Finding critical regions and region-disjoint paths in a network

Stojan Trajanovski, *Student Member, IEEE,* Fernando A. Kuipers, *Senior Member, IEEE,* Aleksandar Ilić, Jon Crowcroft, *Fellow, IEEE, ACM,* and Piet Van Mieghem, *Member, IEEE*

***Abstract*—Due to their importance to society, communication networks should be built and operated to withstand failures. However, cost considerations make network providers less inclined to take robustness measures against failures that are unlikely to manifest, like several failures coinciding simultaneously in different geographic regions of their network.**

Considering networks embedded in a two-dimensional plane, we study the problem of finding a critical region - a part of the network that can be enclosed by a given elementary figure of predetermined size - whose destruction would lead to the highest network disruption. We determine that only a polynomial, in the input, number of non-trivial positions for such a figure need to be considered and propose a corresponding polynomial-time algorithm. In addition, we consider region-aware network augmentation to decrease the impact of a regional failure. We subsequently address the region-disjoint paths problem, which asks for two paths with minimum total weight between a source (s) and a destination (d) that cannot both be cut by a single regional failure of diameter D (unless that failure includes s or d). We prove that deciding whether region-disjoint paths exist is NP-hard and propose a heuristic region-disjoint paths algorithm.

***Index Terms*—Survivability, Geographical failures, Critical regions, Region-disjoint paths, Network augmentation.**

I. Introduction

LINK AND NODE FAILURES in vital infrastructures, such as the Internet, power grids or mobile networks may be caused unintentionally, for instance due to (aged) equipment failure, power failure, natural disasters, or intentionally, for example by terrorist attacks or cyber criminals [1]. The category of multiple network failures at several geographic locations occurring in a short time interval is less likely to take place.

Increasing the robustness of a network usually requires installing redundant resources or over-provisioning, which is very costly. Thus, in general, network providers are only inclined to take preventive robustness measures for events that have a realistic chance of occurring, such as a single, rather than multiple, regional failure. If a network is robust, in terms of connectivity, survivability algorithms are needed to exploit this robustness by quickly rerouting traffic affected by a network failure. The single link- or node-failure scenario has been most studied or assumed by the research community and various link- or node-disjoint paths algorithms have been devised to find two disjoint paths, where one backup path takes over when the other primary path fails (e.g., see [2], [3]). Currently, the fast reroute mechanism, specified in [4] and extended in [5], aims to improve the performance of standard network protocols such as OSPF by pre-determining the alternative (backup) paths. Moreover, the mechanism has already been implemented by several vendors, which shows the need for efficient disjoint-paths algorithms. Typically, attacks or natural disasters often affect a geographical area. For instance, devastations from the 2012 catastrophic hurricane "Sandy" in the US stretched from the East Coast to the Lake Area [6]. Such failure areas cannot always be sufficiently accurately approximated by circular shapes, as often considered in papers on regional failures (e.g., [7]), and require considering other two-dimensional figures, such as ellipses and various polygons.

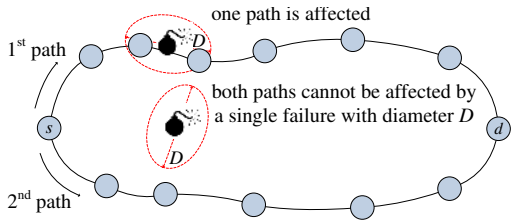


Fig. 1: An example of two region-disjoint paths that cannot both be cut by the failure of a single region with diameter D. The problem holds for any figure with diameter D.

This paper considers the context of a single regional failure by investigating two main problems. We start with the problem of finding a *critical region* in the network identified by a predetermined two-dimensional figure whose failure (of the nodes inside that region) would be most disruptive to the network. The second problem under consideration assumes demands for protected connections arrive in an on-line fashion and asks for two *region-disjoint paths*, between source s and destination d, with a minimum total weight, such that each intermediate node (between s and d) from the first path is on a distance greater than the diameter of a given figure from every intermediate node in the second path. In this case, no single regional failure can destroy both paths unless that failure affects s and d (Fig. 1).

Our main contributions can be summarized as follows:

- We prove that only a polynomial number of figure positions (for ellipses and general polygons) need to be

S. Trajanovski, F.A. Kuipers and P. Van Mieghem are with Delft University of Technology, Faculty of Electrical Engineering, Mathematics and Computer Science, P.O. Box 5031, 2600 GA Delft, The Netherlands (e-mails: {S.Trajanovski, F.A.Kuipers, P.F.A.VanMieghem}@tudelft.nl).

A. Ilić is with Facebook Inc., Menlo Park, CA, USA (e-mail: ailic@fb.com).

J. Crowcroft is with the University of Cambridge, Computer Laboratory, William Gates Building, 15 JJ Thomson Avenue, Cambridge CB3 0FD, United Kingdom (e-mail: Jon.Crowcroft@cl.cam.ac.uk).

considered for finding a critical region and propose a corresponding polynomial-time algorithm;
- We consider network augmentation for improving the robustness against regional failures;
- We prove that the decision variant of the region-disjoint paths problem is NP-hard and propose a polynomial-time heuristic for the region-disjoint paths problem;
- We determine the impact of a critical region failure in real-world networks for different figures of equal area and evaluate our heuristic path selection algorithm.

This paper is structured as follows. An overview of related work is given in Section II. Our formal model and problem statement are defined in Section III. In Section IV we reduce the, in principle, infinite size of the search space of possible locations for the figures to a search space of polynomial size and provide an accompanying algorithm for detecting critical regions. Furthermore, we discuss network augmentation to make the network more resilient against the failure of a region. The NP-hardness of the region-disjoint paths problem is proved and, subsequently, a heuristic is proposed in Section V. Section VI identifies the critical regions and evaluates the proposed algorithms in real-world networks. Concluding remarks are given in Section VII.

II. Related Work

The level to which *connectivity* can be maintained under failures has typically been used as the main metric to characterize network robustness (see [3] for an overview), for instance by finding a given number of nodes/links that reduce connectivity most [8]. Much work on the robustness of a network against geographical failures has been done by Neumayer *et al.* [7], [9], [10] for a different failure model than the one presented in this paper and only for circular and line failures. While most papers confine to the circular failure model, in this paper we consider ellipses and general polygons that allow to better capture reality.

Previous work, e.g. on critical regions and region-disjoint max-flow problems [7], [10], often assumes that the failure of a region affects all nodes and links, even those links that are only traversing and have no terminating nodes in that region. While this may correspond to a realistic scenario in the case of an earthquake, it may be too restrictive for countries not on a fault line or for other scenarios, e.g. for wireless, sensor or radio networks, floods, etc. In the model considered in this paper, we consider that the failure of a region disrupts all nodes and their attached links inside that region, but not any traversing links. For a different problem, the same failure model has been used in [11]. Agarwal *et al.* [12], [13] propose exact and approximation algorithms for a probabilistic geographical failure model that affects a "network component" (e.g., a node, link or lightpath), inversely-proportional to the distance from a point in the network. They also give an overview of the related computational geometry literature. Contrary to their model, in our model the shape and size of the failure figure are predefined. Moreover, not only the failure figure center matters in our model, but also how the predefined figure is positioned (rotated) along that center. Banerjee *et al.* [14] have taken a different application domain, namely that of distributed file storage in a network, in which data resiliency is provided to regional failures.

The problem of finding link- or node-disjoint paths in a network between two nodes has been widely explored under different scenarios, e.g. see [2], [3], [15]–[20]. Suurballe [15] proposed a polynomial-time algorithm to find k node- or link-disjoint paths with minimum total weight (i.e., the min-sum objective). Subsequently, an optimized algorithm was proposed by Suurballe and Tarjan [16] to find 2 link- or node-disjoint paths from a source to all other nodes. Finding disjoint paths for other objective functions has been studied in [17], [21], [22]. Contrary to the min-sum objective, these objectives typically lead to NP-hard problems. Sen *et al.* [23] have studied a related decision variant of the region-disjoint paths problem. In this paper we study finding the *optimal* pair of region-disjoint paths.

III. Model and Problem Statement

We start with a presentation of our network model and the problems considered.

Model: We represent a network as a weighted (directed or undirected) graph $G(\mathcal{N}, \mathcal{L})$ in a plane consisting of a set $\mathcal{N}$ of N nodes and a set $\mathcal{L}$ of L links. Each node $i \in \mathcal{N}$ has two-dimensional coordinates (x_i, y_i). The Euclidean distance between two nodes u and v is denoted by $d(u, v)$. The weight of a link $(i, j) \in \mathcal{L}$ is denoted by $w(i, j)$. The link weight in a communication network may represent different properties, for instance its capacity, delay, length, cost, or failure probability.

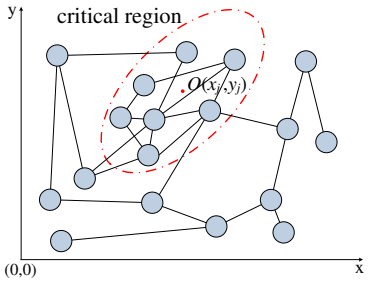


Fig. 2: Critical region.

We define the *critical region* $\mathcal{C}(\mathcal{F}, X)$ as the region covered by the position of the figure $\mathcal{F}$ in the two-dimensional plane for which the removal of all nodes in that area, and the links incident to them, leads to a maximum deterioration in a certain network metric X. The network metric X could for instance represent the number of affected nodes, the average shortest path length, the number of connected node pairs, or some service function like packet loss or average delay. Fig. 2 presents an example of a network for which a critical region is identified. There might be multiple critical regions that affect the metric X to the same degree.

To characterize the position of a figure, we should determine its *center* $O(x_j, y_j)$ and *orientation* φ, which is the angle ($0 \leq \varphi \leq 2\pi$) between the x-axis and an axis of symmetry in the figure as shown in Fig. 3.

We will consider several figures as shown in Fig. 3, namely the circle $\mathcal{F}_{\mathcal{C}}\left(O(x_j, y_j), r\right)$ with radius r, the ellipse $\mathcal{F}_{\mathcal{E}}\left(O(x_j, y_j), \varphi, a, b\right)$ with semi-axes lengths a and b, the rectangle $\mathcal{F}_{\mathcal{R}}\left(O(x_j, y_j), \varphi, a, b\right)$ with side lengths a and b, and general polygons $\mathcal{F}_{\mathcal{P}}$. In the remainder, we use the term *dimensions* to refer to radii or sides. We also denote by D the diameter of an arbitrary two-dimensional figure, which is the maximum distance between two points within the figure. In particular, the diameters of $\mathcal{F}_{\mathcal{C}}\left(O(x_j, y_j), r\right)$, $\mathcal{F}_{\mathcal{E}}\left(O(x_j, y_j), \varphi, a, b\right)$, $\mathcal{F}_{\mathcal{R}}\left(O(x_j, y_j), \varphi, a, b\right)$ and $\mathcal{F}_{\mathcal{P}}$ are $2r$, $2\max\{a, b\}$, $\sqrt{a^2 + b^2}$, and the largest diagonal, respectively.

