Ant Colony Optimization Algorithm for Safest Path Computation in Presence of Correlated Failures in Backbone Networks

Zoltán Tasnádi
Babeş-Bolyai University
Cluj-Napoca, Romania
zoltan.tasnadi@ubbcluj.ro

Balázs Vass
Babeş-Bolyai University
Cluj-Napoca, Romania,
Budapest University of Technology and Economics, Hungary
balazs.vass@ubbcluj.ro

Noémi Gaskó
Babeş-Bolyai University
Cluj-Napoca, Romania
noemi.gasko@ubbcluj.ro

ABSTRACT

Safest path computation with multiple correlated failures is a challenging computational task, with several application possibilities. In communication backbone networks, for example, establishing a path as safe as possible between the two communication endpoints is a crucial component for achieving the ambitious availability requirements on which emerging technologies like autonomous driving, AR/VR applications, or telesurgery depend. In this paper, after proving the $\mathcal{NP}$-hardness of the problem, we propose the Safest Path Ant Colony Optimization (SP-ACO) algorithm to solve the problem. The proposed algorithm is based on the Max-Min Ant System. Numerical experiments conducted on both real-world and synthetic inputs prove the effectiveness of the proposed approach. The proposed SP-ACO algorithm typically provides at least as safe paths as the state-of-the-art algorithms, even outperforming them in a significant share of the parameter settings. This grants a place for the SP-ACO among the best solutions for safest path finding in the presence of correlated failures.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**.

KEYWORDS

safest path problem; correlated failures; Ant Colony Optimization

ACM Reference Format:

Zoltán Tasnádi, Balázs Vass, and Noémi Gaskó. 2025. Ant Colony Optimization Algorithm for Safest Path Computation in Presence of Correlated Failures in Backbone Networks. In *Genetic and Evolutionary Computation Conference (GECCO '25), July 14–18, 2025, Malaga, Spain.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3712256.3726346

1 INTRODUCTION

Studying computational network problems gained huge interest in recent decades due to their large application possibilities, for example, the community detection problem in E-Commerce [1] or link prediction in network completion problems [2]. A concrete, well-studied problem is to find a shortest path between two nodes s and t of a graph G. If the aim is to compute an st-path using the least edges possible, a simple Breadth-First Search (BFS) can be used. For non-negative edge weights, Dijkstra's algorithm [3] is the best solution, while for arbitrary weights, a Bellman-Ford [4] suits well. The task of finding a *safest* st-path in some simple settings proves to be solvable by using a shortest path finding subroutine, after some transformations [5]. However, the case when the failures of the network elements are correlated is not well understood yet. Thus, the primary goal of this study is to assemble a performant safest path finding algorithm in the presence of correlated failures. While our real-world numerical problem inputs are on communication backbone networks paired with seismic hazard data, we believe the algorithm designed in this paper can be efficiently used on a wide range of problem inputs.

1.1 Safest paths in backbone networks

Computing safest paths and evaluating availability between two network nodes assuming independent single-element failures has a long history, see e.g., former studies focusing on the protection of communication backbone networks [6–11]. Remaining at the concrete use-case of communication networks, dealing with multiple failures has its traditions, relying on the concept of Shared Risk Groups (SRGs) (e.g.,[6–9, 12, 13]). An SRG consists of a few network elements that are considered to have a high chance of failing together (e.g., links traversing the same bridge). In many of the applications, the failure of a network node v has the same effect as the failure of all the links incident to v. In these cases, it is enough to deal with a list of Shared Risk *Link* Groups (SRLGs), each SRLG consisting only of links.

Probabilistic extensions of SRLGs were also investigated [14–18]. Lately, [18] proposed a straightforward unified terminology related to the Probabilistic SRLGs (PSRLGs) that will be adopted by this paper.

A natural approach (also taken by this paper) is to take the disaster scenarios as input [19], that have been carefully precomputed by dedicated approaches, e.g., based on historical hazard data. Much of the work in this field tackled disaster modeling more heuristically in their own way to address their given problem in network planning. Some examples are determining the most vulnerable network part [20–23], estimating the damage in the network if a random disaster hits [24–26], (re)routing connections in order to minimize the impact of disasters [27, 28], and resiliency-aware network design and extension [29–32].

GECCO '25, July 14–18, 2025, Malaga, Spain

ACM ISBN 979-8-4007-1465-8/2025/07
https://doi.org/10.1145/3712256.3726346

By now, efficient methods for computing and storing the correlated link failures are available [18] (instead of limiting the set of disasters to a small number, or assuming the independence of link failures [33–35]). These methods are already in use in complex frameworks for disaster resilience [36, 37].

From an algorithmic point of view, several variants of the problem, and different solution concepts were proposed. In [38], two correlated link weight models are proposed (link weights can have different meanings, e.g., delay, failure probability), and the shortest path problem is studied on these models based on the Dijkstra algorithm. [39] presents some characterization of correlated failures and the k-shortest path problem is solved. In [40] single and double link failures are studied and an Integer Quadratic Programming problem is formulated. Generally, considering link correlations in shortest-path algorithms can significantly impact network reliability and performance.

1.2 Nature-inspired algorithms for safest path computation

Nature-inspired algorithms [41] are powerful optimization tools for solving hard optimization problems. Regarding variants of the safest route problem and nature-inspired solving methods, in [42], a multi-objective genetic algorithm is proposed for evacuation route planning, considering three objective functions: evacuation distance, evacuation time, and the safety of the evacuation route.

Considering the variant of the safest path problem studied in this article - to the best of our knowledge - nature-inspired algorithms were not studied. The use of Ant colony optimization algorithms in this kind of graph-based combinatorial optimization problem is straightforward. Current literature studied simple variants of shortest path problems, in [43] a running time analysis of different ACO systems is presented for the base shortest paths problem. In [44] an ACO algorithm is presented to solve the bi-objective shortest path problem. The authors of [45] study a stochastic shortest path problem with ACOs, where the weights of edges are subject to noise (meaning delays or uncertainty). The article [46] presents an ant colony system algorithm for finding the shortest path with preferred edges.

