Erasmus University Rotterdam

Bachelor Thesis

Minimizing the spread of infections in local and metapopulation networks by removing a set of links

Author:
Aniek Markus

Student number:
383550

Supervisor:
Evelot Duijzer

Second assessor:
Prof. dr. Albert Wagelmans

Double degree programme Econometrics & Economics

Erasmus School of Economics

July 2, 2017

Abstract

This thesis evaluates the performance of different link removal methods to minimize the spread of infections in local and metapopulation networks. The aim is to replicate the results of Nandi and Medal (2016) and to extend the analysis to a metapopulation network. For the local network, we study generated scale-free networks and for the metapopulation network we study a global airline network. We examine the link removal methods *MinConnect*, *MinAtRisk*, *MinPaths* and *MinWPaths* proposed by Nandi and Medal (2016) and compare these with the *Random* method. As these methods use indirect metrics, the spread of infection is measured using simulation. We find that *MinConnect* is in general most effective to maximize the time to infect half of the susceptible nodes in a scale-free network, whereas *MinAtRisk* is the best method to minimize the number of new infections. Most results are in line with the findings of Nandi and Medal (2016). One exception is that *MinWPaths* performs worse than expected in slowing down the speed of spread. For the global airline network, *MinWPaths* performs best. We conclude that the performance of the link removal methods strongly depends on the network structure, the type of infection and some other factors.

Contents

1 Introduction

The spread of undesirable things such as infectious diseases, computer viruses and rumors is a common problem but difficult to control in practice. For example, a recent report of the World Health Organization (2016) showed the spread of the well-known infectious disease HIV/AIDS is still not under control. Although the number of people dying due to AIDS decreased with 45% over the last ten years, there are more people living with HIV nowadays and there are still two million new infections per year. Another example is the constant threat of computer viruses as we are more and more connected because of the Internet. Finally, the recent shooting at a pizzeria in Washington as a result of fake news, is an example that shows the dramatic consequences the spread of rumors can have (Kang & Goldman, 2016).

This thesis addresses the problem of minimizing the spread of infections by removing a set of links. Many studies investigate this issue in local networks. However, many local networks are part of larger networks or *metapopulations*. A metapopulation is a population consisting of multiple subpopulations with similar dynamics. This thesis evaluates the performance of different link removal methods (1) in local networks, where every node represents e.g. one person, computer or social media account and in (2) metapopulations, where every node represents a local network (see Figure 16). In particular, the goal of this thesis is to replicate the results of Nandi and Medal (2016) and to extend the analysis to a metapopulation network. In particular, this thesis evaluates the spread of a disease through a global airline network. In this case intervening in a local network might not be possible (e.g. by blocking interaction between people a city), but it is possible to intervene on a larger scale (e.g. by canceling flights to prevent the spread between cities).

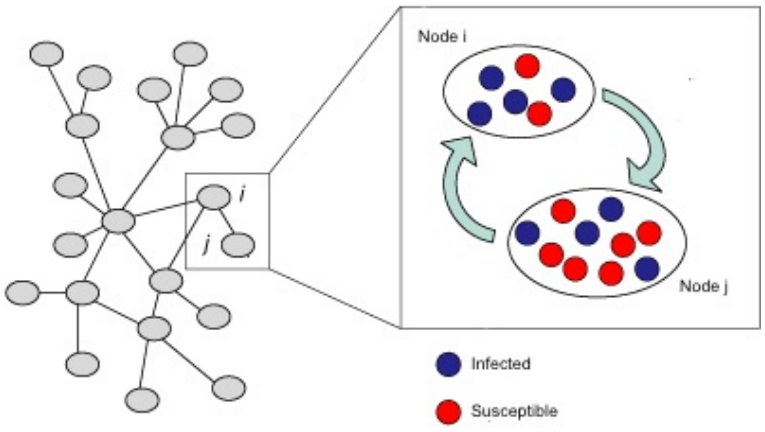


Figure 1: Schematic view of metapopulation network.

The topic studied in this thesis is very important as the spread of infections can result in large social and economical damage. It is of both scientific and societal relevance. From a scientific point of view, this thesis contributes to the existing knowledge on the performance of different link removal methods. Furthermore, a new simulation model to evaluate the spread of infections in a metapopulation network is presented. Moreover, it is an interesting issue from a policy perspective because it provides insight in how to use scarce resources most efficiently.

The remainder of this thesis proceeds as follows. In Section 2 the existing literature on reducing the spread of infections in networks is discussed. The problem formulation is presented in Section 3. Section 4 describes the link removal methods and the methodology used to analyze this problem. The results are summarized in Section 5. Section 6 discusses the findings and we conclude in Section 7.

2 Literature

To provide a clear overview of the existing literature, the related work is evaluated as follows. First, the context of this research is discussed in Section 2.1. Next, two streams in the literature minimizing the spread of infections using network theory are distinguished in Section 2.2: node removal versus link removal methods. Finally, the latter one will be discussed in more detail in Section 2.3.

2.1 Context

The study of the spread of infections in real life networks is a relatively new field of research. However, the study of graphs goes back to the 18th century, when Leonhard Euler published his work that is nowadays known as the problem of the seven bridges of Königsberg (Grimaldi, 2004, p. 109). For a long time graph theory focused purely on generated graphs because not much was known about *complex real life networks*. First, graph theory studied only regular graphs, where each node has the same number of neighbors. Since the 1950's, random graphs were also used to model large-scale networks. In a random graph as defined by Erdös and Rényi the nodes are connected by a random set of edges (Albert & Barabási, 2002). Although random graphs are useful and interesting for theoretical purposes, it seems unlikely that complex real life systems such as the network of chemicals in a cell or the Internet are truly random.

The recent increase in data availability on real life networks, the rapid advancement of computing power and the growing cooperation between different disciplines all contributed to a fast development of *complex network theory* (Albert & Barabási, 2002). This research led to three important observations. First of all, most large-scale networks are *small-worlds*, which means that most nodes are only a small distance away from each other. This is in line with the idea of six degrees of separation first mentioned by writer Frigyes Karinthy in 1929 and later shown by social psychologist Stanley Milgram in 1967. Secondly, most real life networks are more clustered than random graphs suggest as people tend to form groups. Last of all, empirical research shows not all nodes have the same number of links. Instead, it is shown that in most networks a large number of nodes have a small number of links and only a few nodes have a large number of links (Enns, Mounzer, & Brandeau, 2012). Networks with this property are called *scale-free* networks and have a *power law degree distribution* instead of a Poisson distribution for the node degree. In Figure 2 regular, random, scale-free and small-world graphs are shown.

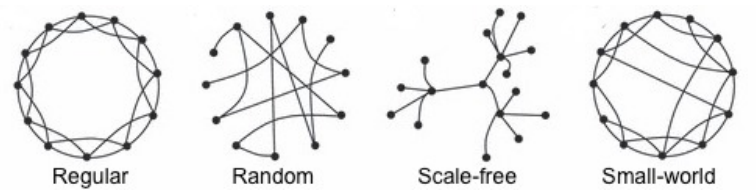


Figure 2: Examples of graphs.

The maturation of complex network theory allowed researchers to study the spread of infections in complex real life networks (Yang, Wu, & Wang, 2013). This resulted in a large increase in the number of researchers studying the spread of infections using network theory in the past decade.

Earlier research into epidemics used *homogeneous-mixing compartmental models* based on the assumption that all individuals have an equal probability to be in contact with each other (Kermack & McKendrick, 1927). Although the models were later extended to incorporate some heterogeneity between individuals, network-based approaches to the spread of infections are more advanced and allow to incorporate contact behavior of people explicitly. However, homogeneous-mixing compartmental models can sometimes be appropriate as Bansal, Grenfell, and Meyers (2007) found that contact patterns among people are not as diverse as expected. Nevertheless, they conclude that using the network-based approach is more intuitive and accurate.

There are a few related problems to minimizing the spread of infections in a network. One is the problem known as the *critical node detection problem* aimed at maximizing the fragmentation of a network or similarly, minimizing the connectivity (Di Summa, Grosso, & Locatelli, 2012; Addis, Di Summa, & Grosso, 2013; Veremyev, Prokopyev, & Pasiliao, 2014). The difference with the problem studied in this paper is the assumption that it is unknown which nodes are infected or susceptible. The critical node detection problem thus focuses on prevention instead of reaction, which makes it more difficult to tackle the spread of infections (Nandi & Medal, 2016). As the methods developed for the critical node detection problem do not use information about the state of the nodes, they will be less effective when used as reactive approach. Network vulnerability and robustness analysis is also a related problem as it is the opposite of the critical node detection problem. Finally, one can also aim to maximize the spread of infections in a network instead of minimizing the spread. This leads to the third related problem, the *influence maximization problem*, where the most influential nodes are searched for initial activation (Domingos & Richardson, 2001; Kempe, Kleinberg, & Tardos, 2003; Chen, Wang, & Yang, 2009). Such a model can be useful for marketing purposes, for example to analyze which individuals to target for a new product line or to find the most influential users on social media for the spread of new ideas.

2.2 Node removal versus link removal methods

In literature, two methods to minimize the spread of infections in a network can be distinguished: node removal and link removal. The latter one is more subtle, as removing all links belonging to one node is similar to removing the node, but not all links have to be removed necessarily in this case (Nandi & Medal, 2016). The link removal method is thus more flexible, and hence expected to be more efficient. Marcelino and Kaiser (2012) for example found in their research that removing connections between certain cities was more effective than closing entire airports to prevent the spread of influenza. However, there might be situations where either node removal or link removal is not possible and the right approach has to be chosen accordingly. For example, it is possible to vaccinate individuals but preventing contact between individuals is not feasible. In this case node removal is the right approach. On the other hand, it might not be possible to determine the location of terrorists in order to eliminate the terrorist network, but it might be possible to block communication between them. This requires a link removal method.

Many researchers studied the node removal problem in the last decade. Callaway, Newman, Strogatz, and Watts (2000) study the deletion of nodes in networks with different node degree distributions and found that networks with a power law degree distribution are robust against random removal of nodes, but vulnerable for targeted removal of the most connected nodes. Newman, Forrest, and Balthrop (2002) examine the spread of computer viruses and evaluate different strategies to minimize the impact of the viruses. They found that removing nodes with the highest degree works well in undirected graphs, which is in line with the findings of Callaway et al. (2000). Moreover, Newman et al. (2002) conclude removing nodes in decreasing order of out-degree can be effective for the case of a directed graph. Interesting applications include the work of Latora and Marchiori (2004) and Brown, Carlyle, Salmerón, and Wood (2006) who study this problem in relation to terrorist attacks, and Boginski and Commander (2009) who analyze protein interaction models using techniques developed for the critical node detection problem. Node removal methods, such as removing nodes in decreasing order of (out-)degree, can thus be very useful in specific situations where link removal methods are not possible.

The link removal problem is also studied extensively. Many researchers proposed new link removal methods, among who Kimura, Saito, and Motoda (2009), Enns et al. (2012) and Tong, Prakash, Eliassi-Rad, Faloutsos, and Faloutsos (2012). They all find their algorithms are better than several existing link removal methods. Kimura et al. (2009) even find their method is 20 to 60 times more effective. Chung, Chew, Zhou, and Lai (2012) find it is more cost effective to remove less busy flight connections than to remove popular flight connections. Yang et al. (2013) analyze the spread of an epidemic using package exchanges to provide insight in traffic dynamics and conclude that removing edges following the out-degree method is more effective than the edge-betweenness method for a *traffic-driven epidemic*. Koch, Illner, and Ma (2013) prove that random edge removal, as expected, always lowers the *basic reproduction number*. The basic reproduction number is defined as the number of infections caused by a single infected node in a susceptible population. Kuhlman, Tuli, Swarup, Marathe, and Ravi (2013) achieve a large improvement to block *simple* and *complex contagions* compared to existing methods. In case of a complex contagion a node will only be infected after at least two interactions with infected nodes. Furthermore, they discover the network structure has a huge impact; it is more difficult to prevent the spread of infections in a network with a smaller average degree and clustering coefficient. Finally, Nandi and Medal (2016) develop four models to minimize the spread of infections with different interdiction metrics. They conclude it is best to choose a model based on the specific problem setting. For low to moderate transmission probabilities, it is most effective to reduce the total weight of the transmission paths between infected and susceptible nodes. For moderate to high transmission probabilities, it is best to minimize the number of susceptible nodes at risk of infection.

