Computing the Vulnerability of Communication Networks to Large-Scale Disasters

J. Oostenbrink

Computing the Vulnerability of Communication Networks to Large-Scale Disasters

by

J. Oostenbrink

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday August 31, 2017 at 01:30 PM.

Student number:	4169263	
Thesis committee:	Dr.ir. F.A. Kuipers,	TU Delft, supervisor
	Prof.dr. K.G. Langendoen,	TU Delft
	Dr. M.T.J. Spaan,	TU Delft

TU Delft

Abstract

Our society is becoming more and more dependent on communication networks. Thus, network operators and designers need to properly prepare themselves against large-scale disasters that could take down a significant amount of networking hardware within a large area.

In this work, we consider the vulnerability of a network to disasters, and we propose an efficient method to compute the distribution of a network performance measure, based on a finite set of disaster areas and occurrence probabilities. Compared to other approaches, our approach is more accurate and gives more information about the vulnerability of the network.

Based on our approach, we have implemented a tool to help visualize the vulnerability of a network to disasters.

We give general methods to obtain finite disaster sets from empirical disaster data and showcase these methods on actual earthquake and hurricane datasets.

Our approach is demonstrated by analyzing the vulnerability of a variety of network topologies to large-scale disasters.

Contents

1 Introduction

Over the last few decades, communication networks have become more and more intertwined with our personal lives, commerce and government. Large failures in our communication systems can now have a very significant impact on both our economy and daily lives.

During and after a disaster, communication networks become even more important, as they are used for timely communication between emergency services and deploying and coordinating relief operations.

In 2006, an undersea earthquake off the coast of Taiwan damaged submarine cables, shutting down communications in several countries in Asia [41]. In 2011, a massive earthquake struck Japan; the earthquake and subsequent tsunami not only resulted in a massive loss of human lives, but also caused extensive damage to telecommunications buildings and equipment, leading to widespread connectivity problems. Meanwhile, peak traffic was 9 times as high as normal. The total cost of emergency restoration and reconstruction of the NTT East network was 80 billion yen (around 1 billion dollars at the time) [16, 40].

It has become clear that network operators should prepare for, and network designers account for, large-scale disasters.

Broadly speaking, disaster (and emergency) management consists of four stages or phases:

1. **Prevention & Mitigation**: this stage involves both the prevention of disasters, and the mitigation of the damage inflicted by disasters. In terms of networks this stage involves protecting hardware, adding backup hardware, spreading hardware around, and moving hardware to a safer location.

2. **Preparation**: not all disasters can be prevented, and not all damage mitigated. Thus one still needs to prepare for the aftermath of a large-scale disaster. This stage mostly involves ensuring the next two stages can be swiftly and efficiently executed when a disaster strikes. For example by training personnel, or acquiring emergency equipment.

3. **Response**: the response phase is the quick immediate response after a disaster. Network operators can set-up an emergency network for rescue operations, and have to set-up emergency routing to make sure important traffic still reaches its destination.

4. **Recovery**: The last stage of disaster management revolves around recovering from the disaster, this includes making repairs and any other actions taken to return the situation to normal. In terms of network management this mostly involves repairing the network itself. Depending on the accessibility of the damaged hardware, this phase can take a very long time to complete.

In the context of communication networks, all four of these stages have seen a large amount of research the past ten years. A large overview of some of the strategies that can be utilized by network operators to protect their networks against large-scale disasters can be found in [37].

Knowledge of the vulnerability of a network (topology) to disasters is essential in both the Prevention & Mitigation and Preparation stages.

In this work, we propose new methods to compute the vulnerability, or survivability, of network topologies to large-scale disasters. This can help in (1) preparing for potential disasters and (2) in designing or modifying a network to become more robust against them.

Although considered as early as 1991 by D. Bienstock [24], research on (the vulnerability of a network to) so-called geographically correlated challenges has only really taken of in the last decade. Often it is assumed that the disaster area takes a fixed shape (e.g., a line or a circle with fixed radius), after which the amount of disasters required to disconnect two nodes (e.g., [53]), the most vulnerable spot(s) of the network against this type of disaster (e.g., [52, 62, 64]), or the impact after a randomly placed disaster (e.g., [51, 59, 60, 64, 65]) are computed. In all aforementioned articles, the vulnerability of the network is reflected by a single value.

We will take a different approach: based on a set of possible disasters (of varying shapes), we compute the distribution of the measure after one of these disasters randomly occurs. We show that this distribution can be efficiently calculated, and that it provides more information to a network operator or designer than any single value, like the expected value, could.

In addition, we demonstrate how to obtain representative disaster sets based on the empirical data of three large organizations studying earthquakes and hurricanes: the (Japanese) National Research Institute for Earth Science and Disaster Resilience (NIED), the United States Geological Survey (USGS) and the National Hurricane Center (NHC).

Our main contributions are as follows:

1. We propose an efficient method to compute the distribution of a network performance measure, based on a finite set of disaster areas and occurrence probabilities.

2. We give approaches for obtaining such a set of disasters, and demonstrate them on real datasets.

3. We describe our tool to compute and visualize these distributions for any network topology and disaster set.

4. We apply our methodology to compute the vulnerability of Japanese and American topologies to earthquakes and hurricanes.

In chapter 2 we go over related work. In chapter 3 we introduce our network and disaster models. Based on these models, we propose the “Vulnerability Distribution Problem” and give efficient methods to solve it in in chapter 4. Next, in chapter 5, we show how to create the set of disaster areas and occurrence probabilities required for our methods. In chapter 6 we describe how our tool visualizes and conveys the results of our vulnerability computations. We analyze the vulnerability of Japanese and American topologies, and discuss the performance of our methods, in chapter 7. We make a brief detour in chapter 8 to give our analysis of and recommendations on SURFnet trouble and incident tickets. In chapter 9 we extend our disaster model to probabilistic disasters and in chapter 10 we give our conclusions.

2
Related Work

Although the concept of designing a network to be robust against geographically correlated natural or anthropogenic challenges is relatively new, it has already been the focus of much research. In this chapter we give an overview of related work on calculating the vulnerability of a (wired) network topology against geographical challenges.

These geographically correlated challenges, such as earthquakes, EMP attacks or fiber cuts, are often termed disasters in the literature, which is also the terminology we will use. Note that this usage of disaster is different from the general definition, which is used to indicate an event that is of such a severity or scale that it overpowers the response capacity of a society, community or nation [39].

While it is possible to research the effect of specific disasters on a network by simulation, e.g. as in [26], we focus on work that calculates the vulnerability of a network to a wide array of disasters by considering one or more survivability/reliability measures.

Disasters are generally modeled in one of two ways: deterministically or probabilistically. In both cases the disaster affects only a certain area of the network, often a circle or line segment. Links outside this area are not affected. In a deterministic disaster model all links and/or nodes in the disaster area fail, while in the probabilistic case links and/or nodes in the area fail with a certain probability, often based on the distance from the epicenter.

Note that the deterministic model is a special case of the probabilistic model, where the probability of failure is taken to be 1. However, by only considering the deterministic case one can often find a polynomial-time exact algorithm, while solutions for the probabilistic case generally are approximated or calculated in exponential time.

We give an overview of work on deterministic and probabilistic disaster models in sections 2.1 and 2.2. In section 2.3 we briefly discuss survivability measures.

2.1. Deterministic Disaster Models

Probably the first paper to consider the effect of geographically correlated failures on a network was [24] by D. Bienstock. He considered a generalized variant of the min-cut problem, where given a plane graph G, s and t vertices of G and a (finite) set of "holes" (with the only requirement that they are homeomorphic to an open disk), one must compute the minimum number of such holes which disconnect s and t. He gave a polynomial-time algorithm for solving this problem and proved that the corresponding generalized max-flow problem is NP-complete.

Neumayer et al. proposed the related "Geographical Min-Cut By Circular Disasters (GMCCD) Problem" and "Geographical Max-Flow By Circular Disasters (GMFCD) Problem" [53]. In these problems the holes are required to be disks of radius r_b. However, instead of only considering a finite amount of possible disks, disks can be centered anywhere in the plane (except inside the disks with radius r_p around s and t). They gave a polynomial-time algorithm for the geographic min-cut problem, but not for the related max-flow problem.

Y. Kobayashi et al. improved upon this work by giving a polynomial-time algorithm for the geographical max-flow problem and proving that $\mathrm{MAX-FLOW} \leq \mathrm{MIN-CUT} \leq \mathrm{MAX-FLOW}+1$ [43]. In addition, they gave a simpler polynomial-time algorithm for the geographical min-cut problem.

A max-flow is often equal to min-cut in practice. K. Otsuki et al. proposed some changes to the algorithms from [43] which give a significant speed-up in these cases [54].

Neumayer et al. considered line and circle cuts in [52]. For both of these types of cuts, they gave polynomial-time algorithms to find the most vulnerable part of the network, i.e. the position where the disaster does the most damage, in terms of 4 different measures: the total expected capacity of the intersected links (TEC), the fraction of node pairs that remain connected, the maximum flow between a given pair of nodes, and the average value of the maximum flow between all pairs of nodes. When calculating TEC, every link (i, j) can have a probability p_{ij} of existing between 0 and 1, while when using the other metrics, p_{ij} should either be 1 or 0 (i.e. the link either exists or it does not).

In [62] Trajanovski et al. gave a polynomial-time algorithm that, given any polygon or ellipse, finds its position (and rotation) which leads to the highest network disruption. However, in their model disasters only remove nodes (and their neighboring links) from the network, intersecting links are not removed. The authors also considered the problem of finding two minimum weight paths between a source s and a destination t that cannot both be cut by the same circle disaster. As the corresponding decision problem is NP-hard, they proposed a heuristic algorithm for finding these paths.

O. Gold et al. looked at an interesting variant of the problem of finding the most vulnerable spot of a network [36]. They focused on random networks, modeling a situation in which for example an adversary only has limited statistical information about the network. They gave polynomial-time algorithms to approximate the damage caused by circle cuts at specific locations and for finding the worst-case circle cuts in terms of TEC.

Instead of finding the most vulnerable spot, in [51] the connectivity of a network after a randomly placed line or disk cut was considered. Using geometric probability, methods were given to calculate the expected values of the all terminal reliability and the average two-terminal reliability in polynomial time. In the case of the disk cut, these values are approximations.

X. Wang et al. considered a randomly placed line segment cut in [65]. They proposed an algorithm that approximates the expected capacity of failed links, the average traffic loss between a given source-destination pair or the probability that a pair of given nodes with path protection is disconnected, by evenly partitioning the network into cells and calculating the value of these measures for a single disaster per cell (a method first proposed in [64]).

H. Saito considered a random disaster taking the form of a half plane in [59]. He showed how to calculate the probability that two nodes remain connected for the single route case, ring-network case and a combination of the two.

In [33] M. T. Gardner et al. considered circular disaster regions, which, like in [62], only affect nodes. They gave two algorithms for computing the vulnerable area of a network for either the 2-terminal or all-terminal case. Iff a disaster is centered in the vulnerable area, it disconnects either the source and destination node (2-terminal) or any component of the network (all-terminal). Although the search space is reduced in both these algorithms, their time complexity is still exponential. In [34] integer linear programming and particle swarm optimization approximation approaches were proposed to minimize the total vulnerable area by adding more nodes to the network. Although meant for wireless networks, these heuristics can, with some changes, also be applied to wired networks.

Components of a communication network require power to work, and thus depend on the power network. In [50] a model was proposed to measure the effect of failures in the power network on a dependent communication network: first all power lines that intersect a randomly located disk are removed. This causes cascade failures in the power network. Finally, the effect on the communication network after all failures have occurred is calculated, where a dependent node is considered operational iff the closest power demand node is still delivering power. Via Monte Carlo simulation it was shown that the average two-terminal reliability is significantly lower when power network dependency is considered.

Y. Berezin et al. considered circular (node) failures in pairs of interdependent diluted (by randomly removing nodes) square lattices in [23]. The dependencies between nodes in these networks are random, but constrained to be less than a distance r. They showed that based on r and their average degree, these interdependent networks can be either stable, metastable or unstable. Networks in a stable state do not collapse under any (finite radius) circle disaster. Networks in a metastable state collapse if a circular disaster is greater or equal than a critical size, independent of the network size. Networks in an unstable state collapse spontaneously.

Besides the geographical and general min-cut and max-flow, there are more network metrics specifically related to geographically correlated challenges.

S. Banerjee et al. proposed three new metrics based on the number and size of the connected components of a network after a (worst case) region failure and gave a polynomial-time algorithm to compute them for the circular disaster case [21, 22].

In [35] M. T. Gardner et al. proposed a new metric called the Network Impact Resilience (NIR), which takes the impact of a disaster on the mission of a network into account. They gave an exponential-time algorithm to calculate this metric for circular (nodal) disasters.

The Euclidean Maximum Flow Network Interdiction Problem (E-MFNIP) [61] is an interdiction problem where attacks diminish link capacity based on the distance from the link to the closest disaster. Note that this is another way to generalize the deterministic disaster model.

2.2. Probabilistic Disaster Models

In [19] P. K. Agarwal et al. presented a general probabilistic model for geographically correlated failures. Their model assumes the failure probability of a component is non-increasing in the distance from the epicenter of a disaster. They considered both the single disaster and multiple disaster scenario and gave approximation algorithms for finding the most vulnerable locations in both cases. These algorithms were improved upon in [20].

M. R. Naeini et al. considered the vulnerability of networks to multiple correlated disasters [56]. Examples of these kind of disasters are earthquakes, which can trigger other earthquakes and aftershocks. They modeled the disaster locations as a Strauss point process and the damage probabilities as Gaussian functions. Based on Monte Carlo simulations they were able to approximate some statistical properties of the vulnerability of the network and find the most vulnerable locations.

In [58], methods were proposed to calculate the probability that two nodes are disconnected after a random probabilistic half plane failure using geometric probability theory. It is assumed that any node or link in the disaster area fails with a fixed probability only depending on the node or link itself. This work is extended in [60] to a finite convex disaster area model, but only for ring networks and combinations of ring networks.

In contrast to the previous two models, X. Wang et al. proposed a probabilistic failure model that also takes into account the length of the segment of the link that intersects the disaster area [64]. They modeled a “probabilistic region failure (PRF)” as a set of M consecutive annuluses, each with their own failure probability. The failure probability of any link that intersects these annuluses depends on the length of the link segment intersecting each annulus. An approximation algorithm was given for calculating the remaining link capacity, the pairwise capacity reduction, and the pairwise connecting probability (after a randomly placed disaster) and finding the most vulnerable zones of the network by evenly partitioning the network into cells and calculating the value of these measures for a single disaster per cell.

Note that both the deterministic and the probabilistic models mentioned up until now all assume disasters can happen, and in the case of random failures with uniform probability, anywhere in the network area. L. Ma et al. extended the model from [64] to non-uniformly distributed region failures [46]. They approximated the node failure probabilities and link failure probabilities, and used these to create an integer linear program to optimally place data center networks and content.

In [28, 29], it is assumed that there is a known list of disasters, the failure probability for every network element in case of every disaster is known, and each connection has an associated failure cost. Proactive and reactive traffic engineering solutions are given to minimize the risk from these disasters, where the risk is taken to be the expected cost.

D. L. Msongaleli et al. presented an integer linear program for adding primary and backup cables to a network in [48]. They also assumed a given list of disasters and failure probabilities and try to minimize the expected costs. However, their cost function is more extensive, considering for example the repair cost and deployment cost as well.

In [63], P. N. Tran et al. proposed an (approximation) algorithm to enhance the robustness of a network against earthquakes by adding more links. The goal of their algorithm is to minimize the total end-to-end disconnection probabilities. They also assumed a list of disasters, with corresponding (earthquake) intensities is given, but calculate the probability of link failures by dividing each link into small segments, where they assumed the conditions of the area in which each segment lies is quasi-homogeneous. Assuming independent link segment failure probabilities, the failure probability of a link can then easily be computed based on the segment failure probabilities. The methods in [63] are especially interesting in that they are motivated by the release of, and applied to, real seismic hazard data from the Japan Seismic Hazard Information Station.

2.3. Measures

A wide array of different measures are used to determine the vulnerability of a network after a disaster. Some often used measures are the amount of link capacity lost, the (average) maximum flow and the (average) 2-terminal reliability.

There have been proposals to use measures specifically designed for this use, e.g. [45, 67], but they do not seem to be in wide use.

In [31], Feyessa and Bikdash proposed modifying existing centrality measures, and introduce their own measure, to account for geographic proximity.

S. C. Liew et al. proposed characterizing network survivability by a survivability function, rather than by a singly value (e.g. the value of a measure after a disaster at the most vulnerable location or the expected value after a random disaster) [44]. In essence their survivability function is the probability mass function of a given survivability measure after a random disaster. Some interesting values can easily be derived from this function, for example the worst case survivability, r-percentile survivability or the probability of zero survivability. As far as we know, this concept has not yet been applied to geographically correlated failures. As we believe such a function can give a much better picture of the vulnerability of a network to disasters, we focus on the application of the concept to geographically correlated challenges in this thesis.

3
Model

In this chapter we introduce our network and disaster model.

As networks and disasters are modeled completely separately. Both networks and disasters can be easily interchanged. Thus the vulnerability of a single network can be computed against different types of disasters, or multiple networks can be compared against each other in respect to the same disasters.

3.1. Network Model

We assume the network $G = \{V, E\}$, consisting of nodes V connected by links E, is embedded in the plane, and lies completely in a bounded convex region $R \subseteq \mathbb{R}^2$. The network can either be directed or undirected.

Nodes $v_i \in V$ are modeled as points $p_i \in R$. While the links $e_i = (v_j, v_k) \in E$ are modeled as a (finite) sequence of line segments connecting v_j and v_k. We define N_i as the number of line segments of e_i and $N_e := \sum_{i=1}^{|E|} N_i$.

3.2. Disaster Model

Disasters can take a wide variety of different forms. For example, an earthquake has a completely different disaster area than a hurricane. Even the same type of disaster can still have varying disaster shapes. The ground motion, and thus the disaster area after an earthquake, for example, depends not only on its magnitude, but also on the properties of the rocks and sediments that earthquake waves travel through [4].

Of course, there are also major differences in the size of the impacted area. Consider for example the differences of effect between earthquakes with varying magnitudes,

Network operators need to prepare their networks for all possible types of disasters, and a good disaster model should thus be able to model disasters of wildly varying shapes and sizes.

We model the disaster area as either a circle, line segment, hippodrome, simple polygon, or a finite union of these. These shapes have the property that we can quickly calculate if a line segment intersects it and should cover a large amount of possible disaster areas. However, our model and methods can be used with any shape of disaster area without any major changes, as long as it is possible to calculate if a line segment intersects it.

We take the deterministic approach, and assume all links intersecting the disaster area fail. If a node lies within a disaster area, all of its link must have at least one endpoint in the disaster area as well and will fail. We do not explicitly model node failures, as the failure of all incident links is equivalent to the node failing (in terms of link failures).

We assume we are given a finite set of possible disasters D, we further assume exactly one of these disasters occurs. The probability of multiple (independent) large-scale disasters occurring simultaneously is generally very small, and thus can be safely ignored. Disasters that trigger other disasters can still be modeled, by combining their disaster areas. Each disaster $d \in D$ has a disaster area $A(d) \subseteq \mathbb{R}^2$ and an occurrence probability $P(d)$. Note that $\sum_{d \in D} P(d) = 1$. In chapter 5 we discuss possible methods to obtain such a list of disasters.

In previous work, it was often assumed that the disaster area takes a fixed shape (e.g., a line or circle with fixed radius) and can occur anywhere in the network area (often with uniform probability). Compared to this approach, we model both the disaster location and impact shape more accurately.

In addition, our approach allows for easy collaboration between network operators/designers and experts (e.g. seismologists). These experts can generate representative accurate disaster sets based on the specific needs of the network operators, thus allowing for a more accurate vulnerability analysis.

Finally, as we show in chapter 4, this specific model allows for very efficient vulnerability computations.

In the next section we go over the different types of disaster areas, discussing both their use cases and how to efficiently present these shapes as input to our algorithms.

3.3. Disaster Shapes

3.3.1. Circle

The circular (or disk) disaster area is a good catch-all shape for most types of disasters and can be used when not enough data is available about the shape of the disaster area. In addition, this shape can be used to find vulnerable areas of the network in general. If a group of links get damaged by a circular disaster, they are placed close together, and thus are vulnerable to geographically correlated challenges in general. Of course, for both use cases the choice of circle radius is very important, and can greatly influence the results.

We assume we are given the circle center $c \in \mathbb{R}^2$ and radius $r \geq 0$. As disaster area we take the closed disk, i.e. $\{p \in \mathbb{R}^2 | ||p - c||_2 \leq r\}$. By taking the closed disk, a circle disaster with radius 0 can still affect links that go through c, for example if c is placed directly on a node.

3.3.2. Line Segment

Small line segments can be used to model some anthropogenic (caused by humans) disasters, for example cable cuts by anchors or damage caused by construction work. Larger line segments can model moving disasters, e.g. large storms, when not enough information is available to model a more accurate disaster area. Although in most cases moving disasters can be more accurately modeled by hippodromes.

We define a line segment solely by its two endpoints $p_1, p_2 \in \mathbb{R}^2$. The corresponding disaster area is $\{p \in \mathbb{R}^2 | p = p_1 + c(p_2 - p_1), c \in [0,1]\}$.

3.3.3. Hippodrome

A hippodrome is the area formed by moving a disk from one endpoint of a line segment to the other end. Or alternatively, a hippodrome consists of all points withing a certain distance (its radius) from a line segment.

Where the circle can be used to model general stationary disasters, the hippodrome can model moving disasters, e.g. hurricanes, when not enough data is available on the exact disaster area. Just as with the circle disaster, the choice of radius should be taken carefully, as it can greatly influence results.

We define the hippodrome by its two endpoints $p_1, p_2 \in \mathbb{R}^2$ and its radius $r \geq 0$. As with the circle disaster, we assume the hippodrome is closed. Its disaster area is given by $\{p \in \mathbb{R}^2 | \exists c \in \mathbb{R} s.t. ||p - p_1 + c(p_2 - p_1)||_2 \leq r\}$.

3.3.4. Simple Polygon

The simple polygon is perhaps the most important type of disaster shape in our approach. By taking a union of simple polygons, one can very accurately approximate every disaster area possible. This allows for precise modeling of disaster areas. Simple polygons can also be used to approximate the previous three disaster shapes, so these need not strictly be accounted for in our methods. However, computing if a links intersects a circle/line segment/hippodrome is much more efficient than computing if a link intersects a union of simple polygons approximating these shapes.

We assume we are given a sequence of n (where $n \geq 3$) endpoints $p_1, p_2, \ldots, p_n$, which can be connected by non-intersecting line segments to form the simple polygon. As with both the circle and hippodrome, we assume the area is closed, and consists of both the interior enclosed by the line segments and the line segments themselves.

3.3.5. Union of Areas

Finally, we allow the union of a finite number of disaster shapes. This can be used to more accurately model the disaster area. For example by taking the union of a large number of simple polygons, or by modeling a change in hurricane direction by combining multiple hippodromes.

We assume we are given, in the format described previously, all disaster areas which form the total disaster area. The total disaster area itself can then be formed by simply taking the union of all these areas.

4 Vulnerability Analysis

There are multiple ways to compute the vulnerability, or survivability, of a network to geographically correlated challenges with respect to certain measures. Typically, the vulnerability is given as a single value, most often the expected value or worst-case value of a measure. We argue this significantly limits the amount of usable information that can be obtained from such a computation.

In this work we propose calculating the whole probability distribution of a measure instead of only calculating a single value. This idea was first proposed by Liew and Lu in the form of survivability functions [44], but as of yet has not been applied to geographically correlated challenges.

By computing the distribution of a measure instead of a single value, one can see the complete picture of the vulnerability of the network with respect to a set of disasters, and make better decisions as a result. There is a big difference between a network with a uniform distribution of its vulnerability measure and a network with only high and low impact events, although measures for both networks may have the same expected value. When considering worst case disasters, it is useful to know the probability of such a disaster occurring, and if other disasters exist which will result in similar, but slightly better outcomes. These are just two examples of situations in which a distribution can help give a better picture of the vulnerability of a network.

Additionally, properties like the expected value, or the worst case value, can be easily derived from the distribution itself.

Note that the distributions are always discrete, as the measure can only take up a finite amount of values for a single network.

Our problem statement is as follows:

Definition 1. Vulnerability Distribution Problem Given a network $G = \{V, E\}$ and a disaster set D, each with a disaster area $A(d)$ (either a circle, line segment, hippodrome, simple polygon, or a finite union of these) and an occurrence probability $P(d)$ (as described in chapter 3).

Let M be the random real-valued variable describing the state of the network after exactly one disaster from D randomly occurs.

Compute

$$P(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$$

We define the distribution of M as the vulnerability distribution of network G (based on the disaster set D).

In section 4.1 we show how to compute the distribution over failures states, which we will use in section 4.2 to efficiently compute vulnerability distributions. Finally, in section 4.3 we discuss how to find the vulnerability of groups of links, based on the vulnerability distribution of a network.

4.1. Failure States

We define a failure state as follows:

Definition 2. Failure State Given a network $G = \{V, E\}$, define a failure state of G as a set $s \subseteq E$, where $e_i \in s$ if and only if e_i is down.

By computing the distribution over these failure states first, we can more efficiently compute the distribution over a measure. In addition, this failure state distribution is also useful by itself when preparing for or protecting the network against disasters.

Let S be the random variable indicating the failure state after a random disaster and let $S(d)$ be the failure state after disaster $d \in D$. Thus $S(d)$ is the set of all links intersecting the disaster area $A(d)$.

Because we assume exactly one disaster occurs, we have

$$P(S = s) = \sum_{d \in D | S(d) = s} P(d) \tag{4.1}$$

The distribution over S can now easily be computed as follows:

1. $\forall d \in D$, compute $S(d)$
2. $\forall s \in S[D]$ (the image of S), store $S^{-1}(s) = \{d \in D | S(d) = s\}$
3. $\forall s \in S$, $P(S = s) = \sum_{d \in S^{-1}(s)} P(d)$

Note that $|S[D]| \leq |D|$ (trivially), and can be significantly smaller. In chapter 7 we show experimentally that in most, if not all, cases the amount of possible failure states is much lower than the number of disasters in consideration. The value of the measure given a disaster depends solely on the failure state, and computing this value is often significantly more expensive than computing the state itself. So by iterating over possible failure states instead of disasters, we can significantly reduce computation time when computing the vulnerability distribution.

In addition to computing the vulnerability distribution, the failure state distribution can also be utilized to compute routes with minimum risk or probability of disconnecting and in optimally augmenting the network to increase its survivability against disasters.

In the next sections we discuss each computation step in more detail.

4.1.1. Step 1: Computing Failure States

The first step requires us to compute the failure state of each disaster. As computing these states can be done completely independently, this step can easily be parallelized, or distributed to multiple workers.

To compute the failure state of a disaster d, we iterate over each link $e_i \in E$ and test if it intersects the disaster area $A(d)$. As an efficient means of storage, and to be able to easily compare failure states, we store these failure states as bit vectors $s \in \{0,1\}^{|E|}$, where $s_i = 1$ if and only if e_i is down (i.e. intersects $A(d)$).

We briefly go over how to test if a link intersects a disaster area for each possible shape of disaster area. We assume the reader knows how to compute the distance between two points and of a point and a line segment, and how to test if two line segments intersect.

Circle

To test if a link $e_i \in E$ intersects a disaster $d \in D$ with a circular disaster area with center $c \in \mathbb{R}^2$ and radius $r \geq 0$ we simply iterate over each line segment of e_i. The link intersects the circle if at least one of these segments intersects it.

To test if a segment intersects the circle, we compute the squared distance dist^2 between the segment and c. If $\text{dist}^2 \leq r^2$ the segment intersects d.

Thus it requires $\mathcal{O}(N_i)$ steps to test if link e_i intersects a circle disaster d and $\mathcal{O}(N_e)$ steps to compute the failure state after a circle disaster d.

Line Segment

Testing if a line segment disaster d intersects a link $e_i \in E$ requires a very similar procedure. Iterate over each segment of e_i and test if it intersects $A(d)$.