For a given arbitrary two-dimensional shape, two sets of nodes $\mathcal{A}$ and $\mathcal{B}$ are *region disjoint* if each node $a \in \mathcal{A}$ is on a distance greater than the diameter of the figure from every node in $\mathcal{B}$. Two paths are *region disjoint* if the set of intermediate nodes in the first path is region disjoint with the set of intermediate nodes in the second path. In some cases, the first hop from s may lie within the region of s, in which case no region-disjoint paths could exist. In that case we consider the region of s (and similarly d) and its incoming or outgoing links as a single source node attached to those incoming and outgoing links. We can now define our two main problems.

Critical region problem: For a given network $G(\mathcal{N}, \mathcal{L})$ embedded in a plane, find a *critical region* $C(\mathcal{F}, X)$ with respect to network metric X and two-dimensional figure $\mathcal{F}_{fig}\left(O(x_j, y_j), \varphi, dim\right)$, where *dim* is the vector of dimensions that defines $\mathcal{F}_{fig}$, $fig \in \{\mathcal{C}, \mathcal{E}, \mathcal{S}, \mathcal{P}\}$.

Region-disjoint paths problem: Given a network $G(\mathcal{N}, \mathcal{L})$ with positive link weights and the diameter of an arbitrary two-dimensional figure, find two *region-disjoint paths* from s to d, with a minimum total weight as reflected by the sum of the link weights in the two paths.

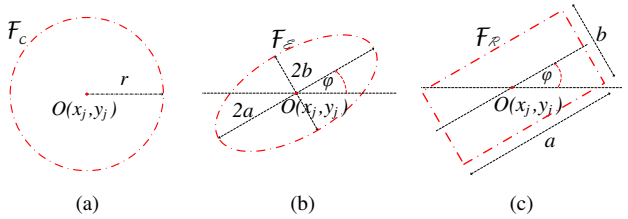


Fig. 3: (a) circle $\mathcal{F}_{\mathcal{C}}\left(O(x_j, y_j), r\right)$; (b) ellipse $\mathcal{F}_{\mathcal{E}}\left(O(x_j, y_j), \varphi, a, b\right)$; and (c) rectangle $\mathcal{F}_{\mathcal{R}}\left(O(x_j, y_j), \varphi, a, b\right)$.

IV. Critical Region Detection and Mitigation

We first demonstrate that finding critical regions of a given two-dimensional figure is solvable in polynomial time for the ellipse and any polygon.

A. Theoretical basis

In this section, we use three kinds of geometric transformations defined in Definition 1.

Definition 1: Translation of a figure is the motion in parallel to a given line (e.g., the x-axis in Fig. 4). *Rotation* of a figure ([24]) along a given node assures that the distance between a point on the perimeter of the figure and that node remains the same (Fig. 4). Finally, we define *sliding* along two nodes as moving the position of the figure such that these two nodes still lie on the perimeter of the figure. The sliding differs for different figures as visualized in Fig. 5.

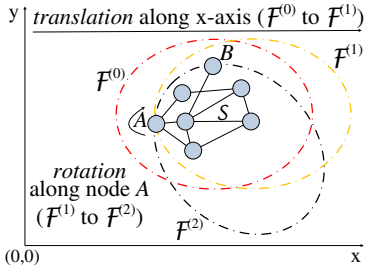


Fig. 4: *Translation* and *Rotation*, illustrated for the ellipse.

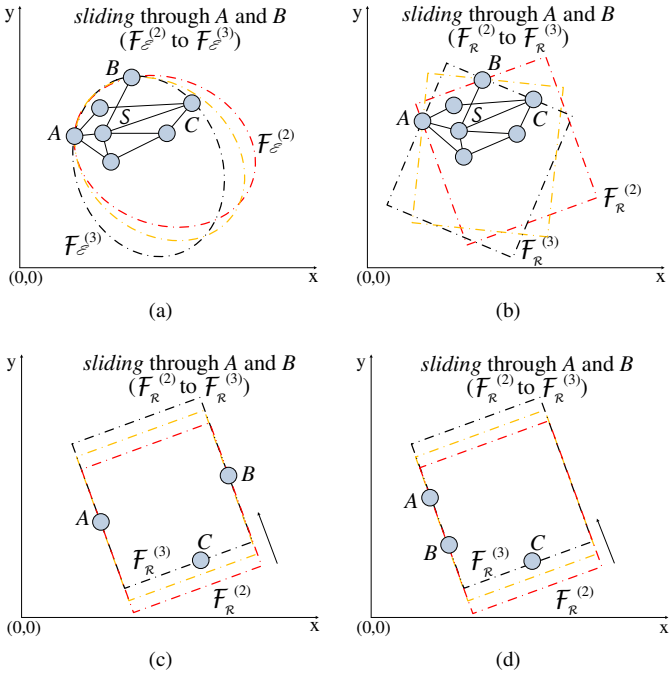


Fig. 5: *Sliding* through A and B for: (a) an ellipse; (b) a rectangle if A and B lie on perpendicular sides; (c) a rectangle if A and B lie on parallel sides; and (d) a rectangle if A and B lie on a same side.

Theorem 1: For any polygon or ellipse, if there exists a two-dimensional figure that covers a set of nodes S, then that same set S can also be covered by the same type of figure that is characterized by and passes through 3 nodes.

Proof: We start from an arbitrary two-dimensional figure $\mathcal{F}$. We will consider four positions for this figure, denoted - for ease of notation - by $\mathcal{F}^{(i)}$, for $i = 0, 1, 2, 3$, with $\mathcal{F}^{(0)}$ the initial position of the figure in which it covers all the points in $\mathcal{S}$. If there is no node that lies on the perimeter of $\mathcal{F}^{(0)}$, then one can *translate* $\mathcal{F}^{(0)}$ parallel to the x-axis until (at least) one node $A \in \mathcal{S}$ hits the perimeter as exemplified in Fig. 4. We denote the position of this figure by $\mathcal{F}^{(1)}$, which still contains all the nodes in S. If A is the only node in S on

the perimeter of $\mathcal{F}^{(1)}$, we *rotate*, either clock-wise or counter-clock-wise until (at least) one node $B \in \mathcal{S}$, different from A, hits the perimeter as shown in Fig. 4. Denoting the position of this figure by $\mathcal{F}^{(2)}$, one can *slide* $\mathcal{F}^{(2)}$ until at least one more node $C \in S$ lies on the perimeter. *Sliding* is illustrated in Fig. 5 for an ellipse and a rectangle. The resulting position $\mathcal{F}^{(3)}$ contains all nodes in S and is characterized by 3 nodes. ■

For the following Theorem 2, we use the term "collinear" and introduce the term "quasi collinear." A set of nodes is collinear if and only if a single line can pass through all those nodes (Fig. 6c). Three nodes are quasi collinear if and only if two nodes can lie on a same side and the third on a parallel side (Fig. 6d) or the three nodes can lie on three different parallel sides (Fig. 6e).

Theorem 2: The positions of ellipses and polygons can be uniquely characterized by three nodes, unless the nodes are collinear or quasi collinear for the polygon.

Proof: The proof relies on figure equations from analytic geometry [24] and appropriate case analysis and can be found in Appendix A. Crucial possibilities for a polygon are visualized in Figs. 6a and 6b. ■

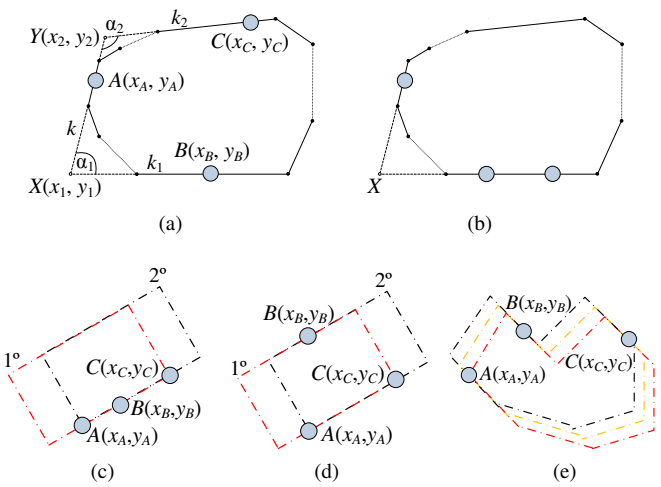


Fig. 6: (a) All three nodes belong to different sides of the polygon; (b) Two nodes lie on the same side and one node is on a different side; (c) two crucial positions for collinear nodes, shown on a rectangle; (d) two crucial positions for a rectangle for quasi-collinear nodes on 2 parallel sides, shown on a rectangle and (e) crucial positions for a polygon for quasi-collinear nodes on 3 parallel sides.

In Theorem 2 collinear or quasi-collinear nodes for polygons were excluded. In our algorithm these exceptions will however also be examined. If a figure covers some set of collinear nodes and at least one node that is not collinear to them, this case will be examined by a figure through the non-collinear node and two other nodes. If there is a set of collinear nodes (even with cardinality greater than 3) that could be covered by some position, but no other non-collinear node can form a figure with them, then it suffices to consider two crucial positions: one of the two "end nodes" lies in a corner of the figure and all the other nodes are positioned on a single side (Fig. 6c). Similarly, for quasi-collinear nodes, in both cases of two nodes on the same side or all three on different sides, a constant number of crucial positions have to be examined: those where an "end node" is in the corner of a polygon (Figs. 6d and 6e). These cases are also considered in our algorithm.

B. Polynomial-time algorithm for detecting critical regions

The algorithm, named FINDCRITICALREGION, is formalized in Algorithm 1. Routine RELAXCRITICAL (Algorithm 2) is used to calculate the change in network metric X and to update the set of critical regions $\mathcal{C}$.

Algorithm 1: FINDCRITICALREGION

input : the network $G(\mathcal{N}, \mathcal{L})$, figure $\mathcal{F}$, metric X
output: critical region(s) $\mathcal{C}$, metric after a failure minVal

$\mathcal{C} \leftarrow \emptyset$;
minVal $\leftarrow \infty$;
foreach *node triple* $\{A, B, C\} \subseteq \mathcal{N}$ **do**
  if *($\mathcal{F} \neq$ ellipse and $\{A, B, C\}$ are collinear/"quasi" collinear)* **then**
    $Q \leftarrow$ positions for $\mathcal{F}$ such that one node of A, B or C is in the corner;
  else
    $Q \leftarrow$ all positions for $\mathcal{F}$ through A, B and C;
  foreach $q \in Q$ **do**
    $G' \leftarrow G(\mathcal{N}, \mathcal{L})$;
    foreach $N \in \mathcal{N}$ **do**
      if $N \in q$ **then** $G' \leftarrow G' - N$; ;
    RELAXCRITICAL$(G', X, q,$ minVal$, \mathcal{C})$;
foreach *isolated node pair* $\{A, B\} \subseteq \mathcal{N}$ **do**
  $G' \leftarrow G' - A$; $G' \leftarrow G' - B$;
  RELAXCRITICAL$(G', X, q,$ minVal$, \mathcal{C})$;
foreach *isolated node* $A \in \mathcal{N}$ **do**
  $G' \leftarrow G' - A$;
  RELAXCRITICAL$(G', X, q,$ minVal$, \mathcal{C})$;

Algorithm 2: RELAXCRITICAL

input : modified network G', the network metric X, region considered q, current minimum minVal, critical region(s) $\mathcal{C}$.
output: current minimum minVal, critical region(s) $\mathcal{C}$.

