve the safest path problem in the presence of correlated failures, whereas [75] proposes the Safest Path Pair Ant Colony Optimization (SPP-ACO) system to obtain a safest path pair.

Connected to the safest path problem in [76], a multi-objective genetic algorithm is proposed for evacuation route planning that considers three objectives: evacuation distance, evacuation time, and route safety.

III. PROBLEM STATEMENT AND HARDNESS RESULTS

A. PROBLEM STATEMENT AND BASIC OBSERVATIONS

The problem's input consists of two essential components. First, we are given a connected graph $G = (V, E)$ together with a designated source–destination pair $s, t \in V$ representing the communicating endpoints. Length $L(e) > 0$ is given for each link $e \in E$. In the decision version of the problem, there is given a length threshold $\mathcal{L}$ and an availability threshold T; with the task being to decide whether there exists an st-path of length at most $\mathcal{L}$ that, under the next disaster, fails with a probability of at most T. To assess the failure probability of a path (i.e., the probability that it does not remain intact after the next disaster), we need the second part of the input, where failure probabilities are stored.

In its second part, the input specifies the probabilities of correlated link failures. More precisely, for any subset of links $S \subseteq E$, we denote by CFP(S) (i.e., the *cumulative failure probability of* S) the probability that, at the occurrence of the next disaster, links in S fail simultaneously (and possibly some other links, too). The collection of these values is stored in a data structure CFP$[G]$, which contains every CFP(S) with strictly positive value (i.e., we omit sets with zero probability). In most practically relevant scenarios, the size of CFP$[G]$ remains tractable [34]. Note that the above terminology follows the notations of [10], [34]. Although CFP$[G]$ explicitly records only link-failure events, it can also encode node-failure probabilities implicitly; see [34, Sec. V.]

for details. In the following, we define cumulative failure probabilities in precise terms.

*Definition 1 (**Cumulative Failure Probability (CFP)**):* Given a set of links $S \subseteq E$, $S \neq \emptyset$, the cumulative failure probability (CFP) of S, denoted by CFP(S), is the probability that all links S fail simultaneously (and possibly other links too).

Observe that whenever a link set S has a positive probability of failure (i.e., CFP(S) > 0), consistency requires that all $2^{|S|} - 1$ non-empty subsets of S must also appear in CFP[G]. Consequently, in scenarios where e.g., only a few disaster events are possible but each affects a large number of links, this representation may become unnecessarily redundant. In such cases, a more succinct encoding of failure hazards is desirable. A direct and provably more space-efficient alternative is the Link Failure State Probability (FP) representation introduced in [34], which we adopt in this work as well.

*Definition 2 (**Link Failure State Probability (FP)**):* Given a set of links $S \subseteq E$, $S \neq \emptyset$, the link failure state probability (FP) of S, denoted by FP(S), is the probability that *exactly* the links of S fail simultaneously (and no other links).

Unless ambiguity arises, we use the term CFP in two closely related senses: (i) to denote the pair $(S, \text{CFP}(S))$ associated with a link set S, and (ii) to refer directly to the numerical value CFP(S). An analogous convention applies to the term FP.

Turning to the optimization version of the problem, one of the optimization objectives is to determine a safest st-path, that is, an st-path minimizing the probability that at least one of its constituent links fails in the next disaster event. The other objective is to minimize the sum of lengths $L(e)$ of links e being part of the st-path.

We introduce two mutually related formulations of the safest path problem under correlated link failures, as Problems 1 and 2, phrased in terms of CFP and FP values as failure probabilities, respectively.

Problem 1: Safest-Shortest Path CFP Problem

Input: A graph $G = (V, E)$, nodes s and t, failure probabilities CFP[G], length threshold $\mathcal{L}$, and failure probability threshold T.

Decision version question: Decide whether there exists an st-path of length at most $\mathcal{L}$ that, due to the next disaster, fails with probability at most T. If such a path exists, the instance is *satisfiable*.

Optimization version objectives: 1) minimizing unavailability while 2) minimizing path length

As we will see in Thm. 3, the above problems are $\mathcal{NP}$-hard, with their optimization versions being even hard to approximate, even if we relax the length objective.

Note that despite the inherent hardness of our *optimization* problem, given the graph G, and the list FP[G] of link sets of positive FP, there is a straightforward way to evaluate the *real* availability $A(P)$ of a given st-path P. Intuitively, a path P fails if a failure event cuts at least one link on the path:

Problem 2: Safest-Shortest Path FP Problem

Input: A graph $G = (V, E)$, nodes s and t, failure probabilities FP[G], length threshold $\mathcal{L}$, and failure probability threshold T.

Decision version question: Decide whether there exists an st-path of length at most $\mathcal{L}$ that, due to the next disaster, fails with probability at most T. If such a path exists, the instance is *satisfiable*.

Optimization version objectives: 1) minimizing unavailability while 2) minimizing path length

$$A(P) = 1 - \sum_{S \cap P \neq \emptyset} \text{FP}(S). \tag{1}$$

Note that while evaluating the above expression, link sets S having a nonempty intersection with path P, that are not in FP[G] can be neglected, since their cumulative failure probability is zero by definition.

Unfortunately, it is not obvious how FP[G] should be used to optimizw the availability of an st-connection. In fact, known optimization methods, including our upcoming S^{2}P-ACO algorithm, take advantage of the cumulative failure probabilities stored in CFP[G], which are possibly much more numerous than the number of failure probabilities in FP[G]. As a prelude to the optimization methods, we note that the *true* availability of a path P can also be expressed via the CFPs, as follows. By iterating through the link sets in CFP[G], and counting their CFP with the right sign, the following sum is calculated:

$$A(P) = 1 + \sum_{S \subseteq P} (-1)^{|S|} \text{CFP}(S). \tag{2}$$

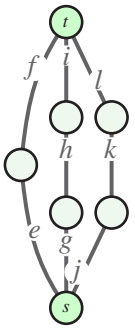
Note that by the inclusion-exclusion principle, this sum correctly assesses the availability.

Fig. 1 depicts a simple example problem input. Here, there are obviously three different st-paths. While *real* availability and real Pareto-optimality are straightforward, their *naive* counterparts need to be introduced here. The traditionally used *Naive* method maximizes the product of the availabilities of the links:

$$A_{\text{naive}}(P_{st}) = \Pi_{e \in P}(1 - \text{CFP}(e)) \tag{3}$$

This way, it assumes the best solution is to go from s to t along links e and f. This estimate turns out to be suboptimal since, in the example, the link failures are correlated. We note that using a well-known arithmetic trick, the optimization (neglecting the length threshold $\mathcal{L}$) here can be done by a simple Dijkstra algorithm [9].

For clarity, we explicitly express here that Eq. (1) and Eq. (2) (used by our algorithm S^{2}P-ACO) define correct assessments of the availability of a given path P; however, Eq. (3) is just a useful, easily optimizable formulation that helps the Naive algorithm to *guess* the best solutions.



FP[G] :
FP(e) =0.2 FP(f) =0.2
FP(g) =0.2 FP(g,h,i) =0.1
FP(j,l) =0.25

CFP[G] :
CFP(e) =0.2 CFP(f) =0.2 CFP(g) =0.3
CFP(h) =0.1 CFP(i) =0.1 CFP(g,h) =0.1
CFP(h,i) =0.1 CFP(g,i) =0.1 CFP(g,h,i) =0.1
CFP(j) =0.25 CFP(l) =0.25 CFP(j,l) =0.25

Link lengths : $L(e)$ =3.6 $L(f)$ =3.6
$L(g)$ =2 $L(h)$ =2 $L(i)$ =2
$L(j)$ =2.82 $L(k)$ =2 $L(l)$ =2.82

Path	Naive availability	Real availability	Length	Naive Pareto optimal	Real Pareto optimal
$\{e,f\}$	0.64	0.6	7.20	✓	✗
$\{g,h,i\}$	0.567	0.7	6.00	✓	✓
$\{j,k,l\}$	0.5625	0.75	7.64	✗	✓

FIGURE 1: A toy example on the input graph G, related failure probabilities stored in either FP[G] or CFP[G]. The table included depicts all possible paths between s and t and their associated availabilities and lengths. The naive algorithm supposes link failures are independent, and as such, it cannot accurately assess the availability of certain paths. Note that, in reality, these values are much closer to 1, thus the difference between optimal and suboptimal solutions is less stark.

B. COMPUTATIONAL HARDNESS

Now we turn to prove the hardness results of the problem at hand.

Theorem 3: For Problems 1 and 2, the following hold, even after relaxing the length objective:

- they are $\mathcal{NP}$-hard to approximate (in terms of minimizing unavailability) within any factor smaller than $10\sqrt{5}-21 \approx 1.3606$, even on planar graphs and dual-connected FPs/CFPs.

Further, Problem 2, after relaxing the length objective:

- is $\mathcal{W}[2]$-hard when parameterized by the number of link sets of FP[G] intersected by the st-path,
- cannot be approximated within a factor of $O(|V|^{1/8})$, unless the 'Dense vs Random' conjecture of [77] fails.

Proof: The ~ 1.36 inapproximability of both problems for planar graphs and so-called dual-connected CFPs/FPs was recently shown in [11, Claim 2], based on `SET COVER`. Here, the dual-connectedness means that the (planar) duals of the set of failed links form a connected link set in the dual graph.

Next, we show the $\mathcal{W}[2]$-hardness of Problem 2. We reduce from the classical *Minimum Label st-Path* problem (`MinLabelPath` for short, Problem 3), also known in the networking literature as *Minimum-Color Path* or *Minimum Constraint Violation Path* [78]–[82].

First, we show a reduction framework for the FP model (Problem 2) from Problem 3.

Claim 4: There is an approximation-preserving and parameter-preserving reduction from `MinLabelPath` problem (Problem 3) to Problem 2 with relaxed length objective. Consequently, Problem 2 with relaxed length objective is at least as hard to approximate as `MinLabelPath`; and at least as hard as `MinLabelPath` parameterized by the number of labels.

Problem 3: MinLabelPath Problem

Input: A graph $H=(V',E')$ with a source s and a sink t, together with a labeling function $\lambda : E' \to \mathcal{C}$ that assigns a single label (color) to each edge.
Objective: Find an st-path P that minimizes the number of distinct labels used on P, i.e., $\big|\{\lambda(e) : e \in P\}\big|$ is minimal.

Proof: Let (H,s,t,λ) be an instance of Problem 3, and let $\mathcal{C}=\{\ell_1,\dots,\ell_m\}$ be the label set. We build an instance of Problem 2 as follows. The underlying graph is $G:=H$, and we take the same terminals s,t. For each label $\ell \in \mathcal{C}$ we introduce one failure scenario $S_\ell := \{e \in E' : \lambda(e) = \ell\}$ that consists of all edges of that label. The family FP[G] consists of these m scenarios, with FP(S_ℓ) := $1/m$ for all $\ell \in \mathcal{C}$. Clearly, the probabilities sum to 1, so this is a valid setting of FP values.