1.3 Main contributions

The main contributions of the paper are as follows:

- We formalize and prove the $\mathcal{NP}$-hardness of the problem of finding a safest path in presence of correlated failures.
- The Ant Colony Optimization algorithm (ACO) is adopted for solving the problem.
- Through extensive simulations, we show that the resulting Safest Path ACO (SP-ACO) algorithm is typically at least on par with the state-of-the-art algorithms, in a significant share of the simulation settings yielding safer paths than former algorithms.

The rest of the paper is structured as follows: Sec. 2 formally defines the problem, and presents some basic results, including the $\mathcal{NP}$-hardness. Sec. 3 describes the proposed Ant Colony Optimization algorithm. Numerical evaluations on both real-world and synthetic inputs are presented in Sec. 4. Finally, Sec. 5 concludes the paper and discusses future research directions.

2 PROBLEM STATEMENT AND PROOF OF $\mathcal{NP}$-HARDNESS

The problem input consists of two main parts. One is a connected graph $G = (V, E)$, along with a communication source-target node pair $\{s, t\} \subseteq V$. The other part of the problem input encodes the probabilities of joint failures link sets. For this, for a link set $S \subseteq E$, in line with [18, 37], we define CFP(S) (that stands for 'cumulative failure probability of S') to denote the probability that *at least* link set S will fail at the next disaster. The second part of the input is CFP$[G]$, which is a data structure containing all the CFP(S) values, where we list CFP(S) only if CFP(S) > 0. Note that in most of the natural settings, CFP$[G]$ has a manageable size [18]. We note that albeit CFP$[G]$ stores only link failures, it is suitable for implicitly storing node failure probabilities, too; see [18, Sec. V.]. The goal is to find a safest path among a node pair s and t, i.e., an st-path with lowest chance that any of its links fails under the next disaster.

Below, we give a more formal definition of the above concept, followed by a proof of $\mathcal{NP}$-hardness of the decision version of the safest path problem. CFPs can be defined as follows.

*Definition 2.1 (**Cumulative Failure Probability (CFP)**).* Given a set of links $S \subseteq E$, 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).

It is easy to see that if for a link set S, CFP(S) > 0, then all the $2^{|S|} - 1$ non-empty subsets of S have to be stored in CFP$[G]$. Intuitively, when there are only a couple of failure scenarios, which, on the other hand, cause the destruction of many network elements, there should be a more compact way of storing the network failure hazard. The straightforward and provably more compact representation is the so-called Link Failure State Probability [18], that we mention in our paper, too:

*Definition 2.2 (**Link Failure State Probability (FP)**).* Given a set of links $S \subseteq E$, 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).

Whenever it does not cause confusion, we will refer as 'CFP' to both 1) the tuple $(S, \text{CFP}(S))$ for a link set S, and 2) simply, to CFP(S). The same goes for 'FP'. Intuitively, FP$[G]$ and CFP$[G]$ are interconnected in a similar way as the density and cumulative density functions.

Next, we define two interconnected versions of the safest path finding problem with correlated link failures:

Problem 1: Safest Path CFP Problem - decision version

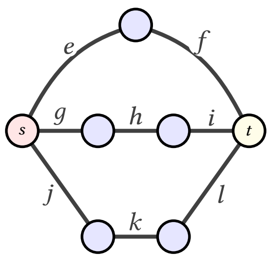
Input: A graph $G = (V, E)$, nodes s and t, a threshold T, and failure probabilities CFP$[G]$.
Question: Decide whether there exists an st-path that, after the next disaster, remains intact with a probability of at least T. If such a path exists, the instance is *satisfiable.*

As we will see in Thm. 2.4, the above problems are $\mathcal{NP}$-hard. Note that the $\mathcal{NP}$-hardness of the *decision* versions indicate the $\mathcal{NP}$-hardness of the *optimization* versions of Problems 1 and 2. That is, it is $\mathcal{NP}$-hard to find a safest st-path if link failures are correlated.

Problem 2: Safest Path FP Problem - decision version

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

Question: Decide whether there exists an st-path that, after the next disaster, remains intact with a probability of at least T. If such a path exists, the instance is *satisfiable*.



$\mathrm{FP}[G]$:
$\mathrm{FP}(e) = 0.2$ $\mathrm{FP}(f) = 0.2$
$\mathrm{FP}(g) = 0.2$ $\mathrm{FP}(g, h, i) = 0.1$
$\mathrm{FP}(j, l) = 0.25$

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

Method	Path yielded	Availability perceived by method			Real availability
		Indep.	E.d.	SP-ACO	
Independent	$\{e, f\}$	0.64	0.6	0.6	0.6
Edge-dual [5]	$\{g, h, i\}$	0.567	0.7	0.7	0.7
SP-ACO (Ours)	$\{j, k, l\}$	0.5625	0.5	0.75	0.75

Figure 1: A toy example on the input graph G, related failure probabilities stored in either $\mathrm{FP}[G]$ or $\mathrm{CFP}[G]$. The table included depicts the supposedly safest paths yielded by different methods in this example, along with the availability perceived by the methods and the real availability of the path. Note that, in reality, availabilities are much closer to 1, thus the mistakes made by Independent and Edge-Dual methods are less obvious.

Note that while we will mainly work with the less compact structure of $\mathrm{CFP}[G]$, the $\mathcal{NP}$-hardness of Problem 2 hints that the hardness does not arise because of an unfortunate phrasing of the input.

Example 2.3. Fig. 1 depicts a simple example of the problem inputs, along with the results of some of the (heuristic) optimization algorithms for finding a safest st-path (that will be briefly presented in the following).