$Y \leftarrow$ CALCULATEMETRIC(G', X);
if $Y <$ minVal **then** /* new critical region */
  $\mathcal{C} \leftarrow q$; minVal $\leftarrow Y$;
else if $Y =$ minVal **then** /* another critical region */
  $\mathcal{C} \leftarrow \mathcal{C} \cup q$;

For an M-polygon, a polygon containing M sides, the complexity of all the possibilities through 3 fixed nodes together with finding the end-points requires $O(M^4)$ time.

According to Theorem 2, at most three points are needed to define a given figure and finding the centers and the orientations of an ellipse or examining all possibilities for the sides (i.e., finding the corners of a general M-polygon) requires time complexity of $O(1)$ and $O(M^4)$, respectively (lines 4-7 in FINDCRITICALREGION). There are $\binom{N}{3}$ triples of nodes and for each triple one needs to consider all possible figures through this triple (lines 3-12 in FINDCRITICALREGION). In a case of a polygon, if a node triple is collinear or quasi collinear, there are infinitely many positions. Fortunately, even in those cases, the number of positions to be examined is constant. If a certain node pair (or isolated node) cannot form a figure with any of the other nodes, then the algorithm for critical region detection considers a figure arbitrarily positioned through these nodes (lines 13-18 in FINDCRITICALREGION).

The worst-case time complexity of this part is $O(N^3)$ for an ellipse and $O(N^3M^4)$ for an M-polygon. Checking whether some of the other nodes lie inside a positioned figure can be done in at most $O(N)$ time. The complexity of determining the change in the value of a metric after the failure of a considered region and the nodes inside is denoted by $O(C)$. The total complexity therefore accumulates to $O(N^4 \cdot C)$ for an ellipse and $O(N^4M^4 \cdot C)$ for an M-polygon in the worst case. General two-dimensional figures can be approximated with arbitrary accuracy by a polygon which is: (i) *circumscribed* (upper bound) or (ii) *inscribed* (lower bound).

1) Improved algorithm for circular critical regions: For a circle, there are $\binom{N}{2}$ possible pairs that form at most two circles. Consequently, an algorithm that considers all possible circles and checks which nodes are inside to find the critical region(s) will have a worst-case complexity of $O(N^3 \cdot C)$. A more efficient algorithm can be designed, when, instead of fixing two nodes, we fix one node A to be the origin in a 2D plane. A center of potential critical region through

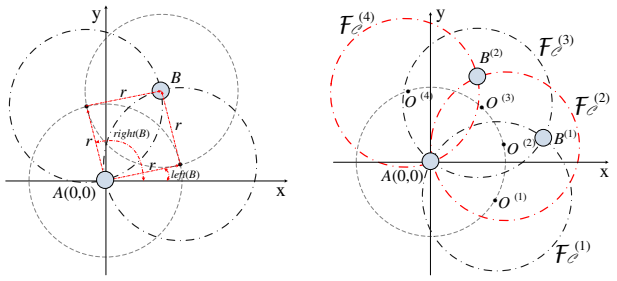


Fig. 7: (a) Definitions of $left(B)$ and $right(B)$ and (b) examination of the potential critical regions.

A and another node will lie on the circumference of the circle with center in A. There are at most 2 circles that pass through both A and B. Their centers lie in the (possible) intersection(s) of the circles with centers in A and B (in Fig. 7a). Let us denote by $left(B)$ and $right(B)$ the central angles between those centers and the positive x axes as shown in Fig. 7a. Further, we sort all the angles in $\{left(B)|B \in \mathcal{N}\backslash\{A\}\} \cup \{right(B)|B \in \mathcal{N}\backslash\{A\}\}$. Now, using those angles, we iterate through this list, which physically means rotating the centers of the potential critical regions (see Fig. 7b). Here, the most important point is that, when moving from one position to the other, exactly one node will leave or enter the critical region. Therefore, all the nodes covered by the circle in the previous iteration remain, except (i) a new node associated to the new circle will enter the circle; or (ii) a node associated to the circle in the previous iteration will leave the circle. Node A can be chosen in N ways, while the sorting of the geographically close v neighbors takes $O(v \log v)$ time, and the angle calculation takes $O(v)$ time, which gives a time complexity of $O(Nv \log v)$. Iterating through each pair of geographically close nodes (knowing which nodes are covered), gives a time complexity of $O(Nv \cdot C)$. Hence, the total time complexity is $O(N \cdot v \log v + N \cdot v \cdot C)$. Because $O(v) \leq O(N)$, we obtain $(N^2 \log N + N^2 \cdot C)$ in the worst case. When we update the nodes that belong to the critical circle, we do not need to calculate the metric X from scratch as only one node is going in or out from the considered circle. Hence, dynamically computing X [25] would make the algorithm even more efficient. We also remark that the "ideal symmetry" of the circle is used, which is not applicable for other figures.

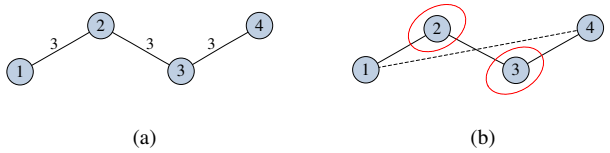
C. Region-critical network augmentation

In order to make the network more robust, we can augment the network by adding k links, preferably of minimum total weight. Since augmenting a network to increase node- or link-connectivity is NP-complete for weighted networks [26] or even APX-hard [27] for certain metrics, our network augmentation problem is also NP-complete for various metrics X. By examining all possible combinations of k links, we could choose the best combination of links to be added or an Integer Linear Programming (ILP) formulation could be used as exemplified in Appendix B. Depending on the computational resources at hand, running-time or memory constraints may still restrict the usability of the ILP. In those cases a heuristic with tunable complexity is desirable, for which we propose an efficient greedy augmentation technique. Since a network provider may add links over time, we will consider adding k links either one at a time, multiple at a time, or all instantaneously. The set of eligible links to be added could be reduced based on for instance cost or distance constraints. Each time one or multiple links are added, all possibilities (from the set of eligible links) are considered, after which the best one (in terms of critical region reduction) is chosen. We will demonstrate that a fast greedy approach that, out of all the available links, only adds that one link which realizes the greatest reduction in network degradation after failure of the critical region, and when iterated k times, is close to adding k links at once. There might be multiple available links that equally reduce the network vulnerability, in which case, if possible, we choose the link that does not connect to any other critical region[1]. The complexity of the greedy augmentation strategy is $O(k \cdot L_{\bar{G}} \cdot C)$, where $O(C)$ is the time complexity of finding the network metric change

[1]In our simulations we take $w(u, v) = d(u, v)$. If different weights are used, one could break ties by choosing the link with lower weight.

after a link addition and $L_{\bar{G}}$ reflects the number of links in the complement graph $\bar{G}$ of the original network. For multiple links at a time, the complexity grows quickly.

Considering the number of connected pairs as our metric X, we demonstrate the link addition strategy in a simple network in Fig. 8a for when only one link needs to be added. The distances between nodes are $d(1,2) = d(2,3) = d(3,4) = 3$ and the diameter of the failure $D = 2$. The network is connected, therefore the initial number of connected pairs is 6. As there is no figure with diameter 2 that passes through two nodes, we detect two critical regions, around nodes 2 and 3, whose single failure results in only one connected pair. There are three possibilities for a link addition: $(1,3)$, $(2,4)$ and (1,4). If we add a link between nodes 1 and 3 to protect against the critical region around node 2, the number of connected pairs would not be decreased as the failure of the critical region around node 3 would lead to only nodes 1 and 2 to be connected, but the number of critical regions is reduced. The same applies to adding the link $(2,4)$. If we add the link $(1,4)$ that is not connected to any critical region then we always have three connected pairs after any single critical region failure as shown in Fig. 8b.



(a) (b)

Fig. 8: Example of the network augmentation technique.

V. Region-disjoint paths problem

We show that deciding whether region-disjoint paths exist is NP-hard. Consequently, the **region-disjoint paths problem** (defined in Section III) is hard (to approximate) as well[2].

Subsequently, we propose a polynomial-time heuristic for the optimal version of the problem.

A. Complexity of the problem

We provide a polynomial time reduction for the 3SAT problem[3], which is known to be NP-complete [28]. We use a graph structure called *lobe* [29] to construct a graph from a 3SAT problem and on which finding two region-disjoint paths would provide a solution to that 3SAT problem. We will assume undirected networks, although directed links could also have been used.

Theorem 3: Establishing whether two region-disjoint paths exist is NP-hard.

Proof: The proof can be found in Appendix C. ■

[2]The problem may be polynomially solvable for certain instances, e.g., if all the nodes in the network are pairwise on a distance greater than D, which could be checked in $O(N^2)$ time. In this case, a polynomial-time algorithm for finding node-disjoint paths [2], [16] gives a solution.

[3]3SAT is a formula satisfiability problem, with an input logical expression $C_1 \wedge C_2 \wedge \ldots C_m$, where C_i are logical clauses of 3 variables (in auxiliary or negated forms) connected by $\vee$, which asks whether it is possible to assign boolean values to the variables so that the formula evaluates to TRUE.

B. Heuristic region-disjoint paths algorithm

Since the region-disjoint paths problem cannot be solved or approximated in polynomial time, unless P=NP, we consider a polynomial-time heuristic, named RegionDisjointPaths.

Algorithm 3: RegionDisjointPaths

input : Network G, diameter D, source s, destination d
output: Region-disjoint paths P_1 and P_2

1 Find node-disjoint paths P_1 and P_2 between s and d, if exists; otherwise **Exit**; Find the set of *critical pairs* $\mathcal{K}$;
2 Initialize the set of "unavailable" nodes $\mathcal{Q} \leftarrow \emptyset$;
3 Divide all the nodes in the network (except s and d) into two disjoint sets $\mathcal{S}_1$ and $\mathcal{S}_2$ in the following way: $i \in \mathcal{S}_1$ if i is the closest to some node in $\mathsf{P}_1 \setminus \{\mathsf{s},\mathsf{d}\}$ or $i \in \mathcal{S}_2$ if i is the closest to some node in $\mathsf{P}_2 \setminus \{\mathsf{s},\mathsf{d}\}$;
4 If the set $\mathcal{K} = \emptyset$ then stop, **region-disjoint paths** are found. Otherwise, find the node $k \in (\mathsf{P}_1 \cup \mathsf{P}_2) \setminus \mathcal{Q}$ that appears in most of the critical pairs in $\mathcal{K}$ (if many the one with a minimum resulting total weight of P_1 and P_2); if such k does not exist then stop, **region-disjoint paths** are not found.
5 $k \in \mathsf{P}_j$, where $j \in \{1,2\}$. Find the shortest path P_x through nodes in $\mathcal{S}_j$ that both do not appear in $\mathcal{Q}$ and in a pair in $\mathcal{K}$, between the first predecessor a and the first successor b of k in P_j that do not appear in $\mathcal{Q}$ and in a pair in $\mathcal{K}$; $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{k\}$;
6 If P_x does not exist: Go to Line 4;
7 Update P_j such that it consists of: the existing part in P_j from s to a, P_x and the current part from b to d in P_j;
8 Update $\mathcal{K}$, $\mathcal{S}_1$ and $\mathcal{S}_2$ based on P_1 and P_2;