Let P be any st-path and denote by $L(P) := \{\lambda(e) : e \in P\} \subseteq \mathcal{C}$ the set of labels used on P. By definition of availability in Eq. (1), P fails in exactly those scenarios S_ℓ with $\ell \in L(P)$, and it survives in all other scenarios. Therefore for the probability $U(P)$ of P failing when the next disaster hits we have:

$$U(P) = \sum_{S \cap P \neq \emptyset} \mathrm{FP}(S) = \sum_{\ell \in L(P)} \mathrm{FP}(S_\ell) = \frac{|L(P)|}{m}.$$

In other words, $U(P)$ is just a fixed constant times $|L(P)|$, that is, the objectives to be minimized in Problems 2 and 3 are constant multiples of each other. Since the underlying graph is the same, in the special case of uniform FP values, (the relaxed) Problem 2 essentially becomes Problem 3 (with an objective value multiplied by constant $1/m$). Consequently, minimizing $U(P)$ (with failure probabilities given by FP values) is at least as hard as minimizing $|L(P)|$. Our reduction preserves the parameter 'number of labels used': for every k, an st-path uses at most k labels if and only if it intersects at most k failure scenarios S_ℓ in the constructed Problem 2 instance. ■

Now we turn to the parameterized hardness of Problem 2. Study [82] investigates several minimum-label problems from the viewpoint of parameterized complexity and shows that MinLabelPath is $\mathcal{W}[2]$-hard when parameterized by the maximum allowed number k of distinct labels on the st-path. Hence, by Claim 4, a hypothetical fixed-parameter algorithm for Problem 2 (with the length objective omitted) with parameter "number of failure sets intersected by P" would imply an FPT algorithm for `MinLabelPath`, contradicting the $\mathcal{W}[2]$-hardness unless $\mathcal{FPT}=\mathcal{W}[2]$.

Lastly, for showing that the unavailability objective of Problem 2 cannot be approximated within a factor of $O(|V|^{1/8})$ (even if we disregard path lengths), we reach out for a version of Problem 3. The version is called the `MinColorPath` Problem, and it differs from Problem 3 in two things: instead of a single label, each edge can have multiple colors, i.e., there is a color function $\chi : E' \to 2^{\mathcal{C}}$, and the objective is to find an st-path P that uses the fewest colors, i.e., $\left|\cup_{e\in P}\chi(e)\right|$ is minimal. Paper [83] claims the `MinColorPath` Problem is inapproximable in $O(|V|^{1/8})$ unless the 'Dense vs Random' conjecture of the highly cited STOC paper [77] fails. It is easy to see that `MinColorPath` is a special case of Problem 2 in an identical way to the `MinLabelPath`. Thus, the hardness result of `MinColorPath` carries over to Problem 2. ■

Note that the absence of Problem 1 from the latter two hardness results of Theorem 3 does not mean it would be an easier problem compared to finding safest paths based on FP values. In fact, while Problems 1 and 2 describe the same real-world instance, CFP lists explode in size with the size of the largest failure set S recorded. In other words, while it is not immediate that Problem 1 is *that* hard of a problem if the possibly failing link sets are big, but then the sheer description size of the problem can be seen as highly suboptimal.

IV. SAFEST-SHORTEST PATH - ANT COLONY OPTIMIZATION ALGORITHM (S^{2}P-ACO)

In this Section, we first design our algorithm S^{2}P-ACO. Then, in Sec. IV-B, we describe the baseline algorithms that will be used in the evaluations. We also discuss some alternative baselines that we decided to omit from presenting the evaluation results for the sake of a better presentation.

A. ALGORITHM S^{2}P-ACO

To compute the safest-shortest paths in the presence of correlated failures, we implemented an algorithm based on Ant Colony Optimization (ACO) [60], more specifically it is inspired by the Max-Min Ant System variant. Ants use pheromone trails to signal the way to food sources to others, and this algorithm uses the same mechanics to find optimal solutions.

In order to use the algorithm to solve this problem, we have to adapt it. This adaptation is done by defining the following mechanisms that the system will use: heuristic information, pheromone setting, and solution generation.

The quick summary is that S^{2}P-ACO is a single colony system that uses a single heuristic matrix based on independent failures and a single pheromone matrix with Pareto-archive-based deposit (pheromone laid on all non-dominated routes of the iteration), with Max-Min Ant System inspired bounds on the pheromone levels.

Heuristic information, sometimes also called a-priori information, is used to guide the ants towards results that initially seem feasible. In the case of ACO, the higher it is, the more likely ants will visit that path, and as such, our algorithm uses the negative logarithm of the probability that the given edge breaks (ignoring probabilities that group multiple edges together). This would malfunction in some cases where there is no associated probability to that edge (the edge cannot break), in those cases, we assign a very small $\epsilon_{\text{prob}} > 0$ probability of breaking. For our tests, we used $\epsilon_{\text{prob}} = 10^{-8}$ as it is insignificant compared to any other CFP value in the data sets. We disallow jumping between nodes that are not connected by setting the heuristic information to 0. This is described in Eq. (4), where $\mathcal{N}_i$ is the set of neighbors of i and $e_{i,j}$ is the edge between nodes i and j. The heuristic information only contains the independent failures, the length and the correlated failures only enter the search exclusively during the Pareto-dominance-based pheromone update stage. This is deliberate since preliminary testing showed that including the length in the heuristic information did not meaningfully improve the results, and calculating the correlated failures a priori is infeasible.

$$\eta_{ij} = \begin{cases} -\log \text{CFP}(\{e_{i,j}\}), & \text{if } j \in \mathcal{N}_i \\ 0, & \text{else.} \end{cases} \tag{4}$$

The pheromone setting is similar to the aforementioned Max-Min Ant System. In order to encourage exploration in the algorithm, the pheromone levels will be limited to be between two values. These values are typically calculated from the best solution we have encountered. Since comparing solutions in multiobjective problems is not straightforward, we had to adapt the algorithm. We use Pareto dominance to compare the solutions that we get, and as such, not all solutions can be strictly ordered. Since there is no "best" solution, our maximum and minimum values are constants named $\tau_{\max}$ and $\tau_{\min}$.

Algorithm 1: Path generation for the S^{2}P-ACO algorithm

Input: Heuristic information ($\eta \in \mathbb{R}^{|V|\times|V|}$), Pheromone information ($\tau \in \mathbb{R}^{|V|\times|V|}$), ACO parameters: α, β nodes s, t
Output: A generated st-path or nothing

```
R := list(s)
v := zeros(|V|) // remember previously visited nodes
v[s] := 1
while R_end ≠ t and |R| < |V| do
    j := 0
    r := s
    while v[r] = 1 and j < maxTries do
        r := new node from N_{R_end} based on Eq. (7)
        j := j + 1
    if j = maxTries then
        return nothing // the ant failed to generate a viable solution
    v[r] := 1
    push(R, r)
if |R| = |V| and R_end ≠ t then
    return nothing // the ant failed to generate a viable solution
return R
```

The pheromone distribution was also changed: in each iteration, a set amount of pheromones is distributed on all paths that are not dominated by previous solutions. The pheromone evaporates at a constant rate described in Eq. (5), where τ_{ij}^{t} is the pheromone level between nodes i and j in iteration t:

$$\tau_{ij}^{(t)} = (1-\rho)\cdot\tau_{ij}^{(t-1)} \tag{5}$$

On the edges in non-dominated routes that the algorithm finds, it places additional pheromones with Eq. (6) describing the complete equation with the evaporation, where c is a constant value during the lifetime of the algorithm:

$$\tau_{ij}^{(t)} = (1-\rho)\cdot\tau_{ij}^{(t-1)} + c \tag{6}$$

When generating a solution, our algorithm places an ant on the starting node and for each move the next node is chosen by considering the heuristic information and pheromone levels from all edges originating from the current node as described in Eq. (7), where τ_{ij} is the pheromone level between nodes i and j, η_{ij} is the heuristic information, and $\mathcal{N}_i$ is the set of the neighbors of i. It would not make sense for an ant to generate a path with cycles in it, thus they are forbidden from revisiting nodes. Only paths that actually reach the target are considered in further calculations. An ant can fail to find a path if it does not reach the target t in a set number of moves. The path generation algorithm is detailed in Algorithm 1.

$$p(i,j) = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{v_q\in\mathcal{N}_i}(\tau_{iq})^{\alpha}(\eta_{iq})^{\beta}}\text{, if } v_j \in \mathcal{N}_i \tag{7}$$

Combining the parts explained above, our proposed algorithm works as follows: The algorithm runs for a set number of iterations; in each iteration, a predefined number of ants is placed on the starting node s. Each ant generates a path as described above, and each generated path is evaluated and compared to a Pareto front created from previously obtained solutions. If they are not dominated by any previous solutions, we increase the pheromone level on the corresponding edges, and if they are not on the Pareto front, they get added to it. Afterwards, the algorithm limits the pheromone levels to the previously defined levels. In the end, the Pareto front of the solutions is returned, paying attention to eliminating solutions that used to be on the front but were later dominated by other solutions. This method is formalized in Algorithm 2.

Algorithm 2: Safest-Shortest Path - Ant Colony Optimization (S^{2}P-ACO)

Input: Graph $G=(V,E)$, cumulative failure probabilities CFP[G], lengths L[G], ACO parameters: α, β, ρ, nrOfAnts; nrOfIterations, nodes s and t
Output: A Pareto front containing st-paths
Initialize pheromone trails
$i := 0$
$P = \emptyset$ `// path set of the Pareto front`
while $i <$ `nrOfIterations` **do**
  $S :=$ list(s)
  repeat
    Construct a new route R based on Algorithm 1
    push(S, R)
  until $|S| =$ `nrOfAnts`;
  Update pheromone trails Eq. (5)
  for $R \in S$ **do**
    if *fitness(R, FP[G], L[G]) is not dominated by any element of P* **then**
      insert(P, (R, fitness(R, FP[G], L[G])))
      Update pheromone trail based on Eq. (6)
  Bound pheromone trails between $\tau_{\min}$ and $\tau_{\max}$
  $i := i + 1$
Reconstruct Pareto front P to eliminate values that are no longer Pareto-optimal
return P

Algorithm 3: Exact algorithm

Input: Graph $G=(V,E)$, cumulative failure probabilities CFP[G], lengths L[G] and nodes s and t
Output: A Pareto front containing st-paths
$Q :=$ queue()
$P = \emptyset$ `// path set of the Pareto front`
push(Q, [s])
while $|Q| > 0$ **do**
  $X :=$ pop(Q)
  if *fitness(X, FP[G], L[G]) is dominated by P* **then**
    continue
  if $X_{end} = t$ **then**
    insert(P, (X, fitness(X, FP[G], L[G])))
    continue
  for $n \in \mathcal{N}_{X_{end}}$ **do**
    if $n \notin X$ **then**
      $Y := X + n$
      push(Q, Y)
Rebuild Pareto front P by eliminating dominated paths.
return P

B. BASELINES

To evaluate the performance of the proposed method, we compare it with three simple algorithms, as discussed in the following.