Note that despite the inherent hardness of our *optimization* problem (to be proved in Thm. 2.4), known the graph G, and the list $\mathrm{FP}[G]$ of link sets of positive FP, there is a straightforward way to evaluate the *real* availability of $A(P)$ a given st-path P, formalized in Alg. 1. Intuitively, a path P fails if a failure event cuts at least one link on the path:

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

Note that it is not a problem that Alg. 1 possibly neglects some link sets S having a nonempty intersection with path P, that are not in $\mathrm{FP}[G]$, since their cumulative failure probability is zero by definition.

Algorithm 1: Computing the availability of a path P

```
Input: Path P = {e_1, ..., e_k} in graph G = (V, E), failure
       probabilities FP[G]
Output: Availability A(P) of path P
A := 1
for S ∈ FP[G] do
    if S ∩ P ≠ ∅ then
        A := A − FP(S)
return A
```

Unfortunately, it is not obvious how $\mathrm{FP}[G]$ should be used for optimizing the availability of an st-connection. In fact, both existing optimization methods (the Independent and the Edge-dual, briefly introduced in the following) and our SP-ACO algorithm take advantage of the cumulative failure probabilities $\mathrm{CFP}[G]$, which are possibly much more numerous than $\mathrm{FP}[G]$. 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 $\mathrm{CFP}[G]$, and counting their CFP with the right sign, the following sum is calculated:

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

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

Returning to Fig. 1, we can see that the three methods return with three different paths. This is because of the following. The traditionally used *Independent* method maximizes the product of the availabilities of the links:

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

This way, it assumes the best solution is to go from s to t along links e and f. This simulation 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 here can be done by a simple Dijkstra algorithm [5].

Taking advantage of some graph transformations, the *Edge-dual* heuristic of [5] takes into account only those CFPs that are connected and have a diameter at most two. Thus, for a path $P = \{e_1, \ldots, e_i\}$, it maximizes the following expression:

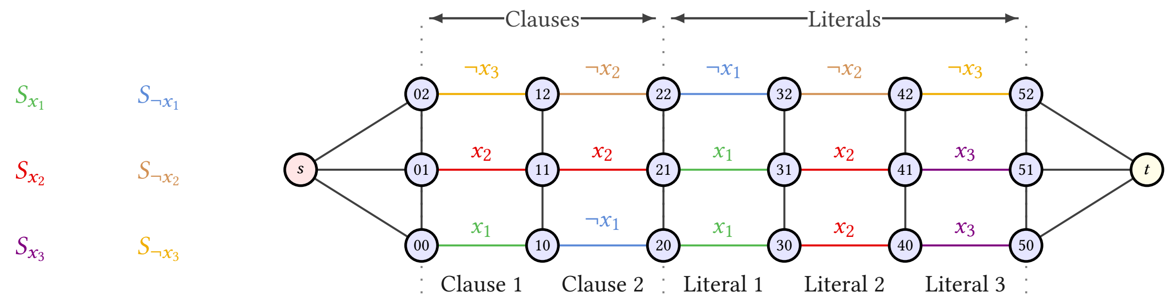
$$A_{\text{edge-dual}}(P) = 1 - \sum_{j=1}^{i} \mathrm{CFP}(\{e_j\}) + \sum_{j=1}^{i-1} \mathrm{CFP}(\{e_j, e_{j+1}\}) \tag{4}$$

In the example of Fig. 1, the edge-dual method fails to find an optimal solution since it underestimates the availability of path $\{j, k, l\}$ by neglecting $\mathrm{CFP}(j, l)$.

Contrary to the above heuristics, the SP-ACO introduced in this paper finds the optimal solution.

For clarity, we explicitly express here that Eq. (1) and Eq. (2) (used by our algorithm SP-ACO) define correct assessments of the availability of a given path P; however, Eq. (3) and Eq. (4) are just useful, easily optimizable formulations that help the Independent and Edge-dual methods *guess* the best solutions.

Now we turn to prove the $\mathcal{NP}$-hardness of the problem at hand. Intuitively, the main idea of the proof is to transform a 3-SAT problem instance f to a safest path problem instance in which there is



(a) Link sets with positive faliure probability.

(b) Graph G created based on 3-SAT instance f

Figure 2: Example of a reduction from a 3-SAT instance $f = (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor \neg x_2)$. The instance has $k = 2$ clauses and $\rho = 3$ literals. There are 2ρ link sets S in FP$[G]$ / CFP$[G]$, each with FP(S) = CFP(S) = $1/2\rho$. Since the 'right side' of the graph, at least 1 out of the 2 link sets related to each literal has to be traversed by each st-path P, thus, $A(P) \le 1/2$. Because of the 'left side' of the graph, $A(P) = 1/2$ is achievable exactly if 3-SAT instance f is satisfiable.

an st-path with availability reaching $1/2$ if and only if f is satisfiable, that is $\mathcal{NP}$-hard to decide.

THEOREM 2.4. *Problems* `Safest Path FP` *and* `Safest Path CFP` *(Problems 2 and 1) are $\mathcal{NP}$-hard.*

PROOF. First, we will provide a polynomial-time reduction from the $\mathcal{NP}$-complete 3-SAT problem [47] to (the decision variant of) the `Safest Path FP` problem. The $\mathcal{NP}$-hardness of `Safest Path CFP` will be proved via an extra observation at the end of this proof.

Suppose we are given a 3-SAT problem instance f, that is a conjunctive normal form, standing of clauses containing exactly three literals each. That is, $f = C_1 \land C_2 \land \cdots \land C_k$, with each C_i ($i \in \{1, \ldots, k\}$) equalling $(l_{i,1} \lor l_{i,2} \lor l_{i,3})$. The 3-SAT problem is to determine whether f is satisfiable. In the following, we construct a `Safest Path FP` problem instance that is satisfiable exactly if f is satisfiable and has an input of size $O(k)$ that can be computed from f in $O(k)$.