RegionDisjointPaths starts by finding two node-disjoint paths P_1 and P_2 with minimum total weight, e.g. by using the Suurballe-Tarjan algorithm [16], if possible, otherwise there are no region-disjoint paths and the algorithm terminates. We denote by *critical pairs* $\mathcal{K}$ the set of pairs of nodes, such that one is in the path P_1 and the other is in P_2 on a distance not greater than D. Then, the algorithm partitions the nodes in the network into two sets $\mathcal{S}_1$ and $\mathcal{S}_2$. A node belongs to the set $\mathcal{S}_1$ if it is closest (in distance) to a node in $P_1 \setminus \{s,d\}$, otherwise the node belongs to $\mathcal{S}_2$, which means that these nodes are closer to P_2. We initialize a set of "unavailable" nodes to an empty set ($\mathcal{Q}$ in Algorithm 3). The algorithm finds a node k that appears in most of the pairs in $\mathcal{K}$, but is not in the set of "unavailable" nodes. Further, if $k \in \mathcal{S}_1$ ($k \in \mathcal{S}_2$), the algorithm makes a local improvement by finding the shortest path through the nodes in $\mathcal{S}_1$ (in $\mathcal{S}_2$) that do not appear in $\mathcal{Q}$ and in a pair in $\mathcal{K}$, between the first predecessor a of k in P_1 (in P_2) and the first successor b of k in P_1 (in P_2) that both do not appear in $\mathcal{Q}$ and in a pair in $\mathcal{K}$. The updated path P_1 (or P_2) comprises the current part from s to a, the newly determined shortest path from a to b and the current part from b to d. RegionDisjointPaths iterates by searching back for a new k that appears in most of the pairs in $\mathcal{K}$, but is not in the "unavailable" nodes, and updates $\mathcal{S}_1$ and $\mathcal{S}_2$. If there

are multiple nodes that appear in most of the pairs in $\mathcal{K}$, it chooses one such that the resulting total weight of P_1 and P_2 is minimal. The pseudo-code of REGIONDISJOINTPATHS is given in Algorithm 3.

REGIONDISJOINTPATHS always terminates, since each node can be picked for removal at most once. The complexity of REGIONDISJOINTPATHS can be determined as follows. Finding node-disjoint paths with a minimum total weight [16] requires a complexity [30] of $O(L + N\log_2 N)$. The shortest path in $\mathcal{S}_1$ or $\mathcal{S}_2$ (after a node k is picked) can be found in $O(L + N\log_2 N)$. The updates of the sets $\mathcal{K}$, $\mathcal{S}_1$ and $\mathcal{S}_2$ all require a worst-case complexity of $O(N^2)$ as $O(N)$ nodes may change in one of the paths, whose distances to the nodes in the second path have to be compared. The number of nodes that appear in pairs in the set $\mathcal{K}$ is $O(N)$, which reflects the number of iterations. Consequently, the total worst-case time-complexity of REGIONDISJOINTPATHS is $O(N^3)$.

Correctness proof for REGIONDISJOINTPATHS. Although the two paths might not be optimal, if the algorithm manages to find two paths then those paths are region-disjoint. Indeed, *critical pairs* set $\mathcal{K}$ being empty ensures that all the intermediate nodes from different paths are on a distance greater than D. If $\mathcal{K} \neq \emptyset$ the algorithm either continues with its "main iteration" or it outputs that region-disjoint paths are not found (ensured by line 4 in REGIONDISJOINTPATHS). Although the intermediate nodes of the two paths are on sufficient distance from each other, their be close or cross in some parts. ■

For comparison purposes, we also deploy a naive algorithm, named DOUBLEDIJKSTRA that uses two iterations of the Dijkstra shortest path algorithm [31]. The first iteration finds the shortest path between s and d. Subsequently, all nodes within a distance D from at least one node in the first path different from s and d are removed. Finally, the second path is found by running Dijkstra's algorithm in the pruned network. Since two Dijkstra algorithm iterations are used, the complexity of DOUBLEDIJKSTRA is $O(L + N\log_2 N)$.

Since the region-disjoint paths problem is NP-hard, we resort to an Integer Linear Program (ILP) to find the exact solution. For each link $(i,j) \in \mathcal{L}$, we define two variables $x_{ij}, y_{ij} \in \{0,1\}$. If a link (i,j) is on path P_1 then $x_{ij} = 1$, otherwise $x_{ij} = 0$ and if a link (i,j) is on path P_2 then $y_{ij} = 1$, otherwise $y_{ij} = 0$. The distances $d(i,j)$ between the nodes can be calculated beforehand in polynomial time. The 0-1 ILP formulation is as follows:

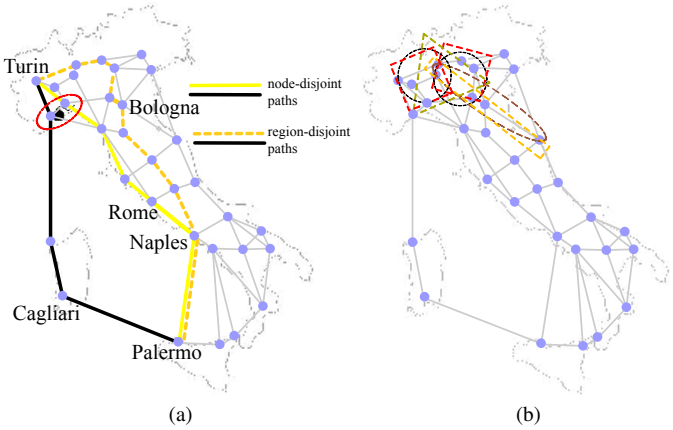
$$\min \sum_{(i,j)\in\mathcal{L}} w(i,j)\cdot(x_{ij}+y_{ij})$$

s.t. (1) $$\sum_{j\in\mathcal{N}}(x_{ij}-x_{ji}) = \begin{cases} 1, & \text{if } i \equiv s \\ -1, & \text{if } i \equiv d \\ 0, & \text{otherwise} \end{cases}$$

(2) $$\sum_{j\in\mathcal{N}}(y_{ij}-y_{ji}) = \begin{cases} 1, & \text{if } i \equiv s \\ -1, & \text{if } i \equiv d \\ 0, & \text{otherwise} \end{cases}$$

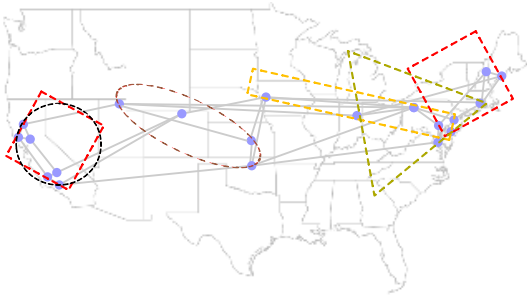
(3) $x_{ij} + y_{kl} \leq 1$, if ($d(i,k) \leq D$, $i \notin \mathcal{M}$, $k \notin \mathcal{M}$) or
($d(i,l) \leq D$, $i \notin \mathcal{M}$, $l \notin \mathcal{M}$) or
($d(j,k) \leq D$, $j \notin \mathcal{M}$, $k \notin \mathcal{M}$) or
($d(j,l) \leq D$, $j \notin \mathcal{M}$, $l \notin \mathcal{M}$), where $\mathcal{M} = \{s,d\}$

(4) $x_{sd} + y_{sd} \leq 1$

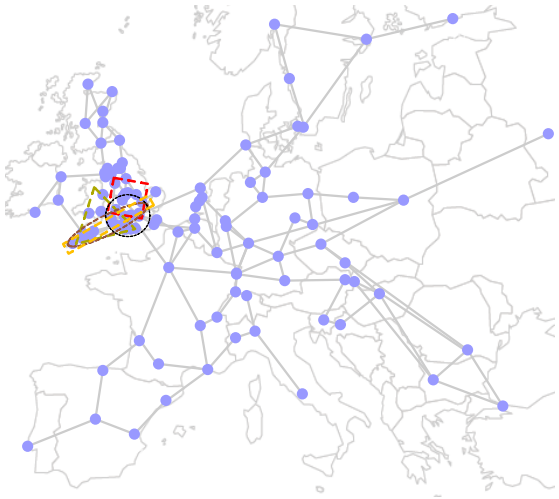


(a) (b)

Fig. 9: Italian backbone network: (a) node-disjoint and region-disjoint paths; and (b) the critical regions (possibly multiple) for the number of disconnected pairs - the area of each figure (circle, ellipse, square, rectangle and triangle) is the same 16895 km^2.

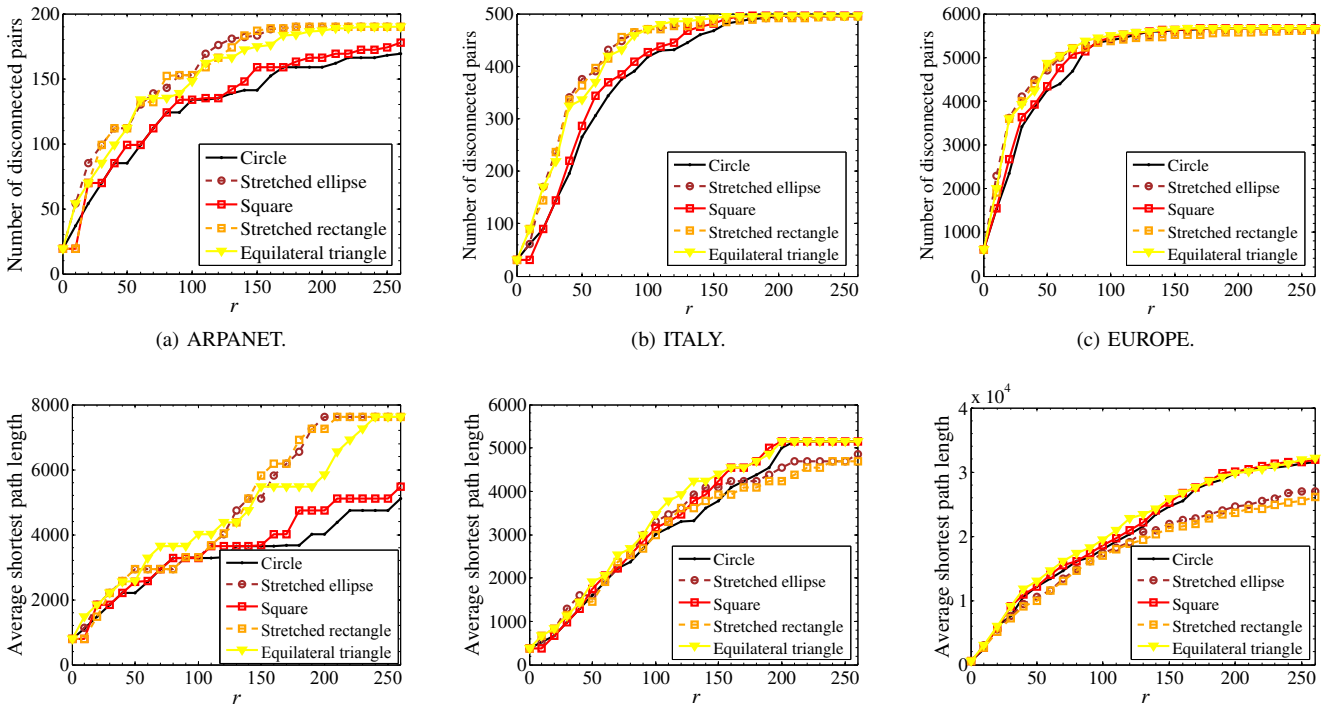


(a) ARPANET.