It requires $\mathcal{O}(N_i)$ steps to test if the link intersects a line segment disaster d and $\mathcal{O}(N_e)$ steps to compute the failure state after a line segment disaster d.

Hippodrome
The distance between two non-intersecting line segments l_1 and l_2 is equal to the minimum of the distances between l_1 and the endpoints of l_2 and the distances between l_2 and the endpoints of l_1.

Thus we can again iterate over each line segment of e_i, compute the squared distance between the link's line segments and the hippodrome's line segment, and compare it to its squared radius.

A slightly more efficient method would compute the distances in such a way as to not compute the distance between the hippodrome line segment and the link line segments endpoints twice, but this only has a slight effect on performance.

Again, it requires $\mathcal{O}(N_i)$ steps to test if link e_i intersects a hippodrome disaster d and $\mathcal{O}(N_e)$ steps to compute the failure state after a hippodrome disaster d.

Simple Polygon
To test if a link $e_i \in E$ intersects a simple polygon area $A(d)$ we iterate over every line segment of e_i.

First we check if the squared distance between the (pre-computed) centroid (or another given center point) and the line segment is greater than the (pre-computed) maximum squared distance between this point and the vertices of the polygon. If so, the line segment does not intersect the disaster area and we can move on to the next segment.

Next, we iterate over all line segments (sides) of the polygon, and test if they intersect the link line segment. If at least one of them does, the link intersects the disaster area.

Finally, there is still the possibility that the line segment is completely covered by the polygon. Thus we count the number of times a vertical line segment (ray) from one of the endpoints of the link line segment to a point above the polygon intersects the polygon sides. If and only if the number of intersected sides is uneven the line segment intersects the disaster area. In the rare case that this ray passes through an endpoint of a polygon side, we only count this intersection if the other endpoint is positioned to the right of the ray, to prevent this single intersection for counting twice (each polygon vertex is the endpoint of two sides).

Let $N(A(d))$ be the number of line segments of simple polygon disaster d. The amount of steps required to test if e_i intersects d is $\mathcal{O}(N_i N(A(d)))$, thus it requires $\mathcal{O}(N_e N(A(d)))$ steps to compute the failure state after simple polygon disaster d.

Union of Areas
Let $A(d)$ be a union of disaster areas. We know $A(d) = A_1 \cup A_2 \cup \cdots \cup A_k$, where each A_i is either a circle, line segment, hippodrome or simple polygon. To test if a link intersects $A(d)$ we iterate over each A_i and test if the link intersects it. If the link intersects one or more of these areas, the link intersects $A(d)$.

Define the function N from valid disasters areas to $\mathbb{N}$ as follows:

$$N(A) := \begin{cases} 1, & \text{if } A \text{ is a circle, line segment or hippodrome} \\ \text{the number of polygon sides,} & \text{if } A \text{ is a simple polygon} \\ \sum_{i=1}^{k} N(A_i), & \text{if } A \text{ is a union of } k \text{ disaster areas} \end{cases} \tag{4.2}$$

It requires $\mathcal{O}(N_i N(A(d)))$ steps to test if e_i intersects $A(d)$ and $\mathcal{O}(N_e N(A(d)))$ steps to compute the failure state after a union of areas disaster d.

Let $N(d) = N(A(d))$ and $N(D) := \sum_{d \in D} N(d)$. Step 1 has a runtime of $\mathcal{O}(N(D)N_e)$.

4.1.2. Step 2: Grouping Disasters by Failure State

We now have the failure state for each disaster in D. To complete step 2, we can iterate over the set of disasters as follows:

```
**for** $d \in D$ **do**
    Let $s$ be the failure state of $d$
    **if** This is the first time we encounter $s$ **then**
        $S^{-1}(s) \leftarrow \{d\}$
    **else**
        $S^{-1}(s) \leftarrow S^{-1}(s) \cup \{d\}$
    **end if**
**end for**
```

The inverse images ($S^{-1}(s)$) can efficiently be stored in a hash table. In this case this step has a runtime of $\mathcal{O}(|D|)$.

4.1.3. Step 3: Computing $P(S = s)$

This step is quite straightforward, as one only needs to sum over the occurrence probabilities of all disasters in $S^{-1}(s)$ to compute $P(S = s)$.

In our implementation, we only compute these values on the fly, during the computation of the vulnerability distribution. Alternatively, one could compute $P(S = s)$ for all $s \in S[D]$, to prevent having to iterate over each $s \in 2^E$ (if $s \notin S[D]$, then $P(S = s) = 0$).

In both cases the time complexity of this step is $\mathcal{O}(|D|)$ and the time complexity of all three steps combined is $\mathcal{O}(N(D)N_e)$.

4.2. Measures

Let $M(d)$ be the value of measure M after disaster d, and $M(s)$ the value of M in network state s. Note that $M(d) = M(S(d))$.

Similarly as in equation 4.1, we have

$$\begin{aligned} P(M = m) &= \sum_{d \in D | M(d) = m} P(d) \\ &= \sum_{s \in S[D] | M(s) = m} \left(\sum_{d \in D | S(d) = s} P(d) \right) \\ &= \sum_{s \in S[D] | M(s) = m} P(S = s) \end{aligned} \tag{4.3}$$

The distribution of M can now be calculated as follows:

1. $\forall s \in S[D]$, compute $P(S = s)$ as described in section 4.1
2. $\forall s \in S[D]$, compute $M(s)$
3. $\forall m \in M[S[D]]$, store $\{s \in S[D] | M(s) = m\}$
4. $\forall m \in \mathbb{R}$, $P(M = m) = \sum_{s \in S[D] | M(s) = m} P(S = s)$

These steps are very similar to those of computing the state distribution. We go over the last 3 steps in the next sections.

4.2.1. Step 2: Computing the Value of M for All Possible Failure States

This step is extremely similar to step 1 of section 4.1. Instead of iterating over disasters, we iterate over each possible failure state (given the disaster set D) and compute the value of M for each state. Thus M needs to be computed $|S[D]| \leq |D|$ times.

The runtime of this step depends mostly on the computation time of M, given a network state s.

In this work we focus on the weighted average 2-terminal reliability, which we define as follows:

Definition 3. Weighted Average 2-Terminal Reliability (WATTR) Let $w_i \in \mathbb{R}$ be the weight of node v_i.

Let $I(i, j) = \begin{cases} 1 & \text{if node } v_i \text{ is connected to node } v_j \\ 0 & \text{otherwise} \end{cases}$

The weighted average 2-terminal reliability (WATTR) is defined as

$$\text{WATTR} := \frac{1}{W} \sum_{v_i \in V} \sum_{v_j \in V - \{v_i\}} w_i w_j I(i, j) \tag{4.4}$$

where

$$W := \sum_{v_i \in V} \sum_{v_j \in V - \{v_i\}} w_i w_j \tag{4.5}$$

By taking a weighted sum of connected node pairs, more importance can be assigned to certain nodes, for example those connecting the network to an Internet Exchange. If all weights are set to 1, WATTR is equivalent to the well-known average 2-terminal reliability (ATTR).

If all weights are greater than 0, WATTR always takes a value between 0 and 1. Additionally, WATTR = 0 if and only if no node pairs are connected (i.e. all links are down) and WATTR = 1 if and only if all nodes are connected to all other nodes.

In an undirected network, WATTR can be computed more quickly by iterating over connected components instead of node pairs. Let $\mathscr{C}$ be the set of all connected components of the network and define

$$\text{sum}(c) := \sum_{v_i \in V} w_i \tag{4.6}$$

for all $c \subseteq V$. Then (for an undirected graph):

$$W = \sum_{v_i \in V} w_i * (\text{sum}(V) - w_i) \tag{4.7}$$

and

$$\text{WATTR} = \frac{1}{W} \sum_{c \in \mathscr{C}} \sum_{v_i \in c} w_i * (\text{sum}(c) - w_i) \tag{4.8}$$

Or alternatively:

$$W = 2\left(\text{sum}(V)^2 - \sum_{v_i \in V} w_i^2\right) \tag{4.9}$$

and

$$\text{WATTR} = \frac{2}{W} \sum_{c \in \mathscr{C}} \text{sum}(c)^2 - \sum_{v_i \in c} w_i^2 \tag{4.10}$$

These sums can be computed in $\mathscr{O}(|V|)$ time (assuming relatively low weights), compared to computing WATTR the naive way, which takes $\mathscr{O}(|V|^2)$ time (ignoring the time it takes to compute all $I(i,j)$). The connected components can found by a simple breadth-first search algorithm, which completes in $\mathscr{O}(|V| + |E|)$ time.

In a directed network, we can iterate over each $v_i \in V$ and find all nodes connected to v_i in $\mathscr{O}(|E|)$ time, so computing W would take $\mathscr{O}(|V|^2)$ time and computing the WATTR would take $\mathscr{O}(|V||E|)$.

In some cases the probability of node pairs disconnecting is very small. If no node pairs disconnect, WATTR = 1, no matter how much bandwidth capacity is lost or how much hardware fails. In these situations it might be helpful to use another survivability measure. One of the more simple survivability measures is the amount of surviving (or failed) links.

We define the Link Survival Ratio (LSR) as follows:

Definition 4. Link Survival Ratio The Link Survival Ratio (LSR) of a network $G = \{V, E\}$ in failure state $s \subseteq E$ is

$$\text{LSR} = \frac{|E| - |s|}{|E|} \tag{4.11}$$

4.2.2. Step 3: Grouping Failure States by Measure

After computing the value of M for each possible failure state, these states need to be grouped by M-value.

This can be done in almost exactly the same manner as disasters were grouped by failure state in section 4.2.1. We iterate over each failure state in $S[D]$, storing $\{s \in S[D] | M(s) = m\}$ in a hash table with key m.

The runtime of this step is $\mathscr{O}(|S[D]|) = \mathscr{O}(|D|)$.

4.2.3. Step 4: Computing the Vulnerability Distribution

Finally, the values of the vulnerability distribution can be computed by simply summing over the failure state probabilities.

Note that it is not required to compute $P(M = m)$ for $m \notin M[S[D]]$, as in this case $P(M = m) = 0$.

Computing $P(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$ in step 4 takes $\mathscr{O}(S[D]) = \mathscr{O}(|D|)$ time.

In practice, computing the distribution over the failure states of G will generally take up significantly more time than step 2 to 4 of computing the vulnerability distribution itself.

4.3. Vulnerable Links

In addition to computing the vulnerability of the whole network, it is also useful to figure out the vulnerability of specific groups of links to disasters. If a specific group of links turns out to be both very vulnerable to a type of disaster and integral to the network function, measures should be taken to either reduce the probability that the groups of links will fail (e.g. by reinforcing them, or moving them out of potential disaster areas) or reduce the impact of their failure on the network (e.g. by pre-installing backup links or re-arranging content throughout the network).

For this purpose it is insufficient to consider only the probability of failure, as groups of links with a high probability of failure, but whose failure does not significantly impact the network need not be considered vulnerable parts of the network. In addition, groups of links whose failure would significantly impact the network, but have a very low probability of failure can also be safely ignored.

Assume the values of M range from 0 to 1, where $M = 0$ indicates a complete network failure and $M = 1$ a fully functioning network. We define the following impact (random) value:

Definition 5. Given a set of links $L \subseteq E$, the impact I_L of these links in state S is defined as

$$I_L = \begin{cases} M(S \setminus L) - M(S) & \text{if } L \subseteq S \\ 0 & \text{otherwise} \end{cases}$$

By computing the distribution of I_L for a group of links L, we get a clear picture of how much this specific group of links might impact M.

If some links in L do not fail, the impact I_L of L is considered to be 0, as the damage to M is not caused by the failure of the complete group L. To further exemplify this, consider two groups of links, which, due to their large distance from each other, can never fail at the same time. Each of these groups can individually have a high impact on M, but as a combined group will never fail at the same time. The combination should never be seen as a single vulnerable group.

For a fixed set of links L, I_L can be considered to be a single measure, and thus easily be calculated as described in section 4.2.

5 Obtaining a Finite Set of Disasters

One often overlooked part of calculating the vulnerability of a network is how to obtain the actual disaster (input) data. Typically it is assumed that this information is already known to the network operator, even though obtaining this data is non-trivial. In this chapter we show how to create the set of disaster areas and occurrence probabilities required for our methods by means of case studies on real datasets.

In section 5.1 approaches are given to obtain a finite set of disaster areas and probabilities that is representative for a given infinite set of disasters. Next, in section 5.2 we demonstrate these approaches on real disaster datasets.

5.1. Infinite Set of Disasters

In many cases the amount of possible distinct disasters is infinite. For example earthquakes could occur anywhere along a fault line, and with different amount of magnitudes. However, many of these possible earthquakes would result in exactly the same links failing in the network, due to their close position to each other and similar magnitude. We can make use of this property to try to find a finite set of disasters that is representative for the (infinite) set of disasters as a whole.

Let D_∞ be the infinite set of possible disasters. Preferable, we would obtain a finite set D such that a function

$$f : D_\infty \to D$$

exists with

$$S(f(d_\infty)) = S(d_\infty)\ \forall d_\infty \in D_\infty \tag{5.1}$$

and

$$P(d) = P(f^{-1}(d))\ \forall d \in D \tag{5.2}$$

i.e. each disaster in D_∞ has a representative disaster in D, and the occurrence probability of $d \in D$ is the probability that any of the disasters it represents will occur. In this case the vulnerability distributions for D would be exactly the same as those for D_∞. More generally this is the case if and only if

$$P(\{d \in D | S(d) = s\} = P(\{d_\infty \in D_\infty | S(d_\infty) = s\}))\ \forall s \in 2^E \tag{5.3}$$

Such a set D always exists:

Theorem 1. *Let D_∞ be an infinite set of possible disaster areas and $G = \{V, E\}$ a network as defined in section 3.1. There exists a finite set $D \subseteq D_\infty$ and a function $f : D_\infty \to D$ such that equations 5.1 and 5.2 hold.*

Proof. The number of links $|E|$ is finite and the number of possible states is $2^{|E|}$. Construct D and f by picking exactly one $d \in \{d_\infty \in D_\infty | S(d_\infty) = s\}$ for all $s \in S[D_\infty]$ and setting $P(d) = P(\{d_\infty \in D_\infty | S(d_\infty) = s\})$ and $f(d_\infty) = d\ \forall d_\infty \in \{d_\infty \in D_\infty | S(d_\infty) = s\}$. $|D| \le 2^{|E|}$ and equations 5.1 and 5.2 hold trivially. □

Remark 1. The construction from this proof is not usable in practice, as it requires $P(\{d_\infty \in D_\infty | S(d_\infty) = s\}) = P(s)$ to be known for all states $s \in 2^E$. However, if these values, i.e. the failure state distribution, are already known, it is not necessary to construct D, as the vulnerability distribution can be calculated directly.

It may not always be practical to find a set satisfying properties 5.1 and 5.2, or even property 5.3. In the next two subsections we go over three methods to find disasters sets which can be used to approximate the distribution over S. In addition, these methods do not depend on the network N, so the resulting disaster set D can be used with any network topology.

5.1.1. Monte Carlo

If the distribution of all possible disasters is known, a Monte Carlo approach can be taken by randomly sampling and adding disasters to D, setting each occurrence probability to $\frac{1}{N}$, where $N = |D|$.

By the law of large numbers $\sum\limits_{d \in D | S(d) = s} \frac{1}{n}$ converges almost surely to $P(S = s)$. Thus the vulnerability distribution can be computed as described in chapter 4 (see equation 4.1).

5.1.2. Historic Dataset

Another possibility, which does not require the distribution to be known, is to take the previous N disasters which actually occurred as D. However, to get accurate results this requires a large amount of historic data, which may not be available.

As in the previous approach, occurrence probabilities should be set to $\frac{1}{N}$, where $N = |D|$.

5.1.3. Scenarios

Instead of randomly selecting disasters, one can also take the exact opposite approach. By choosing a set of representative disaster scenarios as the input set D. These disasters can be chosen, or even created, to represent D_∞ in a way that meets the network operators needs as good as possible. The scenarios do not need to correspond to disasters in D_∞ itself, and full knowledge of (the distribution of) D_∞ is not required.

The disaster area, or hypothetical effect on network infrastructure, of randomly sampled or historical disasters may not be accurately known. But by creating custom tailored scenarios, it is possible to more accurately compute the necessary specifics based on a very specific set of input parameters.

However, when using this method it is important to create or select scenarios in such a way that the final result is still representative for the vulnerability of a network to a (chosen) broader class of interesting (or dangerous) possible disasters. It is easy to select or create scenarios that under- or over-estimate the vulnerability of a network, but these are completely useless for our purpose.

In all three approaches, but especially in the scenarios approach, it can be helpful to employ experts on the type of disasters under review (e.g. seismologists when considering earthquakes) to help in creating the set of representative disasters.

5.2. Case Studies

In this section we go through some case studies on how to create representative disaster sets D for classes of disasters. In section 5.2.1 we consider earthquakes, which due to only occurring on faults and the wealth of detailed maps available on their (potential) intensity are a perfect fit for our disaster model. In 5.2.2 we study how to prepare against a single hurricane which is known to be approaching the network.

5.2.1. Earthquakes

An earthquake is caused by a sudden slip on a fault, releasing energy in waves and causing the ground to shake [2]. Depending on its intensity, this shaking can cause a large amount of damage to network infrastructure.

An earthquake damaging submarine cables can disconnect communications in multiple countries at once. Earthquakes and tsunamis striking coastal or inland network hardware can also inflict significant damage on other infrastructure, making it more difficult to quickly repair the network. In addition, the damage and panic caused by these quakes increases both the usage and importance of communication networks in the area.

Earthquakes typically occur at faults [2], and thus can not occur everywhere in R. In addition, the ground motion, and thus the disaster area after an earthquake, depends on its magnitude, as well as the properties of the rocks and sediments that earthquake waves travel through [4].

Many earthquakes in a similar location affect the same links of the network, even though their exact disaster area may differ. We therefore argue that it makes sense to take a finite representative set of earthquakes and use it to calculate network vulnerability.

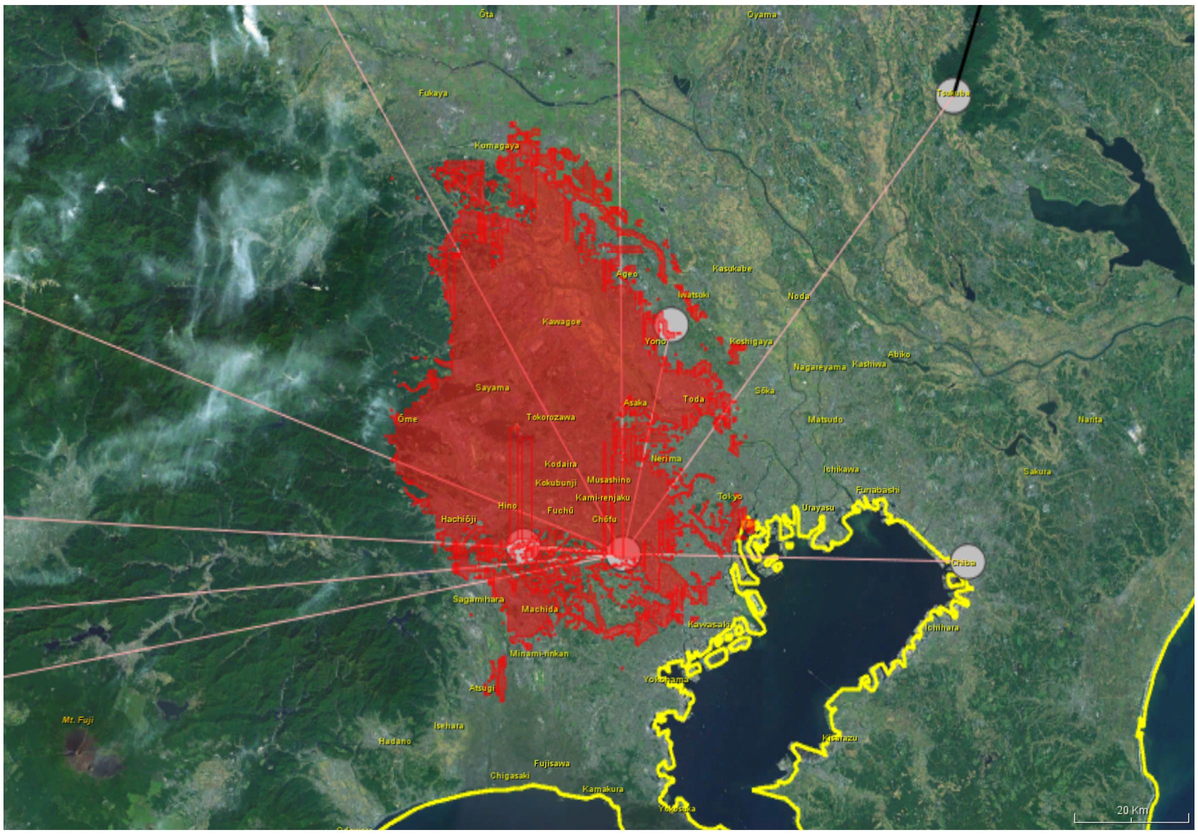


Figure 5.1: Visualization of an earthquake disaster area. The disaster area is painted red, while affected links are colored pink.

A lot of research has been done on computing the probabilities for potential future earthquakes and their impact. As such, many detailed maps are available of the potential (ground shaking) intensity of potential earthquakes.

In this section we consider earthquake datasets from two different organizations: the Japan Seismic Hazard Information Station (J-SHIS) [6] and the United States Geological Survey (USGS) [13].

J-SHIS

Japan has one of the highest earthquake rates in the world and thus needs to be especially prepared for major earthquakes [38]. The National Research Institute for Earth Science and Disaster Resilience (NIED) provides much information about potential earthquakes through the Japan Seismic Hazard Information Station (J-SHIS) [6]. Of particular interest to us are their Seismic Hazard Map and Scenario Earthquake Shaking Maps.

The Seismic Hazard Map gives probabilities for significant ground motion for all of Japan. These probabilities are calculated in a very similar method as our approach: by aggregating over a set of (representative) modeled earthquakes [32]. Unfortunately, as the end result is an aggregation, and the intermediate results are not publicly available, this map was not usable for our purposes.

Instead, we made use of the Scenario Earthquake Shaking Maps. These are maps of the effect of “characteristic earthquakes occurring in major active fault zones, subduction-zone characteristic earthquakes, and characteristic earthquakes that occur in the eastern margin of the Japan Sea subject to long-term evaluation” [32]. Of special interest are the earthquakes occurring in major active fault zones, as these are the highly active fault zones that cause earthquakes that have a large social and economical impact [7].

These scenario maps contain, among other data, (JMA) seismic intensities for each affected grid in Japan using Divided Quarter Grid Square Codes [1, 17]. By converting these to geographical coordinates, and only keeping those grids with an intensity above a specific threshold, a disaster area (of a union of rectangles) can be obtained for every single scenario on the dataset. As the amount of rectangles can be very numerous, we combined adjacent rectangles to form larger simple polygons. The resulting disaster area is not contiguous, as there are gaps where the seismic intensity is below the threshold. An example disaster area can be seen in figure 5.1.

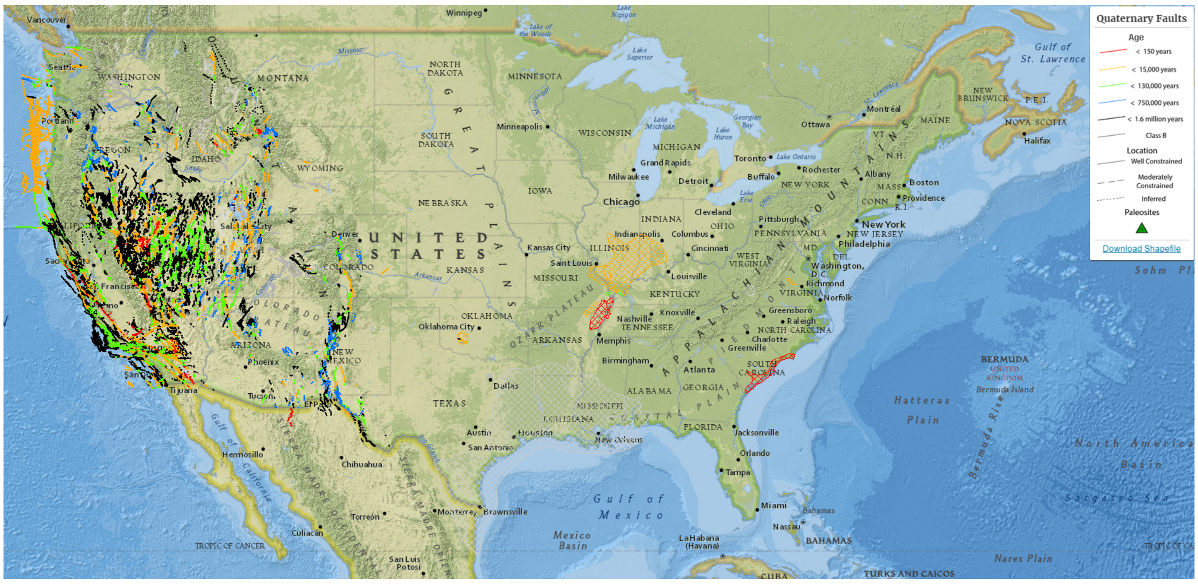


Figure 5.2: Quaternary faults in the contiguous United States [3].

The scenarios do not contain occurrence probabilities. To obtain these probabilities we take the mean recurrence intervals for each fault from the parameter dataset for the Seismic Hazard map. If a fault segment has N scenarios and mean recurrence interval i, the occurrence probability of all its disasters is taken to be

$$\frac{1}{iNT},$$

where T is the sum of the inverses of all recurrence intervals of fault segments with $N > 0$.

USGS

The U.S. contains a large number of active faults. Most of these faults lie in the western part of the U.S., but one should not ignore the risk of earthquakes in the central and eastern parts, as large earthquakes have occurred in both the western [12] and the central and eastern parts of the U.S. [11, 25]. A map of the quaternary faults in the contiguous United States can be seen in figure 5.2.

In California known faults are an accurate predictor of future earthquakes. However, east of the Rockies individual known faults and fault lines are generally unreliable guides to the likelihood of earthquakes. In these parts earthquakes are as likely, or even more likely, to occur on an unknown fault than a known fault. However, most new earthquakes still occur in the general vicinity of past earthquakes. Thus, while potentially less accurate, our model is still applicable.

Similarly to NIED, the United States Geological Survey (USGS) periodically produces hazard maps for the United States. In addition, they also provide earthquake scenario maps of the realization of a hypothetical earthquake by assuming a particular magnitude, location, and (optionally) fault-rupture geometry. Finally, they have a large catalog of (worldwide) historical earthquakes. Their hazard maps are based on a combination of probabilistic seismic hazard analysis and deterministic seismic hazard analysis. A probabilistic analysis accounts for all possible ruptures, while a deterministic analysis only considers a finite representative set of earthquakes.

USGS has created its own tools and format for creating their scenarios and mapping the ground motions of earthquakes, called Shakemap [66]. This tool is used by organizations across the globe, including in France, Italy, Indonesia and Iran. Thus being able to properly process these maps allows us to create earthquake vulnerability distributions for networks in a large number of countries.

One of the output formats of Shakemap is an interpolated ground motion grid ("grid.xml"). This file contains ground motion info, including intensity, of points on a finely-sampled grid. We split the affected region into cells, centered on the grid points. By assuming each cell has a uniform intensity equal to that of its center point we can apply a similar process as used on the J-SHIS scenario maps to find the disaster area for each Shakemap map. The resulting disaster area is again composed of a union of rectangles.

We took two different approaches with the U.S. data. The first approach was to download and convert a set of scenarios, the second was to download and convert a historic set of earthquakes.

As our set of scenarios we took the “Building Seismic Safety Council 2014 Event Set”, which consists of the characteristic earthquakes of all known faults in the coterminous U.S., from the 2014 version of the USGS national seismic hazard maps. Unfortunately, we could not find enough data to base the occurrence probabilities on, so these were all set to $\frac{1}{N}$, where N is the number of scenarios.