There is hardly any research combining node and link removal methods simultaneously. He, Liang, and Yuan (2011) proposed a combined node and link removal method that minimizes the sum of the expected loss caused by infection and prevention costs of removing links or nodes.

From the literature presented in this section it is also clear that some researchers study very general situations, whereas others apply existing techniques or develop specific techniques for a certain application. However, this distinction is not examined further as most techniques, albeit slightly adjusted, can be used in a more general or specific setting. As

explained before, link removal methods are more flexible than node removal methods, hence node removal methods are only better when link removal methods are not possible. Link removal methods are thus preferred to minimize the spread of infections in the global airline network studied in this thesis. The next section therefore focuses on research covering, general and specific, link removal methods.

2.3 Link removal methods

The existing literature on link removal methods is compared based on four aspects: the research objective, the measure for the spread of infection, the methodology and the performance evaluation.

2.3.1 Research objectives

Although the ultimate goal of all researchers studying link removal methods in relation to the spread of infections is to minimize the spread of infections, the research objectives are often focused on a particular subproblem. Some of them propose new link removal methods, whereas others investigate the spread of infections under specific circumstances or take a step back to examine the effect of removing links on a certain infection measure.

New methods are proposed by Kimura et al. (2009), Tong et al. (2012) and Nandi and Medal (2016) to minimize the spread of infections by removing a certain number of edges. Enns et al. (2012) also propose a new link removal method, but have a slightly more specific goal as their aim is to minimize the number of nodes at risk. In other words, this means their goal is to have as many nodes as possible that are not connected to an infected node in any way. If a susceptible node is connected to an infected node through one path, it is already at risk. Kimura et al. (2009), Tong et al. (2012) and Nandi and Medal (2016) consider different measures to minimize the spread of infections, but do not consider this as their final goal.

Others investigate the spread of infections under specific circumstances. Kuhlman et al. (2013) also aim to minimize the spread of infections, but study infection spread both in the case of simple and complex contagion spread. He et al. (2011) aim to minimize the expected total loss by balancing the costs of infection and prevention by using node and link removal methods simultaneously. Chung et al. (2012) and Yang et al. (2013) both use existing link removal methods to study infection spread in a specific situation. Chung et al. (2012) aim to make a trade-off between cost and efficiency on removing connections in real world networks. Yang et al. (2013) investigate the impact of different link removal methods on traffic-driven epidemic spreading. They thus examine the spread of infection through the exchange of packages.

A good example of a study that investigates the effect of removing links on a certain infection measure is the study by Koch et al. (2013). They investigate the effect of removing of a number of edges on the basic reproduction number. Hence, they do not answer the question which edges need to be removed, but instead consider the number of edges that need to be removed to curtail an epidemic.

2.3.2 Measuring the spread of infections

Next, there are also some differences in how the spread of an infection is measured. The spread of infections is a stochastic process as every node is infected with a certain probability. As optimization with the use of simulation is very time demanding, most research uses indirect measures to represent the spread of an infection.

Enns et al. (2012) for example use the number of susceptible nodes at risk of infection, which can easily be computed by counting the number of susceptible nodes that are connected with infected nodes through a path. Kimura et al. (2009) consider worst and average contamination based on the influence degree of a node to capture the worst case and average scenario in terms of the number of contaminated nodes.

The research of Tong et al. (2012) and Chung et al. (2012) uses a less intuitive metric for infection spread. Both evaluate the effectiveness of removing an edge based on the decrease in the eigenvalue of the network adjacency matrix. This idea is based on the finding that the epidemic threshold of any graph depends on the eigenvalue of its adjacency matrix (Wang, Chakrabarti, Wang, & Faloutsos, 2003). The measure used by Koch et al. (2013) is the basic reproduction number as this is the factor of interest in their research. The eigenvalue can be used as the basic reproduction number.

Nandi and Medal (2016) also use indirect metrics for their heuristics, but in contrast to the others they evaluate the performance based on two direct measures of infection spread: the expected number of new infections and the expected time to infect half of the susceptible nodes. They compare four models with different metrics to minimize the spread of infections. For an explanation of these methods, we refer to Section 4.1.

2.3.3 Methods

Most proposed link removal methods are heuristic algorithms. Kimura et al. (2009) propose a greedy strategy that estimates the influence degree of a certain node in a graph using *bond percolation*. The bond percolation process creates a sample of graphs for which the influence degree of the nodes can be determined by randomly determining whether a certain edge is used for information diffusion. Tong et al. (2012) also propose a heuristic algorithm, but they focus on a feasible strategy to remove links based on the leading eigenvalue.

In addition to presenting a heuristic algorithm, some also provide a mathematical formulation. One advantage of having a mathematical model is that many existing solution methods can be used. Furthermore, it can help to develop better heuristic methods in the future. Enns et al. (2012) view the problem as partitioning problem between infected and susceptible nodes and propose a non-linear programming formulation based on two-way graph partitioning. More specifically, the mathematical formulation is a quadratically constrained quadratic program. This problem turns out to be NP-hard, hence the formulation is used to derive a heuristic algorithm. Using exhaustive search, they show for small problem instances that their heuristic method often results in the optimal partition and almost always saves the maximum number of susceptible nodes possible.

Nandi and Medal (2016) also provide a mathematical formulation for all four models. Because they are able to formulate the problems as mixed-integer linear programs they can also solve the models to optimality. Nevertheless, they propose heuristic algorithms

to solve large problem instances. They find that the algorithm for the first two models performs well as it often obtains the optimal solution. This is not the case for the algorithm for the last two models, but the average optimality gap turns out to be small.

2.3.4 Evaluating performance

An important step of the current literature is validating the proposed method by comparing with existing link removal methods using simulation. Commonly used link removal methods for comparison are the *random method*, the *degree method* and the *edge-betweenness method*. The random method as the name indicates, randomly selects a set of edges to remove. If a method does not outperform the random method this indicates the method performs poorly. The degree method has proven to work well in practice and deletes edges between nodes with the largest number of links. Last of all, the edge-betweenness method removes edges based on the largest reduction in the number of shortest paths.

Kimura et al. (2009) compare their proposed algorithm with the above three methods. Enns et al. (2012) do not consider the degree method but instead they evaluate a more specific edge-betweenness method that only counts the shortest paths between susceptible and infected nodes. In addition, they compare their method with a measure based on the clique size of the link and nodes it connects. Tong et al. (2012) also compare their method with the random method and degree method. Next to this, they use two comparative strategies based on *eigen-scores* and *PageRank*. Kuhlman et al. (2013) also compare with the degree method and the method based on eigen-scores. It can thus be concluded that the proposed methods are often compared with various existing methods. Unfortunately, however, it remains usually unclear how the different proposed methods perform relative to each other.

One exception is the work of Nandi and Medal (2016), who compare their method with the proposed methods of Kimura et al. (2009) and Enns et al. (2012). This provides valuable additional insights as it is most useful to know how the proposed methods perform relative to each other. The three methods all take a reactive approach assuming the state of the nodes is known.

After testing the method in small artificial examples, most researchers also try to validate their methods in real life large-scale networks. Networks of very different fields are evaluated. Chung et al. (2012) study a US air transportation network, a collaboration network and a peer-to-peer internet network. Tong et al. (2012) test their proposed method on route graphs. Kuhlman et al. (2013) evaluate the blocking methods based on a local social contact network. Kimura et al. (2009) use a blog network and a Wikipedia network to evaluate the performance of their proposed method. Finally, Enns et al. (2012) investigate a network of residential hotels where drugs are injected. This shows the proposed link removal methods are widely applicable.

3 Problem formulation

The problem is formulated as follows: we model the network as an undirected graph $G = (N, E)$ where N is a set of nodes and E is a set of edges. Then we define $I \subseteq N$ to be the set of *infected nodes* and $S = N \backslash I$ the set of *susceptible nodes*. We aim to find the set of links $L \subseteq E$ of size b that minimizes the spread of an infection. The budget b indicates how many links can be removed. A network can be anything, but in the context of the global airline network studied in this thesis the local network consists of people (i.e. nodes) and relationships between people (i.e. edges or links). Moreover, the metapopulation network consists of cities (i.e. nodes) and flight connections between cities (i.e. edges or links).

4 Methods

This section describes how the performance of the different link removal methods is evaluated. To examine this in local and metapopulation networks, we follow the approach of Nandi and Medal (2016). In short, this method first removes a set of links in a network using a link removal method and then evaluates the spread of an infection in the remaining network using simulation. The details are presented in this section. First, the different link removal methods are introduced and explained in Section 4.1. Section 4.2 and 4.3 provide the details for the case of the local and metapopulation network respectively.

4.1 Link removal methods

In this research the four link removal methods *MinConnect*, *MinAtRisk*, *MinPaths* and *MinWPaths* presented by Nandi and Medal (2016) are used to remove a set of links to minimize the spread of infections. Next to the proposed link removal methods, the *Random* method is used as a benchmark. In this section the link removal methods are shortly introduced. For the mixed-integer linear programming formulation of the proposed methods we refer to the original paper. Although the models can be solved to optimality for small or easy problem instances, the computation time explodes for larger instances. For the *MinPaths* and *MinWPaths* formulation this is already the case for networks larger than 20 nodes. The computation time also highly depends on the problem configuration, hence only a fraction of all problems can be solved to optimality. In addition, it is shown that the heuristics as presented in Nandi and Medal (2016) perform relatively well. As this thesis focuses on larger problems with varying problem configurations, the approximate solution algorithms are used here instead of an exact method to solve to optimality.

The general set up of the four proposed link removal methods is described in Algorithm 1. The input of the heuristics is a network modeled as the undirected graph $G = (N, E)$. The graph G contains information about the status of the nodes and about the probability an infection is transmitted over a certain edge. In every iteration of algorithm, one link is permanently removed from the network. This link is added to the set L consisting of all permanently removed links. To determine which link should be removed, the algorithm temporarily removes all remaining links $E \backslash \{L\}$ one by one to estimate the potential benefit of removing a specific link e. This benefit depends on the objective of the specific link removal method that is used. The difference between the four proposed link removal methods is the infection spread metric used to determine which link should be removed. In total the number of links that can be removed depends on the available budget b. The number of replications of the random part in each algorithm is equal to M.

Algorithm 1 Link removal method: General structure

procedure General Solution Algorithm
  input $G = (N, E)$
  initialization $L = \emptyset$
  while links removed $|L| <$ budget b **do**
    for all remaining links in network $E \backslash \{L\}$ **do**
      temporarily remove one link e from network $\rightarrow E \backslash \{L, e\}$
      compute value infection spread metric for specific link removal method
      if best value so far **then**
        remember this as best link e^*
      end if
      add temporarily removed link e to network $\rightarrow E \backslash \{L\}$
    end for
    permanently remove best link e^* from network $\rightarrow L = L \cup e^*$
  end while
  return remaining links $E \backslash \{L\}$
end procedure

The next sections explain how the value of the infection spread metric is computed for the different methods. The complete heuristic algorithms can be found in Appendix 8.1. Before applying one of the algorithms, all links between infected nodes are removed. Omitting them can reduce computation time and paths between two infected nodes are irrelevant as no second infection can take place. Paths between susceptible nodes are not removed. Although the proposed algorithms only take the initial state of the network into account, these paths can become relevant when one of the nodes becomes infected.

4.1.1 MinConnect

The first proposed method has the objective to minimize the number of connections between infected and susceptible nodes. A pair of nodes has a connection if there is at least one *transmission path*. A path between two nodes is a transmission path if it contains no other infected nodes. Choose to remove the links resulting in the smallest number of remaining connections between infected and susceptible nodes. The number of connections is computed as described in Algorithm 2.