(b) EUROPE.

Fig. 10: For 5 figures (circle, ellipse, square, rectangle and triangle), the critical regions (possibly multiple) as a function of the number of disconnected pairs. The area of each figure per map is the same: (a) 300021 km^2 and (b) 27907 km^2. The stretched ellipse (rectangle) has one semi-axis (side) nine times longer than the other.



(a) ARPANET. (b) ITALY. (c) EUROPE.

(d) ARPANET. (e) ITALY. (f) EUROPE.

Fig. 11: The number of disconnected pairs (a), (b), (c) and the average shortest path length (d), (e), (f), with r in km, for: circle with radius r, ellipse with semi-axes $a = 3r, b = r/3$, square with sides $a = r\sqrt{\pi}$, rectangle with sides $a = 3r\sqrt{\pi}, b = r\sqrt{\pi}/3$ and equilateral triangle with sides $a = 2r\frac{\sqrt{\pi}}{\sqrt[4]{3}}$. For the same r, the areas of the figures are equal in each network.

The objective function represents the total weight of the region-disjoint paths. The equality conditions (1) and (2) are "conservation rules" and ensure that for all the nodes (different from s and d) in both P_1 and P_2, the number of incoming and outgoing links is the same. For the source node s, there is exactly one outgoing link for both P_1 and P_2, while for the destination node d there is exactly one incoming link for both P_1 and P_2. Condition (3) gives the region-disjointness constraint, preserving two nodes different from s and d, one in link $(i, j) \in P_1$ and one in link $(k, l) \in P_2$ to be on a distance at most D. If there is a direct link from s to d, condition (4) states that link will be used by at most one path. Condition (4) is not a sub-case of (3).

C. An example of region-disjoint paths

In Fig. 9a, two region-disjoint paths between Turin and Palermo are depicted for the Italian main backbone network. The (undisplayed) weights in the network are the geographical distances between the nodes. The two node-disjoint paths of minimum total weight (which traverse through Turin-Rome-Naples-Palermo and Turin-Cagliari-Palermo) are depicted in solid lines. However, these paths cannot protect against a failure of a circular region with diameter $D = 6000$. REGIONDISJOINTPATHS returns region-disjoint paths such that one of the paths is the same as in the initial node-disjoint paths, but the other path differs (Turin-Bologna-Naples-Palermo, shown with dashed lines). Moreover, the ILP confirms that this solution is exact. DOUBLEDIJKSTRA could not find a solution.

VI. EVALUATION STUDY

In this section, we conduct simulations to study (i) how certain metrics are affected after the failure of a critical region; (ii) the effect of network augmentation and (iii) the accuracy and running time of our REGIONDISJOINTPATHS algorithm.

A. Used data

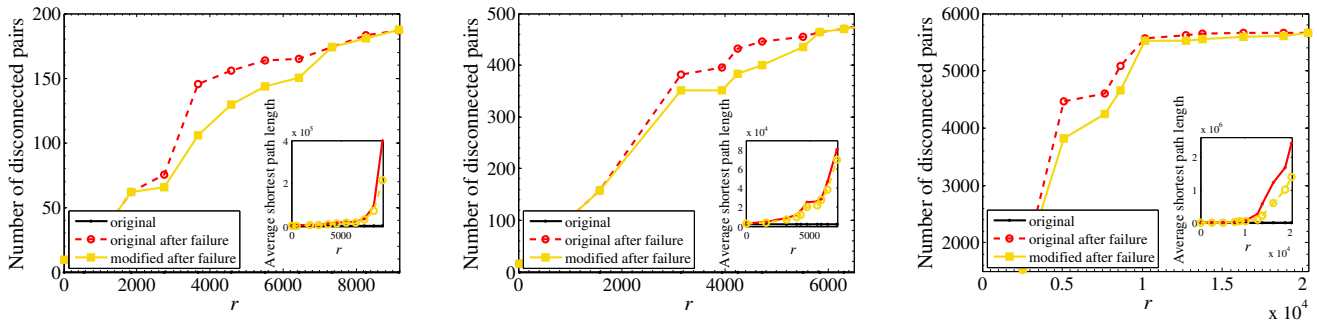
In the evaluation, we use three real-world network data sets: the infrastructure of the ARPANET network [32] (in Fig. 10a), which is often used as a benchmark topology, the Italian main backbone network (in Fig. 9b) and the main backbone fiber connections in Europe [33] (in Fig. 10b). We refer to these networks as: ARPANET, ITALY, and EUROPE, respectively. Through longitude and latitude information, the geographical distances between the nodes can be derived. The properties of the used networks are given in Table I.

TABLE I: Real networks used in the evaluation.

Networks	N	L	Description
ARPANET	20	32	first packet switching network [32]
ITALY	32	62	main fiber connections in Italy
EUROPE	108	151	main fiber connections in Europe [33]

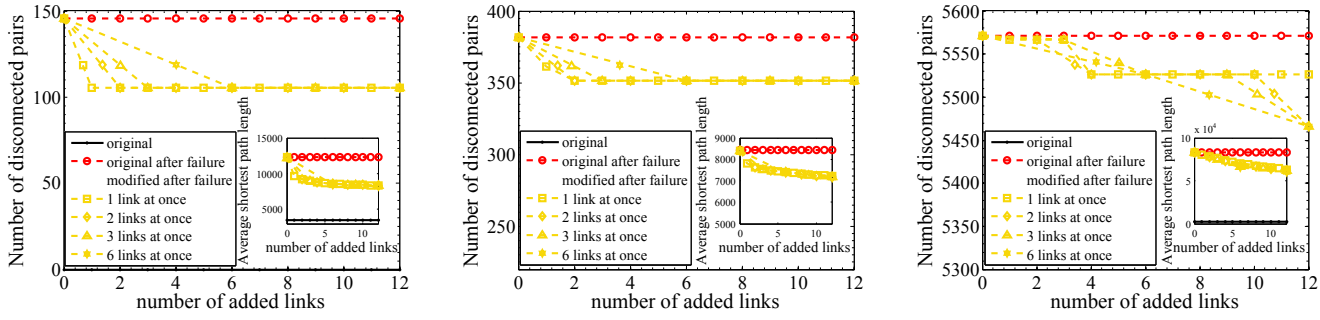
B. Effect of critical regions

We consider five figures namely: the circle, stretched ellipse, square, stretched rectangle and equilateral triangle and three networks (see Table I), for which we detect critical regions.



(a) ARPANET. (b) ITALY. (c) EUROPE.

Fig. 12: The number of disconnected pairs and the average shortest path length (inset) as a function of the distance variable r, which reflects the diameter of failure D. Ten links were added to each of the modified networks. The number of disconnected pairs in the original network is 0. To reflect a realistic scenario, only links between nodes that are on a distance no more than half the maximum distance between two nodes are added.



(a) ARPANET ($r = 3672$). (b) ITALY ($r = 3152$). (c) EUROPE ($r = 10185$).

Fig. 13: The number of disconnected pairs and the average shortest path length (inset) as a function of the number of added links, with a fixed value of r representing the diameter of failure D. The number of disconnected pairs in the original network is 0. All added links are constrained to be no longer than half the maximum distance between two nodes.

In ARPANET, the most critical square and circular regions are those covering the west and east coast, while the most critical stretched figures are located in central USA, because the nodes centered there link the US coasts. In ITALY, the most critical regions are positioned in the northern part of the country; however, the stretched figures touch a part of central Italy. For EUROPE, for a relatively small size of the figure, the most critical regions are situated near London (Figs. 9b and 10).

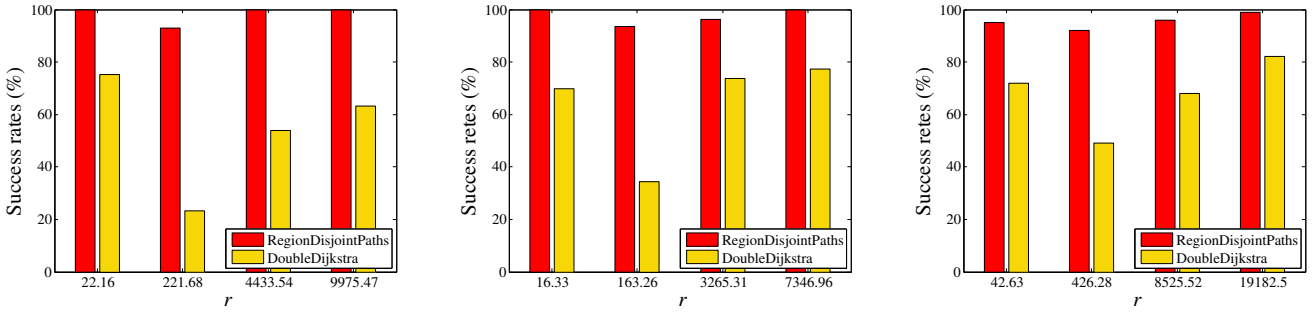
For the same networks, we have also examined the change in two network metrics after the failure of a critical region, namely: (1) the number of disconnected pairs[4] and (2) the average shortest path length for the five different figures. The distance control variable (r) is used, such that for a given r the areas of different figures are the same. Fig. 11 shows that for both metrics the network is not affected equally for different figures. Generally, for the number of disconnected pairs the most critical region is more disruptive for the equilateral triangle and the stretched ones (ellipse and rectangle) than for the circle and the square (Figs. 11a, 11b and 11c).

In particular, for the average shortest path length in ARPANET (Fig. 11d), where there are distant nodes and the most dense areas in terms of nodes are not central, the most critical region is more disruptive for the equilateral triangle and the stretched figures (ellipse and rectangle), than for the circle and the square. The same holds for the number of disconnected pairs (Fig. 11a). Somewhat similar behavior is noticed for ITALY (Fig. 11e). On the other hand, for the average shortest path length in EUROPE (Fig. 11f), where there is a very dense region (United Kingdom), apart from the equilateral triangle, the square and the circle are more disruptive than the stretched figures. When extremely large regions are considered, the metric values for different figures become more similar as most of the nodes in the networks are affected in all cases.