The resulting disaster set leads to inaccurate vulnerability distributions. Not only are the probabilities not accurate, but there are also no earthquakes included to represent the potential risk from unknown faults. The end result is probably an overestimation of vulnerability of the western U.S., as earthquakes with a high magnitude will be given the same probability as earthquakes with a lower magnitude (note that this is also an issue with the J-SHIS scenario set), and an underestimation of vulnerability in the central and eastern part of the U.S., as earthquakes there often occur on unknown faults.

Fortunately, USGS also provides Shakemap maps for (some) historical earthquakes. When using these maps we do not require any information on probabilities. Unfortunately, the catalog of shakemaps does not go back very far (the oldest earthquake in our dataset occurred in 1952). However, this problem will be solved automatically as both older earthquakes and newly occurring ones are converted to the Shakemap format and added to the catalog.

Our historical dataset was created by downloading and converting all 2844 available shakemaps in the USGS catalog [14] of earthquakes with a magnitude larger or equal than 2.5 having occurred before 2017 in the contiguous U.S. [1]

Tsunamis

A large part of the destruction caused by the 2011 earthquake of the coast of Japan was not caused by the ground motions themselves, but by the resulting tsunami. We did not take tsunamis into account in the previous sections, as they are not included in the datasets. However, if necessary, their effect can be included by computing the resulting tsunami disaster area for each earthquake in D, and then adding this area to the earthquake disaster area.

When computing the vulnerability of a real network topology, one should not blindly download and use general datasets, but work together with experts to generate an appropriate disaster set. This includes computing or modifying disaster areas where necessary.

5.2.2. Hurricanes

Hurricanes, typhoons and cyclones are tropical cyclones with maximum sustained winds of 74 mph (±119 km/h) or higher [5]. The name of these storm systems depends purely on location. In the Atlantic Ocean and northeastern Pacific Ocean they are called hurricanes, in the western North Pacific typhoons, and in the Indian Ocean and South Pacific Ocean cyclones. In this report we refer to all these simply as hurricanes.

Hurricanes can inflict a large amount of damage to network infrastructure. Both directly, due to extreme wind speeds, and indirectly, due to the flooding caused by storm surges and rainfall.

Hurricane Katrina caused extensive damage to the U.S. Gulf Coast’s telecommunications infrastructure. Not only was the hardware itself damaged by wind and flooding, there was also an extended electrical outage following the storm. As there was not enough fuel available in the area to supply the backup generators, more facilities soon lost power and went off-line [47].

Incoming Hurricane

Hurricanes are one of the few types of disasters whose arrival can be known days in advance, as storm systems form on the ocean and slowly move their way to land. This gives network operators the ability to compute the impact of a hurricane on their network before it actually arrives. Unfortunately, the prediction of the path a tropical cyclone will take is not perfect. As such, there will be some uncertainty in the results. To be able to properly prepare against the storm, one could use our models and methods to compute the vulnerability of the network against the hurricane.

While the end result will still be the same, this approach is different from what we described until now. Before we assumed disasters in D were (mostly) completely separate disasters. But in this section all disasters in D are different possibilities of the same hurricane.

The National Hurricane Center (NHC), located at Florida International University in Miami, is part of the National Centers for Environmental Prediction (NCEP). Their mission is “to save lives, mitigate property loss,

[1]The area positioned between (24.6N, 125W) and (50N, 65W).

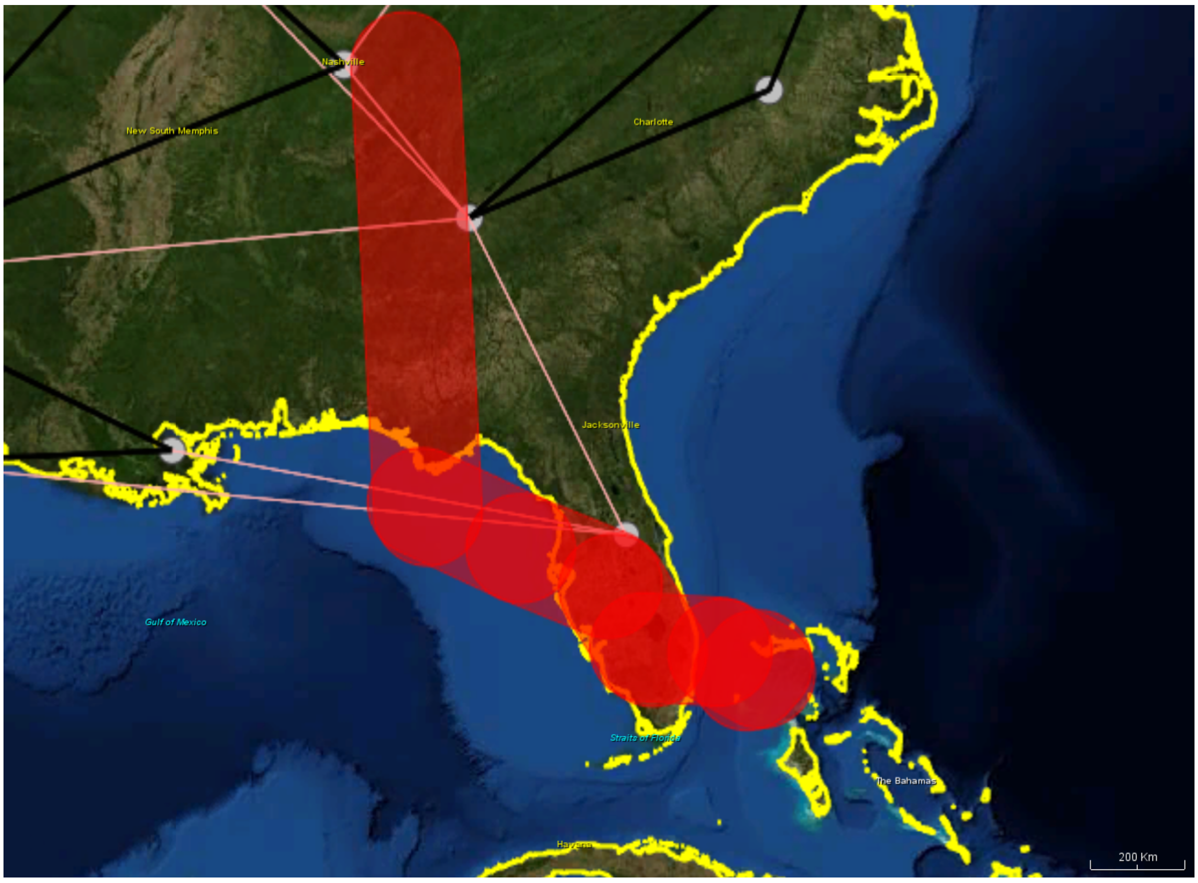


Figure 5.3: Realization of Hurricane Katrina, based on the “5 AM EDT THU AUG 25 2005” forecast. The disaster area is painted red, while affected links are colored pink.

and improve economic efficiency by issuing the best watches, warnings, forecasts, and analyses of hazardous tropical weather and by increasing understanding of these hazards” [8]. In the United States, almost all hurricane forecast information that public officials and the public at large receive is a (repackaged form of) NHC data [57].

When a tropical cyclone is active, the NHC publishes new “advisory packages” every 6 hours. These packages include the predicted track the storm system will follow. NHC forecasters try to predict the track and intensity of tropical cyclones by subjectively combining the results of several models [10]. The track is formed by predicting the position of the storm center at 12, 24, 36, 48, 72, 96 and 120 hours in the future and connecting these positions with a straight line segment [49, 57].

Two of the more interesting products the NHC creates are the “Potential Storm Surge Flooding Map” and “Tropical Cyclone Surface Wind Speed Probabilities”. The former shows the areas where flooding from storm surges could occur and how high above ground the water could reach in those areas and the latter the likelihood of sustained winds meeting or exceeding specific thresholds at particular locations. Both of these products are created by performing many simulations in a Monte Carlo approach based on the predicted hurricane track and intensity and the historical errors in these predictions. Our advice for computing the vulnerability of a network against a hurricane is to use the same (or similar) software and models as used by the NHC to create these products to run multiple simulations computing the potential flooded areas and areas with high wind speeds. Each simulation can be considered as a separate disaster in D, taking the affected areas as the disaster area.

To exemplify and test our models and methods we propose a more simple hurricane model based on the NHC Track Forecast Cone. The “Tropical Cyclone Track Forecast Cone” shows the probable path of the center of a tropical cyclone. The cone is formed by simply placing a circle around each predicted track position and connected them. The size of each circle is set so that two-thirds of historical official forecast errors over a 5-year sample fall within the circle. The basic idea is that the actual track can be expected to completely remain within the cone roughly 60-70% of the time.

We assume the actual track positions (in our 2D model) are distributed around the predicted positions according to a bivariate Normal distribution. This distribution is composed of normal distributions for the horizontal and vertical positions, each with a standard deviation of $\sqrt(\frac{r^2}{\ln(10000/1225)})$, where r is the radius of the corresponding circle, to ensure 65% of samples lie inside the cone.

This standard deviation was derived as follows:

Proof. Suppose we have two normally distributed random variables X and Y, the horizontal and vertical parts of a track position. Let $\mu = (\mu_x, \mu_y)$ be the predicted track position and σ the standard deviations of both X and Y. μ_x is the mean of x, and μ_y the mean of y.

The squared distance D from the track position to the predicted position is $D = (X-\mu_x)^2 + (Y-\mu_y)^2$. Note that $\frac{D}{\sigma^2} \sim \chi^2(2)$. We require that $P(D \le r^2) = 0.65$. Thus $0.65 = P(D \le r^2) = P(\frac{D}{\sigma^2} \le \frac{r^2}{\sigma^2}) = 1 - \exp(-\frac{r^2}{2\sigma^2})$. By simple manipulation it follows that $\sigma = \sqrt(\frac{r^2}{\ln(10000/1225)})$. □

We can randomly sample hurricane tracks for our own Monte Carlo approach by sampling the track positions and then connecting them with a straight line segment. This only leaves us with the problem of computing the disaster area based on a hurricane track.

The strike circle of a hurricane, based on the typical extent of hurricane force winds, is a circle with diameter 231.5 km, centered 23.15 km to the right of the hurricane center (based on its motion) [9]. In our approach we take this circle as the disaster area. Because the hurricane moves through the network area, the complete disaster area of each sampled track takes the form of a union of hippodromes.

Thus the complete approach is as follows:

1. Sample N sets of track positions.
2. For each track: compute the resulting disaster area.
3. Set all occurrence probabilities to $\frac{1}{N}$.
4. Compute the vulnerability distribution of the network, given this disaster set.

Figure 5.3 shows an example of the disaster area of one possible realization of Hurricane Katrina. Constructed using the above approach for the “5 AM EDT THU AUG 25 2005” forecast.

6 Computation and Visualization Tool

Without proper visualization of the vulnerability of a network, and of the reasons a network is vulnerable, network designers can not design disaster-resistant networks and network operators can not properly prepare for disasters. We have created a tool to both compute and visualize the vulnerability of a network. This tool can be used with any network topology and disaster set. In this chapter we describe how we convey our results and in which ways these visualizations can be used to analyze the network vulnerability.

The advantage of only computing a single vulnerability value, say the expected value of a measure, is that the result is clear, and different network topologies can be easily compared to each other. By computing both the distribution over failure states, and the vulnerability distribution, much more information is given about the vulnerability of a topology. This can help in decision-making, but does require the results to be properly visualized.

Vulnerability distributions can be plotted as a histogram of the cumulative distribution function (CDF), for example as in figure 6.1. The rightmost bar will always have length 1, but all other bars should preferably be as small as possible. The leftmost part of the histogram shows the probabilities that large parts of the network will become disconnected.

One can clearly see that for example $P(\text{At least 20\% of all node pairs disconnect}) = P(\text{ATTR} \leq 0.8) \approx 0.20$. The histogram gives a good first impression of the vulnerability distribution of a network.

However, it does not properly convey the details of the vulnerability distribution. Exact probabilities are hard to extract from such a graph, and very low probabilities are not visible in these plots. For example, the worst case result for this specific example is an ATTR of 0.091. But because $P(\text{ATTR} = 0.091) = 2.00 * 10^{-4} \approx 0$ this can not be extracted from the graph itself.

To help analyze the distribution in more detail, we also allow the user to request the value of any CDF values themselves. In addition, we show the value of M in the worst case outcome (and its probability of occurring), the expected value of M, and its variance.

Although all of this helps in studying the vulnerability of a network topology, it does not show why a network is or is not vulnerable to large-scale disasters. The CDF values do not show which groups of links have to fail to cause specific values of M, and which disasters cause these links to fail or what the probability is that a specific combination of links fails.

We can show these properties by coupling our intermediate results (the failure state distribution, and the hash table of disasters resulting in each failure state) and our final results (the vulnerability distribution) together. To visualize this coupling we combine the vulnerability distribution, failure state distribution and disasters themselves in a tree structure. In addition, using the NASA World Wind library [15], we draw both the topology and a selection of disasters on a map to clearly show which links are affected by a disaster and where they lie in the disaster area. This has been demonstrated, for a different example topology and disaster set, in figure 6.2.

At the top level one can see and select the values of the measure with their corresponding probability. Their child nodes show the probabilities of the states resulting in these values. Finally, at the lowest level are the individual disasters causing these states. By selecting one or more of these tree nodes, all corresponding disaster areas are drawn in red on the map. Failing links are colored pink.

In figure 6.2, we first expanded all failure states which result in an Average 2-Terminal Reliability of 0.87512. Subsequently, we expanded a specific failure state with 4 downed links. This failure state is the result of either

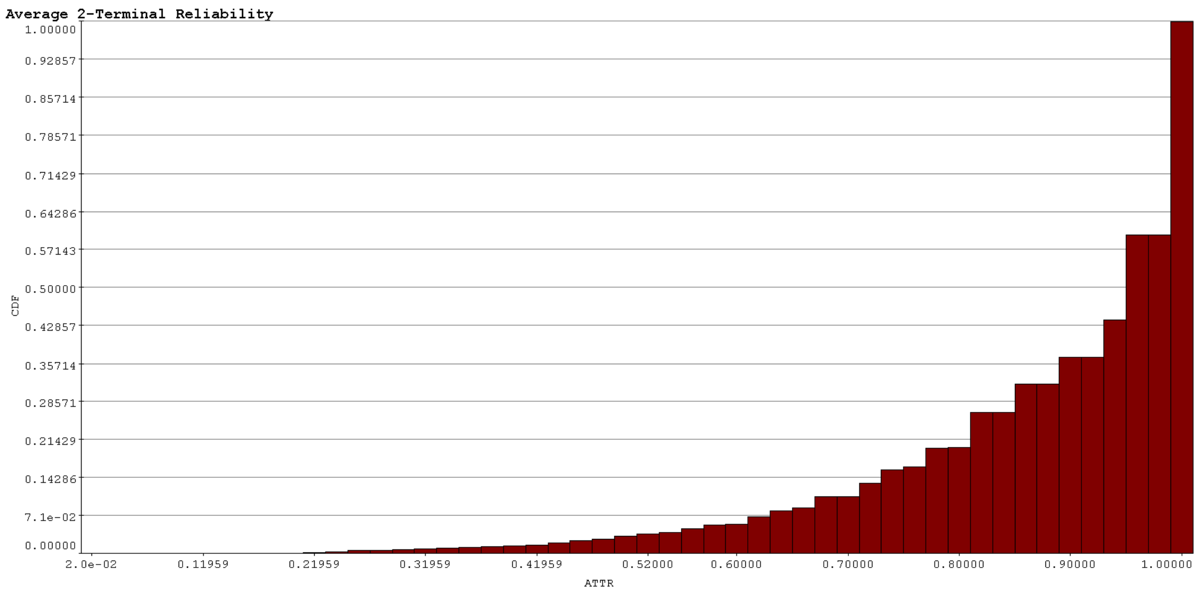
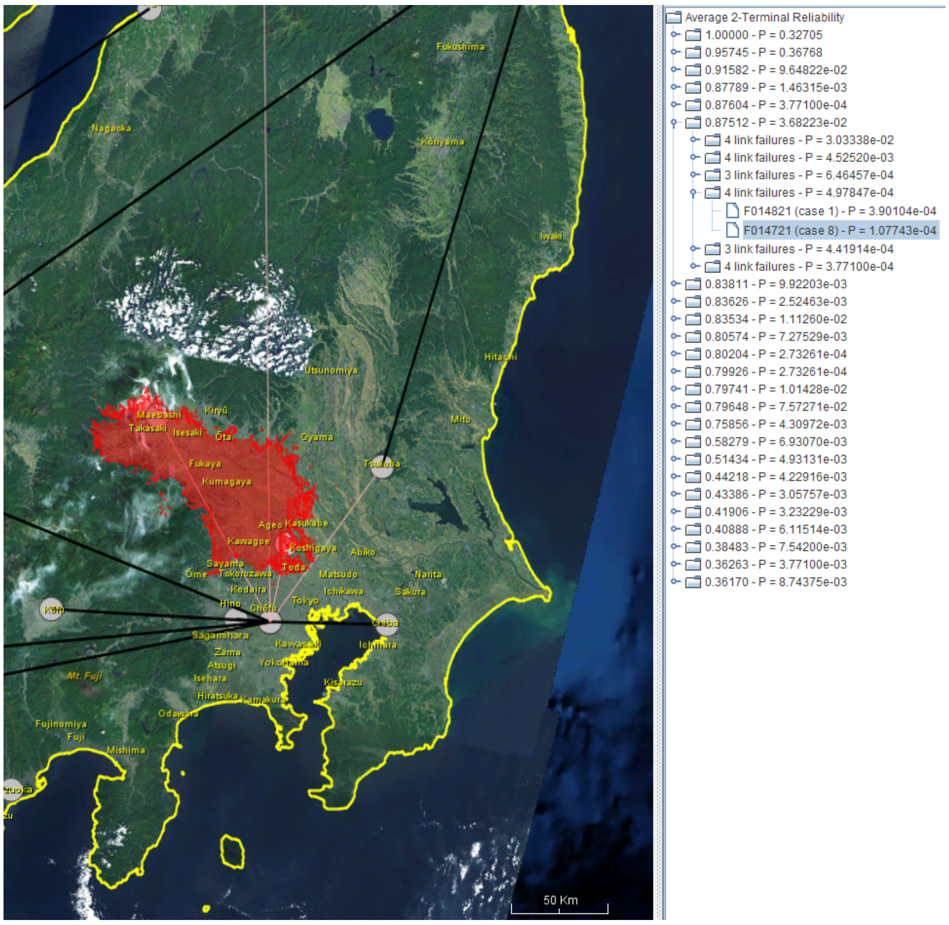


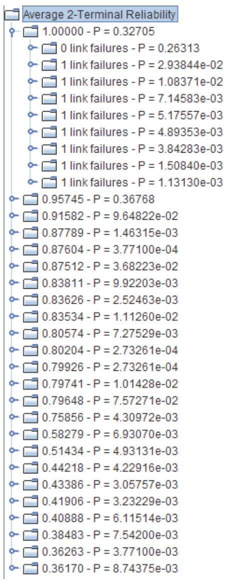
Figure 6.1: Histogram of the CDF of a (ATTR) vulnerability distribution.

disaster “F014821 (case 1)” or “F014721 (case 8)”. “F014721 (case 8)” was selected and is drawn on the map.

If, for example, we want to know the probability that no links are downed, we can expand the tree at ATTR = 1 and read the probability of “0 link failures” (see figure 6.3). Alternatively, we can draw a worst case disaster area by expanding the bottom-most ATTR node (in this case 0.36170) and selecting one of its disasters.



```
Average 2-Terminal Reliability
    1.00000 - P = 0.32705
    0.95745 - P = 0.36768
    0.91582 - P = 9.64822e-02
    0.87789 - P = 1.46315e-03
    0.87604 - P = 3.77100e-04
    0.87512 - P = 3.68223e-02
        4 link failures - P = 3.03338e-02
        4 link failures - P = 4.52520e-03
        3 link failures - P = 6.46457e-04
        4 link failures - P = 4.97847e-04
            F014821 (case 1) - P = 3.90104e-04
            F014721 (case 8) - P = 1.07743e-04
        3 link failures - P = 4.41914e-04
        4 link failures - P = 3.77100e-04
    0.83811 - P = 9.92203e-03
    0.83626 - P = 2.52463e-03
    0.83534 - P = 1.11260e-02
    0.80574 - P = 7.27529e-03
    0.80204 - P = 2.73261e-04
    0.79926 - P = 2.73261e-04
    0.79741 - P = 1.01428e-02
    0.79648 - P = 7.57271e-02
    0.75856 - P = 4.30972e-03
    0.58279 - P = 6.93070e-03
    0.51434 - P = 4.93131e-03
    0.44218 - P = 4.22916e-03
    0.43386 - P = 3.05757e-03
    0.41906 - P = 3.23229e-03
    0.40888 - P = 6.11514e-03
    0.38483 - P = 7.54200e-03
    0.36263 - P = 3.77100e-03
    0.36170 - P = 8.74375e-03
```



```
Average 2-Terminal Reliability
    1.00000 - P = 0.32705
        0 link failures - P = 0.26313
        1 link failures - P = 2.93844e-02
        1 link failures - P = 1.08371e-02
        1 link failures - P = 7.14583e-03
        1 link failures - P = 5.17557e-03
        1 link failures - P = 4.89353e-03
        1 link failures - P = 3.84283e-03
        1 link failures - P = 1.50840e-03
        1 link failures - P = 1.13130e-03
    0.95745 - P = 0.36768
    0.91582 - P = 9.64822e-02
    0.87789 - P = 1.46315e-03
    0.87604 - P = 3.77100e-04
    0.87512 - P = 3.68223e-02
    0.83811 - P = 9.92203e-03
    0.83626 - P = 2.52463e-03
    0.83534 - P = 1.11260e-02
    0.80574 - P = 7.27529e-03
    0.80204 - P = 2.73261e-04
    0.79926 - P = 2.73261e-04
    0.79741 - P = 1.01428e-02
    0.79648 - P = 7.57271e-02
    0.75856 - P = 4.30972e-03
    0.58279 - P = 6.93070e-03
    0.51434 - P = 4.93131e-03
    0.44218 - P = 4.22916e-03
    0.43386 - P = 3.05757e-03
    0.41906 - P = 3.23229e-03
    0.40888 - P = 6.11514e-03
    0.38483 - P = 7.54200e-03
    0.36263 - P = 3.77100e-03
    0.36170 - P = 8.74375e-03
```

Figure 6.3: By expanding node ATTR = 1, we can see that $P(\text{No Link Failures}) = 0.26313$

Figure 6.2: Visualization of distributions and disasters. The disaster area is painted red, while the affected links are colored pink.

7 Experimental Results

In this chapter we demonstrate our methods by computing the vulnerability distributions of 2 Japanese and 2 American network topologies, using the disaster sets described in chapter 5.

These topologies were downloaded from the Topology Zoo, a collection of over 250 physical network topologies [42]. As these files only contain (broad) geographical coordinates for the nodes, and not the links, all links are assumed to be straight line segments directly connecting their endpoints. The Mercator projection was used to map all geographical coordinates (both for the networks and disasters) to the 2-dimensional plane. Nodes without any geographical information were ignored.

Except where mentioned otherwise, we chose to use the average 2-terminal reliability (ATTR), the number of connected node pairs divided by the total amount of node pairs (i.e. WATTR with all weights set to 1), as our measure.

In section 7.1 we analyze the vulnerability of the topologies to earthquakes based on realistic datasets, and compare this to a similar analysis based on randomly placed circular disasters. Next, in section 7.2, we compute the vulnerability of the American networks to Hurricane Katrina based on two forecasts and using our model as described in section 5.2.2. In section 7.3 we analyze the runtime of our methods.

7.1. Earthquakes

7.1.1. Japanese Networks

Average 2-Terminal Reliability

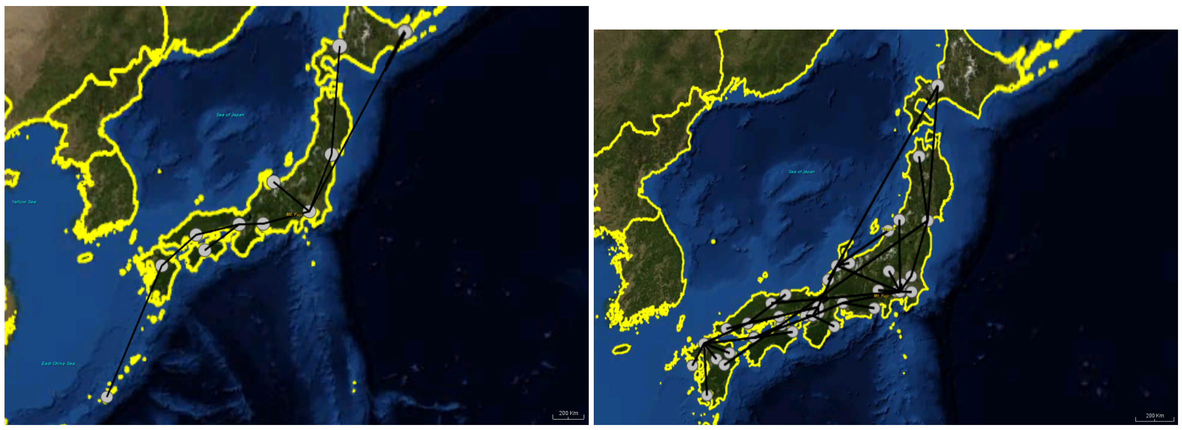
In this section we compute the vulnerability of two Japanese network topologies, JGN2plus-Japan (figure 7.1a) and Sinet (figure 7.1b), to earthquakes. The disaster set was created from the 2016 J-SHIS earthquake scenarios dataset, as described in section 5.2.1. These are a total of 655 scenarios for 189 fault segments. The JMA seismic intensity threshold was set to 5.5.

Although JGN2plus-Japan spans almost all of Japan, it only has 11 nodes and 10 links. In contrast, Sinet spans a slightly smaller area, but consists of 47 nodes connected by 49 links.

In figure 7.2 the cumulative distribution function of the ATTR of both networks after one of the earthquake scenarios has been plotted. One can immediately notice a significant difference between the two: while JGN2plus-Japan has a much lower probability of becoming disconnected than Sinet (0.352 and 0.673, respectively), its probability of incurring a large ATTR impact is much higher than for Sinet. $P(ATTR \leq 0.7)$ is 0.224 for JGN2plus-Japan and 0.049 for Sinet.

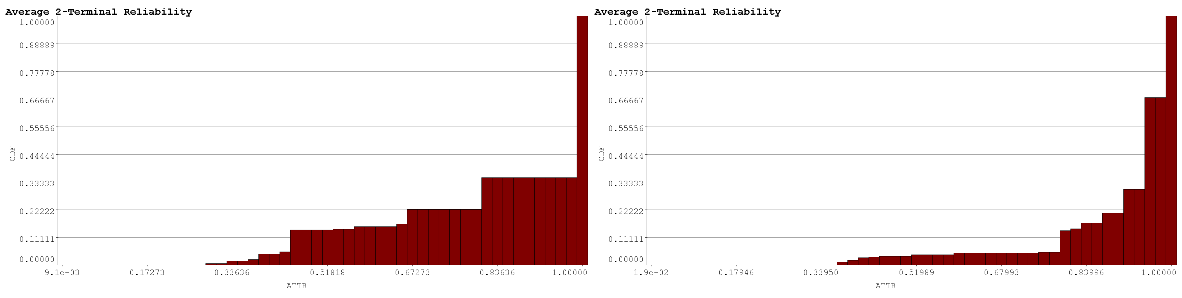
This is probably caused by the large difference in network size between both networks. As JGN2plus-Japan consists of fewer nodes and links, it has a higher probability that it will not be hit by the earthquake at all. However, in the case that the network does get hit, it lacks the backup paths to keep most of its connections. We can confirm this by inspecting P(No Link Failures). Indeed, the probability of all links of JGN2plus-Japan being unaffected is 0.648, and there are no possible states in which any link fails, but the network stays connected. The comparatively low P(No Link Failures) of Sinet is 0.263.