Algorithm 2 Link removal method: MinConnect

procedure Compute Number of Connections (C)
  input $G = (N, E \backslash \{L, e\})$
  for $j \in 1, ..., M$ **do**
    randomly select $R = b - |L| - 1$ links $\in E$
    temporarily remove them $\rightarrow E \backslash \{L, e, R\}$
    determine graph components of $G = (N, E \backslash \{L, e, R\})$
    for all combinations of $i \in I$ and $s \in S$ **do**
      if i and s belong to same component **then**
        update $C = C + 1$
      end if
    end for
    add temporarily removed links R to network $\rightarrow E \backslash \{L, e\}$
  end for
  return value C
end procedure

4.1.2 MinAtRisk

The second proposed method minimizes the nodes at risk of infection. Equivalently, the goal is to have as many nodes as possible that are not connected to an infected node in any way. To achieve this, the method tries to isolate as many susceptible nodes as possible. The number of nodes at risk of infection can be computed in a similar way as the number of connections. The difference is that we do not count with how many infected nodes a susceptible node is connected, but instead we count the number of susceptible nodes that has a connection with an infected node. The number of nodes at risk of infection is computed as described in Algorithm 3.

Algorithm 3 Link removal method: MinAtRisk

procedure Compute Number of Nodes at Risk of Infection (A)
  input $G = (N, E \backslash \{L, e\})$
  for $j \in 1, ..., M$ **do**
    randomly select $R = b - |L| - 1$ links $\in E$
    temporarily remove them $\rightarrow E \backslash \{L, e, R\}$
    determine graph components of $G = (N, E \backslash \{L, e, R\})$
    for all $s \in S$ **do**
      if s belongs to same component as any $i \in I$ **then**
        update $A = A + 1$
      end if
    end for
    add temporarily removed links R to network $\rightarrow E \backslash \{L, e\}$
  end for
  return value A
end procedure

4.1.3 MinPaths

The third proposed method maximizes the number of transmission paths removed. Or in other words, the goal is to find a network with a minimum number of transmission paths between infected and susceptible nodes. Choose to remove the links resulting in the largest reduction in the number of transmission paths.

During the initialization, this algorithm computes the initial total number of transmission paths. For this, the algorithm first randomly generates M *trees* of the graph G. A tree is a connected, acyclic graph with $|N| - 1$ edges connecting all nodes N. Random trees of G can be generated by using a simple random walk as described in Algorithm 4. One important problem to take into account is the possibility that a node n is not connected to any other node $k \in N \backslash n$. In this case the algorithm does not terminate, because we will never visit node n in the random walk. Therefore, we check beforehand if all nodes have at least one connection. If this is not the case, we omit this node and construct a tree T on the remaining graph. This has no consequences for computing the number of transmission paths as the node n will never be part of a transmission path.

Algorithm 4 Link removal method: MinPaths/MinWPaths

```
procedure GENERATE TREES
    input G = (N, E)
    check if all nodes have a connection
    for j ∈ 1, ..., M do
        random walk in graph
        choose random starting node n
        while not all nodes visited do
            randomly select next node n + 1 connected to n
            if first visit node n + 1 then
                add link (n, n + 1) to tree T_j
            end if
        end while
    end for
    return trees T
end procedure
```

For all trees constructed we compute the number of transmission paths as described in Algorithm 5, resulting in the initial total number of transmission paths. In each step of the algorithm, the heuristic then counts the number of transmission paths that contain the specific link e to compute how many transmission paths would be removed if you choose to remove that link. If the link e^* is permanently removed, also remove the transmission paths containing that link as these paths no longer exists.

Algorithm 5 Link removal method: MinPaths

```
procedure COMPUTE NUMBER OF PATHS (P)
    input T
    for s ∈ S do
        for i ∈ I do
            find shortest path between node s and i
            if path exists & path does not contain any infected nodes then
                update P = P + 1
            end if
        end for
        for s + 1 ∈ S do
            find shortest path between node s and s + 1
            if a path exists & path does not contain any infected nodes then
                update P = P + 1
            end if
        end for
    end for
    return value P
end procedure
```

4.1.4 MinWPaths

The last proposed method is a weighted version of the *MinPaths* method. The objective of this method is to minimize the total weight of the transmission paths between infected and susceptible nodes. The weight of each path is determined by multiplying the transmission probabilities of the links on that path. During the initialization, this algorithm computes the initial total weight of transmission paths by generating random trees of G as described before (see Algorithm 4). The initial total weight of transmission paths for the constructed trees is computed as described in Algorithm 6. In each step of the algorithm, the heuristic

considers the total reduction in weight if you would remove link e. If the link e^* is permanently removed, also remove the transmission paths containing that link as these paths no longer exists.

Algorithm 6 Link removal method: MinWPaths

```
procedure Compute number of weighted paths (W)
    input T
    for s ∈ S do
        for i ∈ I do
            find shortest path between node s and i
            if path exists & path does not contain any infected nodes then
                compute weight w of path: multiply transmission probabilities of links
                update W = W + w
            end if
        end for
        for s + 1 ∈ S do
            find shortest path between node s and s + 1
            if a path exists & path does not contain any infected nodes then
                compute weight w of path: multiply transmission probabilities of links
                update W = W + w
            end if
        end for
    end for
    return value W
end procedure
```

4.1.5 Random

In addition, we compare the proposed methods with the random link removal method. The random method randomly removes a set of links of size b from the network and is performed M times for each generated network to get reliable performance measures.

4.2 Local network

This section describes how the local network is obtained and how the performance of the link removal methods is evaluated in the local network.

4.2.1 Generating scale-free networks

For the case of a local network the four methods will be compared on generated scale-free networks. Random networks are not considered here as the interest is in real life networks, which are often scale-free but seldom random.

The random scale-free networks are generated using the method of Barabási and Albert (1999). This method starts with a small number of nodes m_0[1]. These nodes are linked with $|m_0| - 1$ links, hence the starting point is a connected graph (Barabási, 2016). Then one node is added at a time until the desired network size is obtained. Connect each node with m ($< m_0$) already existing nodes when you add it. The probability a new node is connected to a certain existing node i depends on the connectivity k_i of that node. The connectivity k_i is defined as the degree of node i and the probability as $\pi(k_i) = \frac{k_i}{\sum_j k_j}$, where the sum is over all existing nodes j. This is called *preferential attachment*, as a new

[1]This thesis uses $m_0 = 2$, 5 and 10 for networks of size 30, 100 and 150 respectively.

node is more likely to connect to higher degree nodes. The result is a scale-free network with $t + m_0$ vertices, $mt + m_0$ edges and a power law degree distribution with an exponent $\gamma \approx 2.9$. Links are added until the desired number of links is obtained, which is $2|N|$ for a network with average node degree of 4.

4.2.2 Simulation

A simulation model is used to evaluate the performance of the link removal methods. This is necessary because the methods are not able to minimize the spread of infections directly. Instead all methods use indirect objectives targeting the connectivity of the graph.

We use two types of simulations: a *susceptible-infectious* and a *susceptible-infectious-recovered simulation*. The SI simulation is used to calculate the expected time to infect half of the susceptible nodes. The SIR simulation is used to calculate the expected number of new infections. In the SIR simulation nodes have the possibility to recover from infection, which is more realistic in most situations. Nevertheless, we use the SI simulation to analyze the speed of the infection spread because it is possible in the SIR simulation that the infection spread stops before half of the nodes is infected.

Both simulations are built up as follows: in each iteration, we draw a random number for every link between an infected and susceptible node. Then, for each of these links we compare the random number to the probability of transmission. If the random number is smaller than the probability of transmission, the infected node infects its susceptible neighbor node. In addition, we draw a random number for each infected node in the SIR simulation, if the random number is smaller than the probability of recovery, the infected node recovers and is immune for the infection. In each iteration we assume the infection is first able to spread through the network, before the node will recover. The SI simulation is stopped when half of the susceptible nodes is infected and the SIR simulation terminates automatically when all nodes are either recovered or susceptible. It is possible, however, that the SI simulation will not terminate. For example, when there are no links left between infected and susceptible nodes. This would happen if the algorithm is performing really well. Therefore, we also stop the SI simulation if there are no links connecting infected and susceptible nodes. If this happens, we take the maximum value of the time to infect half of the susceptible nodes for this generated network as replacement value. Both simulations are performed M times for each generated network and link removal method to get reliable results.

4.2.3 Comparison methods

The four methods will be compared for different values of the transmission probability τ_I, the recovery probability ρ, the initial fraction of infected nodes i and the fraction of links removed r. To limit computation times we investigate scale-free networks of size $s = 30$. More specifically, to evaluate the performance of the different methods in a local network the following steps are performed:

1. Randomly generate a scale-free network
2. Find a set of links to remove from the network using one of the link removal methods
3. Remove the links from the network

4. Examine the infection spread on the remaining network in terms of the expected number of new infections and the expected time to infect half of the susceptible nodes using simulation

All steps will be replicated 50 times for each link removal method and combination of parameters to get reliable results. Furthermore, the number of replications M of the simulation and random parts of the link removal methods equals 100.

We investigate the performance of the link removal methods in several scenarios (see Table 1). For each scenario, we let the transmission probability τ_I vary over the values 0.05, 0.15, 0.30 and 0.9, while all other parameters remain constant. Set $i = 0.2$, $r = 0.2$ and $\rho = 0.15$ in the base case. Next, we consider several changes in parameters. First, we investigate the case of a higher initial fraction of infected nodes. Second, we examine a change the fraction of removed links. Last of all, we investigate the effect of a change in the probability of recovery. Let all other parameters stay the same as in the base case.

Table 1: Different combinations of parameter values investigated in local network

Scenario	Initial fraction of infected nodes i	Fraction of removed links r	Recovery probability ρ
1. Base case	0.2	0.2	0.15
2. Higher infected	0.3	0.2	0.15
3. Lower removed	0.2	0.1	0.15
4. Higher infected & lower removed	0.3	0.1	0.15
5. Higher removed	0.2	0.3	0.15
6. Higher recovery	0.2	0.2	0.3

4.3 Metapopulation network

This section describes how the metapopulation network is obtained and how the performance of the link removal methods is evaluated in the metapopulation network.

4.3.1 Global airline network

For the case of a metapopulation network the four methods will be compared on one specific type of network, the global airline network obtained from the paper of Marcelino and Kaiser (2012). The network consists of the 500 most frequently used airports that together represent more than 95% of global flight traffic. It is constructed using information about all flights in the period from July 1, 2007 to July 30, 2008.

For this study, we select a subset of 30 airports with the highest yearly number of passengers (Airports Council International, 2013). The resulting network needs several adjustments before it can be used for this study. First of all, the adjacency matrix corresponding to the resulting network is not symmetric. This means there are flights included that only go in one direction. However, this analysis requires an undirected graph and hence the adjacency matrix should be symmetric. Therefore, return flights are added for flights that only go in one direction. Secondly, not all nodes are connected. There is one airport (HND) that has no connection with any other airport in the network. This is probably because the data used to create the network is outdated. To adjust this, links between that airport and the other airports are added based on recent flight information. Last of all, it is important to observe that the resulting network is not a scale-free network. Instead, it

is a highly connected graph as most airports have a large number of links. To limit the number of links, connections with a flight distance larger than 12,000 km are removed. Flights longer than this distance are most likely no direct flights and the computation time of the methods is drastically increasing in the number of links. For details about the obtained global airline network we refer to Appendix 8.2.

Next, we want to determine the traffic intensity between each of these airports to account for a difference in transmission probability between pairs of cities. Ideally, the yearly number of passengers between two airports is used as the traffic intensity between them. However, as this information is not publicly available we generate traffic flow using the method presented by Capar and Kuby (2012). Hence, the traffic intensity f_{od} between city o and d is defined as:

$$f_{od} = \frac{p_o * p_d}{d_{od}^2} \tag{1}$$

where p_o and p_d are the yearly number of passengers of the airport in city o and d respectively. Furthermore, d_{od} is the flight distance between city o and d. The traffic intensity thus increases with the number of passengers of both airports, but decreases more the further the airports are located from each other. Note that the traffic intensity is symmetric, which means that the traffic from city o to d is assumed to be similar to the traffic from city d to o.