Going into details, known formula f that has k clauses, and ρ variables, we define graph $G = (V, E)$ as follows. Let the set of nodes be

$$V := \{v_{i,j} \mid i \in \{1, 2, 3\}, j \in \{1, k + \rho + 1\}\} \cup \{s, t\}.$$

The set of edges is

$$E := \{\{v_{1,j}, v_{2,j}\}, \{v_{2,j}, v_{3,j}\} \mid j \in \{1, k + \rho + 1\}\} \cup$$
$$\cup \{\{v_{i,j}, v_{i,j+1}\} \mid i \in \{1, 2, 3\}, j \in \{1, \ldots, k + \rho\}\} \cup \{\{s, v_{1,1}\},$$
$$\{s, v_{2,1}\}, \{s, v_{3,1}\}, \{v_{1,k+\rho+1}, t\}, \{v_{2,k+\rho+1}, t\}, \{v_{3,k+\rho+1}, t\}\}.$$

Intuitively, G can be drawn as a $3 \times (k + \rho)$ grid graph, with s and t appended to the nodes on the first and last columns, as depicted in Fig. 2. Remaining at this example drawing of G, intuitively, the links that may fail at the next disaster are exactly the horizontal ones, as explained in the following.

Each link set S that has a positive failure probability (FP$(S) > 0$) is set to have a failure probability FP$(S) = \frac{1}{2\rho}$. For each variable x_σ ($\sigma \in \{1, \ldots, \rho\}$), we define two such link sets (this makes a total of 2ρ link sets of positive failure probability; thus the failure probabilities add up to 1 as expected). The first set, S_σ contains each link $\{v_{b,\sigma}, v_{b,\sigma+1}\}$ ($b \in \{1, 2, 3\}, \sigma \in \{1, \ldots, \rho\}$) for which, in the σ^{th} clause, the b^{th} literal is x_σ (as a positive literal); and, in addition, S_σ contains link $\{v_{1,k+\sigma}, v_{1,k+\sigma+1}\}$. The second link set associated to x_σ, denoted by $S_{\neg\sigma}$, contains each link $\{v_{b,\sigma}, v_{b,\sigma+1}\}$ ($b \in \{1, 2, 3\}, \sigma \in \{1, \ldots, \rho\}$) for which in the σ^{th} clause the b^{th} literal is $\neg x_\sigma$ (as a negative literal); and, in addition, $S_{\neg\sigma}$ contains links $\{v_{2,k+\sigma}, v_{2,k+\sigma+1}\}$ and $\{v_{3,k+\sigma}, v_{3,k+\sigma+1}\}$ (see Fig. 2).

The probability threshold T is set to $\frac{1}{2}$. Since any st-path P has to pass at least one of links $\{v_{i,k+\sigma}, v_{i,k+\sigma+1}\}$, $i \in \{1, 2, 3\}$, it has to intersect either S_σ or $S_{\neg\sigma}$. Thus, the survival probability of path P cannot be higher than $\frac{1}{2}$. Further, we can observe that the availability of P can be (at least) $\frac{1}{2}$ precisely if the 3-SAT problem instance f is satisfiable. This proves the $\mathcal{NP}$-hardness of the `Safest Path FP` problem (Problem 2).

In the construction presented above, for each S, FP(S) = CFP(S), thus the same construction is also a proof of the $\mathcal{NP}$-hardness of problem `Safest Path CFP` (Problem 1). □

We note that the above proof of $\mathcal{NP}$-hardness resembles a reduction scheme that, in different versions, can be found in various places, like the proof of hardness in [48].

3 PROPOSED ALGORITHM: THE SAFEST PATH - ANT COLONY OPTIMIZATION ALGORITHM (SP-ACO)

Our proposed algorithm for computing safest paths in presence of correlated failures is based on the Ant Colony Optimization (ACO) algorithm [49], more precisely, its Min-Max Ant System variant. ACO is a nature-inspired algorithm that is based on the communication techniques employed by ant colonies. Individual ants communicate through the use of pheromone trails in a form of indirect communication.

The ACO algorithm is governed by the following settings: heuristic information, pheromone setting, and solution generation.

As heuristic information, our algorithm uses the negative logarithm of the probability that the given edge breaks (ignoring probabilities that group multiple edges together). In case an edge does not have an associated probability, it is given a very small $\epsilon_{\text{prob}} > 0$ probability of breaking, since logarithms cannot work with zeros.

This ϵ has to be significantly lower than other failure probabilities in the original list of failure probabilities CFP$[G]$. For our tests, we used $\epsilon_{\text{prob}} = 10^{-8}$. In case two nodes are not neighbors, the heuristic information will be set to 0 in order to prevent ants from choosing them. This is described in Eq. (5), where $\mathcal{N}_i$ is the set of neighbors of i and $e_{i,j}$ is the edge between nodes i and j.

$$\eta_{ij} = \begin{cases} -\log \text{CFP}(\{e_{i,j}\}), & \text{if } j \in \mathcal{N}_i \\ 0, & \text{else.} \end{cases} \tag{5}$$

The pheromone setting is mostly governed by the rules of the Max-Min Ant System. In order to reduce the chance of getting stuck in local maxima, after the first iteration the pheromone levels will be limited to be between two values calculated from the best solution that we have encountered. This is described in Eq. (6) and (7), where f_{gb} is the fitness of the best solution found, ρ is the evaporation coefficient, and ϵ is the pheromone proportion coefficient:

$$\tau_{\max} = f_{gb}/(1-\rho) \tag{6}$$

$$\tau_{\min} = \epsilon \cdot \tau_{\max} \tag{7}$$

We only distribute pheromones on the best path taken within the given iteration. The combination of pheromone placement and evaporation is described in Eq. (8), where τ_{ij} is the pheromone level between nodes i and j:

$$\tau_{ij}^{(t)} = (1-\rho) \cdot \tau_{ij}^{(t-1)}. \tag{8}$$

On the best route found, the pheromone update is described with Eq. (9), the negative logarithm of the probability of the given path failing is considered:

$$\tau_{ij}^{(t)} = (1-\rho) \cdot \tau_{ij}^{(t-1)} - \log(1 - A(P)). \tag{9}$$

The solution generation in our algorithm places the ant on the chosen start point and moves it to other vertices based on the calculated probabilities until it either reaches the endpoint or exceeds the maximum available moves that it has. If it cannot reach the endpoint, the path that it produces will be discarded. The calculation of the aforementioned probabilities is described in Eq. (10), 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:

$$p(i,j) = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{v_q \in \mathcal{N}_i} (\tau_{iq})^{\alpha}(\eta_{iq})^{\beta}}, \text{ if } v_j \in \mathcal{N}_i \tag{10}$$

The SP-ACO algorithm works as follows. All ants are placed in the starting point s. Based on the heuristic information and pheromone level they choose the next node. Nodes can be visited only once, the last visited node must be t. If an ant cannot reach the endpoint, no path will be returned. The path generation algorithm is detailed in Algorithm 3.