The worst-case disasters for JGN2plus-Japan all occur around Tokyo, resulting in an ATTR of 0.291 with probability 0.007. The worst-case disasters for Sinet are located around Osaka and Kyoto, and result in an ATTR of 0.362 with probability 0.009. Instances of these disasters are depicted in figure 7.3.



(a) JGN2plus-Japan (b) Sinet

Figure 7.1: Japanese network topologies.



(a) JGN2plus-Japan (b) Sinet

Figure 7.2: ATTR distributions.

JGN2plus-Japan has an expected ATTR value of 0.866 with a variance of 0.044 and Sinet an expected ATTR value of 0.920 with a variance of 0.016.

Based on this analysis, the network operator of JGN2plus-Japan might want to invest in backup links, to improve the size of its connected components after being hit by an earthquake.

As adding or protecting cables can be quite expensive, the operators of Sinet would have to consider the benefit of further protecting their network against the cost of doing so. While one can never fully prevent the network from disconnecting (as nodes directly in the disaster area always become disconnected), there is clearly still room for improvement on Sinet. For example, the impact of the worst-case earthquakes could be alleviated by connecting fewer nodes (only) to Osaka, and if possible by protecting the link between Sendai and the solitary node in Kyushu.

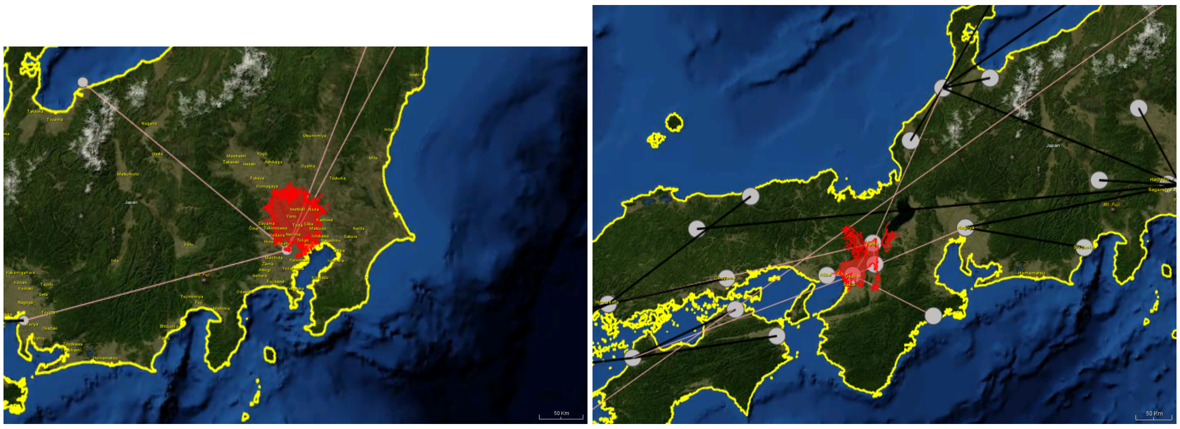
For both networks, only computing the ATTR for each possible failure state, instead of for each disaster, had a large effect on performance, reducing the number of times ATTR had to be computed from 655 to 22 and 93 for JGN2plus-Japan and Sinet respectively.

Weighted Average 2-Terminal Reliability

As Tokyo is the capital of Japan, and its largest prefecture, disconnecting nodes in this area should be considered to be worse than disconnecting other nodes. To model this when computing the vulnerability of Sinet, we give nodes “Tokyo DC1”, “Tokyo DC2”, “Tokyo DC3” and “U Tokyo” a weight of 2. These nodes are all located on exactly the same position: (35.61488 N, 139.5813 E). All other nodes are given a weight of 1.

The results can be seen in figure 7.4. By taking the weighted ATTR, Sinet’s vulnerability is considered to be (slightly) lower than by just taking the unweighted ATTR as before. We can infer that the Tokyo nodes are disconnected less frequently than other nodes in the network.

Although the probability of at least one node disconnecting from the network is (obviously) still the same, the impact of these disconnections is considered to be lower than before. The expected WATTR value is 0.924 (with a variance of 0.015).



(a) JGN2plus-Japan (b) Sinet

Figure 7.3: Worst case earthquake instances. The disaster area is painted red. Affected links are colored pink.

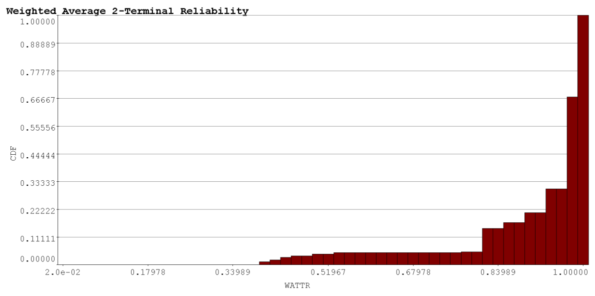


Figure 7.4: WATTR distribution of Sinet.

Interestingly, the worst case disasters are still exactly the same as before. Only the effect itself has changed, from an ATTR of 0.362 to a WATTR of 0.388.

7.1.2. American Networks

In this section we discuss the earthquake vulnerability of 2 network topologies located in the United States: IBM (figure 7.5a) and ATT North America (figure 7.5b). The IBM topology consists of 18 nodes and 24 links. ATT North America is bigger, containing 25 nodes connected by 57 links.

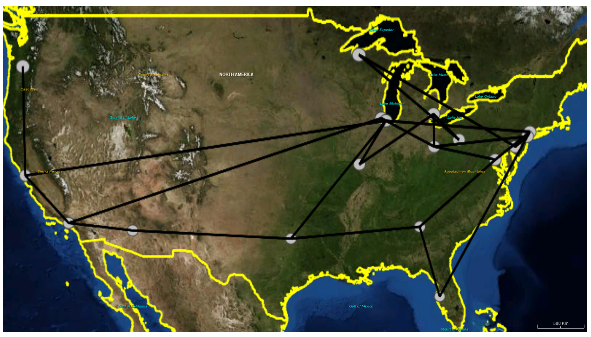
We consider two separate disaster sets: the “Building Seismic Safety Council 2014 Event Set” of 796 characteristic earthquake scenarios and a historic set of 2844 earthquakes. Both of these are described in more detail in section 5.2.1. In all cases the intensity (in terms of Modified Mercalli Intensity) threshold was set to 7.

Scenarios

As can be seen in figure 7.6, not many node pairs get disconnected after these earthquakes. The expected ATTR value of IBM is 0.986 (with a variance of 0.002) and the expected ATTR value of ATT North America is 0.992 (with a variance of 0.001). The probability of one or more node pairs disconnecting is 0.104 for IBM and 0.095 for ATT North America. Even the worst case scenarios still result in relatively high ATTR values: 0.784 (with probability 0.023) for IBM and 0.843 (with probability 0.004) for ATT North America.

The low vulnerability of these topologies to these scenarios can be attributed to two factors. Firstly, and most importantly, the United States is significantly larger than for example Japan. Thus the links of networks spanning the contiguous US are placed far apart from each other. If an earthquake hits the network, only a relatively small amount of links fail. In many cases where one path between two nodes fails, they remain connected through some links comfortably far away from the disaster area.

Secondly, as mentioned in section 5.2.1, this scenario set contains a high amount of earthquakes in the western part of the U.S. compared to other areas. However, the nodes of both the IBM and the ATT North America topology are mostly concentrated in the central and eastern U.S.


(a) IBM

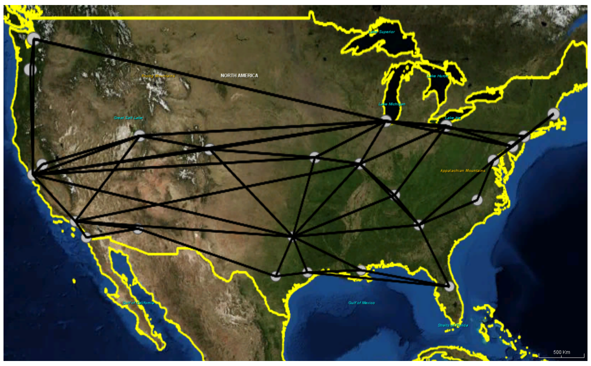
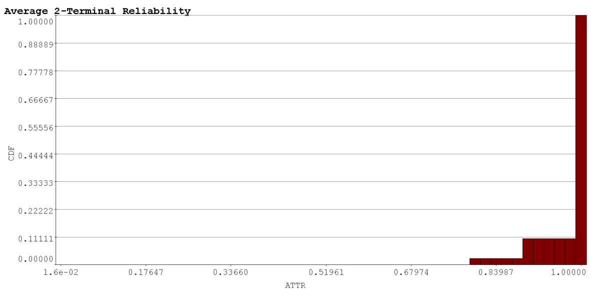
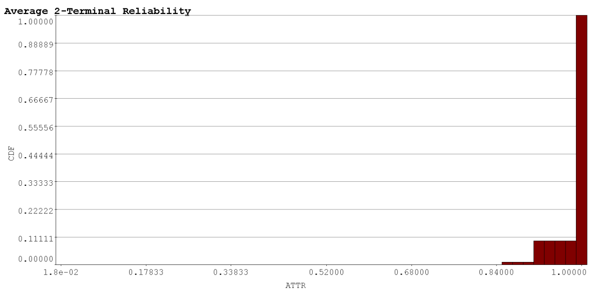

(b) ATT North America

Figure 7.5: American network topologies.



(a) IBM



(b) ATT North America

Figure 7.6: ATTR distributions based on earthquake scenarios.

As with the Japanese networks, only computing the ATTR for each possible failure state, instead of for each disaster, had a large effect on performance, reducing the number of times ATTR had to be computed from 796 to 14 and 75 for IBM and ATT North America respectively.

To get a better idea of the impact of earthquakes on these topologies one should consider using a different measure not related to (dis)connectivity. For example lost link capacity or the average maximum flow between node pairs. We recompute the vulnerability in terms of a very simple measure: the amount of surviving links divided by the total amount of links (i.e. the Link Survival Ratio (LSR)).

Figure 7.7 shows the distribution of the LSR measure of IBM and ATT North America based on the earthquake scenarios. As noted before, not many links get hit at the same time by these quakes.

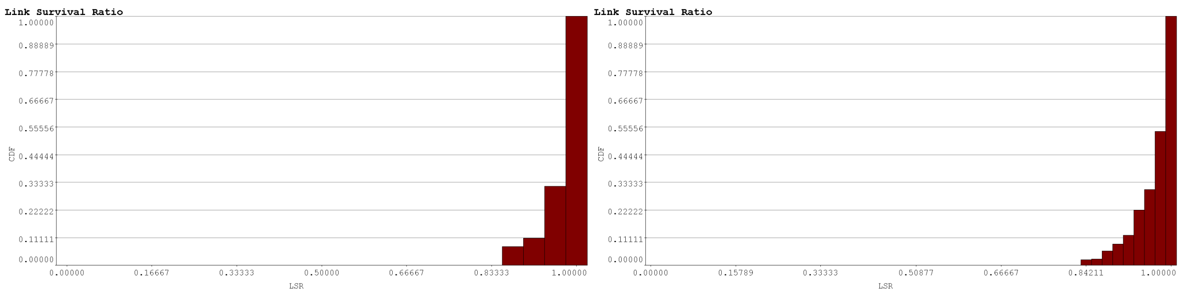
In the worst cases 3 (12.500%) of IBM's links (with probability 0.075), or 9 (15.789%) of ATT's links (with probability 0.023), fail simultaneously. The earthquakes resulting in these failures all occur around San Francisco (for both topologies) or Los Angeles (in IBM's case only). Examples of worst case scenarios are shown in figure 7.8.

Historic Dataset

Our historic dataset does not contain many earthquakes with large impacts, as it only spans a (in geological terms) very small amount of time. Thus the amount of node pairs getting disconnected by even the worst case earthquakes is very small. In addition, the set contains many earthquakes with completely empty disaster areas. These disasters would not get included in a scenario set, but are included in the historic set, significantly inflating the resulting $P(\text{No Link Failures})$.

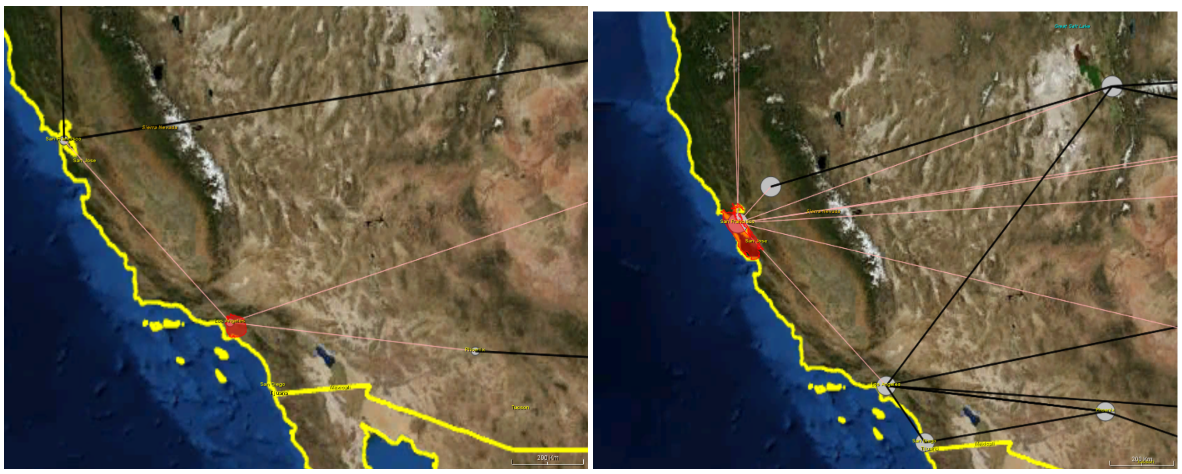
The worst case earthquakes in our historic set only disconnect one ATT node or 2 IBM nodes. It makes sense to compute the vulnerability of these networks in terms of link survival ratio instead of ATTR again. Additionally, we filter out all disasters with an empty disaster area, resulting in a disaster set with only 99 historic earthquakes.

The LSR distributions based on this smaller disaster set are shown in figure 7.9. Just as with the scenarios dataset, links only tend to fail in very small numbers. In addition, even after filtering out all no disaster area earthquakes, $P(\text{No Link Failures})$ is still very high for both topologies: 0.869 for IBM and 0.808 for ATT North



(a) IBM (b) ATT North America

Figure 7.7: Link Survival Ratio distributions based on earthquake scenarios.



(a) IBM (b) ATT North America

Figure 7.8: Worst case earthquake scenario instances (with respect to the amount of links downed). The disaster area is painted red. Affected links are colored pink.

America.

Interestingly, the worst case historic earthquakes are located in the same areas as the worst case scenarios were (San Francisco and Los Angeles), and result in the same amount of downed links (3 IBM links or 9 ATT links). These two worst case earthquakes are depicted in figure 7.10.

The set of 99 earthquakes only results in 7 distinct IBM failures states and 10 distinct ATT North America failure states.

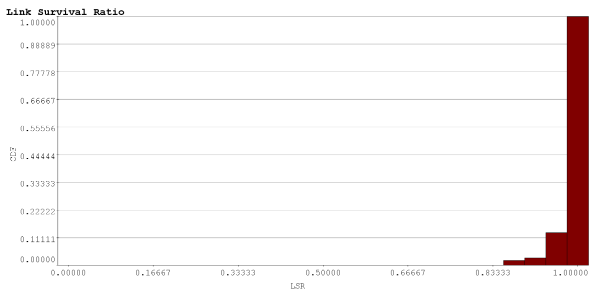
7.1.3. Comparison to Randomly Placed Circular Disasters

In many previous works, instead of computing the vulnerability of a network topology to disasters based on empirical disaster location and area data, authors compute this vulnerability with respect to a circular or line (segment) cut disaster randomly (almost always with uniform probability) placed in the network area (e.g. [50, 51, 64, 65]) or compute the worst-case position for such disasters (e.g. [21, 22, 36, 52, 64]).

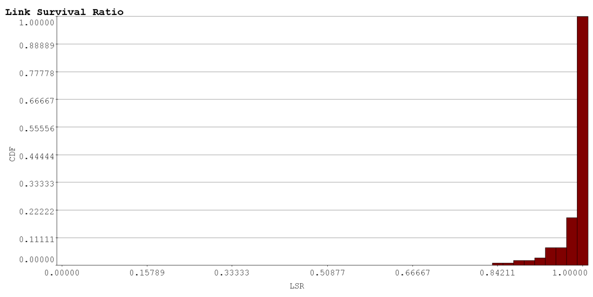
In this section we briefly compare the results of this approach to ours. We assume the network area is rectangular. This assumption is often made in the execution of the before-mentioned works, as this makes assigning the network area and computing the vulnerability much more practical. To model a random failure model, we take a Monte Carlo approach and uniformly sample 1,000,000 locations from the network area as centers of circular disasters for the disaster set. The worst-case disaster (location) can then be found directly as a result of our calculations, as with the empirical datasets.

Japanese Networks

As our Japanese network area we take the area positioned between (25.8N, 126.6 E) and (44.4N, 145.9E). This area covers all parts of Japan spanned by the JGN2plus-Japan and Sinet topologies.

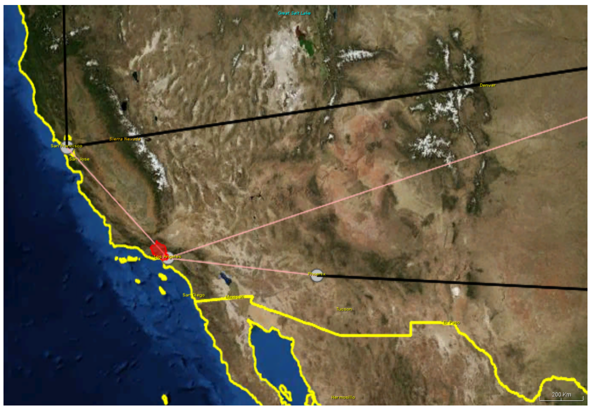


(a) IBM

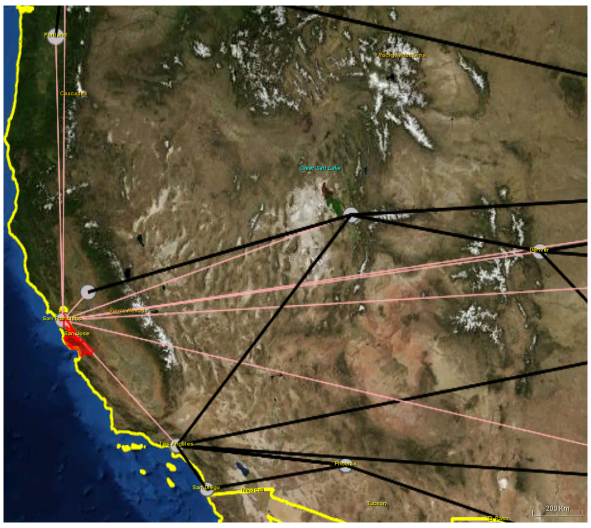


(b) ATT North America

Figure 7.9: Link Survival Ratio distributions based on historic earthquakes.



(a) IBM



(b) ATT North America

Figure 7.10: Worst case historic earthquakes (with respect to the amount of links downed). The disaster area is painted red. Affected links are colored pink.

We base the radius of the circular disaster areas on the size of the disaster areas of the earthquake scenarios. The weighted (by probability) sum of the number of grid squares of all J-SHIS scenario disaster areas is 16220.972. As each square is around 250m by 250m in size, the 'average' scenario impacts an area of around $1,013,810,724 m^2$. Thus we assume the circular disasters have a radius of $17,964$ meters, which is approximately $\sqrt{\frac{1,013,810,724}{\pi}}$ meters.

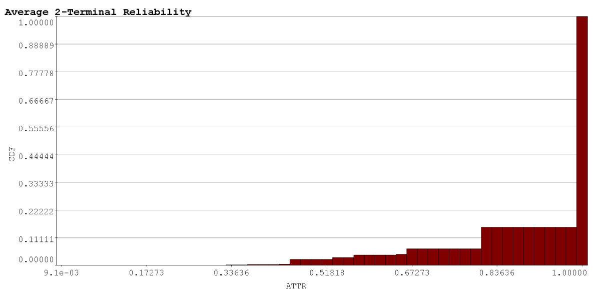
The random circle approach results in a severe underestimation of the vulnerability of both networks. This is mostly caused by the shape of the network area. Japan does not nicely fit within a rectangular area, so a large part of the network area does not contain Japan itself, but the sea or other countries, and many disasters spawned here do not affect the network at all.

To prevent the shape of the network area to have such a large effect on our comparison, we only sample disasters from the Japanese islands in the network area. Essentially, we are changing our network area to these islands. However, note that this is typically not possible when computing the vulnerability of a network to randomly placed circular disasters by some other approach than random sampling, as these approaches require the network area to be convex, or even rectangular.

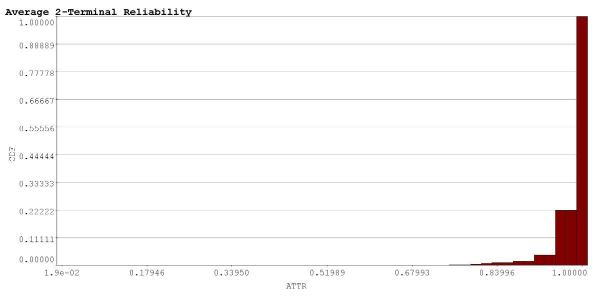
We have plotted the cumulative distribution functions of the ATTR of JGN2plus-Japan and Sinet after a random circular disaster in figure 7.11. Even with the smaller network area, the vulnerability of the networks is clearly underestimated.

The worst-case results are much more accurate. The worst case circular disasters for JGN2plus-Japan are all positioned near Tokyo (as in figure 7.12a), as is the case for the worst case J-SHIS scenarios. They also result in exactly the same ATTR value (0.291), albeit with a lower probability (0.001).

The worst cases circles for Sinet are positioned around Kyoto, centered slightly higher than the worst case

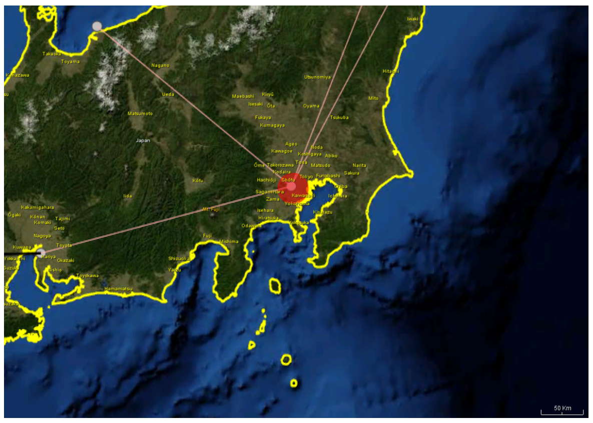


(a) JGN2plus-Japan

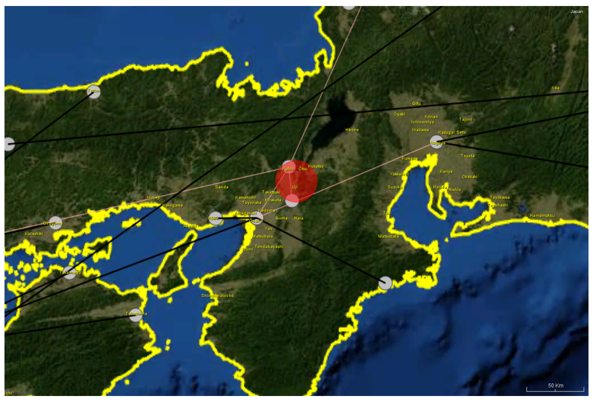


(b) Sinet

Figure 7.11: ATTR distributions after a random circular disaster.



(a) JGN2plus-Japan



(b) Sinet

Figure 7.12: Worst case circular disasters. The disaster area is painted red. Affected links are colored pink.

earthquake scenarios (as in figure 7.12b). They result in an ATTR value of 0.442 (with probability $1.39 * 10^{-4}$), which is higher than the worst case ATTR based on the earthquake scenarios. This is caused by the circle not being able to cover as much links in the area as the scenario disaster areas do.

This in turn can be attributed to two factors. The first being the shape of the circles themselves. Circles are not as 'efficient' in covering much links as some of the scenario disaster areas are. Secondly, the scenario disaster areas differ in size, while the circular disaster areas all have the same fixed size. Thus some scenarios have a larger disaster area than the circular disasters, and thus can affect more links. This is especially noticeable for the worst case disasters, as these often have larger areas.

American Networks

As American network area we take the same area we filtered our historical earthquake locations on: the area positioned between (24.6N, 125W) and (50N, 65W). Thus the shape of the network area should not negatively affect the results.

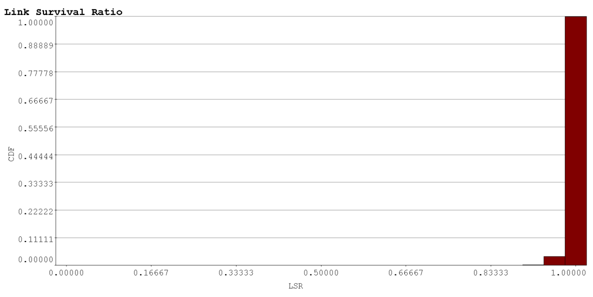
It is a bit more difficult to compute the average area of these historical earthquakes, as their areas are not split into cells with a fixed size. However, the average area of the historical disasters after transforming them to the 2D plane (and filtering out empty disaster areas) is around $769{,}075{,}238m^2$. Thus we set the radius of the circular disasters to $15{,}646$ meters, approximately $\sqrt{\frac{769{,}075{,}238}{\pi}}$ meters.

We computed the resulting LSR distribution for both the IBM and ATT North America topology. The results can be found in figure 7.13.

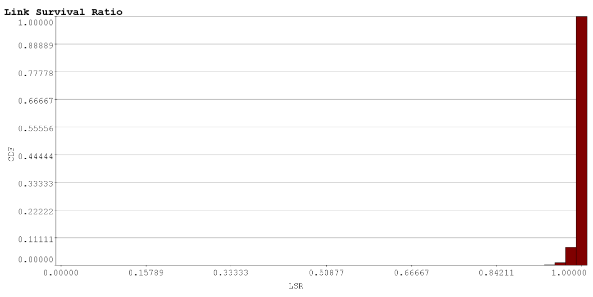
Again, the survivability based on these computations is higher than the survivability as computed based on the historical set of earthquakes. The probability of no link failures are 0.965 and 0.928 for IBM and ATT North America respectively.

Interestingly, the worst case circles for IBM are located in New York and Chicago, instead of along the west coast. A circular disaster placed here downs 5 links (e.g. as in figure 7.14a).

The worst case disaster location for ATT North America has moved to Dallas. These disasters result in 10

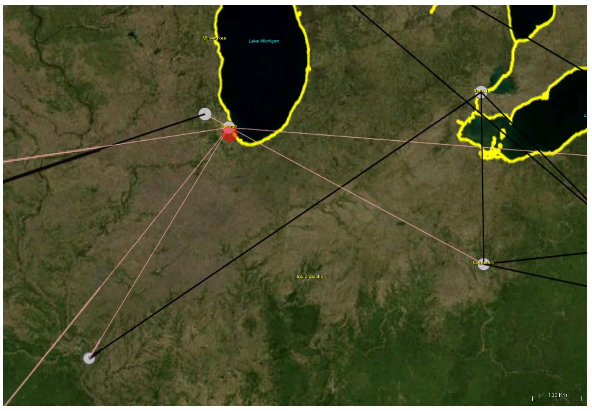


(a) IBM

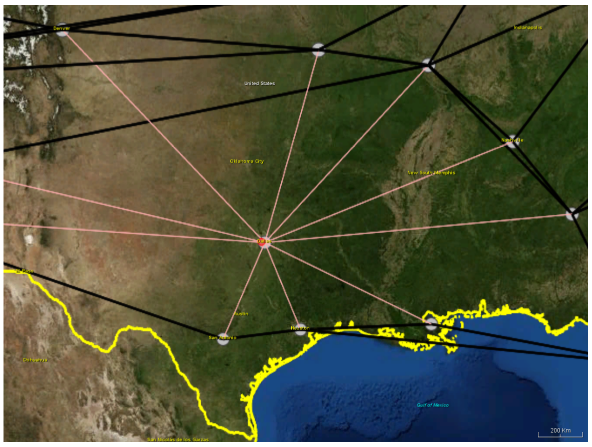


(b) ATT North America

Figure 7.13: LSR distributions after a random circular disaster.