The traffic intensity will be scaled and represented as a number between 0 and 1, e.g. between city o and d it is defined as traffic intensity $f_{od}^* = \frac{f_{od}}{\max_{i,j}(f_{ij})}$. The result is a network with $N = 30$ nodes and a number of edges E, where each edge e has a weight equal to the traffic intensity f_{od}^*. This weight is used for *MinWPaths* in the metapopulation network. However, the spread of an infection between cities does not only depend on the traffic intensity, but also on the stage of infection of the cities. This is explained in the next section.

4.3.2 Homogeneous-mixing compartmental model

This thesis assumes for the metapopulation network that the population within a node is homogeneous. In other words, we do not incorporate contact patterns between people in a city but only between cities. Under the assumption that contact patterns are homogeneous we can use a homogeneous-mixing compartmental model to estimate the spread of an infection in the local network. The number of infected I, susceptible S or recovered R individuals can be described by the following set of differential equations as presented by Bansal et al. (2007):

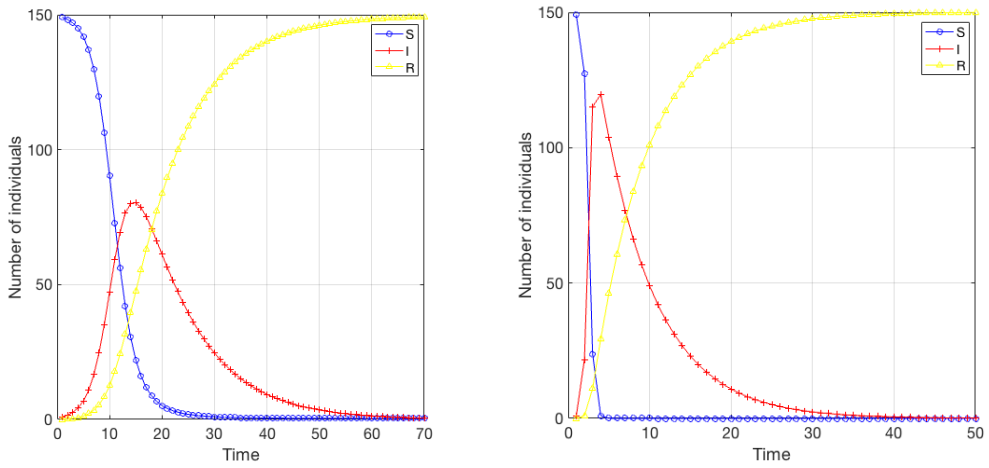
$$\frac{dS(t)}{dt} = -\phi S(t) \tag{2}$$

$$\frac{dI(t)}{dt} = \phi S(t) - \rho I(t) \tag{3}$$

$$\frac{dR(t)}{dt} = \rho I(t) \tag{4}$$

where ϕ is the force of infection and ρ the probability of recovery. The force of infection ϕ is the rate at which new individuals are infected, it is defined as follows: $\phi = \alpha * \frac{I(t)}{|N|} * \tau_I$. The force of infection thus depends on the amount of contact α, the current fraction of infected individuals $\frac{I(t)}{|N|}$ and the transmission probability τ_I. Let $\alpha = 4$, the average number of links of a node in the case of a local network. The system of differential equations allows us to make a prediction of the evolution of an infection in a city over time as can be seen

in Figure 3. This is derived by making small steps t in time, calculating the change in each step starting from the initial situation $S(0) = 149$, $I(0) = 1$, $R(0) = 0$.



(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.10$.

(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 3: Development of infection within city over time for two different scenarios.

This thesis assumes every city goes through the same stages of infection $1, ..., Q$, but each city o can be in a different stage of infection at a certain point in time. Let the degree of infection $q_{o,i}$ of city o in stage of infection i be defined as:

$$q_{o,i} = \tau_I \frac{I(i)}{\max I(i)} \qquad \forall i = 1, ..., Q. \tag{5}$$

Furthermore, it is assumed that every city can only be infected once. Hence, every city that is infected through the global airline network starts in the first stage of infection with degree of infection $q_{o,1}$. The degree of infection for the different stages is determined at the start of the simulation based on the outcome of equations (2)-(4) and will be a number between 0 and τ_I. More specifically, in the stage where the number of infected individuals is at the maximum, the degree of infection is equal to the transmission probability between individuals. For the rest of the stages the degree of infection decreases proportionally with the number of infected individuals. The number of stages Q depends on the duration of the infection and also follows from equations (2)-(4).

4.3.3 Simulation

The simulation model to evaluate the performance of the link removal methods for the metapopulation network is build up as follows. In each iteration, draw a random number for every link between an infected and non-infected city. Then, for each link compare the random number to the probability of transmission τ_{od} between city o and d. If the random number is smaller than the probability of transmission, the infected city infects the other city. The probability of transmission τ_{od} is defined as:

$$\tau_{od} = f^*_{od} * q_{o,i} \tag{6}$$

where f^*_{od} is the traffic intensity between city o and d and $q_{o,i}$ is current degree of infection of the infected city o in stage of infection i. The traffic intensity and the degree of infection

are two important factors that positively influence the probability an infection spreads between two cities. The newly infected city d starts in the first stage of infection with degree of infection $q_{d,1}$. At the end of each iteration cities continue to the next stage of infection with probability π. The transition between stages of infections is thus a simple Markov chain, as can be seen in Figure 4.

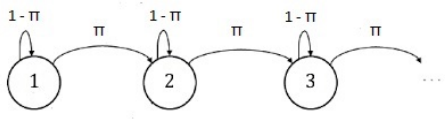


Figure 4: Transition between stages of infection

Let each iteration in the simulation represent a day. In this context, the transmission probability τ_I and the recovery probability ρ have more meaning than before. If the recovery probability is ρ, an individual will recover in approximately $\frac{1}{\rho}$ days. Similarly, a city going through Q stages of infection is infected for at least Q days.

The simulation stops when all nodes in the metapopulation network are either recovered or susceptible. First, we examine how often there is an outbreak. Then, we investigate the spread of the infection in case of an outbreak. An outbreak occurs if there is at least one newly infected city. We evaluate the spread in case of an outbreak in terms of the expected number of new infected cities and the expected time to infect a quarter of the cities. In addition, we investigate the expected fraction of times a specific city is infected to determine the risk of infection for each city. For the remainder of this thesis, we define the fraction of times an event occurs as the number of times the event occurs for a fixed number of replications, expressed as a fraction of the number of replications. The simulation is performed M times for each link removal method to get reliable results.

4.3.4 Comparison methods

The four methods will be compared for two different scenarios (see Table 2). Scenario I represents a low contagious infection with a low transmission probability and a recovery probability of $\rho = 0.1$ (≈ 10 days). Scenario II represents a high contagious infection with a high transmission probability and a recovery probability of $\rho = 0.15$ (≈ 7 days).

Table 2: Two scenarios investigated in metapopulation network

Scenario	Transmission probability τ_I	Recovery probability ρ
I. Low contagious infection	0.15	0.1
II. High contagious infection	0.9	0.15

Let the transition probability between stages of infection be $\pi = 0.8$. For both scenarios, we investigate the performance of the methods for varying fraction of links removed r and consider two origins of infection. More specifically, let $r = 0.1$, 0.2 and 0.3. We select a very highly connected airport (AMS) and a less connected airport (MAD) as initial starting points of infection. Also, we evaluate the spread of an infection without canceling flights $r = 0$.

To evaluate the performance of the different methods in a metapopulation network the following steps are performed:

1. Construct global airline network with a specific origin of infection
2. Find a set of links to remove from the global airline network using one of the link removal methods
3. Remove the links from the global airline network
4. Examine the infection spread on the remaining network in terms of the expected fraction of times an outbreak occurs, the expected number of new infected cities in case of an outbreak, the expected time to infect a quarter of the cities in case of an outbreak and the expected fraction of times a specific city is infected in case of an outbreak using simulation

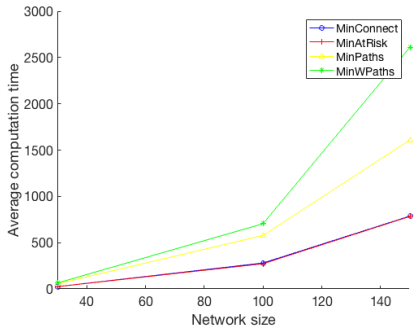
To get reliable results from the simulation, repeat these steps 5 times for each link removal method and combination of parameters. Note, we do not have to replicate all steps as many times as in the local network as we do not generate random networks here. Instead, it is more important that the simulation is replicated more often. Therefore, the number of replications M of the simulation equals $1,000$. For the random parts of the link removal methods M still equals 100 as before.

5 Results

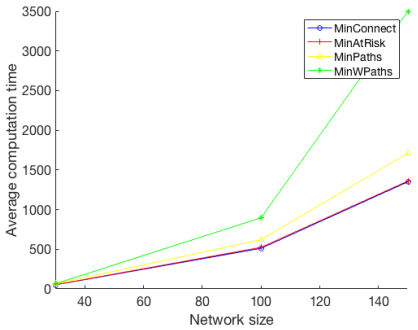
The results are presented in this section. In Section 5.1 we show the computation times of the link removal methods. Next, the results of the relative performance of the methods are presented in Section 5.2 for local networks and in Section 5.3 for metapopulation networks.

5.1 Computation time

First, we compare the computation time of the various link removal methods. All computation times are determined on a computer with an Intel core i5 1.7 GHz processor and 4GB RAM. As can be seen in Figure 5, the computation time of all methods increases significantly with the network size. Moreover, the relative computation time of the different link removal methods changes with the fraction of links removed. The higher the fraction of removed links, the faster *MinPaths* and *MinWPaths* are compared to the *MinConnect* and *MinAtRisk*. This can be explained because *MinPaths* and *MinWPaths* take most time during the initialization, whereas *MinConnect* and *MinAtRisk* have a computationally intensive while loop. When the problem size increases this loop takes relatively more and more time. *MinWPaths* still takes significantly more time than *MinPaths* because it needs to calculate the weight of each path as well.



(a) Fraction of links removed $r = 0.05$.



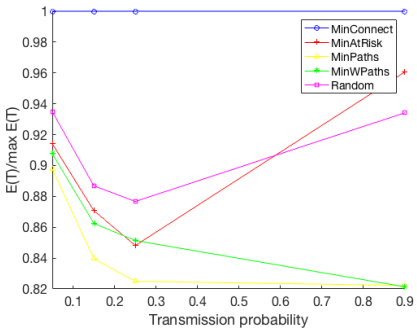
(b) Fraction of links removed $r = 0.1$.

Figure 5: Average computation times of the heuristics with respect to network size. The average computation times are computed over 5 problem instances with initial fraction of infected nodes $i = 0.2$ and transmission probability $\tau_I = 0.15$.

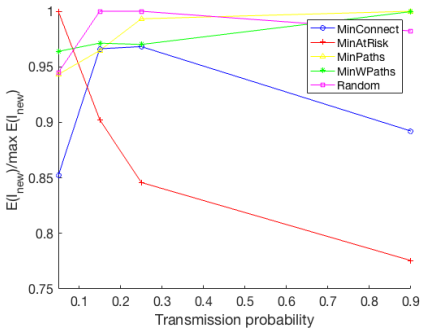
5.2 Local network

In this section the relative effectiveness of the link removal methods in a local network is evaluated. As described in Section 4.2.3, the performance of the link removal methods is investigated for six combinations of parameter values. The plots show the performance of the methods for different transmission probabilities in terms of the expected time to infect half of the susceptible nodes $E(T)$ and the expected number of new infections $E(I_{new})$.

The base case is presented in Figure 6. It is clear that under these circumstances *MinConnect* is most effective in maximizing the time to infect half of the susceptible nodes. For all values of the transmission probability it outperforms the other methods by far. Furthermore, it is remarkable that *Random* is the second best performer as you expect that targeted link removal is at least more effective than random link removal. The results are completely different, however, when considering the number of new infections. Here, *Random* performs worst and *MinAtRisk* is most effective in minimizing the number of new infections for moderate to high values of transmission probability. When the transmission probability is low, *MinConnect* performs best again. The fact that the effectiveness of *MinAtRisk* is increasing with the transmission probability can be explained because this method tries to isolate as many susceptible nodes as possible. This is most helpful when the total spread of infection is large and this is more likely to be the case for higher transmission probabilities.