1) Used baselines

The first baseline is an exact algorithm. Since we know that adding an edge to a path will increase both its length and failure probability, the Pareto front can be found by exploring each path starting from the source node until that path either reaches the destination or becomes dominated by other previously discovered paths. This is a straightforward, albeit slow algorithm. Further description is found in Alg. 3.

The second proposed algorithm - the Naive algorithm - is based on maximizing a linear combination of the independent failure probabilities and length values, both normalized. This is also called the weighted sum method. It constructs a new graph from the combined values and runs Dijkstra's algorithm on it. To construct a Pareto front, the algorithm repeats this calculation on a wide range of blending values as seen in Alg. 4. The set of blending values is the following: $B = \{0.1\cdot x \mid x \in [0,10]\cap\mathbb{N}\}$.

We proposed the third algorithm to study whether the S^{2}P-ACO algorithm was appreciably better than random chance, and it is called random_walk. It is very similar to

the proposed S^{2}P-ACO algorithm, but as the name suggests, instead of relying on pheromones and heuristic knowledge, the agents choose their next node randomly (but still not repeating nodes). The formalization is the same as in Alg. 2 with the difference that during path generation (Alg. 1) instead of using Eq. (7) we use a random selection from the set of neighbors.

2) Omitted baselines

Some of the baselines not included in our evaluation results are as follows.

NSGA-II

Since we are working with a multiobjective optimization problem, we have to take into account the state of the art algorithms that were developed to deal with this problem. Initially, we opted to use the implementation provided by the Metaheuristics.jl package [84]. The algorithm uses a fixed length chromosome one value for each edge that either turns it on or off. This is turned into a path by running Dijkstra's algorithm on the induced subgraph and evaluating its fitness. Although NSGA-II performed close to the ACO algorithm in the preliminary tests, it only managed to recover the optimal Pareto front in 97.93% of the test cases compared to 99.06% of the ACO algorithm on graphs italy_it7 and 22_optic_it7. It also incurred a substantial runtime cost averaging around 13.47 seconds on the relatively simple italy_it7 graph for each node pair and 31.47 seconds on the 22_optic_it7 for each node pair, resulting in a slower runtime than the exact algorithm and an order of magnitude slower than the ACO algorithm. The evaluation budgets are comparable, the NSGA-II algorithm stops due to a convergence criterion after 20 iterations using 3900 fitness evaluations which is right around the 4000 mark used by the ACO algorithm. Due to these results where the NSGA-II performed worse and ran much slower than the ACO algorithm we opted to omit it from further testing.

Naive using at most i links

A safest st-path (according to the naive availability measure) consisting of at most i links can be computed in $O(i|V|^2)$, e.g., with the help of a slight variation of the Bellman-Ford algorithm [5]. The proposition underpinning this claim is that, for any input graph, after i repetitions of the outermost for loop of the Bellman-Ford algorithm, the computed st-path is a shortest st-path with at most i edges, if there is an st-path with at most i edges at all. Again, since this algorithm, at its core, uses the imprecise availability measure A_{naive}, which results in suboptimal solutions, we decided to skip this baseline in the simulations.

V. NUMERICAL EVALUATION

A. SIMULATION SETTINGS

1) Implementation

To conduct our extensive simulation experiments, each algorithm (Exact, Naive, random_walk, and S^{2}P-ACO) was implemented in the same programming language, namely Julia. Most tests were run on a system with AMD Ryzen Threadripper PRO 5955WX 16-Cores, but limited only to 16 threads in order not to interfere with other tests running on the shared system, 252 GB of RAM, Arch Linux on kernel version 6.12.57-2-LTS, Julia version 1.12.1.

Algorithm 4: Naive algorithm

Input: Graph $G = (V, E)$, cumulative failure probabilities CFP[G], lengths L[G], blending values B and nodes s and t
Output: A Pareto front containing st-paths

```
P = ∅        // path set of the Pareto front
L := normalize(L[G])
CFP := normalize(CFP[G])
for b ∈ B do
    for i ∈ V do
        for j ∈ V do
            if a_{i,j} ∈ E then
                L'_{i,j} := b * L_{i,j} + (1 − b) * CFP_{i,j}
    Run Dijkstra with root s and weights L'
    X := st-path provided by Dijkstra
    if X ∉ P then
        insert(P, (X, fitness(X, FP[G], L[G])))
return P
```

2) Benchmarks

The basic properties of the real-world networks are presented in Table 1. The number of nodes, the number of edges, and the source of the graphs (standing for internet backbones) are detailed in this table. For the numerical experiments, we used a variety of seismic hazard inputs taken from [34]. We assumed that all links and nodes of the input graphs had *intensity tolerance* values of either VI or VII according to the MCS scale [85] are considered, denoted by $it6$ and $it7$. Vaguely speaking, this means that we suppose that the network equipment may fail in case of a 'strong' and 'very strong' shaking, with peak ground accelerations exceeding around 1.2 and 2.2 $[\mathrm{m/s^2}]$, respectively. For simplicity, in the rest of this section, we call the tuple of a graph (named `graph`) with a chosen intensity tolerance `it` a *network*, with name `graph_it`.

3) Pareto-optimality measures

In order to test the performance of the proposed S^{2}P-ACO we use seven indicator metrics, capturing different properties of the proposed algorithm:

TABLE 1: Real-world problem inputs

Graph name	$\|V\|$	$\|E\|$	Graph from	Failure data
22_optic	22	45	[28]	[33], [34]
italy	25	34	[33]	[33], [34]
cost266	37	57	[28]	[33], [34]
janos_us	26	42	[28]	[33], [34]
39_optic	39	61	[28]	[33], [34]
79_optic	79	108	[28]	[33], [34]

TABLE 2: Parameter tuning for the 22_optic_it6 network for two path pairs. Average values and standard deviation are presented over 20 independent runs. Best result is depicted in bold, while a * indicates that, based on the Wilcoxon rank sum test ($\alpha = 0.05$), no significant difference exists between the best and the certain configuration.

			22_optic_it6	$2 \rightarrow 3$					
Conf#	α	β	AHD	DR	GD	(1 - HR)·10^5	IGD	MPFE	PQAD
1	0.5	0.5	46.42 ± 35.58	0.02 ± 0.04	1.39 ± 2.87	2.89 ± 3.47	13.4 ± 10.27	14.55 ± 29.62	594.47 ± 1128.55
2	1	0.5	$\mathbf{1.27 \pm 5.66}$	0.00417 ± 0.02*	0.22 ± 0.97*	$\mathbf{0.06 \pm 0.25}$	$\mathbf{0.37 \pm 1.63}$	2.59 ± 11.59*	75.68 ± 338.43*
3	1.5	0.5	2.01 ± 9.01*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.06 \pm 0.28}$	0.58 ± 2.6*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
4	2	0.5	18.89 ± 25.1	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.95 ± 1.5	5.45 ± 7.24	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
5	0.5	1	60.06 ± 65.43	0.05 ± 0.07	2.55 ± 6.09	5.27 ± 5.84	17.34 ± 18.89	23.17 ± 50.82	1019.68 ± 1535.06
6	1	1	4.03 ± 12.4*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.13 ± 0.39*	1.16 ± 3.58*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
7	1.5	1	6.04 ± 14.76*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.19 ± 0.47*	1.74 ± 4.26*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
8	2	1	82.97 ± 268.49	0.00909 ± 0.03*	0.22 ± 0.67*	5.85 ± 22.22	23.95 ± 77.51	2.39 ± 7.35*	219.0 ± 674.06*
9	0.5	1.5	43.39 ± 35.46	0.06 ± 0.1	2.52 ± 3.64	4.01 ± 5.1	12.53 ± 10.24	25.91 ± 36.67	1043.32 ± 1406.07
10	1	1.5	2.01 ± 9.01*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.06 \pm 0.28}$	0.58 ± 2.6*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
11	1.5	1.5	2.01 ± 9.01*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.06 \pm 0.28}$	0.58 ± 2.6*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
12	2	1.5	160.81 ± 257.24	0.04 ± 0.1	11.33 ± 46.24	8.65 ± 14.89	46.42 ± 74.26	67.69 ± 277.23	813.27 ± 1823.24
13	0.5	2	35.87 ± 31.77	0.05 ± 0.08	2.72 ± 4.21	2.94 ± 4.23	10.35 ± 9.17	26.09 ± 38.5	908.11 ± 1407.77
14	1	2	2.01 ± 9.01*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.06 \pm 0.28}$	0.58 ± 2.6*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
15	1.5	2	4.03 ± 12.4*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.13 ± 0.39*	1.16 ± 3.58*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
16	2	2	15.78 ± 24.06	0.005 ± 0.02*	0.12 ± 0.53*	0.62 ± 1.23	4.56 ± 6.94	1.19 ± 5.34*	131.35 ± 587.4*
				$4 \rightarrow 17$					
Conf#	α	β	AHD	DR	GD	HR·10^5	IGD	MPFE	PQAD
1	0.5	0.5	5.08 ± 12.35	0.00938 ± 0.03*	0.66 ± 2.05*	4.35 ± 11.97	1.27 ± 3.09	8.77 ± 26.99*	97.64 ± 305.21*
2	1	0.5	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
3	1.5	0.5	2.47 ± 6.86	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.82 ± 3.55	0.62 ± 1.72	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
4	2	0.5	7.19 ± 16.76	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	6.6 ± 18.58	1.8 ± 4.19	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
5	0.5	1	2.49 ± 6.71	0.00625 ± 0.02*	0.55 ± 1.69*	1.06 ± 3.22	0.62 ± 1.68	8.77 ± 26.99*	81.21 ± 249.97*
6	1	1	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
7	1.5	1	2.26 ± 6.95*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.8 ± 3.57*	0.56 ± 1.74*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
8	2	1	8.79 ± 12.11	0.01 ± 0.04*	0.45 ± 2.0*	4.45 ± 9.59	2.06 ± 2.77	4.39 ± 19.61*	96.59 ± 431.98*
9	0.5	1.5	0.11 ± 0.5*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.00822 ± 0.04*	0.03 ± 0.12*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
10	1	1.5	1.12 ± 5.01*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	0.79 ± 3.55*	0.28 ± 1.25*	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
11	1.5	1.5	3.34 ± 8.16*	0.00625 ± 0.02*	0.47 ± 1.46*	1.29 ± 4.04*	0.83 ± 2.04*	7.48 ± 23.4*	81.31 ± 250.28*
12	2	1.5	87.44 ± 83.82	0.04 ± 0.07	1.18 ± 2.66	61.57 ± 67.65	21.86 ± 20.95	11.87 ± 25.83	676.99 ± 1205.26
13	0.5	2	1.1 ± 4.9*	0.00312 ± 0.01*	0.27 ± 1.23*	0.52 ± 2.34*	0.27 ± 1.23*	4.39 ± 19.61*	40.61 ± 181.6*
14	1	2	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
15	1.5	2	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
16	2	2	12.09 ± 11.28	0.00357 ± 0.02*	0.09 ± 0.38*	5.9 ± 8.44	3.02 ± 2.82	1.19 ± 5.34*	80.16 ± 358.5*

- Averaged Hausdorff distance (AHD) [86] - is based on the GD and IGD metrics;
- dominated ratio (DR) - is a cardinality measure;
- Generational distance (GD) [87] - is a convergence indicator, it calculates the average distance between the found Pareto front and the true Pareto front;
- Hyperarea ratio (HR) [88] - is a convergence and distribution indicator;
- Inverted Generational Distance (IGD) [89] - is a convergence and distribution indicator, which considers the average distance between the discrete representation of the true Pareto front and the obtained front;
- Maximum Pareto front error (MPFE) [87] - is a convergence indicator, that computes the maximum of the minimum distances between the obtained front and the true Pareto front;
- p,q-averaged distance (PQAD) [90] - measures both convergence and distribution;

4) Parameter tuning

To test the proposed S^{2}P-ACO algorithm, we performed a parameter test for the following parameters: $\alpha \in \{0.5, 1, 1.5, 2\}$ and $\beta \in \{0.5, 1, 1.5, 2\}$ in order to determine experimentally optimal values of pheromone strength and heuristic information strength. In total 16 configurations were tested on the 22_optic_it6 and cost266_it6 networks. An excerpt of these results are presented in Tables 2 and 3 showing the metrics for two *st* pairs for two graphs each (22_optic_it6 and cost266_it6). Mean values and standard deviations are presented for 20 independent runs. In every metric, lower values are better, the lowest values are indicated with bold font type, every result that is not statistically different (a Wilcoxon rank-sum test was performed) than the best result is marked with a star. Most configurations yielded similar results, but the consistently best performing settings were number 2, 6, 14 and 15.