(a) IBM



(b) ATT North America

Figure 7.14: Worst case circular disasters. The disaster area is painted red. Affected links are colored pink.

downed links (see figure 7.14b).

Our own approach did not find these locations. It is difficult to say if this is because of a lack of disasters in our historical set, or if the possibility of large earthquakes hitting these locations is around 0.

Note that due to the large space between links, and small disaster areas (compared to the size of the network area), the worst case disasters for the two American topologies are in essence node failures, as these are the positions where many links come together. This means that, assuming all nodes can actually be hit by earthquakes, the (small) circle approach can be quite effective in finding the worst-case positions (nodes). Although in this case just iterating over all possible node failures would be much more efficient.

7.2. Hurricane Katrina

In this section we apply our simple hurricane model from section 5.2.2 to two NHC Hurricane Katrina forecasts: the “5 AM EDT THU AUG 25 2005” forecast and the “5 AM EDT SUN AUG 28 2005” forecast. The track positions in these forecasts are shown in table 7.1. This table also includes the post-analysis best track. The best track positions from before “28 Aug, 06:00” included in the table were added to the beginning of the august 28 forecast tracks. We do not add the positions where Katrina was expected to become extratropical to this track. For each forecast, we generate 100,000 hurricane tracks as described in section 5.2.2 using the radii of the 2017 cone forecast circles [49].

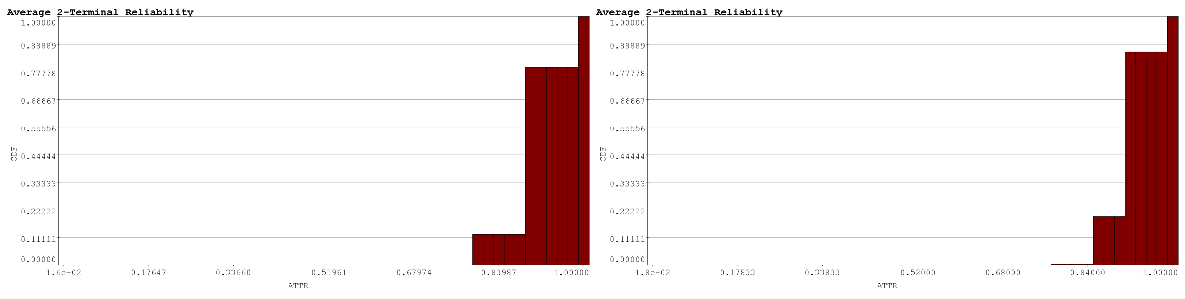
We compute the vulnerability of the same 2 American topologies (IBM and ATT North America) based on these tracks.

7.2.1. August 25 Forecast

The (ATTR) vulnerability distributions based on the 5 AM august 25 forecast can be found in figure 7.15. One can immediately see that there is a very high probability that node pairs will become disconnected, but a very low probability of more than 25% of node pairs disconnecting. Interestingly, this is very much the opposite

Table 7.1: Hurricane Katrina Position Forecasts

Time (UTC+0)	**5 AM EDT THU AUG 25**	**5 AM EDT SUN AUG 28**	**Best Track**
25 Aug, 06:00	-	-	26.1N 78.4W
25 Aug, 09:00	26.2N 78.7W	-	-
25 Aug, 18:00	26.2N 79.4W	-	26.2N 79.6W
26 Aug, 06:00	26.3N 80.4W	-	25.4N 81.3W
26 Aug, 18:00	26.3N 81.4W	-	24.9N 82.6W
27 Aug, 06:00	26.5N 82.5W	-	24.4N 84.0W
28 Aug, 06:00	27.5N 84.0W	-	25.2N 86.7W
28 Aug, 09:00	-	25.4N 87.4W	-
28 Aug, 18:00	-	26.3N 88.4W	26.3N 88.6W
29 Aug, 06:00	29.5N 85.0W	28.0N 89.4W	28.2N 89.6W
29 Aug, 18:00	-	30.0N 89.8W	31.1N 89.6W
30 Aug, 06:00	33.0N 83.0W	32.3N 89.3W	34.1N 88.6W
31 Aug, 06:00	-	37.5N 86.0W	40.1N 82.9W
01 Sept, 06:00	-	42.0N 79.0W (Extratropical)	- (merged with front)
02 Sept, 06:00	-	47.0N 70.0W (Extratropical)	- (merged with front)



(a) IBM (b) ATT North America

Figure 7.15: ATTR distributions based on the 5 AM August 25 Hurricane Katrina forecast.

situation as the vulnerability of the Japanese topologies to earthquakes, where there is a lower probability of one or more nodes disconnecting, but a higher probability that many pairs will be disconnected. Of course, the situation itself is quite different. In this case we are preparing for a specific event we know is coming within a few days, while in the Japanese case the vulnerability is computed to a broad group of possible earthquakes.

The probability that at least one node pair will disconnect is 0.797 for IBM and 0.858 for ATT North America. The probabilities that no link will be downed at all are only 0.025 and $5.50 * 10^{-4}$ for IBM and ATT North America respectively. Examples of tracks which would not result in any damage to the network are given in figure 7.16. Note that if the forecast would give predictions for latter dates, these tracks would probably extend to impact the network as well. As it is, the lack of future data makes us underestimate the vulnerability of these networks to Hurricane Katrina.

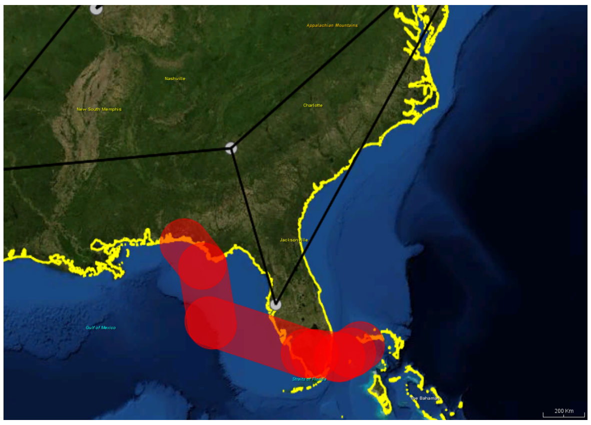
The expected ATTR is 0.899 for IBM (with a variance of 0.004) and 0.916 for ATT North America (with a variance of 0.002).

A single track is both the worst case track for IBM and ATT North America. This track goes all the way to Detroit, and is shown in figure 7.17. This path would result in an ATTR of 0.248 for IBM and 0.503 for ATT North America. This specific failure state only has a $1.00 * 10^{-5}$ probability of occurring according to this computation, but there are a few similar outliers crossing a large part of the U.S.

Just as when computing the vulnerability against earthquakes, computing the failure state distribution before computing the vulnerability distribution greatly speeds up computations. The set of 100,000 disasters is reduced to 36 IBM and 74 ATT failure states.

7.2.2. August 28 Forecast

Figure 7.18 shows the vulnerability distributions based on the 5 AM 28 august forecast. The vulnerability of IBM to Katrina is adjusted to be much lower than based on the older forecast, and the vulnerability of ATT


(a) IBM

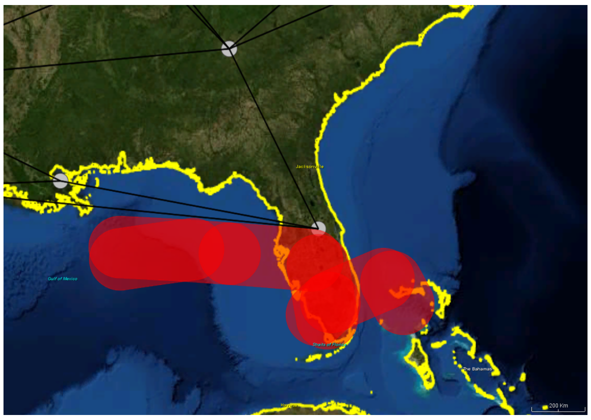

(b) ATT North America

Figure 7.16: Examples of tracks that do not impact the network, based on the 5 AM August 25 Hurricane Katrina forecast. The disaster area is painted red.

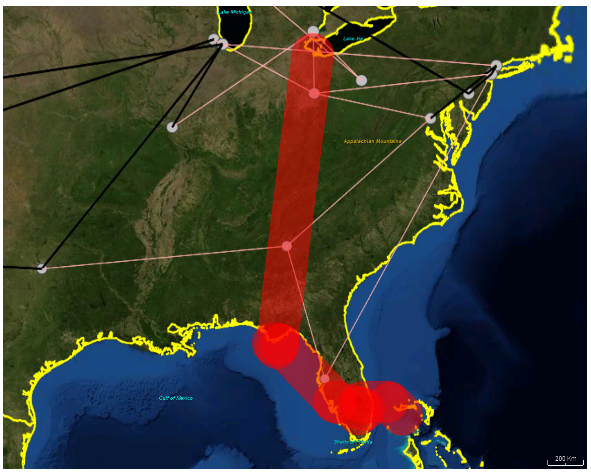


Figure 7.17: Worst case track for IBM and ATT North America based on the 5 AM August 25 Hurricane Katrina forecast. The disaster area is painted red. Affected (IBM) links are colored pink.

North America has been slightly increased. For both topologies every track realization we have generated affects at least one link.

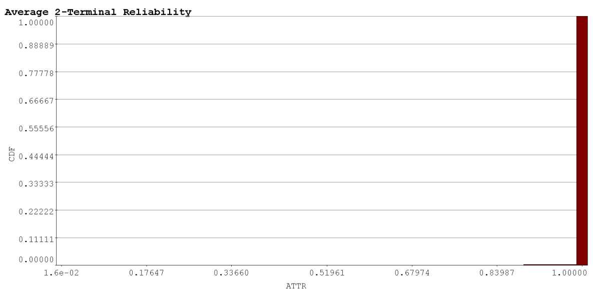
$P(\text{ATTR} < 1)$ is only 0.004 for IBM. In 98.5% of realizations only one IBM link is downed (see figure 7.19). The probability that at least one ATT pair will disconnect is 0.955.

The expected ATTR values are 1.000 (with variance $7.06 * 10^{-5}$) for IBM and 0.876 (with variance 0.002) for ATT North America.

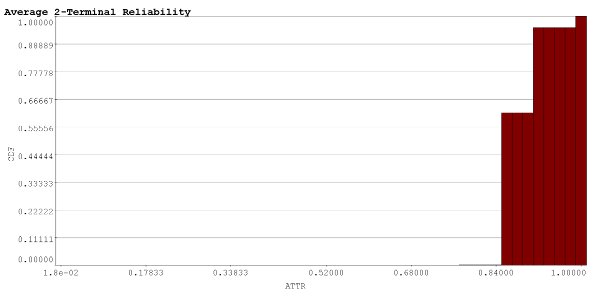
Again, the worst case track is the same for both topologies. It is depicted in figure 7.20 on the ATT North America topology and results in an ATTR value of 0.359 for IBM and 0.423 for ATT North America.

This analysis is a good example of why only computing the expected value, or only computing the worst case disaster gives an incomplete picture of the vulnerability of a network. Based on the expected value one would expect the IBM network to be almost completely secure against Katrina, but based on the worst case disaster one would conclude that the IBM topology is extremely vulnerable. After considering the distribution itself, one can see that in most cases no node pairs will be disconnected at all, but there is a very slight possibility that the hurricane will take a different path and disconnect a large part of the network.

As with the previous forecast, the number of possible failure states is significantly lower than the number of disasters: 14 and 58 for IBM and ATT North America respectively.



(a) IBM



(b) ATT North America

Figure 7.18: ATTR distributions based on the 5 AM August 28 Hurricane Katrina forecast.

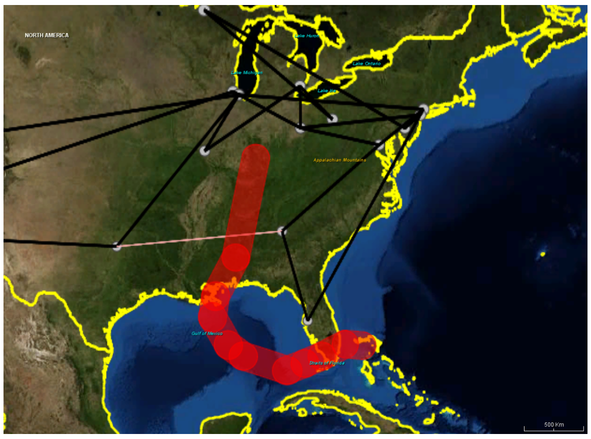


Figure 7.19: 98.5% of track realizations based on the 5 AM August 28 Hurricane Katrina forecast only damages this single IBM link. The disaster area is painted red. The affected link is colored pink.

7.3. Runtime

In this section we analyze the runtime of our methods. All experiments were executed on an (4-core) Intel i5-4670K CPU.

Tables 7.2 and 7.3 show the average runtime, after 10 runs, when computing the ATTR vulnerability distribution based on 3 of our American datasets on 1, 2 and 4 threads. The runtimes have been split up into three categories: computation time of the distribution over failure states (see section 4.1), computation time of the vulnerability distribution (after having computed the failure state distribution, see section 4.2), and total computation time.

Even on the largest dataset, containing 100,000 disasters, computing the vulnerability distributions takes less than a second when utilizing all 4 threads. The most time-consuming steps of our methods can efficiently be executed in parallel. Doubling the amount of threads in use almost doubles the total computation speed.

As mentioned before, computing the failure state distribution takes up much more time than computing the vulnerability distribution afterwards. As states are computed for each disaster, and the ATTR is only computed once for each possible state.

To examine the computation time when the number of disasters grows, we focus on a ‘smaller’ network: SURFnet (see figure 7.21). As this topology contains many nodes and links (50 nodes and 73 links) packed very closely together, we expect the amount of possible failure states to be much higher than that of the American, or even Japanese, topologies. Thus our methods should be less efficient on SURFnet than on most other topologies.

For each run, we generate a disaster set by randomly placing 50 km radius circle disasters between the lower-left node and the upper-right node. We then compute the vulnerability distribution based on this disaster set on a single thread.

The average runtime of our methods compared to the disaster set size is shown in figure 7.22. As expected, the computation times of both the failure state and vulnerability distribution seem to increase linearly with the number of disasters. Even on a set with 1,000,000 disasters the average total computation time is still less

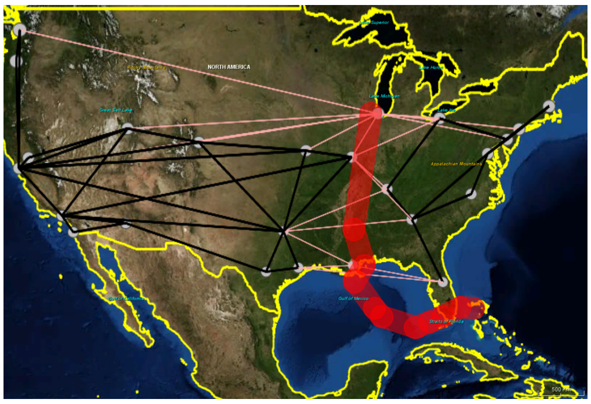


Figure 7.20: Worst case track for IBM and ATT North America based on the 5 AM August 28 Hurricane Katrina forecast. The disaster area is painted red. Affected (ATT) links are colored pink.

Table 7.2: Average Runtimes on IBM (18 nodes, 24 links) topology.

	Failure State Distribution	**Vulnerability Distribution**	**Total**
Earthquake Scenarios			
1-threaded	48.43 ms	0.46 ms	48.89 ms
2-threaded	22.64 ms	0.12 ms	22.76 ms
4-threaded	12.11 ms	0.10 ms	12.21 ms
Historic Earthquake Dataset			
1-threaded	0.8393014 ms	0.0888452 ms	0.93 ms
2-threaded	0.43 ms	0.05 ms	0.47 ms
4-threaded	0.29 ms	0.04 ms	0.33 ms
Katrina (August 25)			
1-threaded	1257.32 ms	10.42 ms	1267.74 ms
2-threaded	633.95 ms	6.32 ms	640.27 ms
4-threaded	325.81 ms	5.29 ms	331.11 ms

than 3.5 seconds.

We have plotted the average number of possible failure states in figure 7.23. The number of possible failures states rapidly increases for low number of disasters, but only slightly increases for higher number of disasters. Based on this graph one might expect the computation time of the vulnerability distribution to be sub-linear, as it should only depend on the number of possible states. However, as mentioned in section 4.1.3, the values of $P(S = s)$ are computed on the fly during the vulnerability distribution computation, which requires $\mathcal{O}(|D|)$ time.

The amount of line segments per link in our topologies is 1. To accurately model each cable this number should be much higher. The number of link segments has a large impact on the failure state computation time.

To model an increase in the number of line segments we fix the disaster set size to 50,000 and increase the number of line segments by repeatedly dividing each line segment into two segments (thus doubling the amount of line segments). The resulting failure states distribution computation times are plotted in figure 7.24. We did not plot the computation times of the vulnerability distribution, as those do not depend on the number of line segments.

The computation time seems to increase linearly with the number of line segments. At 37,376 segments the average failure state distribution computation time is $18,595.36$ ms. Although this is significantly longer than the time it takes to compute the distributions for networks with less line segments, the computation still finishes in a reasonable time.

Table 7.3: Average Runtimes on ATT North America (25 nodes, 57 links) topology.

	Failure State Distribution	**Vulnerability Distribution**	**Total**
Earthquake Scenarios			
1-threaded	101.67 ms	1.16 ms	102.83 ms
2-threaded	52.01 ms	0.35 ms	52.36 ms
4-threaded	27.42 ms	0.22 ms	27.64 ms
Historic Earthquake Dataset			
1-threaded	1.74 ms	0.14 ms	1.87 ms
2-threaded	0.88 ms	0.06 ms	0.95 ms
4-threaded	0.57 ms	0.04 ms	0.61 ms
Katrina (August 25)			
1-threaded	2818.24 ms	11.55 ms	2829.78 ms
2-threaded	1416.71 ms	6.38 ms	1423.10 ms
4-threaded	725.30 ms	4.95 ms	730.25 ms

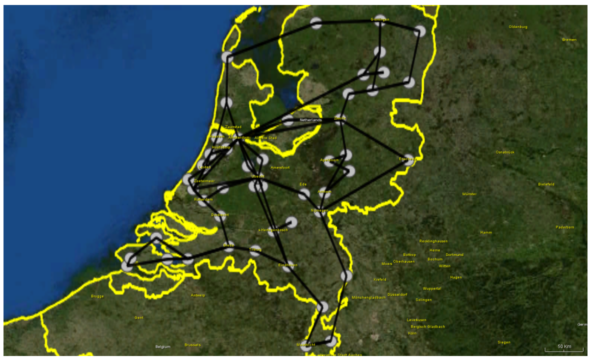
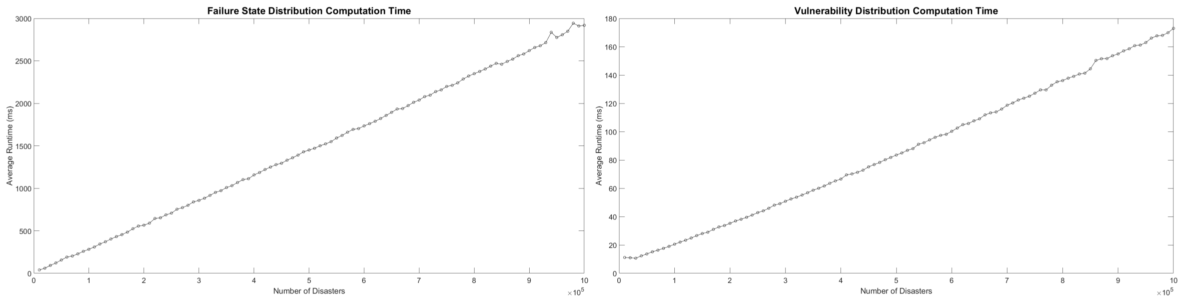


Figure 7.21: SURFnet topology



(a) Failure state distribution computation time (b) Vulnerability distribution computation time

Figure 7.22: Average computation times on SURFnet with randomly sampled circle disaster sets.

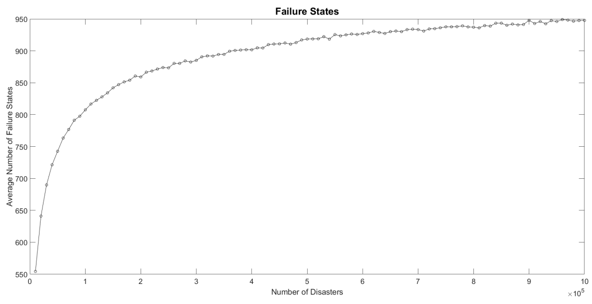


Figure 7.23: Average number of possible failure states of SURFnet with randomly sampled circle disaster sets.

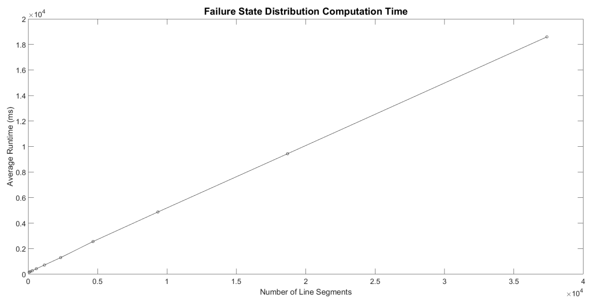


Figure 7.24: Average failure state distribution computation time on SURFnet with randomly sampled circle disaster sets.

8 Analysis of SURFnet Problem and Incident Tickets

SURFnet operates a network connecting Dutch research and education institutions to each other and to other networks. Currently, they are using an issue tracking system called JIRA to keep track of problems and incidents.

At the moment, SURFnet only uses their (problem and incident) tickets to keep track of, and keep their employees in the loop of, current problems and incidents. They are interested in the possibility of analyzing past tickets to gain insight in incident trends and properties.

To this end, we have been given access to their archive of JIRA issues. This chapter contains a preliminary analysis of this issue set. The purpose of this analysis was to (1) discover any relevant insights this data can give us and (2) to research how SURFnet's current system can be improved to become more usable for future more exhaustive analysis.

In the context of our geographical research, we were interested in obtaining a set of characteristic geographically correlated failures based on past events. These could then be used to compute the vulnerability of the network, and to predict possible future events. For example, by analyzing past failures during maintenance, it might be possible to predict the impact of other future maintenance operations. Unfortunately, after studying the dataset, we do not believe the tickets contain enough accurate information to make this possible. It might be better to study hardware logs for this purpose, possibly in combination with the tickets dataset.

The main use of the ticket system is to keep track of, and keep employees up to date about, current issues and problems with the network. It is not setup to be a monitoring or archiving tool, or to be machine-readable.

Section 8.1 contains the results of our analysis, and in section 8.2 we give our recommendations based on this analysis.

8.1. Analysis

Since 2014 SURFnet has used JIRA for issue tracking. Tickets created in their old system were imported as "ARS-S Tickets", while new tickets are called "SURFnet Trouble Tickets". We are not interested in all issues, only in tickets. As such, we limit our analysis to these "SURFnet Trouble Tickets" and "ARS-S Tickets". [1]

SURFnet has provided us with their complete dataset of issues spanning from January 2010 to mid-March 2017. We further limited our dataset to only include those tickets which were already closed, and were not declined, canceled, or duplicates. The resulting tickets were filtered on ticket type (keeping only incidents and problems) where possible (not all imported tickets have a ticket type).

The final dataset consists of 3725 "SURFnet Trouble Tickets" and 6209 "ARS-S Tickets", for a total of 9934 tickets.

[1]Some ARS-S tickets are incorrectly labeled as "SURFnet Trouble Tickets". In our analysis, we treat these as "ARS-S Tickets", not as "SURFnet Trouble Tickets".

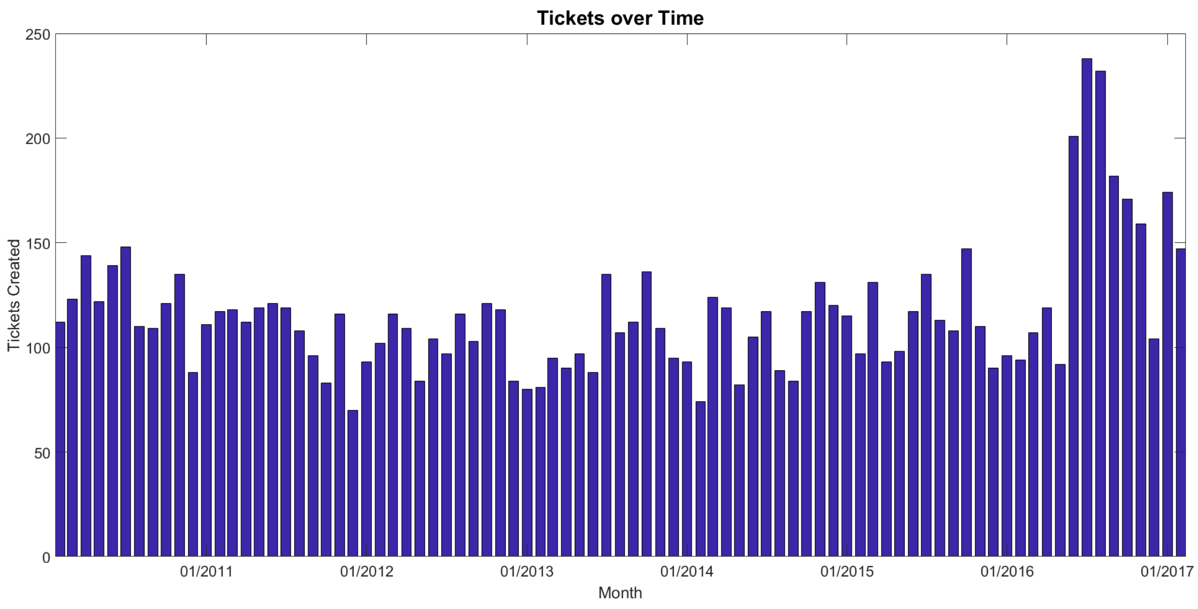


Figure 8.1: Tickets created per month.

8.1.1. Analysis of Ticket Creation Times

At the suggestion of SURFnet itself, we start our analysis by studying the creation dates and times of tickets. In this section we consider the amount of created tickets over time, number of tickets created in each month, and number of tickets created in each hour of the day.

Tickets over Time

We have grouped all tickets by creation month. For the purpose of this analysis, the first and last month of our dataset were excluded.

In figures 8.1 and 8.2 one can see the amount of tickets created per month. One can clearly see that the amount of tickets created each month was relatively stable until June 2016. From this month onwards there is a sharp increase in the amount of tickets created. This coincides with, and is probably indirectly caused by, the change in operational network management at June 15th 2016 [18].

Unfortunately, it is difficult to evaluate if the increase is caused by an increase of problems in the network, or if the new network management just makes more use of the ticket system. The amount of tickets does not correspond directly with the amount of problems in the network, or even the amount of issues raised by customers. For example, even after filtering on ticket type, the dataset contains numerous test tickets, tickets about internal issues (e.g. with JIRA itself), and requests (e.g. to test if an e-mail address change has been properly processed)

Splitting tickets by impact should give at least some insight in what kind of tickets have seen an increase since June 2016. In figure 8.3 tickets creation amounts are plotted separately for each impact category [2].

Typically, impact categories range from P3 (small or no impact) to P1 (large impact). However, issues with an international impact and major outages get assigned to “P1 Internationaal” and “Grote Storing” respectively.

While all impact categories saw an increase in tickets around June 2016, the largest increase by far was that of the P3 tickets, the lowest impact a ticket can be assigned. Furthermore, since 2010 P2 have been slowly decreasing in frequency. This could imply SURFnet has been improving robustness somewhat, but it could also mean that tickets get assigned to other categories more often instead.