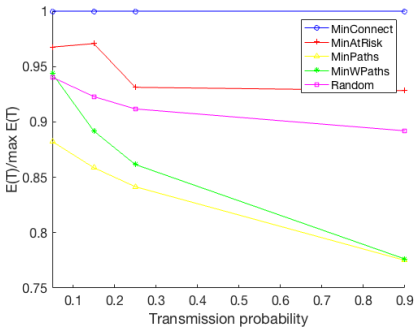
(a) Time to infect half of the susceptible nodes.



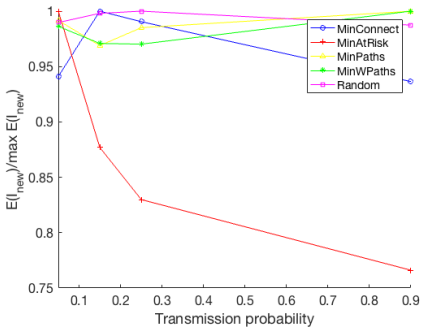
(b) Number of new infections.

Figure 6: Performance link removal methods for varying transmission probability τ_I. Initial fraction of infected nodes $i = 0.2$, fraction of links removed $r = 0.2$, recovery probability $\rho = 0.15$.

The result for the second combination of parameters, with a higher initial fraction of infected nodes ($i = 0.3$), is shown in Figure 7. Again, *MinConnect* is best in slowing down the spread. However, *MinAtRisk* performs relatively better than before. This shows that complete isolation of susceptible nodes can be effective to slow down the speed of spread, but only when many nodes are infected and there are limited resources to remove links. The relative performance of *MinAtRisk* in terms of the number of new infections improved even more and again increases with transmission probability. All other methods perform roughly similar for varying transmission variables.



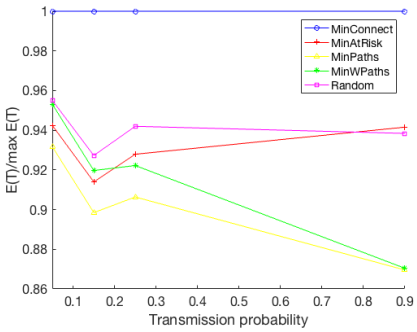
(a) Time to infect half of the susceptible nodes.



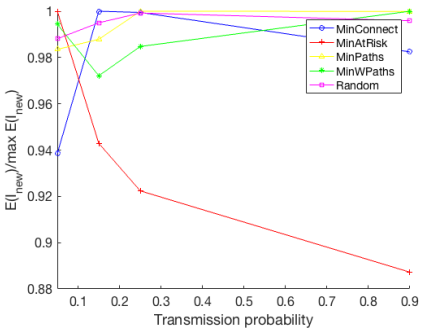
(b) Number of new infections.

Figure 7: Performance link removal methods for varying transmission probability τ_I. Initial fraction of infected nodes $i = 0.3$, fraction of links removed $r = 0.2$, recovery probability $\rho = 0.15$.

The next combination of parameters has a lower fraction of removed links ($r = 0.1$) than the base case. This result is shown in Figure 8. In this case there is less deviation in the performance between link removal methods, but the order of effectiveness is still the same as before. The variation in performance is largest for high transmission probabilities.



(a) Time to infect half of the susceptible nodes.

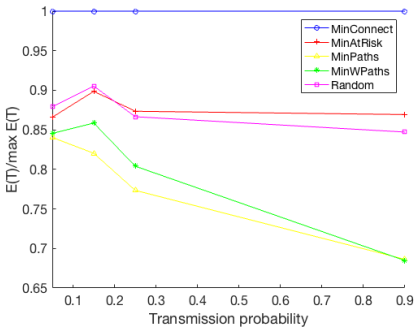


(b) Number of new infections.

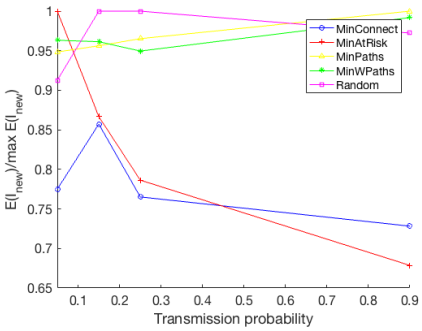
Figure 8: Performance link removal methods for varying transmission probability τ_I. Initial fraction of infected nodes $i = 0.2$, fraction of links removed $r = 0.1$, recovery probability $\rho = 0.15$.

The result for the fourth combination of parameters, with a higher initial fraction of infected nodes ($i = 0.3$) and a lower fraction of links removed ($r = 0.1$), is similar to the findings in Figure 7 and 8. Therefore, this figure can be found in Appendix 8.3.

The fifth combination of parameters has a higher fraction of links removed ($r = 0.3$) compared to the base case. This result is shown in Figure 9. The expectation is that the higher the fraction of links removed, the better the link removal methods should work compared to the random method. This is indeed the case when we consider the number of new infections. We see *Random* is the least effective method and the other methods are all better. The relative performance of the methods improved compared to *Random*. Most notably, the performance of *MinConnect* increased significantly. Apparently, if the fraction of links that can be removed is large enough, the method can effectively decrease the connections between infected and susceptible nodes. For a high fraction of removed links, it is thus an effective way of minimizing the number of new infections. When we consider the time to infect half of the susceptible nodes, however, *MinPaths* and *MinWPaths* still perform badly compared to *Random*.



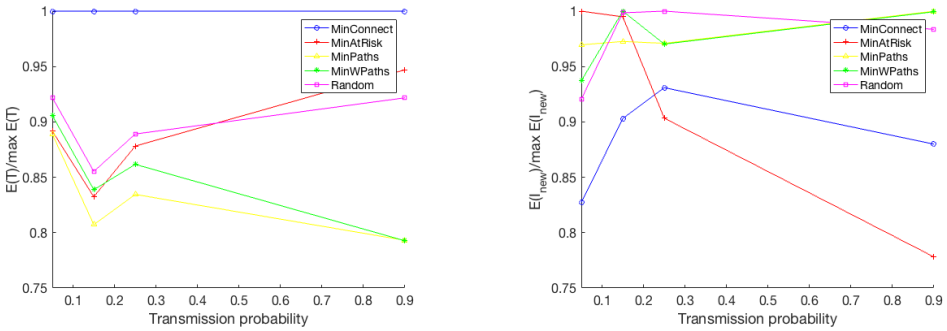
(a) Time to infect half of the susceptible nodes.



(b) Number of new infections.

Figure 9: Performance link removal methods for varying transmission probability τ_I. Initial fraction of infected nodes $i = 0.2$, fraction of links removed $r = 0.3$, recovery probability $\rho = 0.15$.

The result for the final combination of parameters, with a higher recovery probability ($\rho = 0.3$), is shown in Figure 10. As in the base case, *MinConnect* is most effective in maximizing the time to infect half of the susceptible nodes. But, *MinConnect* is now more effective than *MinAtRisk* for moderate transmission probabilities compared to before. Hence, when nodes are likely to recover faster it is less efficient to spend all resources to isolate susceptible nodes.



(a) Time to infect half of the susceptible nodes.

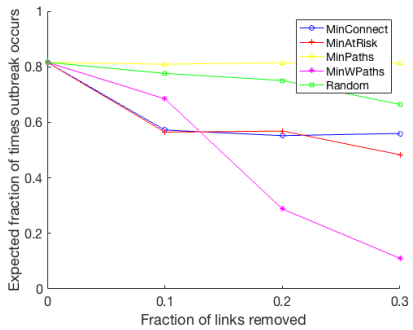
(b) Number of new infections.

Figure 10: Performance link removal methods for varying transmission probability τ_I. Initial fraction of infected nodes $i = 0.2$, fraction of links removed $r = 0.2$, recovery probability $\rho = 0.3$.

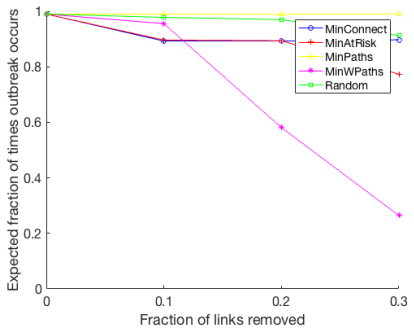
5.3 Metapopulation network

In this section the relative effectiveness of the link removal methods in a metapopulation network is evaluated. As described in Section 4.3.4, the performance of the link removal method is investigated for two scenarios. The plots show the performance of the methods for a varying fraction of removed links in terms of the fraction of times an outbreak occurs, the expected time to infect a quarter of the cities in case of an outbreak $E(T)$, the expected number of new infected cities in case of an outbreak $E(I_{new})$ and the fraction of times a specific city is infected in case of an outbreak.

First, we evaluate the performance of the link removal methods in terms of the fraction of times an outbreak occurs. This is presented in Figure 11 with Amsterdam as the origin of infection. It can be seen that the number of outbreaks decreases the more links are removed. When considering an outbreak of a low contagious infection in Amsterdam, it is transmitted to other cities approximately 80% of the times without intervention. For a high contagious infection this is nearly 100%. Hence, there is a higher number of outbreaks in the case of a very contagious infection. These findings are as expected. As can be seen *MinWPaths* is most effective of the proposed link removal methods to limit the number of outbreaks. It is able to decrease the number of outbreaks in both scenarios by about 80% when 30% of the flight connections is removed.



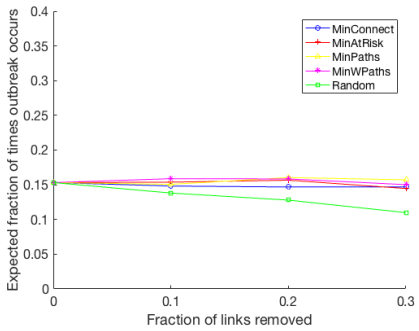
(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.



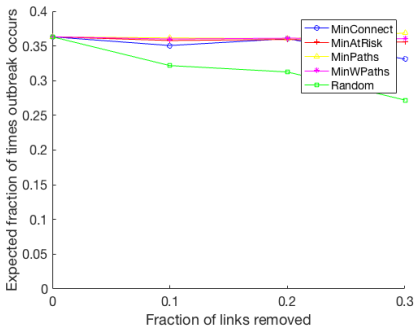
(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 11: Performance link removal methods in terms of the average number of outbreaks for varying fraction of removed links r. Initially infected city: Amsterdam.

In a similar way, the fraction of times an outbreak occurs in Madrid is presented in Figure 12. The difference is that Amsterdam is connected to about twice the number of cities compared to Madrid. This becomes clear immediately, as there are much fewer outbreaks when Madrid is the initially infected city. Although the number of outbreaks reduces slightly if the fraction of removed links is higher, the impact of removing links is small. Furthermore, it can be seen that the link removal methods are not able to outperform *Random*, at least not to reduce the number of outbreaks. However, they might still be able to affect the size or the speed of the infection spread in case of an outbreak.



(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

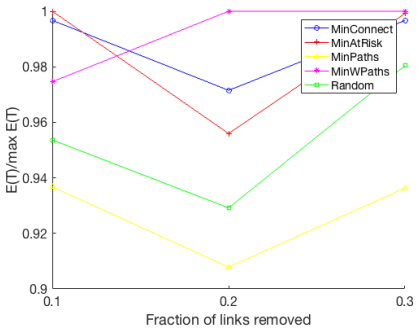


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

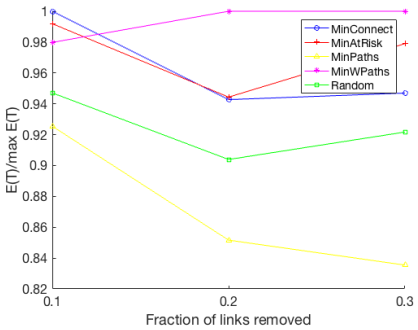
Figure 12: Performance link removal methods in terms of the average number of outbreaks for varying fraction of removed links r. Initially infected city: Madrid.