Ultimately, configuration 6 was selected, where $\alpha = 1.0$, $\beta = 1.0$, because in most cases (all but one in the presented tables) it was not significantly worse than the best result. Other parameters used for the rest of the experiments are: 20 ants, 200 iterations, $\rho = 0.1$. Although it is rarely reached, *maxTries* is set to 100. These settings are based on [75].

The pheromone trails were initialized to $\tau_{\max} = c = 1$, and $\tau_{\min}$ was set to 0. We note that this renders the lower bound of the Max-Min scheme inactive, our preliminary tests showed that this aggressive optimization performed best. Settings with a larger minimum were also tested without providing any consistent improvement.

B. RESULTS AND DISCUSSION

TABLE 3: Parameter tuning for the cost266_it6 network for two node pairs. Average values and standard deviation is presented over 20 independent runs. Best result is depicted in bold, while a * indicates that based on the Wilcoxon rank sum test ($\alpha = 0.05$) no significant difference exists between the best and the certain configuration.

			cost266_it6	$1 \rightarrow 3$					
Conf#	α	β	AHD	DR	GD	HR·10^5	IGD	MPFE	PQAD
1	0.5	0.5	149.49 ± 187.68	0.18 ± 0.15	40.79 ± 67.14	10.99 ± 10.6	18.13 ± 12.26	320.77 ± 563.16	1709.97 ± 1119.36
2	1	0.5	11.97 ± 28.1	0.03 ± 0.1*	0.58 ± 1.85*	1.66 ± 5.06	3.99 ± 9.37	4.69 ± 14.35*	221.71 ± 589.02
3	1.5	0.5	5.0 ± 15.46*	0.02 ± 0.08*	0.58 ± 2.21*	0.87 ± 2.86*	1.67 ± 5.15*	5.14 ± 19.57*	157.69 ± 512.91
4	2	0.5	396.58 ± 836.8	0.2 ± 0.24	104.52 ± 310.06	41.72 ± 78.13	128.89 ± 276.65	261.91 ± 724.76	1990.23 ± 2494.05
5	0.5	1	125.55 ± 155.97	0.21 ± 0.19	31.26 ± 58.28	12.46 ± 11.94	20.88 ± 13.48	237.03 ± 434.76	1544.57 ± 1138.43
6	1	1	28.46 ± 94.11*	0.03 ± 0.09*	9.31 ± 41.22*	1.24 ± 4.36*	4.93 ± 13.1*	46.86 ± 205.87*	235.11 ± 835.25
7	1.5	1	29.19 ± 116.37	0.04 ± 0.12*	2.82 ± 11.67*	5.41 ± 20.28	9.73 ± 38.79	6.7 ± 23.63*	437.47 ± 1371.38
8	2	1	335.89 ± 837.74	0.15 ± 0.17	7.91 ± 22.05	17.86 ± 29.17	107.06 ± 280.13	64.52 ± 203.33	1313.74 ± 1514.57
9	0.5	1.5	126.84 ± 153.4	0.16 ± 0.17	39.23 ± 64.6	10.25 ± 10.98	19.88 ± 15.37	287.07 ± 433.0	1360.34 ± 1188.36
10	1	1.5	8.1 ± 26.29*	0.00556 ± 0.02*	0.09 ± 0.39*	0.31 ± 1.25*	2.7 ± 8.76*	0.79 ± 3.55*	53.28 ± 238.28
11	1.5	1.5	8.36 ± 16.7	0.03 ± 0.06	0.71 ± 1.71	1.54 ± 2.9	2.79 ± 5.57	5.68 ± 13.3	249.86 ± 519.66
12	2	1.5	545.64 ± 964.54	0.25 ± 0.26	16.35 ± 27.27	38.47 ± 35.69	175.87 ± 323.92	99.9 ± 217.73	2065.82 ± 1836.72
13	0.5	2	116.31 ± 112.72	0.26 ± 0.22	24.94 ± 47.32	14.24 ± 12.49	27.4 ± 16.83	170.26 ± 325.71	1870.51 ± 1282.13
14	1	2	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
15	1.5	2	2.56 ± 10.25*	0.00556 ± 0.02*	0.09 ± 0.39*	0.31 ± 1.25*	0.85 ± 3.42*	0.79 ± 3.55*	53.28 ± 238.28
16	2	2	157.83 ± 615.47	0.07 ± 0.12	1.31 ± 2.72	7.62 ± 20.98	52.61 ± 205.16	10.18 ± 20.7	528.96 ± 810.39
				$2 \rightarrow 6$					
Conf#	α	β	AHD	DR	GD	HR·10^5	IGD	MPFE	PQAD
1	0.5	0.5	164.16 ± 152.37	0.22 ± 0.13	39.47 ± 52.07	36.72 ± 27.19	27.68 ± 14.83	371.58 ± 567.71	2725.04 ± 1119.6
2	1	0.5	13.26 ± 24.17*	0.02 ± 0.05*	0.7 ± 2.51*	1.8 ± 3.63*	3.97 ± 7.26*	5.93 ± 19.67*	236.14 ± 609.88*
3	1.5	0.5	19.77 ± 39.58*	0.01 ± 0.04*	0.43 ± 1.61*	2.28 ± 5.05*	5.96 ± 11.93*	3.28 ± 11.5*	232.42 ± 749.28*
4	2	0.5	96.81 ± 83.82	0.07 ± 0.16	9.25 ± 34.3	15.72 ± 21.43	26.4 ± 17.22	56.56 ± 204.79	949.32 ± 1572.1
5	0.5	1	156.9 ± 144.56	0.23 ± 0.15	43.3 ± 52.19	26.12 ± 16.91	19.69 ± 11.83	393.88 ± 459.94	2292.25 ± 1254.88
6	1	1	5.16 ± 20.6	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.31 \pm 1.09}$	$\mathbf{1.55 \pm 6.21}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
7	1.5	1	88.69 ± 148.68	0.03 ± 0.08	1.51 ± 5.65	13.81 ± 42.36	26.74 ± 44.83	6.19 ± 17.47	563.3 ± 1504.79
8	2	1	86.26 ± 70.68	0.03 ± 0.09*	5.71 ± 22.28*	7.26 ± 8.66	22.5 ± 12.43	51.04 ± 205.42*	473.69 ± 1159.86*
9	0.5	1.5	207.01 ± 245.73	0.17 ± 0.15	51.99 ± 85.55	24.7 ± 22.88	24.68 ± 12.01	518.27 ± 842.43	2067.66 ± 1416.87
10	1	1.5	11.51 ± 29.16*	0.00455 ± 0.02*	0.07 ± 0.32*	0.89 ± 2.38*	3.47 ± 8.79*	0.79 ± 3.55*	61.07 ± 273.11*
11	1.5	1.5	41.94 ± 45.11	0.00909 ± 0.03*	0.14 ± 0.44*	3.15 ± 4.12	12.65 ± 13.6	1.59 ± 4.88*	122.14 ± 375.94*
12	2	1.5	177.98 ± 122.8	0.26 ± 0.2	23.5 ± 32.73	53.73 ± 45.78	49.09 ± 34.28	135.65 ± 214.62	3194.83 ± 1969.61
13	0.5	2	111.4 ± 118.18	0.11 ± 0.15	18.72 ± 40.53	13.24 ± 16.52	21.6 ± 15.59	181.56 ± 380.59	1125.03 ± 1529.47
14	1	2	22.62 ± 37.94*	0.00909 ± 0.03*	0.14 ± 0.44*	1.77 ± 3.12*	6.82 ± 11.44*	1.59 ± 4.88*	122.14 ± 375.94*
15	1.5	2	44.27 ± 46.23	0.01 ± 0.03*	0.38 ± 1.41*	3.1 ± 4.34	13.35 ± 13.94	3.28 ± 11.5*	184.91 ± 604.46*
16	2	2	82.73 ± 49.26	0.08 ± 0.13	3.99 ± 11.66	15.48 ± 23.93	24.57 ± 14.49	27.3 ± 74.94	1015.94 ± 1414.61

TABLE 4: Mean, standard deviation of the means for GD and IGD values compared to the exact result and the percentage of tests that resulted in the optimal Pareto front. Bold means lowest mean in row according to GD and IGD, respectively.