Interestingly, P3 tickets have seen an increase in frequency since well before June 2016, this increase has been mostly compensated by a decrease in other impact categories.

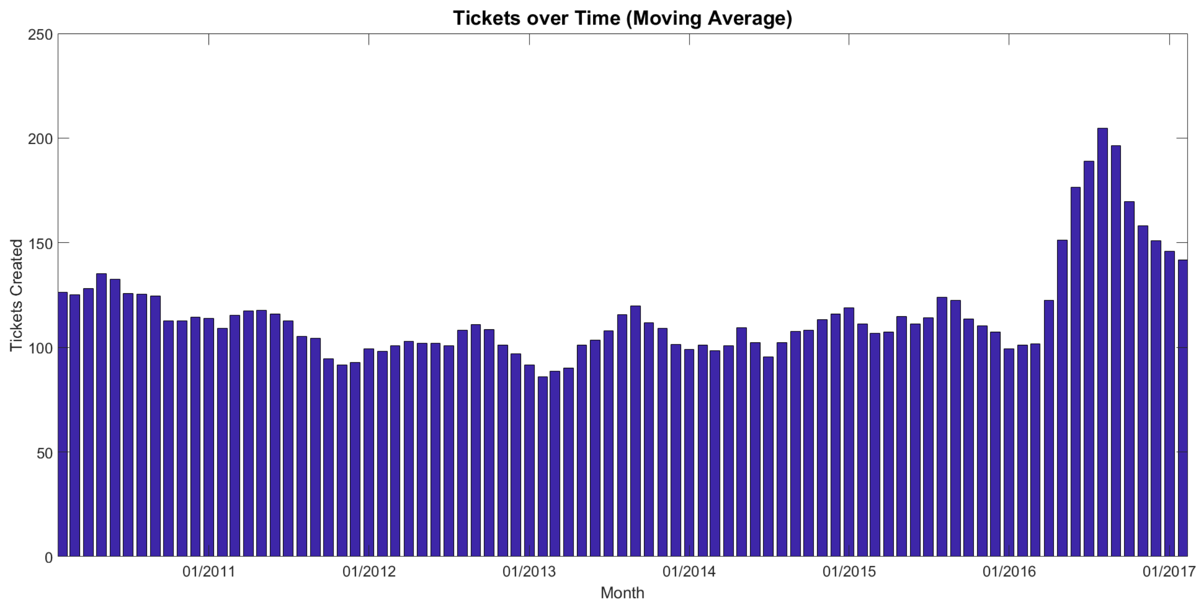


Figure 8.2: Central moving average (with a window size of 5 months) of tickets created per month.

Table 8.1: Percentage of tickets created in each month

Month	2010-2016	2010-2015	2016
January	7.25%	7.68%	5.35%
February	7.06%	7.45%	5.30%
March	8.48%	9.04%	5.98%
April	8.18%	8.53%	6.61%
May	7.26%	7.72%	5.24%
June	9.09%	8.62%	11.16%
July	10.30%	9.59%	13.44%
August	9.11%	8.22%	13.04%
September	8.22%	7.80%	10.08%
October	9.25%	9.25%	9.28%
November	9.08%	9.14%	8.77%
December	6.73%	6.95%	5.75%

Monthly Ticket Frequency

It might be interesting to know which months see an increase or decrease in ticket frequency. For example for more efficiently assigning resources.

Table 8.1 shows tickets frequencies for each month. While there clearly are months with more issues (e.g. July) and months with less issues (e.g. December), there does not seem to be a clear pattern.

In table 8.2 we only considered high impact tickets (“P1”, “P1 Internationaal”, or “Grote Storing”). December has a relatively large amount of high impact tickets compared to the overall amount of tickets. Interestingly, July has both a very high percentage of overall tickets and high impact tickets.

There is not much we can conclude from these tables. Although it is clear that July is a month in which many issues occur. Additionally, in 2016 the summer saw a much larger amount of tickets than one would expect based on historical data.

Hourly Ticket Frequency

Finally, we consider how many tickets get created on average for each hour of the day. As can be seen in figures 8.4 and 8.5 this mostly coincides with working time.

[2]Some ARS-S tickets are not assigned an impact category, thus the sum of tickets over all impact categories is not necessarily equal to the total amount of tickets before 2015.

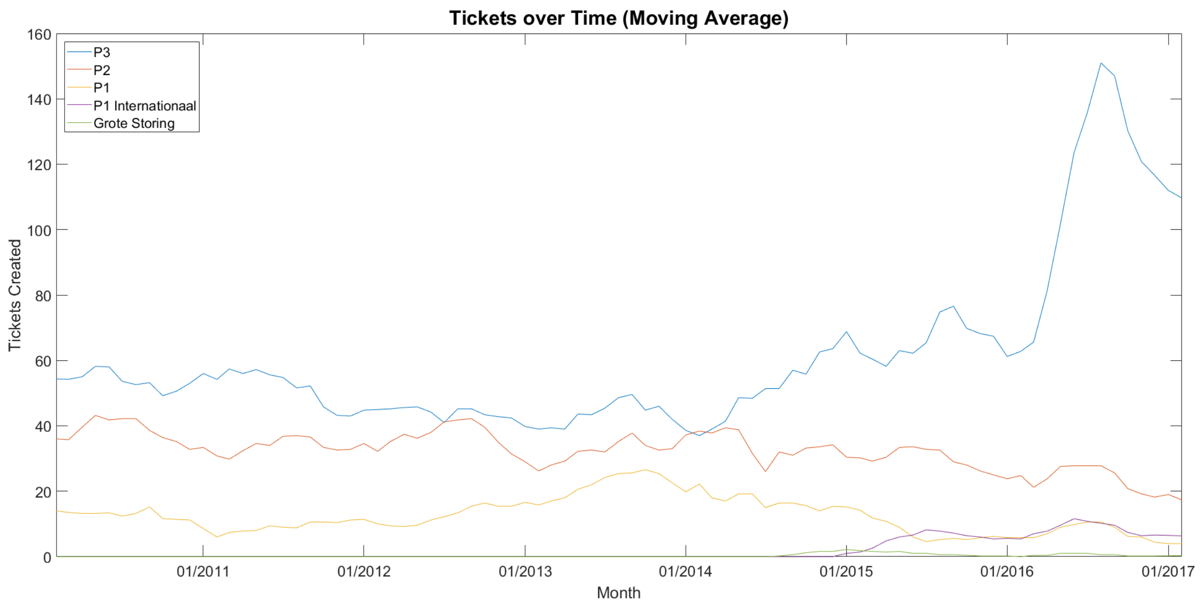


Figure 8.3: Central moving average (with a window size of 5 months) of tickets created per month per impact category.

Table 8.2: Percentage of high impact (“P1”, “P1 Internationaal”, or “Grote Storing”) tickets created in each month

Month	**2010-2016**	**2010-2015**	**2016**
January	6.85%	6.95%	6.28%
February	7.65%	7.89%	6.28%
March	7.41%	7.61%	6.28%
April	8.21%	8.55%	6.28%
May	7.73%	7.42%	9.42%
June	7.81%	7.14%	11.52%
July	11.39%	10.24%	17.80%
August	9.16%	8.36%	13.61%
September	7.33%	7.52%	6.28%
October	9.08%	9.49%	6.81%
November	9.40%	10.06%	5.76%
December	7.97%	8.74%	3.66%

8.1.2. Resolution Times and Open Tickets

In this section we consider the time it takes to resolve a ticket after its creation. We measure this by comparing the “resolutiondate” of a ticket with its creation time. We filter out all ARS-S tickets, as this field has only been filled in for ARS-S tickets with a very long resolution time (resolved after the transfer to JIRA). Out of 3725 remaining tickets, 3697 are assigned a resolution date.

Table 8.3 shows the median and maximum resolution times by impact category. A typical resolution time is 16 hours. Median resolution times are lower for non-P3 tickets, with the exception of “P1 Internationaal” and “Grote Storing” tickets, presumably because these are more difficult to resolve. For example, “P1 Internationaal” issues involve more international cooperation and often require SURFnet to rely on another party to resolve an issue (i.e. sub-oceanic fiber cuts).

It is surprising how many tickets have a resolution time of more than 30 days. Most of these tickets are “P3” tickets, and probably had such a low impact other tickets were given a higer priority. However, 3 “P1” tickets took more than 30 days to resolve.

Note that this means it is a distinct possibility that a ticket opened more than a few months ago is still open. We filter out all open tickets. This means that we are underestimating the number of tickets created in some months (especially those in 2017) in section 8.1.1.

In addition, we are also underestimating median (and potentially maximum) resolution times, as recent tickets with a potentially high resolution time have not been closed yet.

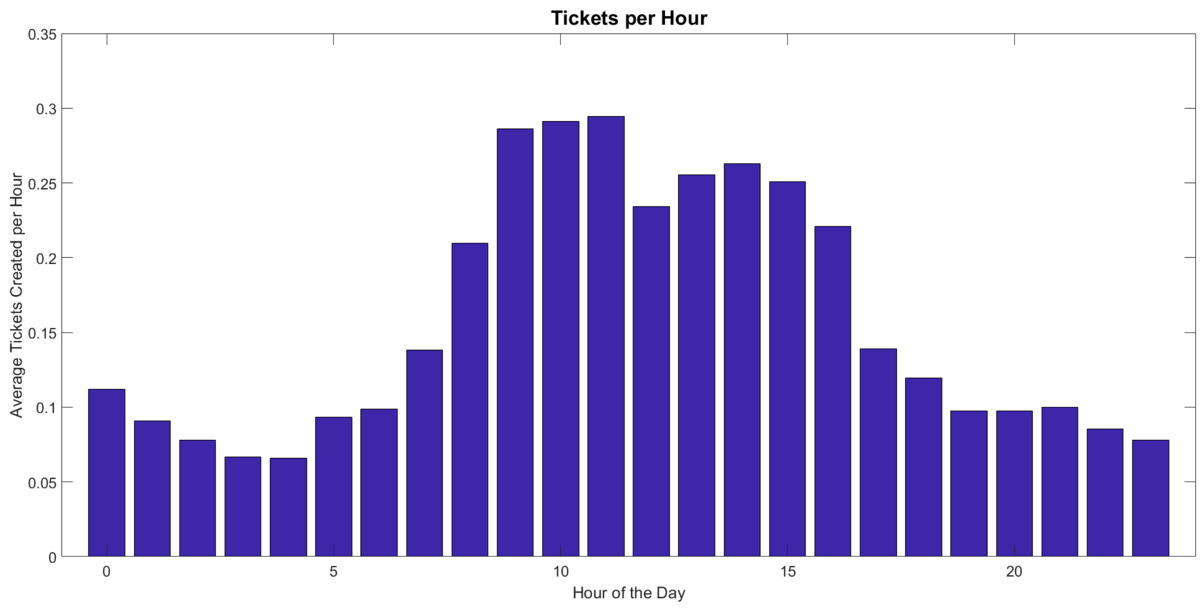


Figure 8.4: Average amount of tickets created per hour per day.

Table 8.3: Resolution Time in Hours

Impact	Median Resolution Time	Maximum Resolution Time	# > 30 Days
P3	18	11496	177
P2	9	5569	19
P1	9	1101	3
P1 Internationaal	20	429	0
Grote Storing	45	455	0
All tickets	16	11496	199

Table 8.4 shows the number of open tickets. Most of the 64 open tickets currently have a “Waiting” status. No tickets created before 2013 are still open.

8.1.3. Causes

In this section we try to analyze the causes of tickets. This information can be very helpful in reducing the number of problems in the future. As mentioned in the introduction, the ticket tracking system is not setup to be a monitoring or archiving tool, or to be machine-readable. Unfortunately, this makes processing the causes of tickets very difficult, if not impossible.

Tickets in SURFnet’s setup can have two fields which could potentially be of help: “Solution” and “Root cause”. Unfortunately, both fields are not mandatory, so many tickets do not include them. Out of all 9934 tickets, only 2707 include a solution field, and only 807 a root cause field.

Employees can freely fill in the solution and the root cause of a ticket themselves in these fields, and there is no standard format they have to adhere to. Thus the fields are not easily machine-interpretable.

Finally, the fields are not always used as they are supposed to. Especially the solution field often contains remarks such as “Probleem is opgelost” (meaning problem has been solved).

It is fair to say that automatically (and accurately) assigning causes to tickets in the dataset is very difficult, or even impossible.

Thus we first consider only tickets assigned the worst possible impact category: “Grote Storing” (Major Failure). This subset only contains 27 tickets, so we can manually assign each of them a cause (category), based on their summary, description, comments, solution, and root cause.

The result can be found in table 8.5. 1 ticket contains no mention of the cause (and is closed as the customer decided to fix the problem himself). 2 tickets are duplicates of other tickets. As mentioned before, tickets which are closed as duplicates or clones are automatically filtered out. Unfortunately, this does not include tickets which were purposefully created to be duplicates (to continue discussing a previously ‘resolved’ issue).

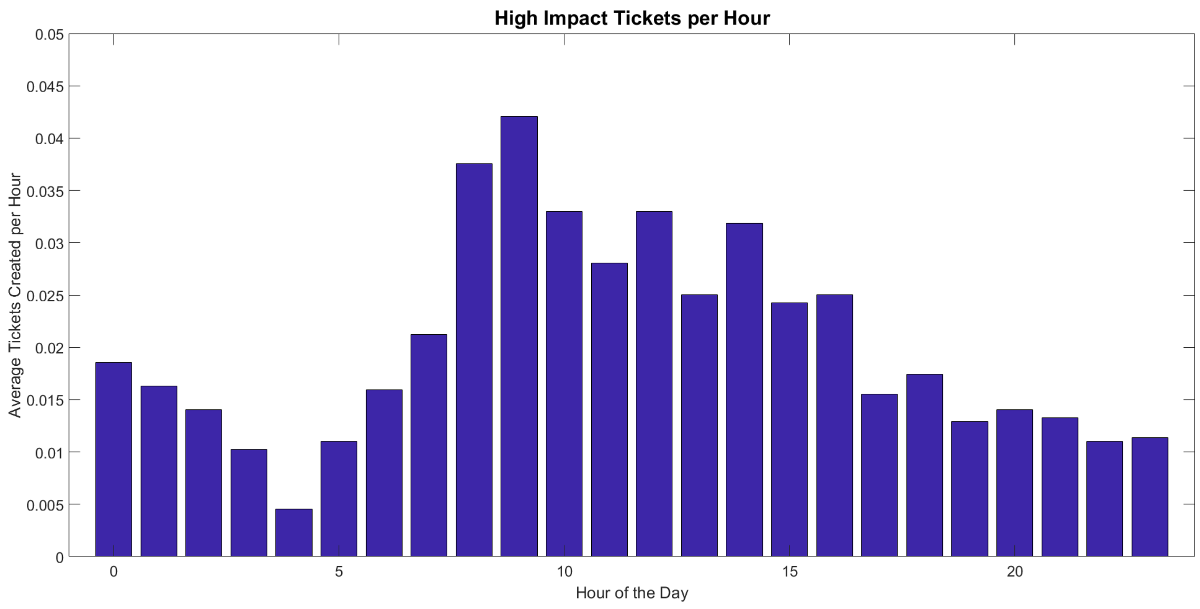


Figure 8.5: Average amount of high impact (“P1”, “P1 Internationaal”, or “Grote Storing”) tickets created per hour per day.

Table 8.4: Open Tickets

Creation Date	Open Tickets
2013-2017	64
2013-2015	5
2016	27
01/2017	9
02/2017	6
03/2017	17

8 out of 27 major failures have been caused by a single node failure. This seems very high, as one would expect a network to be resilient enough to be able to endure a single failure.

4 of the outages were caused by fiber failures. All fiber failures resulting in “grote storingen” were the result of mistakes by a fiber provider during maintenance or repairs.

Of special interest to us are losses of power, as these kind of failures are geographically correlated. 5 out of 27 major failures were caused by a power cut. Additionally, 3 out of 27 were caused by a loss of power due to a hardware failure at the point of presence. As almost 30% of “grote storingen” are geographically correlated, this gives an increased incentive to continue our research on this subject.

Loss of power is also perhaps the only category (of those in table 8.5) we can automatically assign tickets to by searching for key words. To get an idea of the total amount of issues caused by a loss of power we counted all tickets containing one of the following keywords: “powerfailure”, “powercut”, “poweroutage”, “powerloss”, “blackout”, “stroomstoring”, “stroomuitval”, “power failure”, “power-failure”, etc.

Table 8.6 shows the amount of ‘power loss tickets’ per impact category. As these tickets have been labeled automatically the result is probably not completely correct, but it should be fairly accurate. In particular, we can see that 8 out of 27 “Grote Storing” tickets have been labeled as being caused by a loss of power. This matches the results of our manual cause assignments.

With the exception of “P1 Internationaal”, higher impact categories contain a larger percentage of tickets caused by a loss of power. This is to be expected, as a loss of power generally affects a lot of hardware, and thus has a large impact on the network. 45.54% of tickets caused by a loss of power have an impact of “P1” or worse.

We can conclude that power outages have had a significant impact on the SURFnet network. As many as 15.88% of high impact tickets (“P1”, “P1 Internationaal”, or “Grote Storing”) have been caused by a loss of power.

Table 8.5: Causes of “Grote Storingen” (Major Failures)

	Number of Tickets
Node issues/failure	8
Fiber(s) down	4
Power failure/cut	5
Loss of power because of a hardware failure	3
Human error	2
Third party issues	2
Unmentioned	1
Duplicate Ticket	2
Total	27

Table 8.6: Tickets caused by a loss of power (based on an automatic search for keywords)

Impact	Caused by a loss of power	Total Tickets	Percentage caused by loss of power
P3	120	5287	2.27%
P2	119	2786	4.27%
P1	194	1083	17.91%
P1 Internationaal	2	175	1.14%
Grote Storing	8	27	29.63%
All tickets	448	9934	4.51%

8.1.4. Hardware and Sites

Each SURFnet site and network device is assigned an unique ID. The ID of failed hardware, or SURFnet site(s) where failures are occurring, is required to do in-depth geographical analysis of tickets. In particular, this information is necessary for predicting future events or computing the vulnerability of the network. Additionally, information on hardware failures can tell us which specific type/brand of devices performs better, or worse, than others.

Tickets do not contain a failed hardware field, thus the ID of the failed hardware needs to be extracted from the text of the tickets themselves. Unfortunately, there are a number of problems with this approach:

- Sites and devices are typically mentioned when a connection to them fails (or has lost its redundant path).
- As tickets contain communication between employees and between SURFnet and customers and providers, they also contain references to hardware that has not failed. For example, employees can ask if a certain device is affected, mention that a device or point of presence is not affected or compare the problem to a similar issue of another device.
- Tickets can be created when no hardware has failed at all, but still refer to a site or device. In particular, all e-mails sent to the SURFnet helpdesk automatically result in a ticket.

Although these problems can be mitigated somewhat, for example by filtering out connection IDs (which contain device/PoP IDs), analyses of hardware logs will still be much more accurate. These logs lack the context tickets can provide, but we have already shown that for example the root cause is very difficult to accurately extract from tickets. As such, at the moment we propose using hardware logs instead of tickets for geographical analysis of past network failures. If possible, combined analysis of tickets and logs might provide more detailed results.

8.2. Recommendations

Tickets are used to keep track of, and keep employees up to date about, current issues and problems. Thus certain choices were made that while useful for their purpose, makes the analysis of tickets less accurate and more difficult. We briefly go over some of the problems we encountered while analyzing the tickets dataset.

Many tickets in the dataset are wrongly classified as problems and incidents. One common example are test tickets, created simply to test if the ticket system functions correctly or to learn a new employee how to use the ticket system. However, the biggest source of misclassified tickets are e-mails sent to the helpdesk. The helpdesk converts all incoming e-mails to trouble tickets. This includes questions from customers, mail from providers and even mail sent by SURFnet employees who preferred contacting the helpdesk directly over creating an issue in JIRA.

This leads us to a related problem. When e-mails are converted to tickets, the subject line is used as the ticket summary. The result of this is that many ticket summaries do not accurately summarize the ticket. For example, the dataset contains a ticket simply summarized as “Storing.” (meaning Outage). This type of summary gives insufficient information about the issue itself. Additionally, the e-mail itself did not describe an outage at all (only an authentication issue), and should have been sent to SURFconext instead. In contrast to the previous problem, this is also a detriment to the actual functionality of tickets, as employees need to read the ticket descriptions to figure out what the underlying issues are, instead of simply glancing at the ticket summaries.

Tickets themselves contain all communication towards customers, which can quickly add up and make manually reading through tickets much less manageable. These communications include all e-mail traffic. For example, some tickets contain tens of automated e-mails informing SURFnet that persons they are trying to inform of an issue is currently on holiday. Mail is included as is, and are not made more readable before being added to the ticket. Because a reply can contain the complete conversation, often the entire mailing history is copied multiple times in the same ticket.

A large issue is the usage of free fields. It is difficult to automatically extract useful information from these fields. In addition, there are not many fields which are obligatory to fill in. This has been a deliberate choice by SURFnet, because they found they often needed to add more options to multiple choice fields, or add more custom fields, as there was always a new situation they did not account for beforehand. Evidence of SURFnet’s problems with custom fields can be found by looking through the list of all custom fields. Many fields saw (almost) no use. In addition, some fields are extremely similar to other fields.

Finally, it seems as if filling in tickets is deemed much less important after a problem has been solved, as evidenced by the lack of root causes and solution given in tickets. Keeping in mind the purpose of tickets (tracking current problems/incidents), this is to be expected.

We propose introducing a separate system for archiving past incidents for analysis, as archiving past incidents for analysis and keeping each other up to date about current incidents are seemingly incompatible. This system should take up as little time as possible from SURFnet employees, to keep them willing to properly update the system and to prevent wasting time.

The system can be included in the current JIRA setup by creating a new type of JIRA issue. This avoids having to introduce completely new software. We propose creating a new issue after resolving a incident or problem with significant impact. By limiting by impact, we avoid wasting time by creating issues for all small problems and incidents, which might quickly tire out employees. Note that these new issues should only be created for actual problems and incidents, and not for misclassified tickets.

The issues should contain the following fields:

- Related ticket ID(s)
- Impact
- Summary
- Brief description
- Timestamp of the start of the problem
- Timestamp of the resolution of the problem
- Root cause (preferable multiple choice)
- Failed Device/PoP ID(s)
- List of affected connections

Most of this is readily available and easily filled in. Only the summary and description might take some time to fill in. These fields are not important for automatic analysis, so they could potentially be omitted.

If possible, relevant hardware logs could be attached to these issues to provide additional information. This can potentially be automated based on device ID and timestamps.

9 Probabilistic Disaster Model

In this section we discuss how to extend our disaster model to include probabilistic disasters.

In section 9.1 we introduce a simple probabilistic disaster model, and give some properties of this model. The most important being that, in contrast to the deterministic model, computing the (W)ATTR vulnerability distribution becomes NP-hard.

Next, in section 9.2 we propose a more extensive probabilistic disaster model and introduce the Probabilistic Vulnerability Distribution Problem. We show how to create probabilistic disaster sets based on earthquake data and give both an exact algorithm and a randomized approximation algorithm for the Probabilistic Vulnerability Distribution Problem.

9.1. Simple Model

A simple, but naive, method to incorporate probabilistic failures into our model is to assign a link failure probability $P_l(d)$ to each disaster $d \in D$. If any link intersects the disaster area $A(d)$, it fails with probability $P_l(d)$. If a link does not intersect the disaster area, it has failure probability 0. Link failures are assumed to be mutually independent.

Definition 6. Simple Probabilistic Vulnerability Distribution Problem Given a network $G = \{V, E\}$ and a disaster set D, each with a disaster area $A(d)$, occurrence probability $P(d)$ and link failure probability $P_l(d)$.

Let M be the random real-valued variable describing the state of the network after exactly one disaster from D randomly occurs.

Compute

$$P(M = m) \quad \forall m \in \mathbb{R} \; s.t. \; P(M = m) \neq 0$$

Even with this simple model, computing the vulnerability distribution for the weighted average 2-terminal reliability (WATTR), or even the average 2-terminal reliability (ATTR), becomes an NP-hard problem.

Theorem 2. *The Simple Probabilistic (W)ATTR Vulnerability Distribution Problem is NP-hard.*

Proof. Suppose we are given an undirected network $G = \{V, E\}$, and each link of G has mutually independent failure probability p. Computing the probability that G remains connected is a well-known NP-hard problem [55]. We will reduce this problem to the Simple Probabilistic Vulnerability ATTR Distribution Problem.

Let the network area R be a disk with radius 2. Assign positions $p_i \in R$ to each $v_i \in V$. Model each link $(v_j, v_k) \in E$ as a single line segment connecting v_j and v_k. Let d be a circle disaster with radius 2 and link failure probability $P_l(d) = p$, and set $D = \{d\}$. By solving this instance of the Simple Probabilistic Vulnerability ATTR Distribution Problem we solve the above instance of the Connectedness Reliability Problem, as the probability that G remains connected is $P(\text{ATTR} = 1)$.

We have reduced the NP-hard Connectedness Reliability Problem to the Simple Probabilistic (W)ATTR Vulnerability Distribution Problem and can conclude that the Simple Probabilistic (W)ATTR Vulnerability Distribution Problem is NP-hard. □

However, this is not the case for all measures. Consider for example the link survival ratio (LSR).

Theorem 3. *Simple Probabilistic LSR Vulnerability Distribution Problem* $\in P$.

Proof. Suppose we are given a network $G = \{V, E\}$ and a disaster set D. The LSR is the number of surviving links divided by $|E|$. Note that if a disaster $d \in D$ occurs, the number of failing links is binomially distributed, with 'success' probability $P_l(d)$. Thus we can compute the LSR vulnerability distribution as follows:

1. $\forall d \in D$, compute $\#(d) = |\{e \in E | e \text{ intersects } A(d)\}|$
2. for $k = 0, 1, \ldots, |E|$, $P(\text{LSR} = \frac{|E|-k}{|E|}) = \sum\limits_{d \in D | \#(d) \geq k} P(D)\binom{\#(d)}{k} P_l(d)^k (1 - P_l(d))^{\#(d)-k}$

□

Remark 2. While the problem is polynomial in $|V|$ and $|E|$, it is linear in $|D|$ (and $|N(D)|$).

In contrast to the deterministic case, where the number of possible failure states was upper-bounded by $|D|$, the number of possible failure states in the probabilistic case can be as high as $2^{|E|}$, so it is not efficient anymore to compute the failure state distribution as an intermediate step towards computing the vulnerability distribution.

Consider the following more simplified problem:

Definition 7. Simple Probabilistic Disaster Problem Given a network $G = \{V, E\}$ and a disaster d, with disaster area $A(d)$, occurrence probability $P(d)$ and link failure probability $P_l(d)$.

Let M be the random real-valued variable describing the state of the network after d occurs.

Compute

$$P(M = m) \quad \forall m \in \mathbb{R} \; s.t. \; P(M = m) \neq 0$$

Let $X \leq Y$ indicate the existence of a polynomial-time reduction from X to Y.

Theorem 4. *For any given measure:*

Simple Probabilistic Disaster Problem ≤ Simple Probabilistic Vulnerability Distribution Problem

and if all instances of the Simple Probabilistic Disaster Problem can be solved in $\mathcal{O}(f(|V|, N_e, N(d)))$ time, then all instances of the Simple Probabilistic Vulnerability Distribution Problem can be solved in $\mathcal{O}(|D| f(|V|, N_e, N(D)))$ time. In particular:

Simple Probabilistic Vulnerability Distribution Problem ∈ P ⇔ Simple Probabilistic Disaster Problem ∈ P

Proof. Simple Probabilistic Disaster Problem ≤ Probabilistic Vulnerability Distribution Problem is trivial, as the Simple Probabilistic Disaster Problem is a special case of the Probabilistic Vulnerability Distribution Problem.

So we only need to prove that if all instances of the Simple Probabilistic Disaster Problem can be solved in $\mathcal{O}(f(|V|, N_e, N(d)))$ time, then all instances of the Simple Probabilistic Vulnerability Distribution Problem can be solved in $\mathcal{O}(|D| f(|V|, N_e, N(D)))$ time.