Until the stopping criterion is fulfilled, which in this case is the reaching of the maximum number of iterations, the next steps are repeated: Global and iteration-best paths are computed. In each step, the new pheromone limits are calculated, the ant with the highest fitness may place pheromones on their chosen path, while it evaporates at a constant rate from edges that are not part of that path. The resulting method is formalized in Algorithm 2.

Algorithm 2: Shortest Path - Ant Colony Optimization (SP-ACO)

Input: Graph $G = (V, A)$, cumulative failure probabilities CFP[G], ACO parameters: $\alpha, \beta, \rho, \epsilon$, nrOfAnts; nrOfIterations, nodes s and t
Output: A safest st-path found
Initialize pheromone trails
$i := 0$
while $i <$ `nrOfIterations` **do**
    $S := \emptyset$
    repeat
        Construct a new path P based on Algorithm 3
        $S := S \cup \{P\}$
    until $|S| =$ `nrOfAnts`;
    Calculate the iteration-best and global-best paths: $P_{\mathtt{ib}}$ and $P_{\mathtt{best}}$, respectively
    Compute pheromone trail limits ($\tau_{\mathtt{min}}, \tau_{\mathtt{max}}$) based on Eq. (6), (7)
    Update pheromone trail on P_{ib} based on Eq. (8)
    $i := i + 1$
return $P_{\mathtt{best}}$

Algorithm 3: Path generation for the SP-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
$P := list(s)$
$v :=$ zeros$(|V|)$ `// remember previously visited nodes`
$v[s] := 1$
while $P_{end} \neq t$ *and* $|P| < |V|$ **do**
    $j := 0$
    $r := s$
    while $v[r] = 1$ *and* $j <$ `maxTries` **do**
        $r :=$ new node from $\mathcal{N}_{P_{end}}$ based on Eq. (10) $j := j + 1$
    if $j =$ *maxTries* **then**
        return nothing `// the ant failed to generate a viable solution`
    $v[r] := 1$
    $P :=$ push(P, r)
if $|P| = |V|$ *and* $P_{end} \neq t$ **then**
    return nothing `// the ant failed to generate a viable solution`
return P

4 NUMERICAL EVALUATION

4.1 Simulation settings

4.1.1 Implementation. For conducting our extensive simulation experiments, each algorithm (Independent, Edge-dual, and SP-ACO) was implemented in the same programming language, namely Julia. Tests were run on a Framework 13 Laptop with an AMD Ryzen™ 5 7640U, 32 GB of RAM, and a Fedora Linux 41 operating system.

4.1.2 Benchmarks. For benchmarks, we use synthetic and real-world problems. Synthetic benchmarks were generated according to the following rule: cumulative failure probabilities were generated randomly for a single link between 0.005 and 0.1 and for two and three links between $5 \cdot 10^{-8}$ and 0.005. All generated networks have a grid structure. Table 1 describes more details about the generated

Table 1: Generated synthetic random networks with associated synthetic failure data - basic properties

Name	$\lvert V\rvert$	$\lvert E\rvert$	Failure data
n20_1	20	31	$\lvert CFP(S_1)\rvert$=31, $\lvert S_1\rvert$=1,$\lvert CFP(S_2)\rvert$=10, $\lvert S_2\rvert$=2,$\lvert CFP(S_3)\rvert$=10, $\lvert S_3\rvert$=3
n20_2	20	31	$\lvert CFP(S_1)\rvert$=31, $\lvert S_1\rvert$=1,$\lvert CFP(S_2)\rvert$=15, $\lvert S_2\rvert$=2,$\lvert CFP(S_3)\rvert$=15, $\lvert S_3\rvert$=3
n20_3	20	31	like for network n20_1, S_2, and S_3 are connected
n20_4	20	31	like for network n20_2, S_2, and S_3 are connected
n40_1	40	67	$\lvert CFP(S_1)\rvert$=67, $\lvert S_1\rvert$=1,$\lvert CFP(S_2)\rvert$=15, $\lvert S_2\rvert$=2,$\lvert CFP(S_3)\rvert$=15, $\lvert S_3\rvert$=3
n40_2	40	67	$\lvert CFP(S_1)\rvert$=67, $\lvert S_1\rvert$=1,$\lvert CFP(S_2)\rvert$=20, $\lvert S_2\rvert$=2,$\lvert CFP(S_3)\rvert$=20, $\lvert S_3\rvert$=3
n40_3	40	67	like for network n40_1, S_2, and S_3 are connected
n40_4	40	67	like for network n40_2, S_2, and S_3 are connected

Table 2: Real-world problem inputs

Network name	$\lvert V\rvert$	$\lvert E\rvert$	Network from	Failure data
22_optic	22	45	[13]	[18]
Italy	25	34	[17]	[18]
cost266	37	57	[13]	[18]
janos_us	26	42	[13]	[18]

networks; in column Type, the number of single, double and triple link failures $|\text{CFP}(S_1)|$, $|\text{CFP}(S_2)|$, and $|\text{CFP}(S_3)|$; and the type of the failures (by default, non-connected, but there are connected variants, as well) are presented, respectively.

Basic properties of the real-world networks are presented in Table 2. The number of nodes, the number of edges, and the source of the network are detailed in this table. For the numerical experiments, we used a variety of seismic hazard inputs taken from [18].