	S^{2}P-ACO			naive			random_walk		
Network	GD	IGD	Opt.	GD	IGD	Opt.	GD	IGD	Opt.
22_optic_it6	$\mathbf{0.0008 \pm 0.012}$	$\mathbf{0.718 \pm 4.324}$	**98.70%**	1.326 ± 20.112	23.721 ± 40.610	63.20%	2.300 ± 12.744	4.216 ± 12.867	87.66%
22_optic_it7	0.0008 ± 0.075	$\mathbf{1.859 \pm 8.251}$	**97.84%**	$\mathbf{0.0 \pm 0.0}$	150.780 ± 243.924	45.89%	2.309 ± 8.726	12.762 ± 33.760	85.74%
italy_it6	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	**100.0%**	21.729 ± 58.075	7.459 ± 23.683	73.00%	0.015 ± 0.160	0.024 ± 0.206	99.78%
italy_it7	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	**100.00%**	17.772 ± 54.503	4.818 ± 32.765	78.67%	0.021 ± 0.218	0.023 ± 0.195	99.78%
cost266_it6	0.1284 ± 2.007	$\mathbf{4.555 \pm 84.559}$	**98.94%**	$\mathbf{0.0 \pm 0.0}$	25.703 ± 102.158	65.77%	18.169 ± 43.510	16.405 ± 79.767	82.39%
cost266_it7	0.1023 ± 1.578	$\mathbf{0.368 \pm 4.677}$	**99.64%**	$\mathbf{0.0 \pm 0.0}$	27.204 ± 73.427	64.41%	27.119 ± 62.219	15.449 ± 49.143	82.12%
janos_us_it6	0.0008 ± 0.146	$\mathbf{1.654 \pm 18.656}$	**99.37%**	$\mathbf{0.0 \pm 0.0}$	45.844 ± 91.766	58.15%	1.596 ± 5.654	4.321 ± 21.311	91.85%
janos_us_it7	0.0055 ± 0.088	$\mathbf{1.630 \pm 19.683}$	**99.49%**	$\mathbf{0.0 \pm 0.0}$	44.913 ± 92.804	59.69%	1.697 ± 12.572	4.207 ± 22.950	92.05%
39_optic_it6	0.762 ± 3.321	31.855 ± 174.62	**91.55%**	$\mathbf{0.553 \pm 2.844}$	49.602 ± 93.717	58.03%	1.485 ± 5.987	$\mathbf{15.237 \pm 80.443}$	86.73%
39_optic_it7	0.806 ± 3.747	36.684 ± 190.318	**91.05%**	$\mathbf{0.563 \pm 2.903}$	53.402 ± 131.170	57.35%	4.509 ± 12.925	$\mathbf{15.275 \pm 76.501}$	86.94%

1) General behaviors

We conducted experiments for all *st* point pairs present in each graph (a total of $\binom{|V|}{2}$ experiments per graph, where $|V|$ denotes the number of nodes). For 10 of the 12 networks the exact solution is known, and, as such, we can compare our results to the correct ones using Generational Distance (GD) and Inverse Generational Distance (IGD) as metrics. In this case, we calculated how many tests resulted in the optimal Pareto front and we averaged the GA and IGD values first over the 20 runs for each node-pair, then over node-pairs to create Table 4. Some concrete examples are extracted to study whether there is a statistically significant difference between the results, these examples are depicted in Table 5.

Since the approach of the naive algorithm is to use a weighted sum of the two objectives, it cannot find any solution that is not on its perceived Pareto front. Thus, it is not surprising that the GD values of the naive algorithm are generally on the lower end. However, they are not always 0, which can be attributed to the fact that the naive algorithm does not perceive the availability correctly (see Eq. (3)) and thus may give solutions that prove to be suboptimal. The naive method also fails to find all paths on the Pareto front, as evidenced by the IGD column, which contains the worst results out of all three algorithms.

The unfocused behavior of the random_walk method is apparent from the fact that its GD values, in most cases, are much higher than those of the other two methods. The quality of the answers varies wildly between runs.

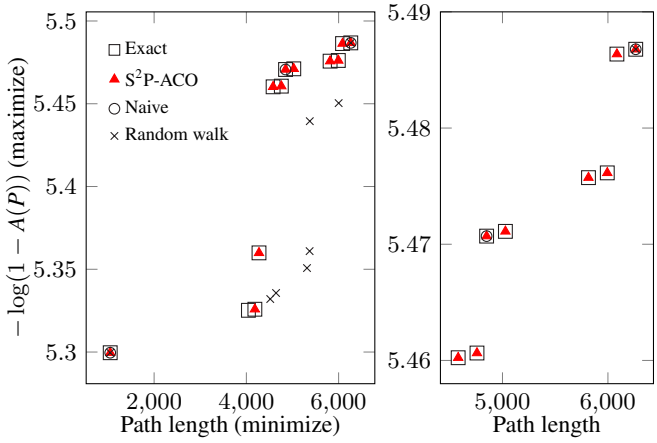
Comparatively, the S^{2}P-ACO algorithm, while far from perfect, generally yields results that are closer to ground truth than other algorithms. This is supported by the fact that both GD and IGD values are very low, even if they are not always the lowest values in the row. The two networks where this is not the case are 39_optic_it6 and 39_optic_it7. Here, the

TABLE 5: Means and standard deviations of GD and IGD values for a given node pair on a given graph. Smallest values are written in bold, while values that are not significantly different are marked with a * ($\alpha = 0.05$).

	S^{2}P-ACO		naive		random_walk	
Node pairs	GD	IGD	GD	IGD	GD	IGD
Graph	22_optic_it6					
$1 \to 6$	$0.185 \pm 0.827*$	$\mathbf{0.867 \pm 2.367}$	$\mathbf{0.0 \pm 0.0}$	103.83 ± 0.0	32.117 ± 58.021	28.793 ± 9.571
$5 \to 6$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{48.0 \pm 67.12}$	$\mathbf{0.0 \pm 0.0}$	$52.55 \pm 0.0*$	$3.365 \pm 15.053*$	90.053 ± 65.982
$5 \to 10$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	306.34 ± 0.0	7.28 ± 0.0	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
Graph	cost266_it6					
$0 \to 2$	$9.30 \pm 41.22*$	$\mathbf{4.929 \pm 13.10}$	$\mathbf{0.0 \pm 0.0}$	113.01 ± 0.0	159.965 ± 138.99	49.01 ± 21.06
$0 \to 3$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
$2 \to 4$	$0.079 \pm 0.354*$	$\mathbf{4.186 \pm 9.713}$	$\mathbf{0.0 \pm 0.0}$	122.92 ± 0.0	176.864 ± 144.654	41.89 ± 15.86
Graph	39_optic_it6					
$0 \to 12$	$4.896 \pm 21.899*$	$\mathbf{13.40 \pm 28.232}$	$\mathbf{0.0 \pm 0.0}$	77.478 ± 0.0	42.950 ± 39.086	25.40 ± 21.13
$2 \to 9$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	103.32 ± 0.0	$7.719 \pm 34.523*$	$7.27 \pm 32.52*$
$2 \to 16$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	10.809 ± 0.0	26.172 ± 0.0	22.742 ± 35.029	38.18 ± 19.45

TABLE 6: Comparisons of the results on graph 79_optic. 20 runs were conducted, and for each run we calculated for how many node pairs did the ACO algorithm produce better, equal and worse results than the other algorithms. We calculated the averages and standard deviations of those numbers. *HA* represents the hyperarea that is dominated by the Pareto front found by the algorithm. For the *DR* results we calculated the ratios of dominated points when comparing the two algorithms. One Pareto front is considered to be better if it dominates a greater ratio of the others points than the vice versa.

Graph $F = HA$	$F(P_{\text{S}^2\text{P-ACO}}) > F(P_{\text{random_walk}})$ mean $\pm$ std	$F(P_{\text{S}^2\text{P-ACO}}) = F(P_{\text{random_walk}})$ mean $\pm$ std	$F(P_{\text{S}^2\text{P-ACO}}) < F(P_{\text{random_walk}})$ mean $\pm$ std	$F(P_{\text{S}^2\text{P-ACO}}) > F(P_{\text{naive}})$ mean $\pm$ std	$F(P_{\text{S}^2\text{P-ACO}}) = F(P_{\text{naive}})$ mean $\pm$ std	$F(P_{\text{S}^2\text{P-ACO}}) < F(P_{\text{naive}})$ mean $\pm$ std
79_optic_it6	789.3 ± 13.81	2025.25 ± 14.38	266.45 ± 15.22	1443.65 ± 13.28	1389.1 ± 7.12	248.25 ± 12.02
79_optic_it7	784.8 ± 13.46	2027.15 ± 11.14	269.05 ± 15.84	1115.6 ± 11.02	1693.3 ± 6.47	272.1 ± 13.62
$F = DR$						
79_optic_it6	524.6 ± 13.50	2504.3 ± 12.62	52.1 ± 8.67	281 ± 3.32	2687.45 ± 9.76	112.55 ± 9.46
79_optic_it7	560.75 ± 9.74	2467.05 ± 11.04	53.2 ± 6.60	125.2 ± 1.93	2836.3 ± 6.77	119.5 ± 6.67



(a) All solutions (b) Top right corner of (a)

FIGURE 2: Pareto fronts found by different algorithms on graph 22_optic_it6 between nodes 1 (Rome) and 2 (Athens)

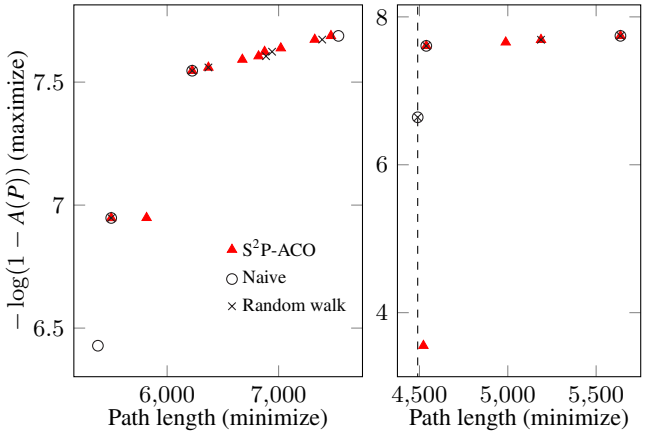
GD values are very low for the S^{2}P-ACO algorithm because it focuses too hard on a portion of Pareto optimal results, nevertheless, it misses a sizeable number of them, whereas the random_walk algorithm produces some not Pareto optimal solutions, and having any solution is considered better than having no solutions. Although this happens rarely, the IGD between the fronts is large enough to inflate the mean. The effectiveness of S^{2}P-ACO is also underlined by the fact that it was able to retrieve the optimal solution in more test cases in every network than the other two algorithms.

An concrete example of the efficacy of the algorithm is presented in Figure 2 which displays the Pareto front obtained from a single run for a single *st* pair for the 22_optic_it6 network asking for a path between Rome and Athens. In this case S^{2}P-ACO found 11 of the 12 paths present on the Pareto front, far outperforming the naive and random_walk algorithms, which found 3 and 2, respectively.

For the 79_optic_it6 and 79_optic_it7 networks, due to the sheer size of the underlying graph, it was infeasible to calculate the exact results and as such it necessitates a different comparison method based on metrics that do not require knowledge of the real Pareto front. The first metric (marked as HA) is based on calculating the hyperarea of the Pareto fronts. In this case, a Pareto front is considered to be better if its hyperarea is larger than the other's. The hyperarea is calculated between 0 and $20 + L_{max}$, where L_{max} is the length of the longest route found by any of the algorithms. The extra value of 20 was added to ensure that the last point in the Pareto front still has an effect on the hyperarea. The second metric (denoted DR) is calculated by determining the ratios of dominated points in both ways. This ratio is described in Eq. 8, where A, B are the two Pareto fronts that are compared, where B' is the set of points from B that are dominated by A. Based on this metric, a Pareto front A is deemed better than B if $DR(A,B) > DR(B,A)$.

The results are reported in Table 6 as the average and standard deviation of the number of instances in which one algorithm is better, equal or worse than the other.

$$DR(A,B) = {}^{|B'|}/_{|B|} \tag{8}$$



(a) Between nodes 0 (Vancouver) and 27 (Palm Bay)

(b) Between nodes 0 (Vancouver) and 69 (Jackson)

FIGURE 3: Pareto fronts found by different algorithms on 79_optic_it6 between different node pairs. On (b), the dashed line clarifies that the naive and random_walk solutions dominate the ACO solution below.