Next, we investigate the performance of the link removal methods in terms of the expected time before a quarter of the cities is infected $E(T)$ and the expected number of new infected cities $E(I_{new})$ given that an outbreak occurs is shown in Figure 13 and 14 with Amsterdam as initially infected city. As the results for Madrid are very similar, these figures can be found in Appendix 8.3.

From Figure 13, it is clear that *MinWPaths* is most effective to maximize the time to infect a quarter of the cities when a substantial fraction of links is removed. For very small values of removed links, *MinConnect* and *MinAtRisk* perform slightly better. The results are very similar for both scenarios.



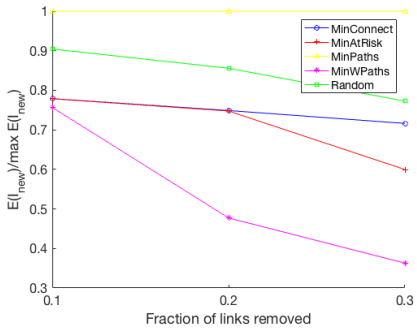
(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.



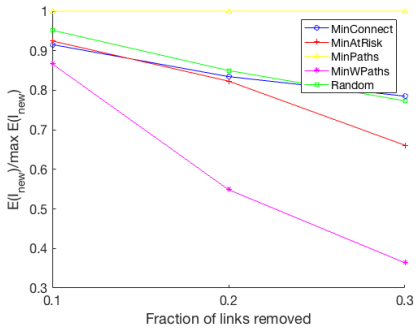
(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 13: Performance link removal methods in terms of time to infect a quarter of the susceptible nodes for varying fraction of removed links r. Initial infected city: Amsterdam.

Moreover, *MinWPaths* is also the best method to minimize the number of new infected cities. Again this method is followed in performance by *MinConnect* and *MinAtRisk*. The results are shown in Figure 14. The reason why *MinWPaths* performs so well for this particular network is probably because this is the only link removal method that takes the likelihood of transmission over each path into account. This is very effective as the traffic intensity for each link in the global airline network varies a lot.



(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

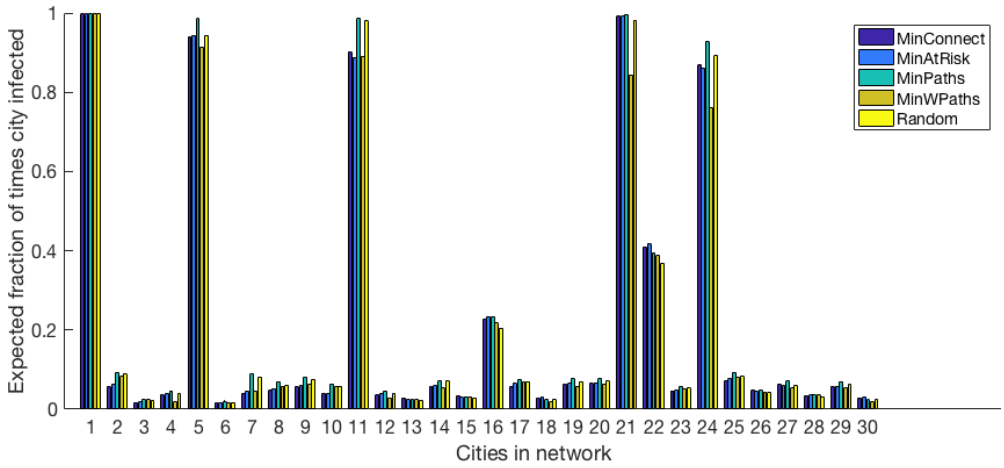


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

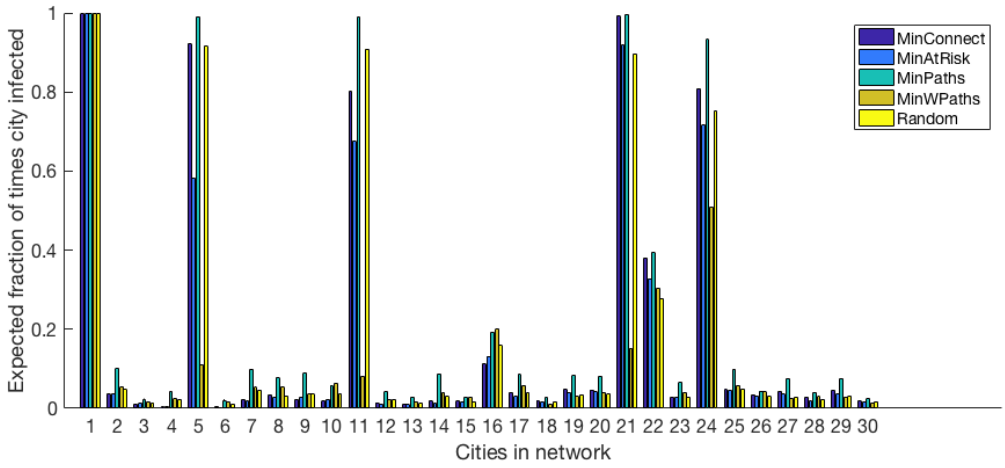
Figure 14: Performance link removal methods in terms of number of new infections for varying fraction of removed links r. Initial infected city: Amsterdam.

In other words, for a low contagious infection starting in Amsterdam, *MinWPaths* is able to reduce the expected number of new infected cities from about 7 to 5 when 10% of the connections is removed and from about 7 to 2.5 when 30% of the connections is removed. For a high contagious infection, a reduction from 12 to 10 and from 12 to 4.5 can be achieved respectively. This is a reduction in the expected number of new infected cites of roughly 60% when 30% of the connections is removed. In addition, it can slow down the spread of infection by postponing the expected time when a quarter of the cities are infected with at least 10 days. For Madrid these numbers are very similar. This suggests that although less outbreaks occur because the starting city is less connected, the spread of an infection in case of an outbreak is similar. The corresponding figures can be found in Appendix 8.3.

Finally, we examine the performance of the link removal methods in terms of the expected fraction of times a specific city is infected in case of an outbreak. This is shown in Figure 15 for a high contagious infection starting in Amsterdam (1). It can be seen that many cities are rarely infected whereas a few other cities are infected very often. Intuitively, this is understandable as cities closer to Amsterdam have a higher probability to be infected. Indeed, among the cities with most risk to be infected are Paris (5), London (21) and Munich (24). When 30% of the links is removed with *MinWPaths*, the risk to be infected in many of the cities that were first likely to be infected decreases significantly. However, this decrease in risk is not equally spread over the cities. For example, Madrid (22) and Munich (24) are still infected quite often after 30% of the links is removed with *MinWPaths* whereas Paris (5) and Frankfurt (11) are much less likely to be infected. Similar figures for a low contagious infection and Madrid as initially infected city are presented in Appendix 8.3.



(a) Fraction of removed links $r = 0.1$.



(b) Fraction of removed links $r = 0.3$.

Figure 15: Performance link removal methods in terms of fraction of times a certain city is infected for different fractions of removed links r. Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$. Initial infected city: Amsterdam.

6 Discussion

In this section the results presented in the previous section will be discussed. In particular, we discuss the similarities and differences with the results of Nandi and Medal (2016). In addition, the aim is to give insight in the possible causes of these differences.

First of all, there is a substantial difference in computation time of the link removal methods. The running times obtained in this thesis are longer. Most remarkable is that *MinPaths* and *MinWPaths* solve in a few seconds for Nandi and Medal (2016), whereas these algorithms take longest in our case. However, as they presented hardly any details needed to replicate the generated random trees, there is probably a big difference in implementation. Furthermore, the computation time highly depends on the specific problem configuration. As the generated scale-free networks are probably not the same this might have influenced the computation time as well. Finally, they used a computer with a faster processor and more memory. To be able to get results in limited time, this thesis examines scale-free networks of a smaller size.

Despite the fact that our computation time obtained for networks of size 30 was much lower than their computation time for networks of size 150, we could not perform as many replications for each combination of parameters as Nandi and Medal (2016). To get reliable results they accepted extremely long computation times ($\approx$ 4.6 days for one combination of parameters), which was impossible within the scope of this thesis due to time constraints. This has to be taken into account when analyzing the results presented in this thesis. This analysis can, however, easily be extended to more replications.

In addition, there is some ambiguity in the implementation details of Nandi and Medal (2016). For example, it is unclear what the order of the spread of new infections and the recovery of nodes is in the susceptible-infectious-recovered simulation. Furthermore, it is not stated how they deal with exceptional cases, such as a node that is not connected to the graph. This can happen after a network is generated, but also after a link removal method is applied. Moreover, the value used for the recovery probability of nodes is not specified and thus unknown. Last of all, they could have been more precise how they generated the scale-free networks. Although the general method was mentioned, there are many details that were unclear. Therefore, the generated scale-free networks are probably not the same.

All these small choices lead to differences in implementation, which may cause different outcomes. Nevertheless, many results obtained for the local network in this thesis are similar to the ones seen in Nandi and Medal (2016). Indeed, they also find *MinAtRisk* is most effective to minimize the spread of infections. Moreover, they show its effectiveness increases with the transmission probability. Furthermore, when the fraction of initially infected nodes increases compared to the base case, this thesis finds the effectiveness of *MinAtRisk* to maximize the time to infect half of the susceptible nodes increases a lot. This is also shown by Nandi and Medal (2016). Finally, the link removal methods overall perform badly to slow down spread and they also find *Random* is often better than the proposed link removal methods.

Therefore, many dynamics found by Nandi and Medal (2016) are confirmed by the results presented in this thesis. There is only one major difference, *MinWPaths* performs much worse than expected for a scale-free network. Whereas *MinWPaths* is found to be most

effective to maximize the time to infect half of the susceptible nodes by Nandi and Medal (2016), in this thesis we find that *MinConnect* performs best. This can be due to a slightly different implementation of the algorithm or a difference in the generated scale-free network which makes the method less efficient.

Next to replicating the results of Nandi and Medal (2016), this thesis examined several other effects in a local network such as a change in recovery probability and a higher fraction of links removed. Furthermore, this thesis investigated a new type of network, a metapopulation network. For the two scenarios investigated in the global airline network we do find *MinWPaths* performs best to maximize the time to infect a quarter of the cities. It is also best to minimize the spread of infections and the number of outbreaks. This method is very effective for this type of network, because this link removal method is the only one that takes the differences in traffic intensity into account. Unfortunately, *MinWPaths* fails to decrease the risk of infection evenly for all cities.

7 Conclusion

This thesis investigated the problem to minimize the spread of infections in local and metapopulation networks by removing a set of links. For that purpose, we first evaluated the performance of the link removal methods *MinConnect*, *MinAtRisk*, *MinPaths*, *MinWPaths* and *Random* on generated scale-free networks to replicate the results of Nandi and Medal (2016). Then, we investigated the performance of the link removal methods in a metapopulation network consisting of multiple local networks. To be specific, we examined a global airline network consisting of the 30 airports with the highest yearly number of passengers. The performance of the link removal methods is evaluated by means of simulation after removing a set of links with one of the heuristics.

For the local network we found that *MinAtRisk* is most effective to minimize the number of new infections for moderate to high values of transmission probability. The relative performance of this method increases with the transmission probability and when many nodes are infected compared to the number of links that can be removed. For low values of transmission probability, *MinConnect* performs better to reduce the number of new infections. *MinConnect* also performs well when a large number of links can be removed. Furthermore, we found it is best to remove links based on *MinConnect* to slow down the spread of an infection. Slowing down the speed of the infection is desirable as it gives more time for prevention and other interventions. Most results are in line with the findings of Nandi and Medal (2016). One exception is that *MinWPaths* performs worse than expected to maximize the time to infect half of the susceptible nodes. In addition, the running times obtained in this thesis are longer.

For the global airline network studied in this thesis, *MinWPaths* is the most effective link removal method to minimize the number of new infected cities and to maximize the time before a quarter of the cities is infected. When 30% of links is removed, this method is able to decrease the average number of new infected cities by approximately 60% and can postpone the time before a quarter of the cities is infected with at least 10 days. However, the decrease in risk to be infected is not equally spread over the cities. In addition, *MinWPaths* can decrease the number of outbreaks depending on the initially infected city.