4.1.3 Parameter tuning. To test the proposed SP-ACO algorithm, we run 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 heuristic information strength and pheromone strength. The parameter tuning was performed on a real-world network, the janos_us graph. Results are presented in Table 3. Mean values, standard deviations, and maximum values are presented over 10 independent runs. The table also presents the generation number, where the last improvement takes place (globally from the ten runs). Most configurations resulted in similar results, and the configuration that was ultimately used is $\alpha = 1.0$, $\beta = 1.5$. Other parameters used for the rest of the experiments are: 25 ants, 200 iterations, $\epsilon = 0.1$, $\rho = 0.3$.

4.1.4 Comparisons with other methods. For comparisons, we use two methods: the Independent Method (indep) [5] that for each edge e, assigns $-\log(\text{CFP}(\{e\}))$ as its weight, and uses a simple Dijkstra to obtain the safest path. The Edge-Dual Method (edge-dual) was proposed in [5], it constructs a so-called edge-dual graph, and assigns edge weights calculated on the basis of Equation (4). After that, it also uses the Dijkstra to compute a (supposedly) safest path.

Table 3: Results of parameter tuning (ten independent runs) on janos_us graph containing the mean, standard deviation, best result (max), and the last iteration where an improvement was found by the SP-ACO algorithm (out of 200 generations)

α	β	$A(P_{\text{SP-ACO}})$ mean ± std	max	last generation of change
0.5	0.5	**20 ± 0.00**	**20**	190
1	0.5	**20 ± 0.00**	**20**	193
1.5	0.5	**20 ± 0.00**	**20**	107
2	0.5	**20 ± 0.00**	**20**	177
0.5	1	**20 ± 0.00**	**20**	200
1	1	**20 ± 0.00**	**20**	123
1.5	1	**20 ± 0.00**	**20**	84
2	1	**20 ± 0.00**	**20**	180
0.5	1.5	**20 ± 0.00**	**20**	200
1	1.5	**20 ± 0.00**	**20**	**50**
1.5	1.5	**20 ± 0.00**	**20**	143
2	1.5	**20 ± 0.00**	**20**	186
0.5	2	**20 ± 0.00**	**20**	200
1	2	**20 ± 0.00**	**20**	66
1.5	2	**20 ± 0.00**	**20**	144
2	2	**20 ± 0.00**	**20**	178

4.2 Results and discussion

4.2.1 General behaviors. For each network, experiments were conducted for each possible *st* pair (totally $\binom{|V|}{2}$ experiments, where $|V|$ denotes the number of nodes).

Let $P_{\text{SP-ACO}}$, P_{indep}, and $P_{\text{edge-dual}} = P_{\text{e-d}}$ denote the safest paths obtained by SP-ACO, the Independent, and the Edge-Dual method, respectively. Thus, $A(P_{\text{SP-ACO}})$, $A(P_{\text{indep}})$, and $A(P_{\text{edge-dual}})$ denote their availabilities.

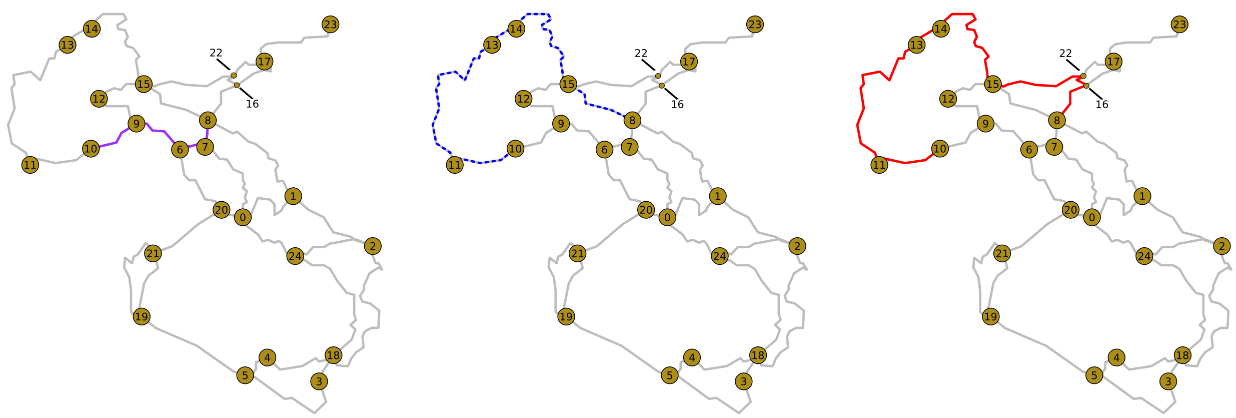
Table 4 presents the results obtained for both the synthetic and real-world problem inputs. In the case of the synthetic inputs with graphs having 20 nodes (n20_3, n20_4) there were no significant differences between the three methods. When the set of failure probabilities was not connected (n20_1 and n20_2) in a few cases the SP-ACO algorithm found better paths. For the synthetic networks with 40 nodes, the behavior of SP-ACO was different: for all four networks our algorithm found improvements regarding the independent method, and for the first three networks regarding the Edge-Dual method.

For setting n40_4, from the 780 different cases, for 716 cases obtained the same results (considering the best SP-ACO path), and for 64 cases, the Edge-Dual method found safer paths as the SP-ACO algorithm. At the same time, for setting n40_1, the tendency is reversed: the SP-ACO outperformed the Edge-Dual method for 86 cases, and for 694 cases, there was no difference between them. An explanation for this could be that SP-ACO can deal better with non-connected failure probabilities.

Regarding the nine real-world graph- failure data tuples, SP-ACO outperformed the independent method in eight cases and the Edge-Dual method in three cases. For two settings (22_optic_it7 and cost266_it7), it is significant that the result of the SP-ACO: SP-ACO obtained the better results as the Edge-Dual method in 59.74%, respectively 11.41% of the total cases (231, respectively 666 node pairs).