C. Evaluation of region-critical network augmentation

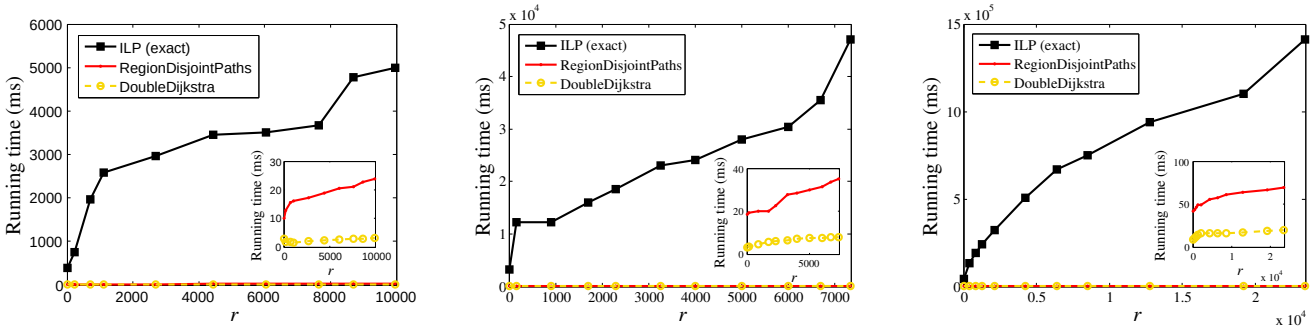
We examine the effect of our network augmentation strategy on the number of disconnected pairs (i.e., the number of connected pairs of the original network minus the number

[4]A related metric, called *average two-terminal reliability* A2TR has been used in [7], which is a normalization of the number of connected pairs divided by the maximum possible. Consequently, A2TR=1 $-$ $\frac{\text{number of disconnected pairs}}{\binom{N}{2}}$.



(a) ARPANET. (b) ITALY. (c) EUROPE.

Fig. 14: Success rates. All pairs of nodes are requests. Distance variable (r) reflects the failure diameter D.



(a) ARPANET. (b) ITALY. (c) EUROPE.

Fig. 15: Average running times of the algorithms over all pairs of nodes. Distance variable (r) reflects the failure diameter D.

of remaining connected pairs) and the average shortest path length of the connected pairs. In Fig. 12, we present the numbers of disconnected pairs (main plot) and the average shortest path lengths (inset) for (i) the original network (the solid, black, horizontal line), (ii) the original network after a critical region failure and (iii) the modified network after failure of a critical region as a function of the diameter of that region. The network augmentation strategy has added ten links, one at a time. In order to reflect realistic scenario, each of the added links is constrained to be no longer than half of the maximum distance between two nodes. The results show a substantial reduction in the number of disconnected pairs and in the average shortest path lengths, as a result of the network augmentation. The curve for the number of disconnected pairs of the modified network approaches that of the original for high values of the diameter, because most of the nodes are then affected, leaving no effective protection.

On the other hand, in Fig. 13, we examine the effect on the number of disconnected pairs and the average shortest path length as a function of the number of added links when added one at a time or in multiples. Again, added links are shorter than half of the maximum distance. For small networks (ARPANET in Fig. 13a and ITALY in Fig. 13b), only few links are required to reduce the number of disconnected pairs and further link additions would not have an effect as the remaining disconnected pairs are attributed to nodes residing inside a critical region (which therefore cannot be protected against the failure of that region). For EUROPE (in Fig. 13c), more links are required to reduce the number of disconnected pairs. In all three networks, the average shortest path lengths are shortened as more links are added, where a few added links already realize a significant reduction. Moreover, there are small differences, in general k times adding 1 link is faster and quite close in performance to adding k links at once.

D. Evaluation of region-disjoint paths algorithms

The accuracy of REGIONDISJOINTPATHS is evaluated by a comparison with the exact ILP solution and the naive DOUBLEDIJKSTRA algorithm. For each network, the requests consist of all the possible pairs of distinct nodes. Fig. 14 depicts the comparison of the heuristic algorithms and the exact ILP shown by the *success rates*[5] of the heuristic algorithms. The results vary for different values of the distance variable $r = D/2$, where D is the diameter of the failure, but the algorithm REGIONDISJOINTPATHS correctly finds region-disjoint paths in a significant number of cases. On the other hand, DOUBLEDIJKSTRA has a much worse performance, because greedily removing a first shortest path may jeopardize finding a second path.

The average running times of the three algorithms over all possible requests (pairs of nodes) are shown in Fig. 15. The algorithms have been implemented in the same programming

[5]We define *success rate* of an algorithm as the quotient: $\frac{\text{number of successful requests of the algorithm}}{\text{number of successful requests of the exact ILP}}$.

language, using the same libraries and the simulations have been conducted on the same machine[6]. Fig. 15 shows that solving the ILP requires significant running time, orders of magnitude greater than for the REGIONDISJOINTPATHS and DOUBLEDIJKSTRA algorithms, which is unfeasible when paths need to be computed on the fly for dynamically arriving requests. The running times of REGIONDISJOINTPATHS and DOUBLEDIJKSTRA do not differ much, while the improved accuracy of REGIONDISJOINTPATHS clearly outweighs the slight increase in time over the DOUBLEDIJKSTRA algorithm.

VII. CONCLUSION

This paper has considered two problems in relation to the geographical embedding of a network: finding critical network regions as a function of general polygons or ellipses and finding two region-disjoint paths with a minimum total weight such that they cannot both be cut by a single regional failure with a given diameter, unless that failure affects the source or the destination. Region-disjoint paths are needed to quickly reroute traffic when a regional failure occurs.

First, we have proved that the number of potential locations of critical regions that need to be examined is polynomially bounded by the number of nodes N. Subsequently, we have proposed a polynomial-time algorithm for finding the critical regions for a generic network metric. We have used our algorithm to study the critical regions in three real-world networks by finding the critical regions for a certain figure size. The results show that the equilateral triangle and the stretched figures might be more disruptive than the "centralized" ones when the number of disconnected pairs is chosen as measure for criticality. However, in networks with very dense regions, the circular and the square figures are more disruptive than the stretched figures when the average shortest path length is considered. We have also considered a region-aware network augmentation to increase the network robustness to regional failures. By applying our augmentation strategy to three real networks, adding only a few links may already induce significant robustness gains.

We have discussed the hardness of the region-disjoint paths problem and subsequently proposed an efficient polynomial-time heuristic. The comparison with an exact exponential-time algorithm shows that our proposed algorithm correctly finds region-disjoint paths in most of the cases, while being orders of magnitude faster than the exact algorithm.

ACKNOWLEDGMENT

This research has been partly supported by the EU FP7 Network of Excellence in Internet Science EINS (project no. 288021) and by the GigaPort3 project led by SURFnet.

APPENDIX

A. Proof of Theorem 2

Let us denote the three considered nodes by $A\left(x_A, y_A\right)$, $B\left(x_B, y_B\right)$ and $C\left(x_C, y_C\right)$.

[6]Intel(R) Core 2 Duo T9600 - 2 × 2.80GHz, 4GB RAM memory; using JAVA libraries: JUNG (http://jung.sourceforge.net/) for network representations and algorithms and lpsolve (http://lpsolve.sourceforge.net/) for the ILP.

1) *Ellipse* $\mathcal{F}_{\mathcal{E}}$. The equation of an ellipse with semi-axes lengths a and b, center $O(x_j, y_j)$, orientation φ is given by

$$\frac{(x\cos(\varphi) - y\sin(\varphi) - x_j)^2}{a^2} + \frac{(x\sin(\varphi) + y\cos(\varphi) - y_j)^2}{b^2} = 1 \quad (1)$$

Having three nodes A, B and C that fulfill equation (1) leads to a system of 3 equations with three unknowns: x_j, y_j and φ. Subtracting equations (1) for A, B & A, C and expanding in terms of x_j and y_j results in

$$\begin{aligned} & 2b^2((x_A - x_m)\cos(\varphi) - (y_A - y_m)\sin(\varphi))x_j \\ & + 2a^2((x_A - x_m)\sin(\varphi) + (y_A - y_m)\cos(\varphi))y_j \\ = \; & b^2[((x_A - x_m)\cos(\varphi) - (y_A - y_m)\sin(\varphi))\times \\ & ((x_A + x_m)\cos(\varphi) - (y_A + y_m)\cos(\varphi))] \\ & + a^2[((x_A - x_m)\sin(\varphi) + (y_A - y_m)\cos(\varphi))\times \\ & ((x_A + x_m)\sin(\varphi) + (y_A + y_m)\cos(\varphi))] \end{aligned} \quad (2)$$

where $m \in \{B, C\}$. (2) forms a system of two equations with two unknowns, treating φ as a constant. Finding the solutions (e.g., by calculating determinants), we obtain

$$x_j = \frac{M_3(\cos(\varphi), \sin(\varphi))}{K_2(\cos(\varphi), \sin(\varphi))},\; y_j = \frac{N_3(\cos(\varphi), \sin(\varphi))}{K_2(\cos(\varphi), \sin(\varphi))} \quad (3)$$

where $K_2(\cos(\varphi), \sin(\varphi))$, $M_3(\cos(\varphi), \sin(\varphi))$ and $N_3(\cos(\varphi), \sin(\varphi))$ are homogeneous polynomials in $\cos(\varphi)$ and $\sin(\varphi)$ of degree 2, 3 and 3, respectively. Using (3) in (1), for instance for node C, results in an equation for φ. After expanding, we end up with a homogeneous polynomial in $\cos(\varphi)$ and $\sin(\varphi)$ of degree 6. The resulting equation consists of terms $\sin^i(\varphi)\cos^{6-i}(\varphi)$ for $i = 0, 1, \ldots, 6$. We take all the terms with i even on one side and all the terms with i odd on the other. For the even terms: $\sin^{2l}(\varphi)\cos^{6-2l}(\varphi) = \left(\frac{1-\cos(2\varphi)}{2}\right)^l \left(\frac{1+\cos(2\varphi)}{2}\right)^{3-l}$ for $l = 0, 1, 2, 3$ and for the odd terms $\sin^{2l+1}(\varphi)\cos^{5-2l}(\varphi) = \frac{\sin(2\varphi)}{2}\left(\frac{1-\cos(2\varphi)}{2}\right)^l \left(\frac{1+\cos(2\varphi)}{2}\right)^{2-l}$ for $l = 0, 1, 2$. Finally, squaring both sides and using $\sin^2(2\varphi) = 1 - \cos^2(2\varphi)$ results in an equation of degree 6, solely in $\cos(2\varphi)$. The last can be solved numerically. Because of the squaring, we need to check whether the solution satisfies (1). Finally, we obtain the solution via equation (3).

2) *General polygon* $\mathcal{F}_{\mathcal{P}}$. We have two possibilities as visualized in Figs. 6a and 6b.

(a) All three nodes belong to different sides of the polygon (Fig. 6a). We extend two of the sides, such that they intersect the third side or its extension in points X and Y. Let us denote the coefficient of one of the sides (through A) as k. Because, we know the (pairwise) angles between those three sides, we can express the coefficients of the other two sides (lines) as functions of k, namely $k_i = \frac{k - \tan\alpha_i}{1 + k\tan\alpha_i}$ for $i = 1, 2$. Further, we can express the coordinates of $X(x_1, y_1)$ and $Y(x_2, y_2)$ as functions of k. Because X is in the intersection of the lines through A and B: $k(x_1 - x_A) + y_A = k_1(x_1 - x_B) + y_B$, from which we obtain: $x_1 = \frac{x_B k^2 + (y_B - y_A)k + x_B + \frac{y_B - y_A}{\tan\alpha_1}}{1 + k^2}$. A similar expression could be found for x_2. Because the M-polygon is known, the distance $|XY|$ is also known, hence we have an equation in k: $(x_1 - x_2)^2 + (y_1 - y_2)^2 = (1 + k^2)(x_1 - x_2)^2 = |XY|^2$, from which we obtain an equation of degree at most 4: $[(x_B - x_C)k^2 + (y_B - y_C)k + x_B - x_C + \frac{y_B - y_A}{\tan\alpha_1} + \frac{y_C - y_A}{\tan\alpha_2}]^2 = (1 + k^2)|XY|^2$. Simpler equations

are obtained for specific figures (e.g., quadratic for rectangle and equilateral triangle [34]). The last equation can be solved exactly in constant time and at most 4 solutions are possible, which reflect different possible positions of the M-polygon. For each determined k, we can immediately find X and Y and the end-points of the sides of the three nodes. Thus, we can obtain all the nodes (i.e. the position) of the M-polygon in $O(M)$ time.