To conclude, based on the results presented in this thesis it is clear that the performance of the link removal methods is strongly affected by the network structure and the type of infection (e.g. value of transmission probability and recovery probability). Other important factors are the available budget and the size of the infection at the time of intervention. The direct application of the results in this thesis is limited as we only investigated a small number of parameter combinations with a low number of replications. Besides, we might not have perfect knowledge about the origin of infection, transmission probability or network topology in reality. It would also be recommended to further examine the differences between the results of this thesis and the paper of Nandi and Medal (2016).

For future work, the presented metapopulation simulation should be extended to relax some unrealistic assumptions such as for example the assumption that a city can only be infected once. Besides, as it is very harmful for a city to be infected, the link removal methods need to be adapted to incorporate a certain equality between nodes as it is unfair that all available budget is spend on one or a few nodes. Other interesting directions for further research are the combination of node and link removal methods, taking the uncertainty of the transmission probability into account and investigating other types of networks.

Despite the fact that the practical applicability of the results in this thesis might be limited, it showed that substantial improvements are possible. Furthermore, the methods and analysis presented in this thesis can be used to investigate which link removal method is most effective for other real life networks. The final trade-off that remains is then how much one is willing to pay to reduce the spread of infections.

8 Appendix

8.1 Heuristic algorithms

Algorithm 7 Link removal method: MinConnect

procedure Complete Algorithm
  input $G = (N, E)$
  initialization $N = N^0$, $E = E^0$
  set of links to remove $L = \emptyset$
  budget b
  set of infected nodes I
  set of susceptible nodes $N \backslash I$
  number of replications random process M
  while $|L| < b$ **do**
    $C_{best} = \infty$
    $e_{best} = \infty$
    for $e \in E$ **do**
      $E = E \backslash e$
      $C_e = 0$
      for $j \in 1, ..., M$ **do**
        randomly select $R = b - |L| - 1$ links $\in E$
        $E = E \backslash R$
        $c = 0$
        determine graph components of $G = (N, E)$
        for $i \in I$ **do**
          for $s \in S$ **do**
            if i and s belong to same component **then**
              $c = c + 1$
            end if
          end for
        end for
        $C_e = C_e + c$
        $E = E \cup R$
      end for
      $E = E \cup e$
      if $C_e < C_{best}$ **then**
        $e_{best} = e$
      end if
    end for
    $E = E \backslash e_{best}$
    $L = L \cup e_{best}$
  end while
  return E
end procedure

Algorithm 8 Link removal method: MinAtRisk

```
procedure Complete Algorithm
    input G = (N, E)
    initialization N = N^0, E = E^0
    set of links to remove L = ∅
    budget b
    set of infected nodes I
    set of susceptible nodes N\I
    number of replications random process M
    while |L| < b do
        A_best = ∞
        e_best = ∞
        for e ∈ E do
            E = E\e
            A_e = 0
            for j ∈ 1, ..., M do
                randomly select R = b − |L| − 1 links ∈ E
                E = E\R
                a = 0
                determine graph components of G = (N, E)
                for s ∈ S do
                    q = 0
                    for i ∈ I do
                        if i and s belong to same component then
                            q = 1
                        end if
                    end for
                    a = a + q
                end for
                A_e = A_e + a
                E = E ∪ R
            end for
            E = E ∪ e
            if A_e < A_best then
                e_best = e
            end if
        end for
        E = E\e_best
        L = L ∪ e_best
    end while
    return E
end procedure
```

Algorithm 9 Link removal method: MinPaths

```
procedure Complete Algorithm
    input G = (N, E)
    initialization N = N^0, E = E^0
    set of links to remove L = ∅
    budget b
    set of infected nodes I
    set of susceptible nodes N\I
    number of replications random process M
    check if all nodes have a connection
    TP = 0
    for j ∈ 1, ..., M do
        choose random starting node n
        while not all nodes visited do
            randomly select next node n + 1 connected to n
            if first visit node n + 1 then
                add link (n, n + 1) to tree T_j
            end if
        end while
        for s ∈ S do
            for i ∈ I do
                find shortest path between node s and i
                if path exists & path does not contain any infected nodes then
                    TP = TP + 1
                end if
            end for
            for s + 1 ∈ S do
                find shortest path between node s and s + 1
                if a path exists & path does not contain any infected nodes then
                    TP = TP + 1
                end if
            end for
        end for
    end for
    while |L| < b do
        P_best = 0
        e_best = ∞
        for e ∈ E do
            E = E\e
            count the number of paths removed P_e by counting the total
            number of paths TP containing link e
            E = E ∪ e
            if P_e > P_best then
                e_best = e
            end if
        end for
        E = E\e_best
        L = L ∪ e_best
    end while
    return E
end procedure
```

Algorithm 10 Link removal method: MinWPaths

procedure COMPLETE ALGORITHM
  input $G = (N, E)$
  initialization $N = N^0$, $E = E^0$
  set of links to remove $L = \emptyset$
  budget b
  set of infected nodes I
  set of susceptible nodes $N \backslash I$
  number of replications random process M
  check if all nodes have a connection
  TW = 0
  for $j \in 1, ..., M$ **do**
    choose random starting node n
    while not all nodes visited **do**
      randomly select next node $n + 1$ connected to n
      if first visit node $n + 1$ **then**
        add link $(n, n + 1)$ to tree T_j
      end if
    end while
    for $s \in S$ **do**
      for $i \in I$ **do**
        find shortest path between node s and i
        if path exists & path does not contain any infected nodes **then**
          compute weight w of path: multiply transmission probabilities of links
          $TW = TW + w$
        end if
      end for
      for $s + 1 \in S$ **do**
        find shortest path between node s and $s + 1$
        if a path exists & path does not contain any infected nodes **then**
          compute weight w of path: multiply transmission probabilities of links
          $TW = TW + w$
        end if
      end for
    end for
  end for
  while $|L| < b$ **do**
    $W_{best} = 0$
    $e_{best} = \infty$
    for $e \in E$ **do**
      $E = E \backslash e$
      compute the weight of paths removed W_e by computing the total
      weight of paths TW containing link e
      $E = E \cup e$
      if $W_e > W_{best}$ **then**
        $e_{best} = e$
      end if
    end for
    $E = E \backslash e_{best}$
    $L = L \cup e_{best}$
  end while
  return E
end procedure

Table 3: Selected subset of airports

	Airport (code)	Number of passengers (yearly)	Number of connections
1.	AMS	52,569,200	26
2.	ATL	94,431,224	17
3.	BKK	51,363,451	17
4.	CAN	52,450,262	15
5.	CDG	62,052,917	25
6.	CGK	60,137,347	11
7.	CLT	43,457,471	14
8.	DEN	52,556,359	15
9.	DFW	60,470,507	17
10.	DXB	66,431,533	18
11.	FRA	58,036,948	29
12.	HKG	59,588,081	18
13.	HND	68,906,509	17
14.	IAH	39,799,414	17
15.	ICN	41,679,758	24
16.	IST	51,304,654	16
17.	JFK	50,423,765	23
18.	KUL	47,498,127	15
19.	LAS	40,933,037	15
20.	LAX	66,667,619	23
21.	LHR	72,368,061	24
22.	MAD	39,717,850	15
23.	MIA	40,562,948	17
24.	MUC	38,672,644	21
25.	ORD	66,777,161	23
26.	PEK	83,712,355	23
27.	PHX	40,341,614	13
28.	PVG	47,189,849	21
29.	SFO	44,945,760	22
30.	SIN	53,726,087	15

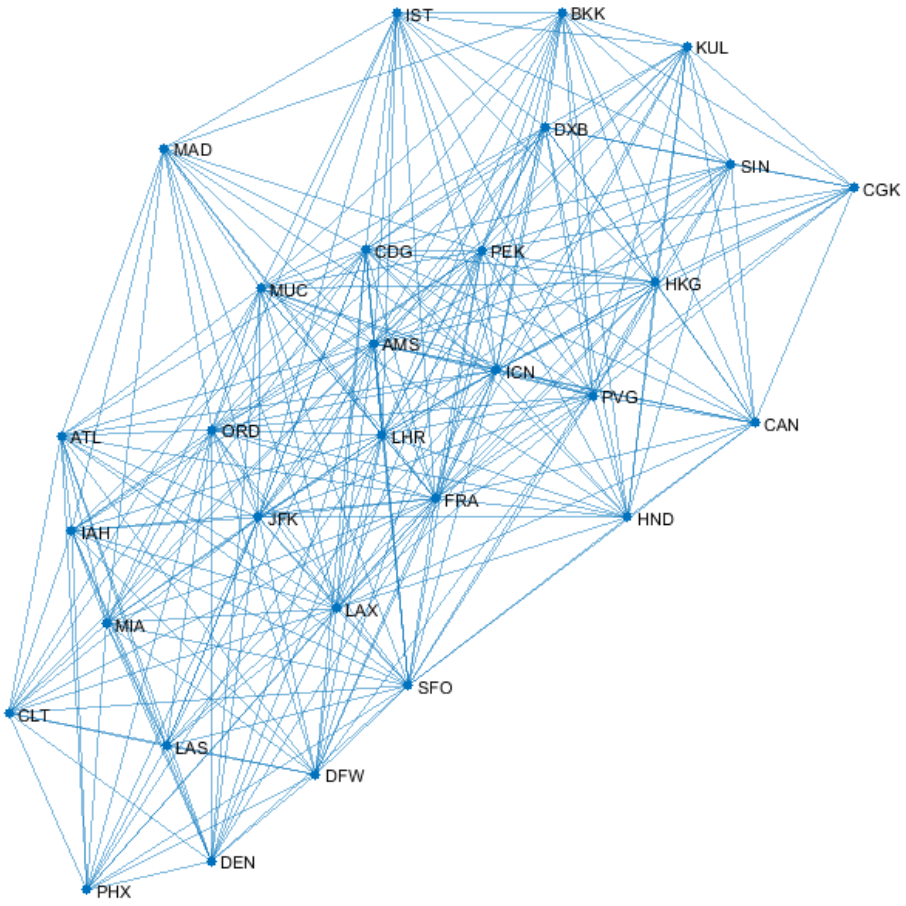
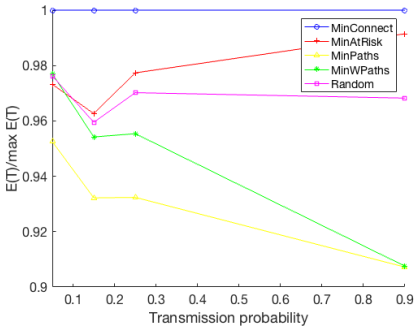


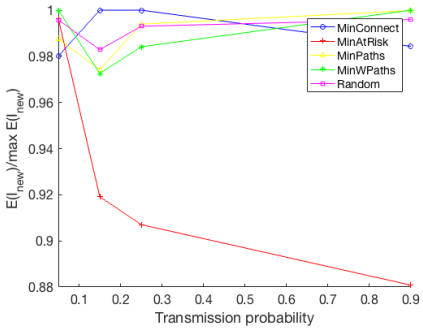
Figure 16: Selected subset of airports

8.3 Results - extra figures

Local Network



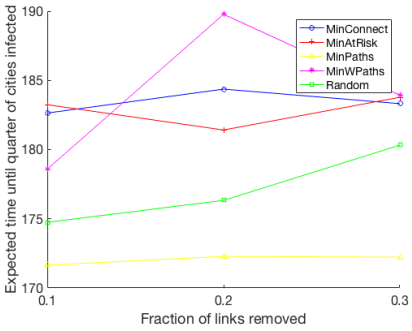
(a) Time to infect half of the susceptible nodes.



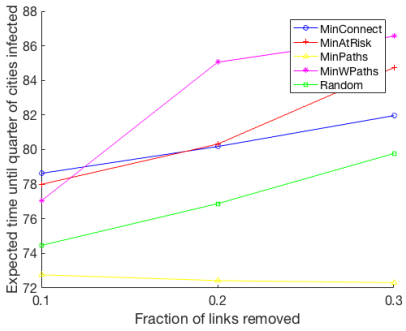
(b) Number of new infections.