Table 4: Comparisons of the availability of the st-paths. Results of synthetic benchmarks and real-world networks, based on 30 independent runs, the mean, std, and max represent the number of node-pairs on which the SP-ACO algorithm performed better than the independent, the equal symbolizing the ones where it was equal, while the next column shows the number of node-pairs where the independent outperformed the mean SP-ACO result. In the table, the $P^*_{\text{SP-ACO}}$columns symbolize the results when compared to the best result of the SP-ACO algorithm. The fifth column shows the number of node pairs where the Edge-Dual method outperformed the independent method. The last three columns compare the SP-ACO algorithm with the edge-dual method in the same way: number of node-pairs, when SP-ACO outperformed the Edge-Dual method, when it was equal, and in the last column, the Edge-Dual method outperformed our proposed method.

Graph+failure setting	$A(P_{\text{SP-ACO}}) >$ $A(P_{\text{indep}})$		$A(P_{\text{SP-ACO}})=$ $A(P_{\text{indep}})$		$A(P_{\text{SP-ACO}}) <$ $A(P_{\text{indep}})$		$A(P_{\text{e-d}}) >$	$A(P_{\text{SP-ACO}}) >$ $A(P_{\text{e-d}})$		$A(P_{\text{SP-ACO}})=$ $A(P_{\text{e-d}})$		$A(P_{\text{SP-ACO}}) <$ $A(P_{\text{e-d}})$	
	mean ± std	$P^*_{\text{SP-ACO}}$	mean	$P^*_{\text{SP-ACO}}$	mean	$P^*_{\text{SP-ACO}}$	$A(P_{\text{indep}})$	mean	$P^*_{\text{SP-ACO}}$	mean	$P^*_{\text{SP-ACO}}$	mean	$P^*_{\text{SP-ACO}}$
n20_1	2 ± 0	2	188	188	0	0	0	2	2	188	188	0	0
n20_2	4 ± 0	4	186	186	0	0	0	1	1	189	189	0	0
n20_3	0 ± 0	0	190	190	0	0	0	0	0	190	190	0	0
n20_4	0 ± 0	0	190	190	0	0	0	0	0	190	190	0	0
n40_1	84.76 ± 0.77	86	692	694	4	0	0	84	86	685	694	11	0
n40_2	7.8 ± 0.40	8	769	772	6	0	12	1	1	763	775	16	4
n40_3	6.0 ± 0.00	6	771	774	3	0	5	5	5	767	775	8	0
n40_4	2.33 ± 0.54	3	771	777	10	0	67	0	0	684	716	96	64
22_optic_it6	1.00 ± 0.00	1	229	230	1	0	0	0	0	230	230	1	0
22_optic_it7	78.00 ± 0.00	78	153	153	0	0	1	138	138	93	93	0	0
janos_us_it6	1.57 ± 0.82	3	321	322	1	0	3	0	0	321	325	4	0
janos_us_it7	0.00 ± 0.00	0	321	325	5	0	0	0	0	320	325	5	0
janos_us_995	20.00 ± 0.00	20	305	305	0	0	20	0	0	321	325	4	0
cost266_it6	5.03 ± 0.18	6	657	660	3	0	6	0	0	662	666	4	0
cost266_it7	11.00 ± 0.00	11	654	655	1	0	5	76	76	589	590	1	0
cost266	33.00 ± 0.00	33	632	633	1	0	33	0	0	665	666	1	0
italy_it6	31.00 ± 0.00	31	269	269	0	0	31	0	0	300	300	0	0
italy_it7	58.00 ± 0.00	58	242	242	0	0	58	0	0	300	300	0	0
italy_995	30.00 ± 0.00	30	270	270	0	0	21	9	9	291	291	0	0



(a) **Unavailability of $P_{\text{least hop}} = P_{\text{geo. shortest}}$: 145.44% of best found**

(b) **Unavailability of $P_{\text{indep}} = P_{\text{e-d}}$: 100.67% of best found**

(c) **Unavailability of $(P_{\text{SP-ACO}})$: best found**

Figure 3: Routes found by the different algorithms on the italy_995 graph between Bologna (id 8) and Monaco (id 10), alongside their relative unavailabilities. The unavailability of a path P is just $1 - A(P)$.

As a general conclusion, we can mention that SP-ACO outperformed the independent method in almost all settings. The SP-ACO did not found better st-paths as by the Independent method only in 3 out of the 19 tested settings (namely, in $n20\_3$, $n20\_4$, and janos_us_it7). Regarding the Edge-Dual method for eight settings our algorithm found better solutions, but almost in all cases found similar results.

As an advantage of the SP-ACO algorithm we can emphasize the stability of the algorithm, over the 30 independent runs the standard deviation in most cases is equal to 0.

4.2.2 A case study. Figure 3 presents one of the cases where the SP-ACO algorithm found a safer route than the Edge-Dual and independent algorithms. Here, because of the subtle differences between the availabilities of the paths, we investigate their unavailabilities. The unavailability $U(P)$ of path P is defined just as $1 - A(P)$. On a), we can see that, under similar circumstances, the downtime of the custom path through nodes $8 - 7 - 6 - 9 - 10$, which is both a least-hop graph and is the shortest path between Bologna and Monaco, is almost 45% more than the downtime of our best path found. This shortest path could be seen as a reasonable choice if there is no