Assume an algorithm for the Simple Probabilistic Disaster Problem exists with time complexity $\mathcal{O}(f(|V|, N_e, N(d)))$.

Suppose we are given a network $G = \{V, E\}$ and a disaster set D. We compute the vulnerability distribution as follows:

1. $\forall d \in D$, compute and store $P(M = m | d)$ $\forall m \in \mathbb{R}$ $s.t.$ $P(M = m | d) \neq 0$ using the algorithm for the Simple Probabilistic Disaster Problem.
2. $P(M = m) = \sum\limits_{d \in D} P(d) P(M = m | d)$

$N(d)$ is upper-bounded by $N(D)$, so step 1 takes $\mathcal{O}(|D| f(|V|, N_e, N(D)))$ time and we store $\mathcal{O}(f(|V|, N_e, N(D)))$ probabilities for every disaster. Iterating over every possible value for M and computing $P(M = m)$ requires accessing each of these probabilities exactly once, thus this step also takes $\mathcal{O}(|D| f(|V|, N_e, N(D)))$ time.

So if all instances of the Simple Probabilistic Disaster Problem can be solved in $\mathcal{O}(f(|V|, N_e, N(d)))$ time, then all instances of the Probabilistic Vulnerability Distribution Problem can be solved in $\mathcal{O}(|D| f(|V|, N_e, N(D)))$ time. □

When analyzing the time complexity of the probabilistic problem, it might be more useful to consider a related problem where failure probabilities are assigned directly to links, as this is often the case in research into the time complexity of reliability problems.

Definition 8. Simple Probabilistic Network Distribution Problem Given a network $G = \{V, E\}$ (not embedded in the plane) and link failure probability p. i.e. each link of G has independent failure probability p.

Let M be the random real-valued variable describing the state of the network.

Compute

$$P(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$$

Theorem 5. *For any given measure:*

Simple Probabilistic Network Distribution Problem ≤ Simple Probabilistic Vulnerability Distribution Problem

Proof. Suppose we have an instance of Simple Probabilistic Network Distribution Problem: $G = \{V, E\}$ and p.

First embed G in the plane as follows: let the network area R be a disk with radius 2. Assign positions $p_i \in R$ to each $v_i \in V$. Model each link $(v_j, v_k) \in E$ as a single line segment connecting v_j and v_k.

Let d be a circle disaster with radius 2 and link failure probability $P_l(d) = p$, and set $D = \{d\}$.

We now have an instance of the Simple Probabilistic Vulnerability Distribution Problem with the same solution as the Simple Probabilistic Network Distribution Problem instance. So

Simple Probabilistic Network Distribution Problem ≤ Simple Probabilistic Vulnerability Distribution Problem

□

9.2. Extensive Model

The simple probabilistic model is a very straightforward extension of our deterministic model. However, it is not very realistic. For example, the possibility of node failures is completely ignored. This was not an issue in the deterministic model, as the failure of all incident links would essentially give an equivalent result. In a probabilistic model, incident links of nodes in the disaster area only fail with a certain probability, so node failure probabilities also need to be accounted for.

In this section we introduce a more extensive probabilistic disaster model, based on the following requirements:

- The model should account for both link and node failures.
- The failure probability of a link intersecting the disaster area should depend on the length of the intersection.
- Different parts of the disaster area should result in different failure probabilities. This allows for the modeling of earthquake intensity levels for example.
- Different equipment (e.g. aboveground and underground cables) can have different failure probabilities under the same circumstances.

9.2.1. Model

As with the simple model, we assume that exactly one disaster occurs and that given the occurrence of a specific disaster, device failures are mutually independent.

We assume we have a finite set of disasters D. Each disaster $d \in D$ has occurrence probability $P(d)$. We model the disaster area of d as a finite set $A(d) = \{A_1, A_2, \ldots, A_k\}$ of areas. These areas can either be a circle (disk), line segment, hippodrome, simple polygon, or a finite union of these. We say a link (or node) intersects $A(d)$ if and only if it intersects one or more of the areas $A_i \in A(d)$. We assume we are given link failure probabilities $P_l(e, A_i)\ \forall e \in E$ and node failure probabilities $P_n(v, A_i)\ \forall v \in V$ for each $A_i \in A(d)$. To be able to model disasters where intersection length has no effect on the probability of failure, we introduce $f_l : D \to \{0, 1\}$. $f_l(d) = 1$ if the intersection length with disaster d has no effect on the probability of failure, $f_l(d) = 0$ otherwise. Finally, we assume the areas of $A(d)$ are mutually disjoint.

Remember that all nodes $v_j \in V$ are modeled as points $p_j \in R$, where R is the network area. Thus a node $v_j \in V$ intersects A_i if and only if $p_j \in A_i$. Note that a node can intersect at most one area of $A(d)$. If disaster $d \in D$ occurs, any node $v_j \in V$ fails with probability

$$P(v_j \text{ fails}|d) = \begin{cases} 0, & \text{if } v_j \text{ does not intersect } A(d) \\ P_n(v_j, A_i), & \text{if } p_j \in A_i \end{cases} \tag{9.1}$$

If $f_l(d) = 1$, we assume the failure probability of a link $e \in E$ intersecting the disaster area $A(d)$ is equal to $\max_{A_i \in A(d) | e \text{ intersects } A_i} P_l(e, A_i)$.

We base our $f_l(d) = 0$ link failure probability model on the probabilistic region failure model of [64]. Given a link $e \in E$ and disaster $d \in D$. Let $\text{len}(e, A_i)$ be the total length of the intersection of e and $A_i \in A(d)$. If we assume that $P_l(e, A_i)\frac{\text{len}(e,A_i)}{n}$ is the failure probability of an arbitrarily short part of the intersection of A_i and e we can say that the failure probability of the intersection (not of the whole link) is

$$1 - \lim_{n\to\infty}\left(1 - P_l(e, A_i)\frac{\text{len}(e, A_i)}{n}\right)^n = 1 - \exp(-P_l(e, A_i)\text{len}(e, A_i))$$

A link fails if any of its intersections fail, so the failure probability of e is given by

$$1 - \prod_{i=1}^{k} \exp(-P_l(e, A_i)\text{len}(e, A_i)) = 1 - \exp(-\sum_{i=1}^{k} P_l(e, A_i)\text{len}(e, A_i))$$

Thus, if disaster $d \in D$ occurs, any link $e \in E$ fails with probability

$$P(e \text{ fails}|d) = \begin{cases} 0 & \text{if } e \text{ does not intersect } A(d) \\ \max_{A_i \in A(d)|e \text{ intersects } A_i} P_l(e, A_i) & \text{if } f_l(d) = 1 \text{ and } e \text{ intersects A(d)} \\ 1 - \exp(-\sum_{i=1}^{k} P_l(e, A_i)\text{len}(e, A_i)) & \text{otherwise} \end{cases} \tag{9.2}$$

where $k = |A(d)|$ and $\text{len}(e, A_i)$ is the length of the intersection of e and $A_i \in A(d)$.

To be able to efficiently model mutually disjoint areas of varying shapes we allow one more type of area: the difference between two other valid areas. This allows us to for example model a set of annuli with decreasing failure probabilities as the radius increases. In addition, as we exemplify in section 9.2.2, this also allows for more efficient representation of disaster areas.

This model does not allow for the modeling of an continuous decrease (or increase) in disaster intensity based on for example the distance to the epicenter. Failure probabilities can only be decreased (increased) stepwise. However, by decreasing the size of individual areas in $A(d)$, one can get arbitrarily close to a continuous change in failure probabilities.

As in the deterministic case, our model allows for a very accurate approximation of any disaster shape.

Note that in contrast to the deterministic model, a disaster set can only be applied to a single network, as failure probabilities are given for devices of a specific network. To be able to apply disaster sets more broadly, one could assign devices to specific classes (e.g. aboveground cable, underground cable, protected cable) and assign failure probabilities to these classes instead of the devices themselves.

We define the Probabilistic Vulnerability Distribution Problem as follows:

Definition 9. Probabilistic Vulnerability Distribution Problem Given a network $G = \{V, E\}$ and a probabilistic disaster set D, each with a disaster area $A(d)$ as described above.

Let M be the random real-valued variable describing the state of the network after exactly one disaster from D randomly occurs.

Compute

$$P(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$$

For the purpose of computing device failure probabilities we consider link and node failures separately. However, when considering the state of the network itself after any node/link failures we model a node failure as the failure of all its incident links and redefine the failure state as follows:

Definition 10. Failure State Given a network $G = \{V, E\}$, define a failure state of G as a set $s \subseteq E$, where $s = \{e = (v_i, v_j) \in E | e \text{ failed or } v_i \text{ failed or } v_j \text{ failed}\}$.

Thus for example the link survival ratio remains $\text{LSR} = \frac{|E|-|s|}{|E|}$.

9.2.2. Earthquakes

To give an example of how this probabilistic model can be used to model disasters based on empirical data, we show how to adapt our earlier method to create deterministic disaster sets based on J-SHIS and USGS earthquake data to create probabilistic disaster sets.

As part of our conversion progress we converted disaster maps to grid cells and assigned intensity levels to each of these cells. To create a deterministic disaster set, we created a disaster area by taking the union of all cells with an intensity above a certain threshold.

To create a probabilistic disaster set, we require functions $g_l : E \times \mathbb{R} \to [0,1]$ and $g_n : V \times \mathbb{R} \to [0,1]$ that match link and node failure probabilities to intensity levels. Then, we can create the disaster area $A(d)$ as the set of all cells. If the intensity level of cell $A_i \in A(d)$ is I, its failure probabilities are $P_l(e, A_i) = g_l(e, I)$ and $P_n(e, A_i) = g_n(e, I)$.

Unfortunately, storing each disaster area as a set of small cells is incredibly inefficient, and $|A(d)|$ will be very large. It is more efficient to combine cells with similar intensity levels. Assume g_l and g_n are piecewise-constant in intensity, e.g. $g_l(e, I) = \begin{cases} 1 & \text{if } I \geq 10 \\ 0.8 & \text{if } 8 \leq I < 10 \\ 0.4 & \text{if } 4 \leq I < 8 \\ 0 & \text{if } I < 4 \end{cases}$

Let s be a finite sequence of length k such that for all $i < k$

$$g_l(e,a) = g_l(e,b)\ \forall e \in E\ \forall a,b \in \mathbb{R} \text{ s.t. } s_i \leq a < s_{i+1} \wedge s_i \leq b < s_{i+1}$$

$$g_n(v,a) = g_n(v,b)\ \forall v \in V\ \forall a,b \in \mathbb{R} \text{ s.t. } s_i \leq a < s_{i+1} \wedge s_i \leq b < s_{i+1}$$

and

$$g_l(e,a) = g_l(e,b)\ \forall e \in E\ \forall a,b \in \mathbb{R} \text{ s.t.} a < s_1 \wedge b < s_1$$
$$g_l(e,a) = g_l(e,b)\ \forall e \in E\ \forall a,b \in \mathbb{R} \text{ s.t.} a \geq s_k \wedge b \geq s_k$$
$$g_n(v,a) = g_n(v,b)\ \forall v \in V\ \forall a,b \in \mathbb{R} \text{ s.t.} a < s_1 \wedge b < s_1$$
$$g_n(v,a) = g_n(v,b)\ \forall v \in V\ \forall a,b \in \mathbb{R} \text{ s.t.} a \geq s_k \wedge b \geq s_k$$

We can create $A(d) = \{A_1, A_2, \ldots\}$ as follows:

- $A_1 =$ the union of all cells with intensity $I < s_1$
- $A_2 =$ the union of all cells with intensity $s_1 \leq I < s_2$
- ...
- $A_k =$ the union of all cells with intensity $I \geq s_k$

Cells in the same area have exactly the same failure probabilities. As failure probabilities of each area we simply take the failure probabilities of one of its cells.

As noted in chapter 5, cells can be efficiently combined as simple polygons. Unfortunately, areas A_1 to A_{k-1} contain holes created by the absence of the higher intensity cells. These holes make combining cells into simple polygons less efficient. Consider A_1 for example. A_1 is a large rectangle (the entire grid area) minus areas A_2 to A_k. A large rectangle could be presented as a single simple polygon, but A_1 itself cannot.

Note that intensity strongly correlates with distance to the epicenter, so higher intensity areas typically do not contain many holes created by the absence of cells with lower intensity levels.

Presenting areas as set differences to our algorithms would be more efficient. Create the following intermediate areas:

- $H_1 =$ the union of all cells
- $H_2 =$ the union of all cells with intensity $I \geq s_1$
- $H_3 =$ the union of all cells with intensity $I \geq s_2$
- ...
- $H_k =$ the union of all cells with intensity $I \geq s_{k-1}$
- $H_{k+1} =$ the union of all cells with intensity $I \geq s_k$

These areas can all be relatively efficiently presented as unions of simple polygons.

We present $A(d)$ as follows:

- $A_1 = H_1 \setminus H_2$
- $A_2 = H_2 \setminus H_3$
- …
- $A_k = H_{k+1}$

Technically, it was also an issue that neighboring cells, and thus neighboring areas, were not mutually disjoint. In practice this should have no effect on the vulnerability distribution. Nevertheless, our more efficient representation solves this problem as well.

9.2.3. Time Complexity

Note that the Simple Probabilistic Vulnerability Distribution Problem is a special case of the Probabilistic Vulnerability Distribution Problem, where $|A(d)| = 1$, $P_l(e, A_1) = P_l(d)$, $f_l(d) = 1$ and $P_n(v, A_1) = 0$ for all $e \in E, v \in V, d \in D$. So some results of section 9.1 also hold for the extensive model.

In particular:

Theorem 6. *The Probabilistic (W)ATTR Vulnerability Distribution Problem is NP-hard.*
For any given measure:

Probabilistic Network Distribution Problem ≤ Probabilistic Vulnerability Distribution Problem

As before, we consider a related problem where failure probabilities are assigned directly to links and nodes.

Definition 11. Probabilistic Network Distribution Problem Given a network $G = \{V, E\}$ (not embedded in the plane), link failure probabilities $P_l : E \to [0,1]$ and node failure probabilities $P_n : V \to [0,1]$. Assume all failures are mutually independent.

Let M be the random real-valued variable describing the state of the network.

Compute

$$P(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$$

Theorem 7. *For any given measure:*

Probabilistic Network Distribution Problem ≤ Probabilistic Vulnerability Distribution Problem

Proof. Suppose we have an instance of Probabilistic Network Distribution Problem: $G = \{V, E\}$, $P_l : E \to [0,1]$ and $P_n : V \to [0,1]$.

First embed G in the plane as follows: let the network area R be a disk with radius 2. Assign positions $p_i \in R$ to each $v_i \in V$. Model each link $(v_j, v_k) \in E$ as a single line segment connecting v_j and v_k.

Construct a disaster d as follows. $A(d) := \{A_1\}$, where A_1 is a disk with radius 2. Note that this is exactly R, so all links and nodes of G intersect A_1. Set $P_l(e, A_1) = P_l(e)\ \forall e \in E$ and $P_n(v, A_1) = P_n(e)\ \forall v \in V$.

Finally, we set $D = \{d\}$ and $f_l(d) = 1$.

We now have an instance of the Probabilistic Vulnerability Distribution Problem with the same solution as the Probabilistic Network Distribution Problem instance. So

Probabilistic Network Distribution Problem ≤ Probabilistic Vulnerability Distribution Problem

□

Any algorithm for the Probabilistic Network Distribution Problem can also be used for the Probabilistic Vulnerability Distribution Problem:

Theorem 8. *(Re)defining N as*

$$N(A_i) := \begin{cases} 1, & \textit{if } A_i \textit{ is a circle, line segment or hippodrome} \\ \textit{the number of polygon sides,} & \textit{if } A_i \textit{ is a simple polygon} \\ \sum_{j=1}^{n} N(A_{i,j}), & \textit{if } A_i = A_{i,1} \cup A_{i,2} \cup \cdots \cup A_{i,n} \\ N(A_{i,1}) + N(A_{i,2}) & \textit{if } A_i = A_{i,1} \setminus A_{i,2} \end{cases} \tag{9.3}$$

and defining $N(d) := \sum_{A_i \in A(d)} N(A_i)$ *and* $N(D) := \sum_{d \in D} N(D)$.

For any given measure:

If all instances of the Probabilistic Disaster Problem can be solved in $\mathcal{O}(f(|V|,|E|))$ *time, then all instances of the Probabilistic Vulnerability Distribution Problem can be solved in* $\mathcal{O}(|D|f(|V|,|E|)+(|V|+N_e)N(D))$ *time.*

Proof. Assume we have an algorithm that can solve all instances of the Probabilistic Disaster Problem can be solved in $\mathcal{O}(f(|V|,|E|))$ time for some function f. We will show that using this algorithm, we can solve all instances of Probabilistic Vulnerability Distribution Problem in $\mathcal{O}(|D|f(|V|,|E|)+(|V|+N_e)N(D))$ time.

Assume we have an instance of Probabilistic Vulnerability Distribution Problem $G = \{V,E\}$, D and $f_l : D \to 0,1$. We can find its vulnerability distribution as follows:

1. $\forall d \in D$, compute all node and link failure probabilities
2. $\forall d \in D$, compute $P(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$ by giving the algorithm for the Probabilistic Disaster Problem the node and link failure probabilities computed in step 1
3. $P(M = m) = \sum_{d \in D} P(M = m)$

As we will show in section 9.2.4, step 1 takes $\mathcal{O}((|V|+N_e)N(D))$ time.

Step 2 takes $\mathcal{O}(|D|f(|V|,|E|))$ time. This means that per disaster there are $\mathcal{O}(f(|V|,|E|))$ possible (i.e. $P(M = m) > 0$) values for M.

By iterating over all probabilities computed in step 2, and keeping track of the sums of $P(M = m)$ for all possible m, we can compute the vulnerability distribution in $\mathcal{O}(|D|f(|V|,|E|))$ time.

Thus the total computation time required is $\mathcal{O}(|D|f(|V|,|E|)+(|V|+N_e)N(D))$.

So if all instances of the Probabilistic Disaster Problem can be solved in $\mathcal{O}(f(|V|,|E|))$ time, then all instances of the Probabilistic Vulnerability Distribution Problem can be solved in $\mathcal{O}(|D|f(|V|,|E|)+(|V|+N_e)N(D))$ time. □

Theorem 9. *For any given measure:*

$$\textit{Probabilistic Vulnerability Distribution Problem} \in P \Leftrightarrow \textit{Probabilistic Disaster Problem} \in P$$

Proof. This is a direct result of theorems 8 and 7. □

If we ignore node failures, instances of the Probabilistic LSR Vulnerability Distribution Problem can be solved in polynomial time.

Theorem 10. *If all node failure probabilities are zero, the probabilistic LSR vulnerability distribution can be computed in polynomial time.*

Proof. We will first prove that the Probabilistic LSR Network Distribution Problem can be solved in polynomial time if $P_n(v) = 0\ \forall v \in V$.

If $P_n(v) = 0\ \forall v \in V$, the number of failed links is a Poisson binomial distributed variable with success probabilities $P_1 = P_l(e_1), P_2 = P_l(e_2), \ldots,$, so the values $P(|S| = j)$ can be computed in polynomial time for all $j \leq |E|$ [27, 30]. As $P(\text{LSR} = \frac{|E|-j}{|E|}) = P(|S| = j)$, $P(\text{LSR} = m)$ can be computed in polynomial time for all m such that $P(\text{LSR} = m) \neq 0$ by computing $P(|S| = j)$ for $j = 0, 1, \ldots, |E|$.

Note that if all node failure probabilities of an instance of the Probabilistic LSR Vulnerability Distribution Problem are zero, $P(v_j \text{ fails}|d) = 0\ \forall v_j \in V$. Thus we can use our method from the proof of theorem 8 to obtain a polynomial-time algorithm for all instances of the the Probabilistic LSR Vulnerability Distribution Problem with node failure probabilities zero. □

Given that for some measures the Probabilistic Vulnerability Distribution Problem is NP-hard, we might want to approximate the values of the vulnerability distribution instead.

Definition 12. Probabilistic Vulnerability Distribution ϵ-Approximation Problem Given $\epsilon \in \mathbb{R}$, a network $G = \{V,E\}$ and a probabilistic disaster set D, each with a disaster area $A(d)$ as described above.

Let M be the random real-valued variable describing the state of the network after exactly one disaster from D randomly occurs.

Output

$$\hat{P}(M = m) \quad \forall m \in \mathbb{R}\ s.t.\ \hat{P}(M = m) \neq 0$$

such that

$$\hat{P}(M = m) - \epsilon \leq P(M = m) \leq \hat{P}(M = m) + \epsilon \quad \forall m \in \mathbb{R}\ s.t.\ P(M = m) \neq 0$$

Unfortunately, the Probabilistic (W)ATTR Vulnerability Distribution ϵ-Approximation Problem is also NP-hard.

Theorem 11. *The Probabilistic (W)ATTR Vulnerability Distribution ϵ-Approximation Problem is NP-hard.*

Proof. Suppose we are given $\epsilon \in \mathbb{R}$, an undirected network $G = \{V, E\}$, and each link of G has mutually independent failure probability p. Approximating the probability that G remains connected within absolute error ϵ is an NP-hard problem [55]. We will reduce this problem to the Probabilistic (W)ATTR Vulnerability Distribution ϵ-Approximation Problem.

Let the network area R be a disk with radius 2. Assign positions $p_i \in R$ to each $v_i \in V$. Model each link $(v_j, v_k) \in E$ as a single line segment connecting v_j and v_k.

Construct a disaster d as follows. $A(d) := \{A_1\}$, where A_1 is a disk with radius 2. Note that this is exactly R, so all links and nodes of G intersect A_1. Set $P_l(e, A_1) = p\ \forall e \in E$ and $P_n(v, A_1) = 0\ \forall v \in V$.

Finally, we set $D = \{d\}$ and $f_l(d) = 1$.

$P(\text{ATTR} = 1)$ is the probability that the network G remains connected, so $\hat{P}(\text{ATTR} = 1)$ is an approximation of this probability with absolute error $\leq \epsilon$. Thus by solving this instance of the Probabilistic (W)ATTR Vulnerability Distribution ϵ-Approximation Problem we can obtain a solution to the Connectedness Reliability ϵ-Approximation Problem.

We have reduced the NP-hard Connectedness Reliability ϵ-Approximation Problem to the Probabilistic (W)ATTR Vulnerability Distribution ϵ-Approximation Problem and can conclude that the Probabilistic (W)ATTR Vulnerability Distribution ϵ-Approximation Problem is NP-hard. □

9.2.4. Computing Node and Link Failure Probabilities for a Single Disaster

In this section we show how to compute all node and link failure probabilities for a single disaster. This is required in both our exact and approximation algorithm.

Assume we want to compute all node and link failure probabilities for disaster d. That is, we want to compute $P(v \text{ fails}|d)$ for all $v \in V$ and $P(e \text{ fails}|d)$ for all $e \in E$.

To compute the failure probability of a specific node we can simply iterate over each area $A_i \in A(d)$ and test if the node lies within A_i.

If $f_l(d) = 0$, computing the failure probability of a link requires computing all intersection lengths $\text{len}(e, A_i)$. If $f_l(d) = 0$, we can compute the failure probability of a link by iterating over each area $A_i \in A(d)$ and testing if the link intersects A_i.

In the next sections we show how to test if a point lies within an area, compute the intersection length of a link and an area and test if a link intersects an area for each type of area. The link intersection test is only described for the difference of areas area type, as section 4.1.1 already shows how to test if a link intersects an area for all other shapes. Using these results, equations 9.1 and 9.2 can be used to compute all node and link probabilities.

Circle

Let $c = (c_1, c_2)^T$ be the center of the circle area and r its radius.

Trivially, a point p lies within the circular area if $||p - c||_2 \leq r$. Computationally, it is slightly more efficient to test if $(||p - c||_2)^2 \leq r^2$ instead.

The intersection of a link and a circular area can be found by summing the intersection lengths of all its line segments.

Assume we are given a line segment l with endpoints $x = (x_1, x_2)^T$ and $y = (y_1, y_2)^T$. We assume $x \neq y$, because if $x = y$ the line segment is simply a point, and the intersection length of this point and any area will always be 0. The line through these two points intersects the circle at $x + z(y - x)$ for $z \in \mathbb{R}$ such that

$$(c_1 - (x_1 + z(y_1 - x_1)))^2 + (c_2 - (x_2 + z(y_2 - x_2)))^2 = r^2$$

We can rewrite this as

$$(c_1 - x_1)^2 - 2(c_1 - x_1)(y_1 - x_1)z + (y_1 - x_1)^2 z^2 + (c_2 - x_2)^2 - 2(c_2 - x_2(y_2 - x_2)z + (y_2 - x_2)^2 z^2 = r^2$$

or

$$(||c - x||_2)^2 - 2(c - x)^T (y - x)z + (||y - x||_2)^2 z^2 = r^2$$

This equation can be solved using the quadratic formula, giving us

$$z = \frac{(c-x)^T(y-x) \pm \sqrt{((c-x)^T(y-x))^2 - ((||c-x||_2)^2 - r^2)(||y-x||_2)^2}}{(||y-x||_2)^2} \tag{9.4}$$

There can be 0, 1 or 2 intersection points.

If there are less than 2 intersection points, the intersection length of the line segment and circle area is 0.

If there are 2 solutions z_1, z_2, the intersection length can be computed as follows: First replace z_1 and z_2 with values $z_1' := \min(1, \max(0, z_1))$ and $z_2' := \min(1, \max(0, z_2))$, as the line might intersect the circle outside of the line segment itself. Then compute the length itself, which we will denote with $\text{len}(l, A_i)$:

$$\text{len}(l, A_i) = ||(z_2' - z_1')(y-x)||_2 \tag{9.5}$$

It only requires $\mathcal{O}(1)$ time to test if a node lies within a circle area. Computing the intersection length of a circular area and $e_j \in E$ takes $\mathcal{O}(N_j)$ time.

Line Segment

Let $a = (a_1, a_2)^T$ and $b = (b_1, b_2)^T$ be the endpoints of the line segment area. We assume that $a \neq b$, as otherwise the line segment area would be a single point.

A point $p = (p_1, p_2)^T$ lies on this line segment if $(p_1 - a_1)(b_2 - a_2) = (p_2 - a_2)(b_1 - a_1)$, $||p-a||_2 \leq ||b-a||_2$ and $||p-b||_2 \leq ||b-a||_2$. Alternatively, p lies in the line segment if $\exists z \in [0,1]$ s.t. $p = a + z(b-a)$.

To compute the intersection length of this line segment and a link $e_j \in E$ we sum over all segments of e_j again.

Let $x = (x_1, x_2)^T$ and $y = (y_1, y_2)^T$ $(x \neq y)$ be the endpoints of a line segment l of link e_j.

Note that the intersection length of two line segments can only be larger than 0 if the segments are part of the same line. So we first test if all endpoints are collinear. If this is not the case, the intersection length of l and the area is 0.

If the 4 endpoints are collinear, compute $z_1, z_2 \in \mathbb{R}$ such that $a = x + z_1(y-x)$ and $b = x + z_2(y-x)$. As when computing the circle area intersection lengths, we replace z_1 and z_2 with $z_1' = \min(1, \max(0, z_1))$ and $z_2' = \min(1, \max(0, z_2))$ and compute the length by:

$$\text{len}(l, A_i) = ||(z_2' - z_1')(y-x)||_2$$

Again, it only requires $\mathcal{O}(1)$ time to test if a node lies within a line segment area. Computing and summing up all intersection lengths takes $\mathcal{O}(N_j)$ time.

Hippodrome

Suppose we have a hippodrome area A_i with endpoints $a = (a_1, a_2)^T$ and $b = (b_1, b_2)^T$ and radius r. We assume $a \neq b$, as otherwise the hippodrome is a disk and we can utilize our methods for the circle area. Additionally, we assume $r > 0$, because if $r = 0$ the hippodrome would be a line segment.

A point $p \in \mathbb{R}^2$ lies in A_i if and only if the distance between p and the line segment between a and b is less than or equal to r.

Computing the intersection length of a link $e_j \in E$ and A_i is slightly more complicated. As before, we compute the total intersection length by summing the intersection lengths of all line segments l of e_j.