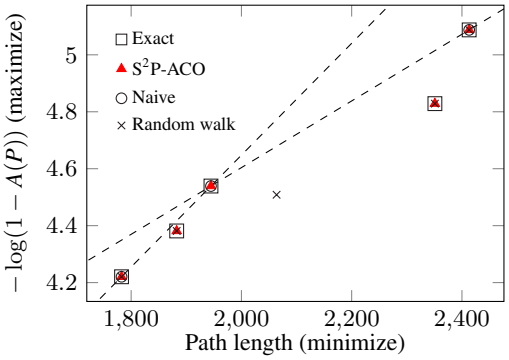


FIGURE 4: Pareto fronts found by different algorithms on graph italy_it7 between nodes 3 (Catania) and 13 (Lausanne). The results that were not found by the naive algorithm actually cannot be found by it in any setting, because the weighted sum method cannot find certain Pareto optimal solutions if the solution space is not convex. Dashed lines symbolize the lines between points found by the naive algorithm. Anything that is under those lines and between the two points cannot be found by the naive algorithm.

Looking at the HA metric, we can see that on average S^{2}P-ACO outperforms the naive algorithm in 41.53% of the cases and is only beaten in 8.44% of them. The comparison with the random_walk algorithm reveals a tighter competition, but still shows a significant amount of 25.55% where S^{2}P-ACO is better and only 8.69% where it is worse. The DR metric is prone to produce equal results; consequently, the win rate for S^{2}P-ACO over the naive algorithm shrinks to 6.59% and the loss rate to 3.77%. The same trend can also be observed in the other comparison based on the *DR* metric; however, the random_walk algorithm performs worse in this test, with 17.61% of the results being worse than S^{2}P-ACO and only 1.71% are better. Two interesting examples are shown in Figures 3a and 3b, the former shows an *st*-pair for which the ACO outperforms the other two algorithms, whereas the latter showcases that it can perform worse. These reinforce the idea that while the S^{2}P-ACO algorithm may not always find the entire Pareto front; most of the paths discovered by it are Pareto optimal.

In conclusion, the results show that the S^{2}P-ACO algorithm outperforms the other two algorithms in most cases and for the networks where we know the optimal results reliably produces them. Out of the 90,520 test cases where the optimal Pareto front is known, our algorithm found it in 87,519, which means a 96.68% success rate. This is greater than in the case of the naive algorithm which stands at 62.12% and the random_walk which is at 87.92%.

2) A case study

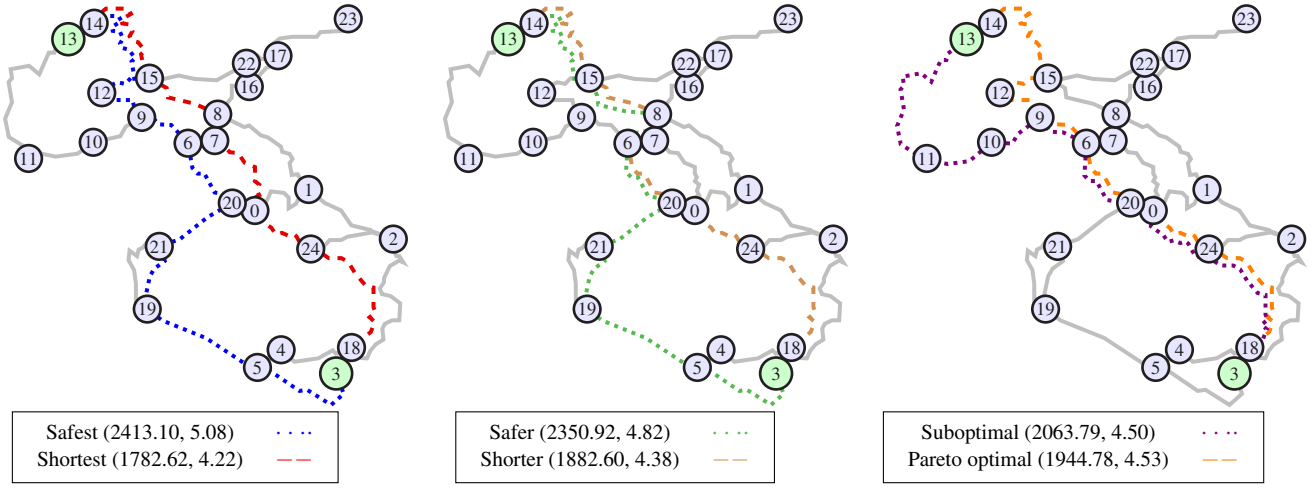
An example that underlines the differences between the algorithms is the path between Catania (node 3) and Lausanne (node 13) in network italy_it7. We chose this network and these two nodes because the results strike a balance between representing the differences in the typical end results of the algorithms and being small enough to be easily interpretable and presentable.

There are 5 Pareto optimal routes between these settlements, and these are depicted in Fig. 4. Both the shortest and the safest paths were found by all algorithms. However, it is worth mentioning that for the naive algorithm, this may not always be the case; as evidenced in [12], a similar naive algorithm misses the safest route. The two aforementioned paths are depicted in Fig. 5a. Comparing the result to the earthquake frequency map depicted in Fig. 6, it is apparent that the blue (dotted) path avoids a large earthquake-prone area between Rome (id 0) and Messina (id 18), resulting in a safer albeit much longer path.

Since the naive algorithm uses a weighted sum method to solve the multiobjective optimization problem, it is unable to find some solutions if the shape of the solution area is not convex [92]. Fig. 5b shows the paths that the naive algorithm was unable to find, but the other algorithms did. This shows the second weakness of this approach, namely that it will not find all valid solutions on the Pareto front.

One drawback of using the random_walk method can be seen in the last case, where it was unable to find a Pareto-optimal solution, resulting in a route that is strongly dominated by the one produced by the other algorithms. The former is the point marked by an X in Fig 4 at $x = 2063.79, y = 4.50$, which is dominated by the solution which has the other markers at $x = 1944.78, y = 4.53$. These paths are depicted in Fig. 5c.

We would like to restate and reinforce the idea that knowing all these Pareto-optimal routes gives flexibility to Internet Service Providers (ISPs). If speed is paramount, the provider may opt for a shorter (i.e. faster), but suboptimally safe route. Similarly, if the aim is to go for the absolute highest availability end-to-end connection, the ISP has to choose the safest option. However, many of these options may also come with additional restrictions; for example, a company may want to opt for the safest option that is faster (equivalently, shorter) than a given threshold. If in the given example, if we place a



(a) The safest (dotted), and shortest (dashed) path. Both of these routes were found by all algorithms.

(b) The two paths that weren't found by the naive algorithm, but were found by all other algorithms. The safer is dotted, and the shorter is dashed.

(c) The suboptimal path found by random_walk algorithm (dotted), and path that dominates it found by the exact, naive and ACO algorithms (dashed).

FIGURE 5: Paths and their lengths and availability-based values $[-\log(1-A(\cdot))]$ between nodes 3 (Catania) and 13 (Lausanne), provided by different algorithms.

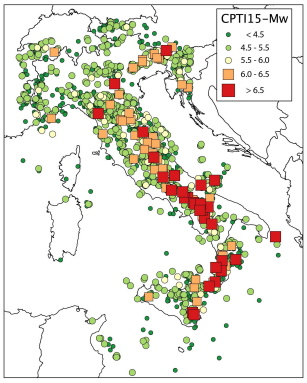


FIGURE 6: Historical earthquakes from a recent version of the historical parametric Italian catalog [91].

hard limit of 2000 on the length, the safest available route is the dashed brown route from Figure 5b, whereas if we have a minimum availability requirement of 99%, the shortest route we can take is the dotted green route from the same figure. Even if these thresholds are not set in stone, having a complete Pareto front available enables making informed decisions efficiently.

3) Runtime analysis

Since each algorithm was implemented in Julia, their runtimes are comparable under similar settings. We ran each algorithm 20 times on each graph and summarized the results in Table 7.

Naturally, the slowest algorithm is the exact algorithm that only managed to calculate 1% of the paths in the case of the 79 node graph, and after 42 days of runtime, we opted to stop the test; hence it is marked as Did Not Finish (DNF). Similarly, in the case of 39_optic_it6, while we did manage to run the exact algorithm a single time to get the results, properly benchmarking the runtime with 20 runs would take an unreasonable amount of time and, as such, it is also marked as DNF.

At the other end of the spectrum, in terms of runtime, sits the naive algorithm, with its maximum runtime not exceeding 1 second even on the largest graph having 79 nodes. The standard deviations of runtimes are also very low here. This behavior is well explained by the fact that this algorithm is based on some arithmetic precomputations that scale linearly with the input size, and then a run of Dijkstra's algorithm for each blending value.

Now we turn to the runtimes of S^{2}P-ACO and random_walk. Fortunately, the typical runtimes of these algorithms are much closer to those of the naive one than to the exact (i.e., they terminated in reasonable time). More concretely for a single st-path, whereas, in the worst case, random_walk slipped just over a minute (with 65.15 [sec]) on network 79_optic_it6, S^{2}P-ACO maxed 40.15 [sec] on the same network.

Comparing the average runtimes of these algorithms shows a more pronounced runtime difference between the two, in favor of S^{2}P-ACO. At first glance, it might be somewhat surprising that the simpler random_walk algorithm runs slower than S^{2}P-ACO. However, we conjecture that the relative speed of S^{2}P-ACO is a consequence of the caching mechanism utilized by both the random_walk and S^{2}P-ACO. Here, in order not to recalculate the fitness of the same path multiple times, the fitness value of each path was stored in a cache at the first encounter at each run of each algorithm. Since the S^{2}P-ACO algorithm explores less wildly than the random_walk algorithm, S^{2}P-ACO could use the cached fitness values more often, thus avoiding costly fitness calculations more frequently.

The rather high cost of fitness evaluation can be observed

TABLE 7: Runtime of algorithms. All values are reported in [sec]. Each algorithm was run 20 times for each unordered node pair in the graph. The runtimes were then averaged by node pairs and then averaged again and that result is presented here. The standard deviation is the deviation of the averages, while the max value is the largest value out of all runs on all node pairs.