(b) Two nodes lie on the same side and one node is on a different side (Fig. 6b). We can exactly determine the side (line) through the nodes on the same side. The angle between this side and the side of the third node is also known, thus we can find the line through the third node as well. Finally, the intersection point X of the two sides or their extensions is exactly known, hence we can determine all the nodes (i.e. the position) of the M-polygon in $O(M)$ time.

In general, we again have finitely many possibilities for fixed 3 nodes, namely: $\binom{M}{3}$ in (a); and $\binom{M}{2}$ in (b). We remark that analytical expressions for large M or specific polygons might be very complex.

B. ILP of the optimal addition of (at most) k new links

We formulate an ILP for optimizing on the shortest path between two dedicated nodes s and d. We start with the following optimization program:

$$\min \Big\{ \max_{\mathcal{F}} \Big[\sum_{(i,j)\in\mathcal{L}} w(i,j)(1-z_{ij}(\mathcal{F}))x_{ij} + \sum_{(i,j)\in\mathcal{N}^2\setminus\mathcal{L}} w(i,j)(1-z_{ij}(\mathcal{F}))y_{ij} \Big] \Big\}$$

s.t.

(1) $\sum_{j\in\mathcal{N}} (x_{ij}+y_{ij} - x_{ji}-y_{ji})(1-z_{ij}(\mathcal{F})) = \begin{cases} 1, & \text{if } i\equiv s \\ -1, & \text{if } i\equiv d \\ 0, & \text{otherwise} \end{cases}$
for each position of $\mathcal{F}$

(2) $\sum_{(i,j)\in\mathcal{N}^2\setminus\mathcal{L}} y_{ij} \le k$

(3) $x_{ij}+y_{ij}\le 1,\ \forall (i,j)\in\mathcal{N}^2$

where the variables are $x_{ij}\in\{0,1\}$, $\forall (i,j)\in\mathcal{L}$ for the links that are present in the network and $y_{ij}\in\{0,1\}$, $\forall (i,j)\in\mathcal{N}^2\setminus\mathcal{L}$ for the links that the network is augmented with, otherwise $x_{ij}=y_{ij}=0$. Variables $z_{ij}(\mathcal{F})=1$, if i or j from $\mathcal{N}$ are covered by the position $\mathcal{F}$ of the figure, otherwise $z_{ij}(\mathcal{F})=0$. In a similar way as for the **Critical region problem**, there are polynomially many, at most $O(N^3)$, candidates for $\mathcal{F}$, hence $z_{ij}(\mathcal{F})$ could be determined/pre-processed in polynomial time before the execution of the optimization problem. Condition (1) ensures the flow conservation constraints, being aware that nodes and links might be cut by $\mathcal{F}$. Condition (2) ensures that at most k links will be added from $\mathcal{N}^2\setminus\mathcal{L}$.

The objective function reflects the shortest path between s and d in the augmented network. The optimization program is not an ILP, so we introduce an additional variable q to replace the part after **min** in the objective function, which leads to the following new objective

$$\min q$$

with additional constraints

(4) $\sum_{(i,j)\in\mathcal{L}} w(i,j)(1-z_{ij}(\mathcal{F}))x_{ij} + \sum_{(i,j)\in\mathcal{N}^2\setminus\mathcal{L}} w(i,j)(1-z_{ij}(\mathcal{F}))y_{ij} \le q$
for each position of $\mathcal{F}$

Similarly, one can formulate an ILP for other metrics, like the average shortest path in the network, where we should have variables x_{sdij} (and y_{sdij}), which are 1, if and only if link (i,j) belongs to the shortest path between s and d.

C. Proof of Theorem 3

We start with the graph construction.

Graph Construction. For a given 3SAT instance, we create a lobe for each variable x_i. Denoting by p_i the number of occurrences of variable x_i (in an auxiliary form x_i or as a negation $\bar{x}_i$), the lobe of x_i contains $(6p_i+2)$ nodes: x_i, x_{i+1}, $u^i_j, v^i_j, w^i_j, \bar{u}^i_j, \bar{v}^i_j, \bar{w}^i_j$, for each $j=1,2,\dots,p_i$. We construct the links: $\left(x_i,u^i_1\right), \left(x_i,\bar{u}^i_1\right), \left(v^i_{p_i},x_{i+1}\right), \left(\bar{v}^i_{p_i},x_{i+1}\right)$; (u^i_j,v^i_j), (v^i_j,w^i_j), (w^i_j,u^i_{j+1}), $(\bar{u}^i_j,\bar{v}^i_j)$, $(\bar{v}^i_j,\bar{w}^i_j)$, $(\bar{w}^i_j,\bar{u}^i_{j+1})$ for each $j=1,2,\dots,p_i$. In Fig. 16, the lobe of variable x_i is depicted, which contains two parallel disjoint branches.

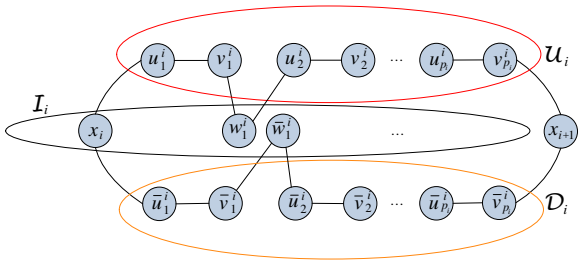


Fig. 16: Lobe of x_i.

All n lobes are connected in series (the i-th lobe is connected to the $(i+1)$-th lobe), starting from x_1 to x_n. Further, for each clause i of the 3SAT instance, we construct two nodes y_i and z_i. A link (z_i, y_{i+1}) is established for each $i=1,2,\dots,m-1$. Additionally, links (s,y_1), (s,x_n), (z_m,d) and (x_{n+1},d) are added. To relate the clauses with the variables, we have the following: (i) if the k-th occurrence of variable x_i exists without negation in clause C_j, then links $\left(y_j,u^i_k\right)$ and $\left(v^i_k,z_j\right)$ are added; or (ii) if the k-th occurrence of variable x_i exists with a negation ($\bar{x}_i$) in clause C_j, then links $\left(y_j,\bar{u}^i_k\right)$ and $\left(\bar{v}^i_k,z_j\right)$ are added. The final graph is shown in Fig. 17.

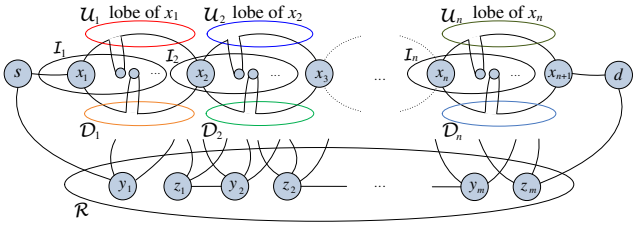


Fig. 17: Constructed graph.

The position of all the nodes in the Euclidean plane is defined in the following way: (i) We set the nodes u^i_j, v^i_j

from the upper part of each lobe x_i such that they are all on a distance not greater than D from each other and they are all on a distance greater than D from all the other nodes in the constructed graph. We denote the union of these nodes by $\mathcal{U}_i = \bigcup_{j=1}^{p_i} u_j^i \cup \bigcup_{j=1}^{p_i} v_j^i$; (ii) For the lower part of each lobe x_i, we set the nodes $\bar{u}_j^i, \bar{v}_j^i$, such that they are all on a distance not greater than D from each other and they are all on a distance greater than D from all the other nodes in the constructed graph. We denote the union of these nodes by $\mathcal{D}_i = \bigcup_{j=1}^{p_i} \bar{u}_j^i \cup \bigcup_{j=1}^{p_i} \bar{v}_j^i$; (iii) Similarly, for a given i, we position nodes x_i, $w_j^i, \bar{w}_j^i$, such that they are all on a distance not greater than D from each other and they are all on a distance greater than D from all the other nodes in the constructed graph. We denote the union of these nodes by $\mathcal{I}_i = x_i \cup \bigcup_{j=1}^{p_i} w_j^i \cup \bigcup_{j=1}^{p_i} \bar{w}_j^i$; (iv) Finally, all the nodes y_i and z_i, $i = 1, 2, \ldots, m$ are on a distance not greater than D from each other and they are all on a distance greater than D from all the other nodes in the constructed graph. We denote these nodes by $\mathcal{R} = \bigcup_{i=1}^{m} y_i \cup \bigcup_{i=1}^{m} z_i$. All the sets $\mathcal{R}, \mathcal{U}_i, \mathcal{D}_i, \mathcal{I}_i$ for $i = 1, 2, \ldots, n$ are pair-wise region disjoint. The construction of the graph can be done in polynomial time.

An example of the aforementioned construction of the graph, for a 3SAT instance is given in Fig. 18. We will

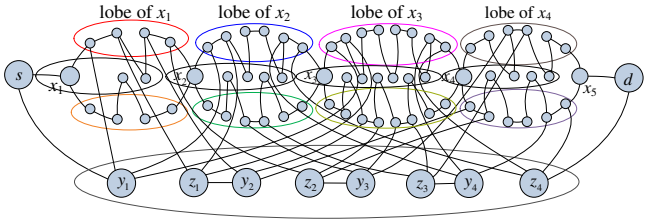


Fig. 18: Constructed graph that corresponds to $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee x_4) \wedge (x_2 \vee x_3 \vee \bar{x}_4)$.

demonstrate that we could solve any instance of the 3SAT problem by solving the region-disjoint paths problem on a graph obtained via the polynomial-time transformation of the 3SAT instance explained before.

3SAT to **Region-disjoint paths**. Let us assume there is an assignment τ, such that all m clauses are satisfied. For each clause C_j: (i) there exists a variable x_i (in non-negated form) such that $\tau(x_i) =$ TRUE in which case we use links (y_j, u_k^i), (u_k^i, v_k^i), (v_k^i, z_j) or (ii) there exists a variable $\bar{x}_i$ in C_j, such that $\tau(x_i) =$ FALSE in which case we use links $(y_j, \bar{u}_k^i)$, $(\bar{u}_k^i, \bar{v}_k^i)$, $(\bar{v}_k^i, z_j)$ for $j = 1, 2, \ldots, m-1$. Together with links (s, y_1), (z_m, d), (z_j, y_{j+1}) $(j = 1, 2, \ldots, m-1)$ they form a path P_1. In addition, there is another path P_2 that traverses through the lower part of x_i's lobe ($\mathcal{D}_i$) if $\tau(x_i) =$ TRUE or the upper part ($\mathcal{U}_i$) if $\tau(x_i) =$ FALSE, for each $i = 1, 2, \ldots, m$. In this way, we have one path (P_1) that passes through $\mathcal{R}$ and parts of the lobes, but not through the lobes parts in $\mathcal{I}_i$ and one path that traverses only through the other branches of each lobe and all $\mathcal{I}_i$. The paths are region-disjoint.