Let $x = (x_1, x_2)^T$ and $y = (y_1, y_2)^T$ $(x \neq y)$ be the endpoints of a line segment l of link e_j.

We first search for $z \in \mathbb{R}$ such that the distance between $x + z(y-x)$ and the line through a and b is exactly r, that is

$$\frac{\left|(b_1 - a_1)(x_2 + z(y_2 - x_2) - a_2) - (b_2 - a_2)(x_1 + z(y_1 - x_1) - a_1)\right|}{||b-a||_2} = r$$

This can be rewritten as

$$\left|(b_1 - a_1)(x_2 - a_2) - (b_2 - a_2)(x_1 - a_1) + ((b_1 - a_1)(y_2 - x_2) - (b_2 - a_2)(y_1 - x_1))z\right| = r||b-a||_2$$

If the lines are not parallel to each other, there are two possible solutions:

$$z = \frac{(b_2 - a_2)(x_1 - a_1) - (b_1 - a_1)(x_2 - a_2) \pm r||b-a||_2}{(b_1 - a_1)(y_2 - x_2) - (b_2 - a_2)(y_1 - x_1)} \tag{9.6}$$

Assuming the lines are not parallel, let z_1 and z_2 be our two solutions. First check if the distance between $x+z_1(y-x), x+z_2(y-x)$ and the line segment are larger than r. If the distance between one (or both) of these points is larger than r, l does not intersect the hippodrome boundary at this point, but at one of the circles around a or b. If this is the case, simply replace z_1 and/or z_2 using equation 9.4.

If the lines are parallel, the line through x and y can only intersect the hippodrome boundary at the circles through the endpoints, so, if they exist, z_1 and z_2 can be found using equation 9.4. If they do not exist, the intersection length of l and the hippodrome is 0.

We have found the intersection points between the line through x and y and the hippodrome boundary. Similarly as before, we can replace these with $z_1' = \min(1, \max(0, z_1))$ and $z_2' = \min(1, \max(0, z_2))$ and compute the intersection length with equation 9.5.

Testing if a node lies in the hippodrome area takes $\mathcal{O}(1)$ time and computing the total intersection length of a link and the area takes $\mathcal{O}(N_j)$ time.

Simple Polygon

Let A_i be a simply polygon area.

As described in section 4.1.1, one can test if a point p lies within a simple polygon by counting the number of times a line from p to outside the polygon intersects the polygon sides.

As before, we compute the length of the intersection between $e_j \in E$ and a simple polygon area by summing the intersection lengths of all its line segments.

Let $x = (x_1, x_2)^T$ and $y = (y_1, y_2)^T$ $(x \neq y)$ be the endpoints of a line segment l of link e. We can find all intersections between the line though x and y and the polygon sides by utilizing equation 9.6 (setting $r = 0$). Let $Z := \{\min(1, \max(0, z)) | x + z(y-x) \text{ lies on a polygon side}\}$. Sort Z in ascending order, such that $z_1, z_2, \ldots, z_n \in Z$ and $z_1 \leq z_2 \leq \cdots \leq z_n$. If $|Z| \leq 1$ the intersection length of l and the polygon is 0. If $|Z| > 1$, we can compute the intersection length as follows:

$$\text{len}(l) = \sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} ||(z_{2k} - z_{2k-1})(y-x)||_2 \tag{9.7}$$

Remember that $N(A_i)$ is the number of sides of simple polygon area A_i. It takes $\mathcal{O}(N(A_i))$ time to test if a node lies in A_i and $\mathcal{O}(N_j N(A_i))$ time to compute the total intersection length of a link $e_j \in E$ and the simple polygon area A_i.

Union of Areas

Suppose A_i is a finite union of circle, line segment and hippodrome areas, i.e. $A_i = A_{i,1} \cup A_{i,2} \cup \cdots \cup A_{i,n}$ with $A_{i,1}, A_{i,2}, \ldots, A_{i,n}$ disks, line segments or hippodromes.

To test if a point p lies in A_i, we can simple iterate over each area $A_{i,k}$ and test if $p \in A_{i,k}$. If the point lies in one of the areas, it lies in A_i.

If all $A_{i,k}$ are mutually disjoint, computing $\text{len}(e_j, A_i)$ is rather simple:

$$\text{len}(e_j, A_i) = \sum_{k=1}^{n} \text{len}(e_j, A_{i,k})$$

However, we do not require all $A_{i,k}$ to be mutually disjoint, so there is a possibility that $\sum_{k=1}^{n} \text{len}(e_j, A_{i,k}) > \text{len}(e_j, A_i)$.

Instead, we will sum the intersection lengths of A_i and every line segment of e_j again.

To compute the intersection length $\text{len}(l, A_i)$ of a line segment l between $x \in \mathbb{R}^2$ and $y \in \mathbb{R}^2$ and A_i, we first compute the intersections between l and all $A_{i,k}$. Every intersection will consist of one or more segments of the line through x and y. Next, we combine these segments into one large intersection of line segments, removing any overlapping parts. Finally, $\text{len}(l, A_i)$ can be computed by summing the lengths of all line segments of the intersection between l and A_i.

Similar to section 4.1.1, let $N(A_i) = \sum_{k=1}^{n} N(A_{i,k})$ (if A_i is a union of areas), where $N(A_{i,k}) = 1$ if $A_{i,k}$ is a circle, line segment or hippodrome. Testing if a node lies in a union of areas A_i takes $\mathcal{O}(N(A_i))$ time and computing $\text{len}(e_j, A_i)$ takes $\mathcal{O}(N_j N(A_i))$ time.

Difference of Areas

Let A_i b the difference of two areas, i.e. $A_i = A_{i,1} \setminus A_{i,2}$.

In this case, $p \in \mathbb{R}$ lies in A_i if and only if $p \in A_{i,1} \wedge p \notin A_{i,2}$.

Computing $\operatorname{len}(e_j, A_i)$ is also quite simple:

$$\operatorname{len}(e_j, A_i) = \operatorname{len}(e_j, A_{i,1} \cup A_{i,2}) - \operatorname{len}(e_j, A_{i,2}) \tag{9.8}$$

Testing if a link $e_j \in E$ intersects A_i is slightly more complicated, as $\operatorname{len}(e_j, A_i) = 0$ does not necessarily mean that e_j and A_i do not intersect. One method is to compute all intersections of the line segments of e_j and $A_{i,1}$ and e_j and $A_{i,2}$. If we remove all points and line segments that make up the intersection of e_j and $A_{i,2}$ from the intersection of e_j and $A_{i,1}$ and the result still contains some points or line segments, then e_j intersects A_i. Otherwise, e_j does not intersect A_i.

Let $N(A_i) = N(A_{i,1}) + N(A_{i,2})$. It takes $\mathcal{O}(N(A_i))$ time to test if a node lies in A_i, $\mathcal{O}(N_j N(A_i))$ to compute the intersection length of e_j and A_i and $\mathcal{O}(N_j N(A_i))$ time to test if e_j intersects A_i.

Redefine $N(d)$ as $N(d) := \sum_{A_i \in A(d)} N(A_i)$.

We can compute $P(v_j \text{ fails}|d)$ in $\mathcal{O}(N(d))$ time and compute $P(e_j \text{ fails}|d)$ in $\mathcal{O}(N_j N(d))$ time[1]. So the total time required to compute all node and link failure probabilities given the occurrence of a specific disaster is $\mathcal{O}((|V| + N_e)N(d))$ and the time required to compute these values for all disasters is $\mathcal{O}((|V| + N_e)N(D))$.

9.2.5. Algorithms

We propose two general-purpose algorithms for the Probabilistic Vulnerability Distribution Problem. The first algorithm computes the vulnerability distribution exactly, but runs in exponential time in $|V|$ and $|E|$. The second is a randomized approximation algorithm, which can compute the ϵ-approximation of the ATTR vulnerability distribution in polynomial time.

For all $c \subseteq V \cup E$, define $S(c)$ as the failure state of the network after all nodes and links in c fail. Thus $S(c) = (E \cap c) \cup \{(v_j, v_k) \in E | v_j \in c \vee v_k \in c\}$. Let $M(c) := M(S(c))$.

Exact Algorithm

We give an exact algorithm for the Probabilistic Network Distribution Problem, which can then be adapted for the Probabilistic Vulnerability Distribution Problem (as in the proof of theorem 8).

The algorithm simply considers all possible combinations of failures to compute the vulnerability distribution:

1. let $C = \{e \in E | P_l(e) > 0\} \cup \{v \in V | P_n(v) > 0\}$
2. $\forall c \subseteq C$, compute $M(c)$ and $P(c) = \prod_{e \in c \cap E} P_l(e) \prod_{v \in c \cap V} P_n(v)$
3. $\forall m \in M[C]$, store $\{c \in C | M(c) = m\}$
4. $P(M = m) = \sum_{c \in C | M(c) = m} P(c)$

Ignoring the time it takes to compute $M(c)$, which depends on the measure, the time complexity of this algorithm is $\mathcal{O}(2^{|E|+|V|}(|E| + |V|))$. So the related algorithm for the Probabilistic Vulnerability Distribution Problem has time complexity $\mathcal{O}(|D| 2^{|E|+|V|}(|E| + |V|) + (|V| + N_e)N(D))$.

Computing the WATTR vulnerability distribution would take $\mathcal{O}(|V|^2 + |D| 2^{|E|+|V|}|E||V| + (|V| + N_e)N(D))$ time.

Monte Carlo

Similar as our Monte Carlo approach to constructing disaster lists, we can also approximate the vulnerability distribution with the Monte Carlo method.

First compute $P(v \text{ fails}|d)$ and $P(e \text{ fails}|d)$ for all $d \in D$ and for all $v \in V$ and $e \in E$.

Then perform the following experiment N times:

1. Sample a random disaster $d \in D$, where $P(d \text{ sampled}) = P(d)$
2. Iterate over each node $v \in V$, and let $v \in c$ with probability $P(v \text{ fails}|d)$

[1]Assuming some fixed upper-bounds on link segment intersection lengths.

3. Iterate over each link $e \in E$, and let $e \in c$ with probability $P(e \text{ fails}|d)$
4. Compute the resulting value of the measure: $M(c)$

Let X_i be the result of experiment i.

Note that $P(X_i = m) = P(M = m)\ \forall m \in \mathbb{R} \forall 0 < i < N$. Let $Y(m) := |\{0 < i \leq N | X_i = m\}|$. We approximate the vulnerability distribution by $\hat{P}(M = m) = \frac{Y(m)}{N}$.

Theorem 12. *Let out(G) indicate the amount of possible distinct values of M, given network G. Given an* $\epsilon > 0$ *and* $0 < \delta \leq 1$.
If

$$N \geq out(G)\frac{0.25}{\delta\epsilon^2}$$

then

$$P(\exists m \in \mathbb{R}\ s.t. |\hat{P}(M = m) - P(M = m)| > \epsilon) \leq \delta$$

Proof. Note that $E(\hat{P}(M = m)) = E(\frac{Y(m)}{N}) = P(M = m)$ and $\text{Var}(\hat{P}(M = m)) = \text{Var}(\frac{Y(m)}{N}) = \frac{p(1-p)}{N}$.
Consider a single $m \in \mathbb{R}$.

$$P(|\frac{Y(m)}{N} - P(M = m)| \geq \epsilon) \leq \frac{p(1-p)}{N\epsilon^2} \text{ (Chebyshev's Inequality)}$$

So

$$P(\exists m \in \mathbb{R} \text{ s.t.} |\hat{P}(M = m) - P(M = m)| > \epsilon) \leq \text{out}(G)\frac{0.25}{N\epsilon^2}$$

So if $N \geq \text{out}(G)\frac{0.25}{\delta\epsilon^2}$, then $(\exists m \in \mathbb{R} \text{ s.t.} |\hat{P}(M = m) - P(M = m)| > \epsilon) \leq \delta$ □

The amount of possible ATTR values is equal to the amount of node pairs in the network ($\frac{|V|(|V|-1)}{2}$), so only $\frac{|V|(|V|-1)}{2}(G)\frac{0.25}{\delta\epsilon^2}$ experiments are required to make sure that

$$P(\exists m \in \mathbb{R} \text{ s.t.} |\hat{P}(\text{ATTR} = m) - P(\text{ATTR} = m)| > \epsilon) \leq \delta$$

Computing all device failure probabilities for all disasters takes $\mathcal{O}((|V| + N_e)N(D))$ time. And a single experiment, computing ATTR, takes $\mathcal{O}(|D| + |V||E|)$ (with a one time cost of $\mathcal{O}(|V|^2)$, see section 4.2.1) time.

So approximating the ATTR vulnerability distribution with probability $\leq \delta$ to exceed an absolute error of ϵ takes $\mathcal{O}((|V| + N_e)N(D) + \delta^{-1}\epsilon^{-2}(|V|^2|D| + |V|^3|E|))$ time using this randomized algorithm.

10 Conclusion

We proposed computing the vulnerability of a network topology to a group of large-scale disasters by taking a finite set of representative disasters and computing the distribution of a measure (e.g. the average 2-terminal reliability) given that exactly one of these disasters randomly occurs.

Compared to only computing one value related to the vulnerability of a network, for example the expected value of a measure, our approach gives much more information.

In addition, by taking a finite set of disasters we can model disaster locations and areas more accurately than by assuming a fixed disaster shape randomly placed in the network area.

We introduced methods to efficiently compute the ‘vulnerability distribution’ and have experimentally shown these methods to efficiently scale to large disaster sets and detailed networks. In addition, these methods can be very efficiently parallelized. In our experiments, doubling the number of threads on which the distribution is computed almost halved the execution time.

We have given some general approaches to obtaining a finite set of representative disasters, and have demonstrated these approaches by generating disaster sets based on the empirical data of three organizations: the NIED, the USGS and the NHC. In addition, we found that all three of these institutions employ a finite representative disaster set approach for at least part of their hazard analysis.

Additionally, we have shown how the vulnerability of a network can be visualized to a network designer or operator.

We have demonstrated our methods by analyzing the vulnerability of Japanese and American network topologies to earthquakes and hurricanes.

We proposed an extension of our model for probabilistic disasters and showed that even in a simple probabilistic disaster model computing the (W)ATTR vulnerability distribution becomes an NP-hard problem. We introduced the Probabilistic Vulnerability Distribution Problem based on a more extensive probabilistic model and gave both an exact algorithm and a randomized approximation algorithm for this problem.

Bibliography

[1] Standard grid square and grid square code used for the statistics (announcement no. 143 by the administrative management agency on july, 12, 1973). www.stat.go.jp/english/data/mesh/02.htm. Accessed: 07-05-2017.

[2] Earthquakes and plate tectonics faqs - what is an earthquake and what causes them to happen? www2.usgs.gov/faq/categories/9827/3343,. Accessed: 07-05-2017.

[3] Interactive fault map. earthquake.usgs.gov/hazards/qfaults/map/,. Accessed: 05-06-2017.

[4] Earthquake hazards 101 - the basics. earthquake.usgs.gov/hazards/learn/basics.php. Accessed: 07-05-2017.

[5] Tropical cyclone climatology. www.nhc.noaa.gov/climo/. Accessed: 07-06-2017.

[6] J-shis. www.j-shis.bosai.go.jp/en/,.

[7] Glossary. www.j-shis.bosai.go.jp/en/glossary,. Accessed: 07-05-2017.

[8] About the national hurricane center. www.nhc.noaa.gov/aboutintro.shtml,. Accessed: 07-06-2017.

[9] Glossary of nhc terms. www.nhc.noaa.gov/aboutgloss.shtml,. Accessed: 07-06-2017.

[10] Nhc track and intensity models. www.nhc.noaa.gov/modelsummary.shtml,. Accessed: 07-06-2017.

[11] 1811-1812 new madrid, missouri earthquakes. earthquake.usgs.gov/earthquakes/events/1811-1812newmadrid/. Accessed: 05-06-2017.

[12] The great 1906 san francisco earthquake. earthquake.usgs.gov/earthquakes/events/1906calif/18april/index.php. Accessed: 05-06-2017.

[13] Usgs. www.usgs.gov/,.

[14] Usgs catalog. earthquake.usgs.gov/earthquakes/search/,.

[15] Nasa world wind. worldwind.arc.nasa.gov.

[16] Special report - the ntt group's response to the great east japan earthquake. www.ntt.co.jp/csr_e/2011report/pdf/en_05-12.pdf, 2011. Accessed: 08-05-2017.

[17] Japan seismic hazard information station (j-shis) file format specification, July 2016.

[18] Succesvolle overdracht operationeel netwerkbeheer. www.surf.nl/nieuws/2016/06/succesvolle-overdracht-operationeel-netwerkbeheer.html, June 2016.

[19] Pankaj K Agarwal, Alon Efrat, Shashidhara K Ganjugunte, David Hay, Swaminathan Sankararaman, and Gil Zussman. The resilience of wdm networks to probabilistic geographical failures. *IEEE/ACM Transactions on Networking (TON)*, 21(5):1525–1538, 2013.

[20] Pankaj K Agarwal, Sariel Har-Peled, Haim Kaplan, and Micha Sharir. Union of random minkowski sums and network vulnerability analysis. *Discrete & Computational Geometry*, 52(3):551–582, 2014.

[21] Sujogya Banerjee, Shahrzad Shirazipourazad, Pavel Ghosh, and Arunabha Sen. Beyond connectivity-new metrics to evaluate robustness of networks. In *High Performance Switching and Routing (HPSR), 2011 IEEE 12th International Conference on*, pages 171–177. IEEE, 2011.

[22] Sujogya Banerjee, Shahrzad Shirazipourazad, and Arunabha Sen. Design and analysis of networks with large components in presence of region-based faults. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–6. IEEE, 2011.

[23] Yehiel Berezin, Amir Bashan, Michael M Danziger, Daqing Li, and Shlomo Havlin. Localized attacks on spatially embedded networks with dependencies. *Scientific reports*, 5:8934, 2015.

[24] Dan Bienstock. Some generalized max-flow min-cut problems in the plane. *Mathematics of Operations Research*, 16(2):310–333, 1991.

[25] GA Bollinger. Historical and recent seismic activity in south carolina. *Bulletin of the Seismological Society of America*, 62(3):851–864, 1972.

[26] Egemen K. Çetinkaya, Dan Broyles, Amit Dandekar, Sripriya Srinivasan, and James P. G. Sterbenz. Modelling communication network challenges for future internet resilience, survivability, and disruption tolerance: a simulation-based approach. *Telecommunication Systems*, 52(2):751–766, 2013. ISSN 1572-9451. doi: 10.1007/s11235-011-9575-4. URL `http://dx.doi.org/10.1007/s11235-011-9575-4`.

[27] Sean X Chen and Jun S Liu. Statistical applications of the poisson-binomial and conditional bernoulli distributions. *Statistica Sinica*, pages 875–892, 1997.

[28] Ferhat Dikbiyik, Abu S Reaz, Marc De Leenheer, and Biswanath Mukherjee. Minimizing the disaster risk in optical telecom networks. In *Optical Fiber Communication Conference*, pages OTh4B–2. Optical Society of America, 2012.

[29] Ferhat Dikbiyik, Massimo Tornatore, and Biswanath Mukherjee. Minimizing the risk from disaster failures in optical backbone networks. *Journal of Lightwave Technology*, 32(18):3175–3183, 2014.

[30] M. Fernandez and S. Williams. Closed-form expression for the poisson-binomial probability density function. *IEEE Transactions on Aerospace and Electronic Systems*, 46(2):803–817, April 2010. ISSN 0018-9251. doi: 10.1109/TAES.2010.5461658.

[31] T Feyessa and M Bikdash. Geographically-sensitive network centrality and survivability assessment. In *System Theory (SSST), 2011 IEEE 43rd Southeastern Symposium on*, pages 18–23. IEEE, 2011.

[32] Hiroyuki FUJIWARA, Shinichi KAWAI, Nobuyuki MORIKAWA Shin AOI, Shigeki SENNA, Nobuaki KUDO, Masahiro OOI, Ken Xiansheng HAO, Kazue WAKAMATSU, Yutaka ISHIKAWA, Toshihiko OKUMURA, Toru ISHII, Shinichi MATSUSHIMA, Yuzuru HAYAKAWA, Nobuhiko TOYAMA, and Akira NARITA. Technical reports on national seismic hazard maps for japan. Technical Report 336, NIED, November 2009.

[33] M Todd Gardner and Cory Beard. Evaluating geographic vulnerabilities in networks. In *Communications Quality and Reliability (CQR), 2011 IEEE International Workshop Technical Committee on*, pages 1–6. IEEE, 2011.

[34] M Todd Gardner, Cory Beard, and Deep Medhi. Avoiding high impacts of geospatial events in mission critical and emergency networks using linear and swarm optimization. In *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2012 IEEE International Multi-Disciplinary Conference on*, pages 264–271. IEEE, 2012.

[35] M Todd Gardner, Rebecca May, Cory Beard, and Deep Medhi. Determining geographic vulnerabilities using a novel impact based resilience metric. *Journal of Network and Systems Management*, 24(3):711–745, 2016.

[36] Omer Gold and Reuven Cohen. Coping with physical attacks on random network structures. In *Communications (ICC), 2014 IEEE International Conference on*, pages 1166–1172. IEEE, 2014.

[37] Teresa Gomes, János Tapolcai, Christian Esposito, David Hutchison, Fernando Kuipers, Jacek Rak, Amaro de Sousa, Athanasios Iossifides, Rui Travanca, Joao André, et al. A survey of strategies for communication networks to protect against large-scale natural disasters. In *Resilient Networks Design and Modeling (RNDM), 2016 8th International Workshop on*, pages 11–22. IEEE, 2016.

[38] Sadaki HORI. From the director. `www.bosai.go.jp/e/research/the_second/earthquake/director.html`. Accessed: 07-05-2017.

[39] Madan Kumar Jha. Natural and anthropogenic disasters: An overview. In *Natural and Anthropogenic Disasters*, pages 1–16. Springer, 2010.

[40] Motoki Kazama and Toshihiro Noda. Damage statistics (summary of the 2011 off the pacific coast of tohoku earthquake damage). *Soils and Foundations*, 52(5):780–792, 2012.

[41] Yasuichi Kitamura, LEE Youngseok, Ryo Sakiyama, and Koji Okamura. Experience with restoration of asia pacific network failures from taiwan earthquake. *IEICE transactions on communications*, 90(11): 3095–3103, 2007.

[42] S. Knight, H.X. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *Selected Areas in Communications, IEEE Journal on*, 29(9):1765 –1775, october 2011. ISSN 0733-8716. doi: 10.1109/JSAC.2011.111002.

[43] Yusuke Kobayashi and Kensuke Otsuki. Max-flow min-cut theorem and faster algorithms in a circular disk failure model. In *INFOCOM, 2014 Proceedings IEEE*, pages 1635–1643. IEEE, 2014.

[44] Soung C Liew and Kevin W Lu. A framework for characterizing disaster-based network survivability. *IEEE Journal on Selected Areas in Communications*, 12(1):52–58, 1994.

[45] Xuelian Long, David Tipper, and Teresa Gomes. Measuring the survivability of networks to geographic correlated failures. *Optical Switching and Networking*, 14:117–133, 2014.

[46] Lisheng Ma, Xiaohong Jiang, Bin Wu, Achille Pattavina, and Norio Shiratori. Probabilistic region failure-aware data center network and content placement. *Computer Networks*, 103:56–66, 2016.

[47] Denise MB Masi, Eric E Smith, and Martin J Fischer. Understanding and mitigating catastrophic disruption and attack. *Sigma Journal*, pages 16–22, 2010.

[48] Dawson Ladislaus Msongaleli, Ferhat Dikbiyik, Moshe Zukerman, and Biswanath Mukherjee. Disaster-aware submarine fiber-optic cable deployment for mesh networks. *Journal of Lightwave Technology*, 34 (18):4293–4303, 2016.

[49] *National Hurricane Center Product Description Document: A User's Guide to Hurricane Products*. National Hurricane Center, June 2017.

[50] Sebastian Neumayer and Eytan Modiano. Assessing the effect of geographically correlated failures on interconnected power-communication networks. In *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, pages 366–371. IEEE, 2013.

[51] Sebastian Neumayer and Eytan Modiano. Network reliability under geographically correlated line and disk failure models. *Computer Networks*, 94:14 – 28, 2016. ISSN 1389-1286. doi: http://dx.doi.org/10.1016/j.comnet.2015.11.025. URL `http://www.sciencedirect.com/science/article/pii/S1389128615004740`.

[52] Sebastian Neumayer, Gil Zussman, Reuven Cohen, and Eytan Modiano. Assessing the vulnerability of the fiber infrastructure to disasters. *IEEE/ACM Transactions on Networking (TON)*, 19(6):1610–1623, 2011.

[53] Sebastian Neumayer, Alon Efrat, and Eytan Modiano. Geographic max-flow and min-cut under a circular disk failure model. *Computer Networks*, 77:117 – 127, 2015. ISSN 1389-1286. doi: http://dx.doi.org/10.1016/j.comnet.2014.10.026. URL `http://www.sciencedirect.com/science/article/pii/S1389128614003880`.

[54] Kensuke Otsuki, Yusuke Kobayashi, and Kazuo Murota. Improved max-flow min-cut algorithms in a circular disk failure model with application to a road network. *European Journal of Operational Research*, 248(2):396 – 403, 2016. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2015.07.035. URL `http://www.sciencedirect.com/science/article/pii/S0377221715006694`.

[55] J Scott Provan and Michael O Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4):777–788, 1983.

[56] Mahshid Rahnamay-Naeini, Jorge E Pezoa, Ghady Azar, Nasir Ghani, and Majeed M Hayat. Modeling stochastic correlated failures and their effects on network reliability. In *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, pages 1–6. IEEE, 2011.

[57] Eva Regnier. Public evacuation decisions and hurricane track uncertainty. *Management Science*, 54(1): 16–28, 2008. doi: 10.1287/mnsc.1070.0764. URL `https://doi.org/10.1287/mnsc.1070.0764`.

[58] Hiroshi Saito. Geometric evaluation of survivability of disaster-affected network with probabilistic failure. In *INFOCOM, 2014 Proceedings IEEE*, pages 1608–1616. IEEE, 2014.

[59] Hiroshi Saito. Analysis of geometric disaster evaluation model for physical networks. *IEEE/ACM Transactions on Networking*, 23(6):1777–1789, 2015.

[60] Hiroshi Saito. Spatial design of physical network robust against earthquakes. *Journal of Lightwave Technology*, 33(2):443–458, 2015.

[61] Kelly M Sullivan and J Cole Smith. Exact algorithms for solving a euclidean maximum flow network interdiction problem. *Networks*, 64(2):109–124, 2014.

[62] Stojan Trajanovski, Fernando A Kuipers, Aleksandar Ilić, Jon Crowcroft, and Piet Van Mieghem. Finding critical regions and region-disjoint paths in a network. *IEEE/ACM Transactions on Networking (TON)*, 23 (3):908–921, 2015.

[63] Phuong Nga Tran and Hiroshi Saito. Enhancing physical network robustness against earthquake disasters with additional links. *Journal of Lightwave Technology*, 34(22):5226–5238, 2016.

[64] Xiaoliang Wang, Xiaohong Jiang, and Achille Pattavina. Assessing network vulnerability under probabilistic region failure model. In *High Performance Switching and Routing (HPSR), 2011 IEEE 12th International Conference on*, pages 164–170. IEEE, 2011.

[65] Xiaoliang Wang, Xiaohong Jiang, Achille Pattavina, and Sanglu Lu. Assessing physical network vulnerability under random line-segment failure model. In *High Performance Switching and Routing (HPSR), 2012 IEEE 13th International Conference on*, pages 121–126. IEEE, 2012.

[66] CB Worden and DJ Wald. Shakemap manual online: technical manual, user's guide, and software guide. online. *URL< http://usgs. github. io/shakemap>. http://dx. doi. org/10.1234/012345678*, 2016.

[67] Xiao Zhang, Xiaoliang Wang, Xiaohong Jiang, and Sanglu Lu. Degree of network damage: A measurement for intensity of network damage. In *Networks and Optical Communications-(NOC), 2014 19th European Conference on*, pages 140–146. IEEE, 2014.