Network	S^2P-ACO		naive		random_walk		exact	
	mean $\pm$ std	max	mean $\pm$ std	max	mean $\pm$ std	max	mean $\pm$ std	max
22_optic_it6	4.82 ± 1.63	9.34	0.30 ± 0.01	0.42	29.42 ± 7.65	50.99	69.98 ± 510.5	6774.2
22_optic_it7	1.10 ± 0.36	2.18	0.06 ± 0.002	0.10	6.20 ± 1.63	10.66	12.99 ± 84.34	1166.4
italy_it6	3.40 ± 1.46	8.64	0.59 ± 0.04	1.01	10.11 ± 4.18	23.64	7.45 ± 8.94	47.84
italy_it7	0.28 ± 0.09	0.63	0.03 ± 0.002	0.05	0.55 ± 0.22	1.88	0.40 ± 0.48	2.6
cost266_it6	3.16 ± 1.35	9.41	0.26 ± 0.009	0.42	15.41 ± 5.27	34.24	19.50 ± 56.87	790.3
cost266_it7	0.58 ± 0.20	1.41	0.30 ± 0.01	0.42	2.05 ± 0.70	4.67	2.75 ± 8.74	153.3
janos_us_it6	0.61 ± 0.23	1.60	0.04 ± 0.002	0.07	2.70 ± 1.00	6.03	18.84 ± 61.74	451.8
janos_us_it7	0.24 ± 0.07	0.61	$0.01 \pm 5.7 \times 10^{-4}$	0.02	0.72 ± 0.26	2.12	5.13 ± 16.67	128.7
39_optic_it6	1.38 ± 0.58	4.08	0.10 ± 0.004	0.18	6.59 ± 2.42	14.56	DNF	DNF
39_optic_it7	0.50 ± 0.17	1.26	0.03 ± 0.001	0.07	1.81 ± 0.65	3.87	112.26 ± 353.54	2551.80
79_optic_it6	7.51 ± 4.04	40.15	0.67 ± 0.031	0.98	26.94 ± 10.81	65.15	DNF	DNF
79_optic_it7	1.18 ± 0.54	5.48	0.09 ± 0.003	0.13	3.41 ± 1.35	8.52	DNF	DNF

by looking at the stark differences between graphs with intensity tolerance 6 and 7. Since higher intensity tolerance means there is less chance of an edge breaking (resulting in less seismic hazard data), it makes calculating the exact availability easier, resulting in lower runtimes.

VI. CONCLUSION

A. CONCLUSION

Safest path computation under correlated failures is a challenging computational task. Taking into account not only the quality of edges or edge combinations, but also the length of the path makes the problem more realistic. In this article, we formally introduce the bi-objective safest shortest path problem, and we derive related computational hardness results. To tackle the problem, we design an ACO algorithm, called the Safest-Shortest Path Ant Colony Optimization (S^2P-ACO) algorithm. For comparisons, we selected three baseline algorithms (an exact algorithm, which can deal only with small networks, a naive approach (supposing independent failures), and a random walk-based approach). A detailed discussion based on the results is conducted, including a runtime analysis and relative performance of the algorithms. We conclude that, across the various multi-objective performance metrics investigated, our proposed algorithm S^2P-ACO outperforms the other algorithms in most relevant cases.

B. LIMITATIONS

This work concentrates on static seismic hazards and does not take into account traffic-dependent failures. For the larger graphs, we do not have the optimal Pareto-front and as such the comparisons may not show the full picture. Furthermore, the available datasets (graphs, and more importantly failure data) for this problem are scarce and limited in size.

The authors have provided public access to their code at: https://gitlab.com/VaranTavers/safest_routes_jl

REFERENCES

[1] H. Huang, H. Shen, Z. Meng, H. Chang, and H. He, "Community-based influence maximization for viral marketing," *Applied Intelligence*, vol. 49, no. 6, pp. 2137–2150, 2019.

[2] E. Wong, B. Baur, S. Quader, and C.-H. Huang, "Biological network motif detection: principles and practice," *Briefings in bioinformatics*, vol. 13, no. 2, pp. 202–215, 2012.

[3] A. Schrijver, "On the history of the shortest path problem," *Documenta Mathematica*, vol. 17, no. 1, pp. 155–167, 2012.

[4] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, pp. 269–271, 1959.

[5] R. Bellman, "On a routing problem," *Quarterly of applied mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[6] H. N. Gabow and R. E. Tarjan, "Faster scaling algorithms for network problems," *SIAM Journal on Computing*, vol. 18, no. 5, pp. 1013–1036, 1989.

[7] A. Bernstein, D. Nanongkai, and C. Wulff-Nilsen, "Negative-weight single-source shortest paths in near-linear time," in *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, 2022, pp. 600–611.

[8] R. Duan, J. Mao, X. Mao, X. Shu, and L. Yin, "Breaking the sorting barrier for directed single-source shortest paths," in *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, ser. STOC '25. New York, NY, USA: Association for Computing Machinery, 2025, pp. 36–44.

[9] B. Vass, P. Revisnyei, and A. Pašić, "Computing safest *st*-paths in backbone networks: Efficiently solvable cases and fast heuristics," in *2024 14th International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2024, pp. 1–8.

[10] A. Pašić, R. Girão-Silva, F. Mogyorósi, B. Vass, T. Gomes, P. Babarczi, P. Revisnyei, J. Tapolcai, and J. Rak, "eFRADIR: An Enhanced FRAmework for DIsaster Resilience," *IEEE Access*, vol. 9, pp. 13 125–13 148, 2021. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9319646

[11] B. Vass, L. Birszki, E. Bérczi-Kovács, P. Babarczi, P. Gyimesi, and J. Tapolcai, "Availability-Aware Routing in Presence of Geographically Correlated Failures," in *IEEE INFOCOM 2026 - IEEE Conference on Computer Communications (INFOCOM 2026)*, Tokyo, Japan, May 2026, p. 10.

[12] Z. Tasnádi, B. Vass, and N. Gaskó, "Ant colony optimization algorithm for safest path computation in presence of correlated failures in backbone networks," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2025, pp. 331–339.

[13] P. Chołda, J. Tapolcai, T. Cinkler, K. Wajda, and A. Jajszczyk, "Quality of resilience QoR as a network reliability characterization tool," *IEEE Network Magazine*, vol. 23, no. 2, pp. 11–19, March/April 2009.

[14] Y. Liu, D. Tipper, and P. Siripongwutikorn, "Approximating optimal spare capacity allocation by successive survivable routing," in *INFOCOM 2001*, 2001, pp. 699–708.

[15] V. Y. Liu and D. Tipper, "Spare capacity allocation using shared backup path protection for dual link failures," *Computer Communications*, vol. 36, no. 6, pp. 666–677, 2013.

[16] W. Fawaz, B. Daheb, O. Audouin, M. Du-Pond, and G. Pujolle, "Service level agreement and provisioning in optical networks," *IEEE Communications Magazine*, vol. 42, no. 1, pp. 36–43, 2004.

[17] J. Gozdecki, A. Jajszczyk, and R. Stankiewicz, "Quality of service terminology in ip networks," *IEEE communications magazine*, vol. 41, no. 3, pp. 153–159, 2003.

[18] G. Iannaccone, C.-n. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an ip backbone," in *Proceedings of the 2nd*

ACM SIGCOMM Workshop on Internet measurment. ACM, 2002, pp. 237–242.

[19] RFP RMC Rajkot Way. Available: scribd.com/document/528871820/Rfp-Rmc-Rajkot-Way-Part-2. Accessed: 2024.

[20] X. Long, D. Tipper, and T. Gomes, "Measuring the survivability of networks to geographic correlated failures," *Optical Switching and Networking*, vol. 14, pp. 117–133, 2014.

[21] D. Zhou and S. Subramaniam, "Survivability in optical networks," *IEEE network*, vol. 14, no. 6, pp. 16–23, 2000.

[22] O. Crochat, J.-Y. Le Boudec, and O. Gerstel, "Protection interoperability for WDM optical networks," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 384–395, 2000.

[23] C. S. Ou and B. Mukherjee, *Survivable Optical WDM Networks*. Springer Science & Business Media, 2005.

[24] A. Somani, *Survivability and traffic grooming in WDM optical networks*. Cambridge University Press, 2006.

[25] G. Aceto, A. Botta, P. Marchetta, V. Persico, and A. Pescapé, "A comprehensive survey on internet outages," *Journal of Network and Computer Applications*, vol. 113, pp. 36–63, 2018.

[26] S. Verbrugge, D. Colle, P. Demeester, R. Huelsermann, and M. Jaeger, "General availability model for multilayer transport networks," in *Design of Reliable Communication Networks (DRCN)*, Lacco Ameno, Italy, Oct. 16-19, 2005.

[27] S. Yang, S. Trajanovski, and F. Kuipers, "Availability-based path selection and network vulnerability assessment," *Wiley Networks*, vol. 66, no. 4, pp. 306–319, 2015.

[28] J. Tapolcai, L. Rónyai, B. Vass, and L. Gyimóthi, "List of shared risk link groups representing regional failures with limited size," in *IEEE INFOCOM*, Atlanta, USA, May 2017.

[29] B. Vass, L. Németh, and J. Tapolcai, "The Earth is nearly flat: Precise and approximate algorithms for detecting vulnerable regions of networks in plane and on sphere," *Networks, Wiley*, 2020.

[30] H.-W. Lee, E. Modiano, and K. Lee, "Diverse routing in networks with probabilistic failures," *IEEE/ACM Trans. Netw.*, vol. 18, no. 6, pp. 1895–1907, 2010.

[31] J. Liu, J. Zhang, Y. Zhao, C. Ma, H. Yang, W. Li, J. Xin, and B. Chen, "Differentiated quality-of-protection provisioning with probabilistic SRLG in flexi-grid optical networks," in *OSA Asia Communications and Photonics Conference*, 2013, pp. AF2G–8.

[32] J. Tapolcai, B. Vass, Z. Heszberger, J. Biró, D. Hay, F. A. Kuipers, and L. Rónyai, "A tractable stochastic model of correlated link failures caused by disasters," in *Proc. IEEE INFOCOM*, Honolulu, USA, Apr. 2018.

[33] A. Valentini, B. Vass, J. Oostenbrink, L. Csák, F. Kuipers, B. Pace, D. Hay, and J. Tapolcai, "Network resiliency against earthquakes," in *2019 11th International Workshop on Resilient Networks Design and Modeling (RNDM)*, Oct 2019, pp. 1–7.

[34] B. Vass, J. Tapolcai, Z. Heszberger, J. Bíró, D. Hay, F. A. Kuipers, J. Oostenbrink, A. Valentini, and L. Rónyai, "Probabilistic shared risk link groups modelling correlated resource failures caused by disasters," *IEEE Journal on Selected Areas in Communications (JSAC) - issue on Latest Advances in Optical Networks for 5G Communications and Beyond*, 2021.

[35] J. Oostenbrink and F. Kuipers, "Computing the impact of disasters on networks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 45, no. 2, pp. 107–110, 2017.

[36] S. Neumayer, G. Zussman, R. Cohen, and E. Modiano, "Assessing the vulnerability of the fiber infrastructure to disasters," *IEEE/ACM Trans. Netw.*, vol. 19, no. 6, pp. 1610–1623, 2011.

[37] M. T. Gardner and C. Beard, "Evaluating geographic vulnerabilities in networks," in *IEEE Int. Communications Quality and Reliability Workshop (CQR)*, 2011, pp. 1–6.

[38] S. Trajanovski, F. A. Kuipers, A. Ilić, J. Crowcroft, and P. Van Mieghem, "Finding critical regions and region-disjoint paths in a network," *IEEE/ACM Trans. Netw.*, vol. 23, no. 3, pp. 908–921, 2015.

[39] X. Wang, X. Jiang, A. Pattavina, and S. Lu, "Assessing physical network vulnerability under random line-segment failure model," in *IEEE High Performance Switching and Routing (HPSR)*, 2012, pp. 121–126.