Region-disjoint paths to **3SAT**. Let us assume there are two region-disjoint paths P_1 and P_2. Both P_1 and P_2 cannot both contain links that pass through $\mathcal{R}$ and

$$\{(v_j^i, w_j^i), (w_j^i, u_{j+1}^i), (\bar{v}_j^i, \bar{w}_j^i), (\bar{w}_j^i, \bar{u}_{j+1}^i) | j = 1, 2, \ldots, m-1\}$$

for each $i = 1, 2, \ldots, n$, otherwise they would both pass through $\mathcal{I}_i$ for some i. Hence, one path (P_1 or P_2) should only contain links (u_j^i, v_j^i) and $(\bar{u}_j^i, \bar{v}_j^i)$ from the lobes and exactly one should pass through (z_i, y_{i+1}) (because of the region $\mathcal{R}$). If there are two region-disjoint paths, one (e.g., P_2) would go to x_1 and the other (e.g., P_1) would go to y_1 from s. Because, P_1 is already in $\mathcal{R}$, P_2 does not go through any of the nodes in y_i and z_i in order to have region-disjoint paths. Hence, P_2 must continue through all of the lobes. Moreover, P_1 could pass through part of the lobes, but not in the parts in $\mathcal{I}_i$, so it can go "up" through one link (but not more) in the lobe and then go "down" again through the nodes y_i and z_i. If P_1 uses links (s, y_1), (z_m, d), (z_i, y_{i+1}) we set $\tau(x_i) =$ TRUE, while if P_1 uses links $(y_j, \bar{u}_k^i)$, $(\bar{u}_k^i, \bar{v}_k^i)$, $(\bar{v}_k^i, z_j)$ we set $\tau(x_i) =$ FALSE. Finally, by this truth assignment, all m clauses become satisfied.

REFERENCES

[1] C. Doerr, R. Gavrila, F. A. Kuipers, and P. Trimintzios, "Good Practices in Resilient Internet Interconnection," June 2012, ENISA report.

[2] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing*. Kluwer Academic Publishers, 1999.

[3] F. A. Kuipers, "An Overview of Algorithms for Network Survivability," *ISRN Communications and Networking*, vol. 2012, no. 932456, 2012.

[4] P. Pan, G. Swallow, and A. Atlas, "Fast Reroute Extensions to RSVP-TE for LSP Tunnels," *RFC4090*, May 2005.

[5] M. Shand and S. Bryant, "IP Fast Reroute Framework," *RFC5714*, January 2010.

[6] NASA, available on: http://www.nasa.gov/mission_pages/hurricanes/archives/2012/h2012_Sandy.html, 2012.

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

[8] Y. Shen, N. P. Nguyen, Y. Xuan, and M. T. Thai, "On the discovery of critical links and nodes for assessing network vulnerability," *IEEE/ACM Trans. on Networking*, vol. 21, no. 3, June 2013.

[9] S. Neumayer and E. Modiano, "Network reliability with geographically correlated failures," in *INFOCOM*. IEEE, March 2010, pp. 1658–1666.

[10] S. Neumayer, A. Efrat, and E. Modiano, "Geographic max-flow and min-cut under a circular disk failure model," in *INFOCOM*. IEEE, March 2012, pp. 2736–2740.

[11] A. Sen, B. H. Shen, L. Zhou, and B. Hao, "Fault-tolerance in sensor networks: A new evaluation metric," in *INFOCOM*. IEEE, 2006.

[12] P. K. Agarwal, A. Efrat, S. Ganjugunte, D. Hay, S. Sankararaman, and G. Zussman, "The resilience of wdm networks to probabilistic geographical failures," *IEEE/ACM Trans. on Networking*, vol. 21, no. 5, Oct. 2013.

[13] P. K. Agarwal, H. Kaplan, and M. Sharir, "Union of random minkowski sums and network vulnerability analysis," in *Proc. of SoCG*. Rio de Janeiro, Brazil: ACM, 2013, pp. 177–186.

[14] S. Banerjee, S. Shirazipourazad, and A. Sen, "On region-based fault tolerant design of distributed file storage in networks," in *INFOCOM*. IEEE, March 2012, pp. 2806 –2810.

[15] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974.

[16] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.

[17] D. Xu, Y. Chen, Y. Xiong, C. Qiao, and X. He, "On finding disjoint paths in single and dual link cost networks," in *INFOCOM*, vol. 1. IEEE, March 2004, pp. 4 vol. (xxxv+2866).

[18] Y. Guo, F. A. Kuipers, and P. Van Mieghem, "Link-disjoint paths for reliable QoS routing," *Int. J. Communication Systems*, vol. 16, no. 9, pp. 779–798, 2003.

[19] R. Andersen, F. Chung, A. Sen, and G. Xue, "On disjoint path pairs with wavelength continuity constraint in wdm networks," in *INFOCOM*, vol. 1. IEEE, March 2004, pp. 4 vol. (xxxv+2866).

[20] A. A. Beshir, F. A. Kuipers, A. Orda, and P. Van Mieghem, "On-line survivable routing in WDM networks," in *21st International Teletraffic Congress (ITC 21)*, Sept. 2009.

[21] Q. She, X. Huang, and J. P. Jue, "Maximum survivability using two disjoint paths under multiple failures in mesh networks," in *GLOBECOM*. IEEE, Dec. 2006.

[22] A. A. Beshir and F. A. Kuipers, "Variants of the min-sum link-disjoint paths problem," in *16th Annual Symposium on Communications and Vehicular Technology (SCVT)*. IEEE, Nov. 2009.

[23] A. Sen, S. Murthy, and S. Banerjee, "Region-based connectivity - a new paradigm for design of fault-tolerant networks," in *Int. Conference on High Performance Switching and Routing*, June 2009, pp. 1 –7.

[24] W. Osgood and W. Graustein, *Plane and solid analytic geometry*. The Macmillan Company, 1921.

[25] P. Narváez, K.-Y. Siu, and H.-Y. Tzeng, "New dynamic algorithms for shortest path tree computation," *IEEE/ACM Trans. Networking*, vol. 8, no. 6, pp. 734–746, Dec. 2000.

[26] G. Frederickson and J. Ja'Ja', "Approximation algorithms for several graph augmentation problems," *SIAM Journal on Computing*, vol. 10, no. 2, pp. 270–283, 1981.

[27] J. Cheriyan, T. Jordán, and R. Ravi, "On 2-coverings and 2-packings of laminar families," in *Algorithms - ESA 99*, ser. LNCS. Springer, 1999, vol. 1643, pp. 510–520.

[28] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[29] A. Itai, Y. Perl, and Y. Shiloach, "The complexity of finding maximum disjoint paths with length constraints," *Networks*, vol. 12, no. 3, pp. 277–286, 1982.

[30] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *J. ACM*, vol. 34, no. 3, pp. 596–615, July 1987.

[31] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959, 10.1007/BF01386390.

[32] "ARPANET Maps," available on: http://www.10stripe.com/featured/map/arpanet/arpanet-1972-aug.php.

[33] Level 3 Communications, "Level 3 Network map," available on: http://www.level3.com/en/resource-library/maps/level-3-network-map/.

[34] S. Trajanovski, F. A. Kuipers, and P. Van Mieghem, "Finding critical regions in a network," in *INFOCOM workshops (5th NetSciCom)*. Turin, Italy: IEEE, April 2013.

Stojan Trajanovski is a PhD candidate at Delft University of Technology, the Netherlands. He obtained a master degree in Advanced Computer Science (*with distinction*, 2011) from the University of Cambridge, United Kingdom. He also holds an MSc degree in Software Engineering (2010) and a Dipl. Engineering degree (*magna cum laude*, 2008) from Ss. Cyril and Methodius University in Skopje, Macedonia. He successfully participated in international science olympiads, winning a bronze medal at the International Mathematical Olympiad (IMO) in 2003. His main research interests include network robustness, clustering analysis of networks and applied graph theory.

Fernando A. Kuipers (SM'10) is associate professor in the Network Architectures and Services group at Delft University of Technology. He received the M.Sc. degree in Electrical Engineering at Delft University of Technology in June 2000 and subsequently obtained his Ph.D. degree (*cum laude*) in 2004 at the same faculty.

His research interests mainly revolve around network algorithms and cover Routing, Quality of Service, Network Survivability, Optical Networks, and Content Distribution. His work on these subjects has resulted in over 50 publications in refereed journals and conferences and includes distinguished papers at IEEE INFOCOM 2003, Chinacom 2006, IFIP Networking 2008, IEEE FMN 2008, IEEE ISM 2008, and ITC 2009.

Aleksandar Ilić holds a PhD in computer science (2011) from, and is part-time assistant professor at the Faculty of Sciences and Mathematics, University of Niš, Serbia. Since July 2011, he has been working as a software engineer in Facebook. He published over 50 papers on chemical, combinatorial and spectral graph theory and design of algorithms, and has two patents regarding social networks. Aleksandar won more than 20 medals and diplomas at international competitions in mathematics and informatics for high school and university students, most notably silver medals on both International Mathematical Olympiad (IMO) and International Olympiad of Informatics (IOI) in 2003. He was also the main organizer and problem proposer for national competitions and IOI.

Jon Crowcroft (SM'95-F'04) received the B.S. degree in physics from Trinity College, Cambridge University, Cambridge, U.K., in 1979 and the M.Sc. degree in computing and the Ph.D. degree from the University College London (UCL), London, U.K., in 1981 and 1993, respectively.

He is a Marconi Professor of communication systems with the Computer Laboratory, University of Cambridge. He is a Fellow of Wolfson College, Cambridge, U.K. Until the end of September 2001, he was a Professor with the Department of Computer Science, UCL. His research interests are communications and multimedia systems, particularly Internet related.

Dr. Crowcroft is a Fellow of the ACM, the British Royal Society, the British Computer Society, the Institute for Ethics and Emerging Technologies, and the Royal Academy of Engineering.

Piet Van Mieghem received the Masters (*magna cum laude*, 1987) and PhD (*summa cum laude*, 1991) degrees in electrical engineering from the K.U. Leuven, Leuven, Belgium.

He is a Professor at the Delft University of Technology and Chairman of the section Network Architectures and Services (NAS) since 1998. His main research interests lie in modeling and analysis of complex networks and in new Internet-like architectures and algorithms for future communications networks. Before joining Delft, he worked at the Interuniversity Micro Electronic Center (IMEC) from 1987 to 1991. During 1993-1998, he was a member of the Alcatel Corporate Research Center in Antwerp, Belgium. He was a visiting scientist at MIT (1992-1993) and a visiting professor at UCLA (2005) and at Cornell University (2009). He is the author of three books: Performance Analysis of Communications Networks and Systems (Cambridge Univ. Press, 2006), Data Communications Networking (Techne, 2006; 2nd ed., 2011), and Graph Spectra for Complex Networks (Cambridge Univ. Press, 2011).

Prof. Van Mieghem currently serves on the editorial board of the OUP Journal of Complex Networks, Computer Communications and the Journal of Discrete Mathematics. He was member of the editorial board of the IEEE/ACM Transactions on Networking (2008-2012) and the Journal Computer Networks (2005-2006).