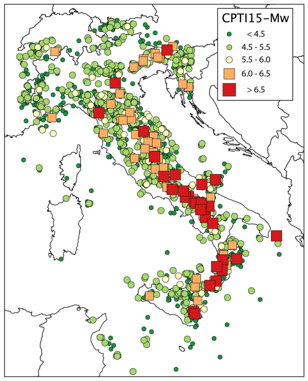


Figure 4: Historical earthquakes from a recent version of the historical parametric Italian catalog [50]. M_w **= moment magnitude.**

knowledge of the possible disasters, as it minimizes the amount of network infrastructures that are exposed to possible disasters. On Fig. 3b), we can see the path using nodes $8 - 15 - 14 - 13 - 11 - 10$ computed by both the Independent and the Edge-dual methods, which achieves a near-best unavailability, exceeding it by 0.67%. Since our disaster data is the one computed by [17], which takes into account the earthquakes affecting *Italy* (based on recorded earthquake history, see Fig. 4), we can see that these methods basically correctly figure out that, starting from node 8, it worths to head north to node 15, and to leave the Italian territory to make a huge detour through Switzerland and France to reach node 10 (i.e., Monaco). We note again that the used disaster data leaves out the earthquakes that are not affecting Italian territory, and as such, it can be seen as incomplete for this network topology; however, it perfectly suits our purposes of exemplification of the outputs of the algorithms studied here. The output of the best-performing algorithm, our proposed SP-ACO, is depicted in Fig. 3c). Basically, it provides the same route as the Independent and the Edge-dual, except that it realizes that instead of edge $\{8, 15\}$ it is worth making a detour through nodes $8 - 16 - 22 - 15$. In fact, the unavailability of the detoured path is more than 1% lower than that of edge $\{8, 15\}$. Note that the significance of this detour is *real* in the sense that the earthquake scenarios that should be taken into account *are* taken into consideration. With this, the resulting path $P_{\text{SP-ACO}}$ has the highest availability, or equivalently, the lowest unavailability among those found, namely $\simeq 1.135 \cdot 10^{-2}$.

In the following, we translate the above differences between the availabilities of paths $P_{\text{indep}} = P_{\text{e-d}}$ and $P_{\text{SP-ACO}}$ availability/ unavailability to yearly downtimes. First of all, in fact, $A(P)$ denotes the probability that path P will fail *when the next disaster strikes*. In Italy, the expected number of earthquakes is $r = 5.53$ (considered events that have a strength of $> 4.5M_w$) [18]. For the sake of estimation, we apply a Mean Time To Repair (MTTR) of 24 hours, equaling 1440 minutes [37] (this MTTR might be an optimistic under-estimation in case of earthquakes). With this, the expected downtime differences of the above paths due to earthquakes in Italy can be calculated as follows:

$$(1 - A(P_{\text{SP-ACO}})) \cdot (0.6657\%) \cdot r \cdot \text{MTTR} \simeq 0.60[\texttt{min/year}].$$

Note that while an expected downtime difference of roughly 36[`sec/year`] might not seem a lot, actually, it alone already violates the Quality of Service requirements, if an availability of six-nines is prescribed (that is a common prescription). This is because, in that case, the connection should be working in 99.9999% of the time, which translates to a maximum allowed downtime of around 31.5[`sec/year`].

Table 5: Runtimes of the compared algorithms on italy_it6 graph from node Rome (id 0) to node Bari (id 2) over 5 runs. The results are presented in seconds.

Method	Runtime [`sec`]: mean ± std
Independent	$(2.17 \pm 0.4) \times 10^{-4}$
Edge-Dual	$(3.93 \pm 0.6) \times 10^{-3}$
SP-ACO	$(6.37 \pm 0.32) \times 10^{0}$

4.2.3 *Runtime analysis.*

Since each algorithm was implemented in Julia, runtimes of the algorithms under similar settings are comparable. The times were measured by calling each algorithm 5 times for italy_it6 and between the same points, namely Rome (id 0) and Bari (id 2). These nodes were chosen to avoid including the JIT compilation time for the calculation of the first route when running the algorithm.

The results are presented in Table 5, where we can see that the ACO algorithm, while still completing in a reasonable amount of time, lags behind the other two algorithms. This is consistent with the general expectations of the difference in the runtime between heuristic algorithms and greedy algorithms. The independent method sets the weights of the graph and runs a Dijkstra algorithm on it, while the Edge-dual builds an auxiliary graph and runs the same algorithm on the second, hence the relatively small but noticeable difference between the runtime of the two algorithms.

Evaluating the fitness function takes a considerable amount of time in each iteration of the ACO algorithm, however, due to caching results, in later iterations, which do not yield a previously not seen path, can be more efficient with times as low as 5.81×10^{-4} compared to the first iteration that averages around 0.635.

5 CONCLUSION AND FUTURE WORK

Safest path computation is a challenging computational task with several application possibilities, for example, in dependable communication networks for emerging and future mission-critical applications. In the case where multiple correlated failures can appear the problem is $\mathcal{NP}$-hard. In this article, we propose an Ant Colony Optimization algorithm (SP-ACO) to solve the problem. The cumulative failure probability values are used in the heuristic information, and in the pheromone update rules. Numerical experiments were conducted on both constructed grid-type synthetic networks and on real-world problems. Comparisons with existing methods (the independent method, and the Edge-Dual algorithm) prove the effectiveness of the proposed approach. Future work will include analysis of different failure types and parallelization of the proposed SP-ACO algorithm to deal with larger networks.

ACKNOWLEDGMENTS

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.

REFERENCES

[1] J. Reichardt and S. Bornholdt, "Clustering of sparse data via network communities—a prototype study of a large online market," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2007, no. 06, p. P06016, 2007.

[2] M. Kim and J. Leskovec, "The network completion problem: Inferring missing nodes and edges in networks," in *Proceedings of the 2011 SIAM international conference on data mining.* SIAM, 2011, pp. 47–58.

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

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

[5] 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).*

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

[7] 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.

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

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

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

[22] 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.

[23] 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.

[24] 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.

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

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

[27] 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.

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

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] A. Pašić, R. G. ao 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

[38] 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.

[39] 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.

[40] 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.

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

[42] 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.

[43] D. Sudholt and C. Thyssen, "Running time analysis of ant colony optimization for shortest path problems," *Journal of Discrete Algorithms*, vol. 10, pp. 165–180, 2012.

[44] 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.

[45] 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.

[46] 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.

[47] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department.* Springer, pp. 85–103.

[48] J. Oostenbrink and F. Kuipers, "Going the extra mile with disaster-aware network augmentation," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–10.

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

[50] 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