[40] H. Saito, "Analysis of geometric disaster evaluation model for physical networks," *IEEE/ACM Trans. Netw.*, vol. 23, no. 6, pp. 1777–1789, 2015.

[41] ——, "Spatial design of physical network robust against earthquakes," *J. Lightw. Technol.*, vol. 33, no. 2, pp. 443–458, 2015.

[42] F. Dikbiyik, M. Tornatore, and B. Mukherjee, "Minimizing the risk from disaster failures in optical backbone networks," *J. Lightw. Technol.*, vol. 32, no. 18, pp. 3175–3183, 2014.

[43] F. Iqbal and F. Kuipers, "Spatiotemporal risk-averse routing," in *IEEE INFOCOM Workshop on Cross-Layer Cyber Physical Systems Security (CPSS)*, 2016.

[44] M. F. Habib, M. Tornatore, M. De Leenheer, F. Dikbiyik, and B. Mukherjee, "Design of disaster-resilient optical datacenter networks," *J. Lightw. Technol.*, vol. 30, no. 16, pp. 2563–2573, 2012.

[45] I. B. B. Harter, D. Schupke, M. Hoffmann, G. Carle *et al.*, "Network virtualization for disaster resilience of cloud services," *IEEE Commun. Mag.*, vol. 52, no. 12, pp. 88–95, 2014.

[46] B. Mukherjee, M. Habib, and F. Dikbiyik, "Network adaptability from disaster disruptions and cascading failures," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 230–238, 2014.

[47] R. Souza Couto, S. Secci, M. Mitre Campista, K. Costa, and L. Maciel, "Network design requirements for disaster resilience in IaaS clouds," *IEEE Commun. Mag.*, vol. 52, no. 10, pp. 52–58, 2014.

[48] P. K. Agarwal, A. Efrat, S. K. Ganjugunte, D. Hay, S. Sankararaman, and G. Zussman, "The resilience of WDM networks to probabilistic geographical failures," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1525–1538, 2013.

[49] P. N. Tran and H. Saito, "Geographical route design of physical networks using earthquake risk information," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 131–137, 2016.

[50] H. Honda and H. Saito, "Nation-wide disaster avoidance control against heavy rain," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, pp. 1084–1097, 2019.

[51] L. Pašić, A. Pašić, F. Mogyorósi, and A. Pašić, "FRADIR meets availability," in *2020 16th International Conference on the Design of Reliable Communication Networks DRCN 2020*. IEEE, 2020, pp. 1–6.

[52] S. Yang, S. Trajanovski, and F. A. Kuipers, "Shortest paths in networks with correlated link weights," in *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2015, pp. 660–665.

[53] M. General and J. E. Pezoa, "Reliable path provisioning in the presence of multiple correlated failures," in *2016 IEEE 37th Sarnoff Symposium*. IEEE, 2016, pp. 1–2.

[54] W. Cui, I. Stoica, and R. H. Katz, "Backup path allocation based on a correlated link failure probability model in overlay networks," in *10th IEEE International Conference on Network Protocols, 2002. Proceedings.* IEEE, 2002, pp. 236–245.

[55] A. Kumar, M. Nadeem, and H. Banka, "Nature inspired optimization algorithms: a comprehensive overview," *Evolving Systems*, vol. 14, no. 1, pp. 141–156, 2023.

[56] K. Gao, Z. Cao, L. Zhang, Z. Chen, Y. Han, and Q. Pan, "A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 4, pp. 904–916, 2019.

[57] R. Alkhalifa, F. Alkhomayes, B. Almazroua, D. Alhaidan, M. Alothman, and J. Almuhaidib, "A comparative review of parallel exact, heuristic, metaheuristic, and hybrid optimization techniques for the traveling salesman problem," *arXiv preprint arXiv:2505.18278*, 2025.

[58] H. Alghamdi, T. Alsubait, H. Alhakami, and A. Baz, "A review of optimization algorithms for university timetable scheduling," *Engineering, Technology & Applied Science Research*, vol. 10, no. 6, pp. 6410–6417, 2020.

[59] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: An autocatalytic optimizing process," *TR91-016, Politecnico di Milano*, 1991.

[60] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[61] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *biosystems*, vol. 43, no. 2, pp. 73–81, 1997.

[62] ——, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[63] T. Stützle and H. H. Hoos, "Max–min ant system," *Future generation computer systems*, vol. 16, no. 8, pp. 889–914, 2000.

[64] D. K. Gupta, Y. Arora, U. K. Singh, and J. P. Gupta, "Recursive ant colony optimization for estimation of parameters of a function," in *2012 1st International Conference on Recent Advances in Information Technology (RAIT)*. IEEE, 2012, pp. 448–454.

[65] S. Tsutsui, "cAS: Ant colony optimization with cunning ants," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2006, pp. 162–171.

[66] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *European journal of operational research*, vol. 185, no. 3, pp. 1155–1173, 2008.

[67] S. Iredi, D. Merkle, and M. Middendorf, "Bi-criterion optimization with multi colony ant algorithms," in *International conference on evolutionary multi-criterion optimization*. Springer, 2001, pp. 359–372.
[68] I. Ariyasinghe and T. Fernando, "A performance study for the multi-objective ant colony optimization algorithms on the job shop scheduling problem," *International Journal of Computer Applications*, vol. 132, pp. 1–8, 12 2015.
[69] G. Leguizamón and C. A. Coello Coello, "Multi-objective ant colony optimization: A taxonomy and review of approaches," in *Integration of swarm intelligence and artificial neural network*. World Scientific, 2011, pp. 67–94.
[70] J. G. Falcón-Cardona, G. Leguizamón, C. A. Coello Coello, and M. G. Castillo Tapia, "Multi-objective ant colony optimization: an updated review of approaches and applications," *Advances in Machine Learning for Big Data Analysis*, pp. 1–32, 2022.
[71] M. A. Awadallah, S. N. Makhadmeh, M. A. Al-Betar, L. M. Dalbah, A. Al-Redhaei, S. Kouka, and O. S. Enshassi, "Multi-objective ant colony optimization," *Archives of Computational Methods in Engineering*, vol. 32, no. 2, pp. 995–1037, 2025.
[72] S.-H. Ok, W.-J. Seo, J.-H. Ahn, S. Kang, and B. Moon, "An ant colony optimization approach for the preference-based shortest path search," *Journal of the Chinese Institute of Engineers*, vol. 34, no. 2, pp. 181–196, 2011.
[73] C. Horoba and D. Sudholt, "Ant colony optimization for stochastic shortest path problems," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 2010, pp. 1465–1472.
[74] K. Ghoseiri and B. Nadjari, "An ant colony optimization algorithm for the bi-objective shortest path problem," *Applied soft computing*, vol. 10, no. 4, pp. 1237–1246, 2010.
[75] Z. Tasnádi, B. Vass, and N. Gaskó, "An ant colony optimization approach for safest path pair computation under correlated failures," in *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 2025, pp. 183–196.
[76] Y. Ikeda and M. Inoue, "An evacuation route planning for safety route guidance system after natural disaster using multi-objective genetic algorithm," *Procedia computer science*, vol. 96, pp. 1323–1331, 2016.
[77] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan, "Detecting high log-densities – an $O(n^{1/4})$ approximation for densest k-subgraph," New York, NY, USA, p. 201–210, 2010. [Online]. Available: https://doi.org/10.1145/1806689.1806719
[78] S. Yuan, S. Varma, and J. P. Jue, "Minimum-color path problems for reliability in mesh networks," in *Proc. IEEE INFOCOM*, 2005, pp. 2658–2669.
[79] R. Hassin, J. Monnot, and D. Segev, "Approximation algorithms and hardness results for labeled connectivity problems," *Journal of Combinatorial Optimization*, vol. 14, no. 4, pp. 437–453, 2007.
[80] P. Zhang, J.-Y. Cai, L.-Q. Tang, W.-B. Zhao, and A. Li, "Approximation and hardness results for label cut and related problems," *Journal of Combinatorial Optimization*, vol. 21, no. 2, pp. 192–208, 2011.
[81] H. Wirth, "Multicriteria approximation of network design and network upgrade problems," Ph.D. dissertation, Universität Würzburg, 2001.
[82] M. R. Fellows, J. Guo, H. F. Moser, and R. Niedermeier, "Some parameterized intractability results for minimum label path and related problems," *Journal of Computer and System Sciences*, vol. 77, no. 4, pp. 799–811, 2011.
[83] N. Kumar, "Computing a minimum color path in edge-colored graphs," in *Analysis of Experimental Algorithms - Special Event*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:109937250
[84] J.-A. M. de Dios and E. Mezura-Montes, "Metaheuristics: A julia package for single- and multi-objective optimization," *Journal of Open Source Software*, vol. 7, no. 78, p. 4723, 2022. [Online]. Available: https://doi.org/10.21105/joss.04723
[85] A. Sieberg, "Erdbeben," *Handbuch der Geophysik*, vol. 4, pp. 552–554, 1931.
[86] O. Schutze, X. Esquivel, A. Lara, and C. A. C. Coello, "Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 504–522, 2012.
[87] D. A. Van Veldhuizen, *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*. Air Force Institute of Technology, 1999.
[88] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms—a comparative case study," in *International conference on parallel problem solving from nature*. Springer, 1998, pp. 292–301.
[89] C. A. C. Coello and N. C. Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic programming and evolvable machines*, vol. 6, no. 2, pp. 163–190, 2005.
[90] A. Vargas and J. Bogoya, "A generalization of the averaged hausdorff distance," *Computación y Sistemas*, vol. 22, no. 2, pp. 331–345, 2018.
[91] A. Rovida, M. Locati, R. Camassi, B. Lolli, and P. Gasperini, "Cpti15, the 2015 version of the parametric catalogue of italian earthquakes," *Istituto Nazionale di Geofisica e Vulcanologia*, 2016. [Online]. Available: https://doi.org/10.6092/INGV.IT-CPTI15
[92] M. Ehrgott, "Multiobjective optimization," *AI Magazine*, vol. 29, pp. 47–57, 12 2008.



ZOLTÁN TASNÁDI received his MSc in data analytics and modeling from Babeș-Bolyai University, Cluj-Napoca, in 2022 and is currently a PhD student at the same university. His research focuses on applying nature-inspired algorithms to difficult graph problems.



BALÁZS VASS received his MSc in applied mathematics from Eötvös Loránd University (ELTE), Budapest, in 2016, and his PhD in informatics from the Budapest University of Technology and Economics (BME) in 2022. He is currently an associate professor at Babeș-Bolyai University. His research focuses on network survivability, combinatorial optimization, and graph algorithms. He has served on the Technical Program Committee of IEEE INFOCOM since 2023 and is a recipient of a 2023 MSCA Postdoctoral Fellowship.



NOÉMI GASKÓ is an Associate Professor at the Faculty of Mathematics and Computer Science, Babeș-Bolyai University, Cluj-Napoca, Romania. Her research interests include game theory, evolutionary algorithms, optimization, and complex networks. Her work focuses on developing computational intelligence methods for solving problems in game theory and network analysis. She has co-authored several scientific publications in international journals and conferences.