Figure 17: Performance link removal methods for varying transmission probability τ_I. Initial fraction of infected nodes $i = 0.3$, fraction of links removed $r = 0.1$, recovery probability $\rho = 0.15$.

Metapopulation Network

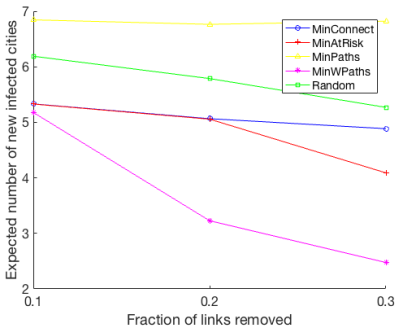


(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

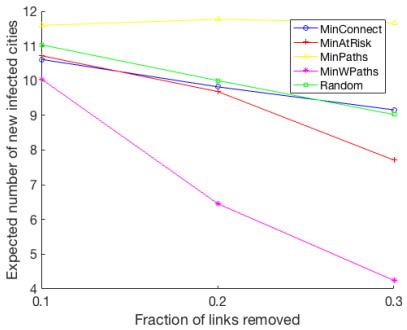


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 18: Performance link removal methods in terms of time to infect a quarter of the susceptible nodes for varying fraction of removed links r. Initial infected city: Amsterdam.

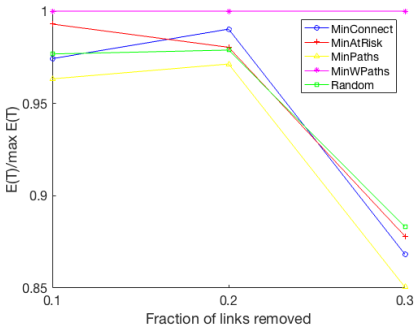


(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

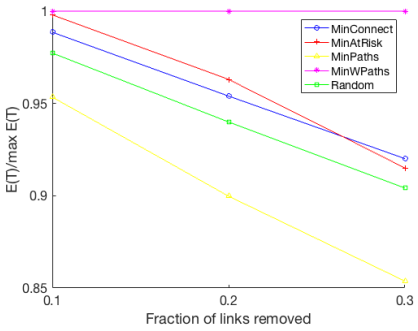


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 19: Performance link removal methods in terms of number of new infections for varying fraction of removed links r. Initial infected city: Amsterdam.

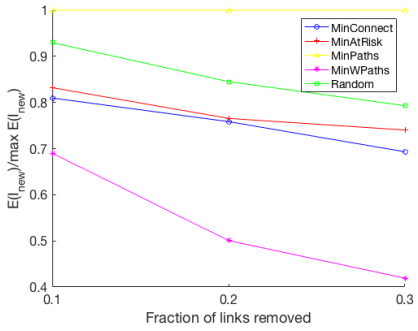


(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

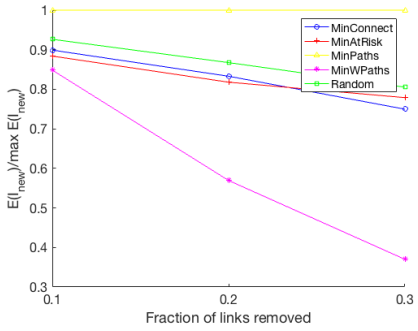


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 20: Performance link removal methods in terms of time to infect a quarter of the susceptible nodes for varying fraction of removed links r. Initial infected city: Madrid.

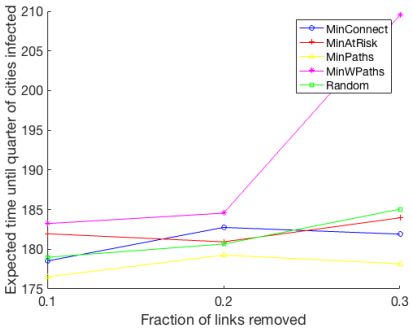


(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

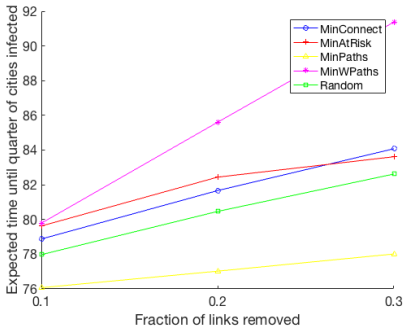


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 21: Performance link removal methods in terms of number of new infections for varying fraction of removed links r. Initial infected city: Madrid.

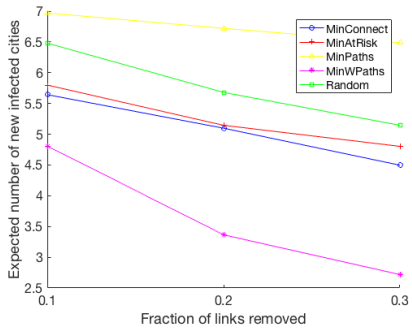


(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

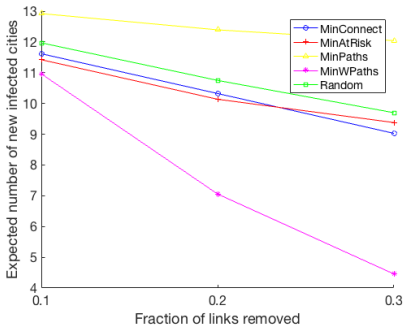


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 22: Performance link removal methods in terms of time to infect a quarter of the susceptible nodes for varying fraction of removed links r. Initial infected city: Madrid.

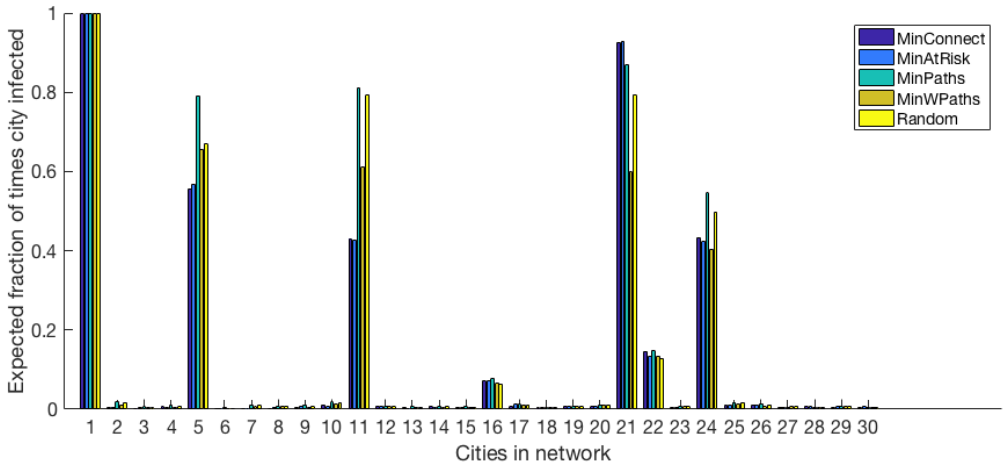


(a) Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.1$.

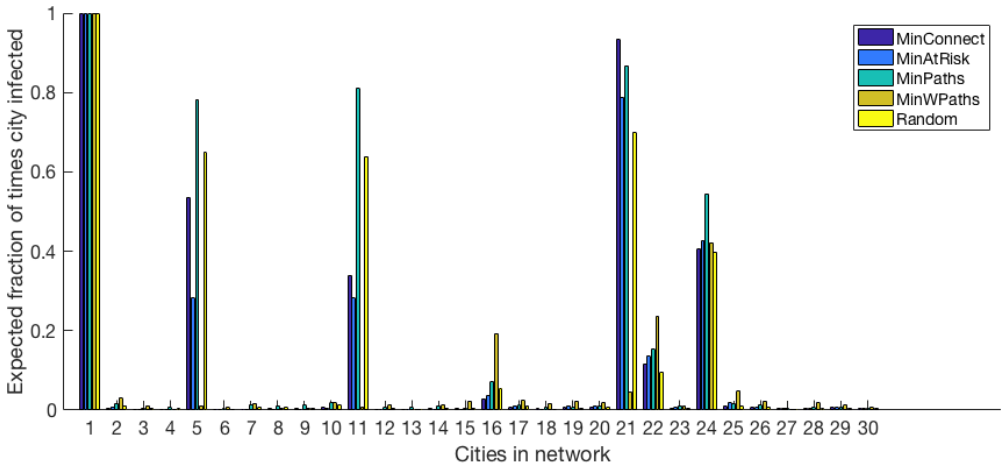


(b) Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$.

Figure 23: Performance link removal methods in terms of number of new infections for varying fraction of removed links r. Initial infected city: Madrid.

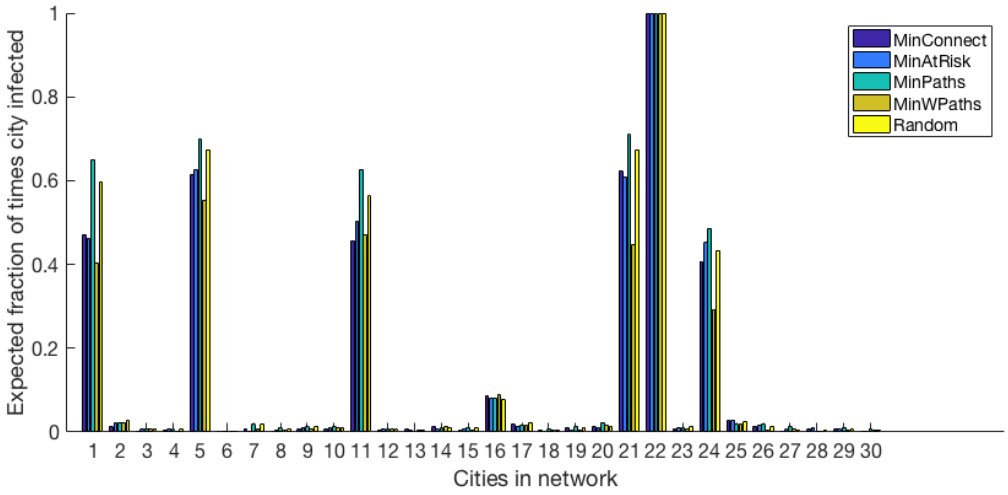


(a) Fraction of removed links $r = 0.1$.

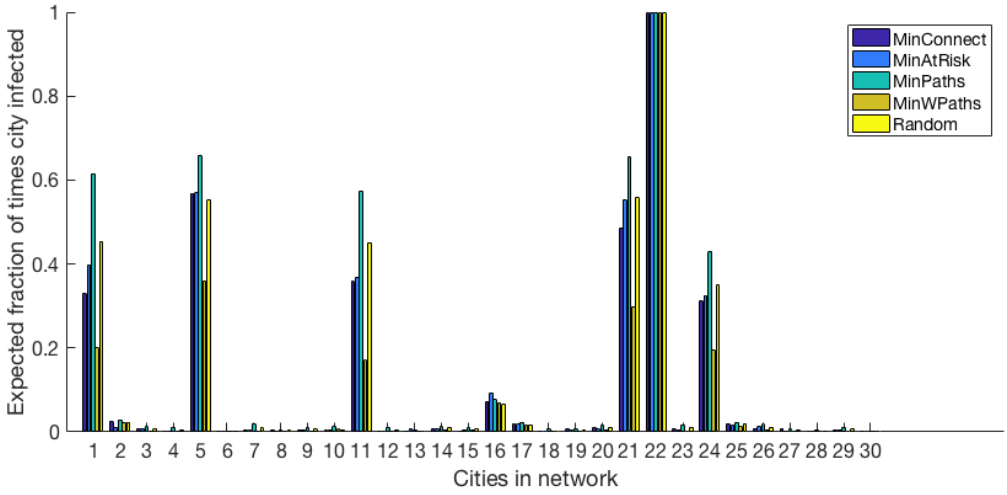


(b) Fraction of removed links $r = 0.3$.

Figure 24: Performance link removal methods in terms of fraction of times a certain city is infected for different fractions of removed links r. Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.10$. Initial infected city: Amsterdam.

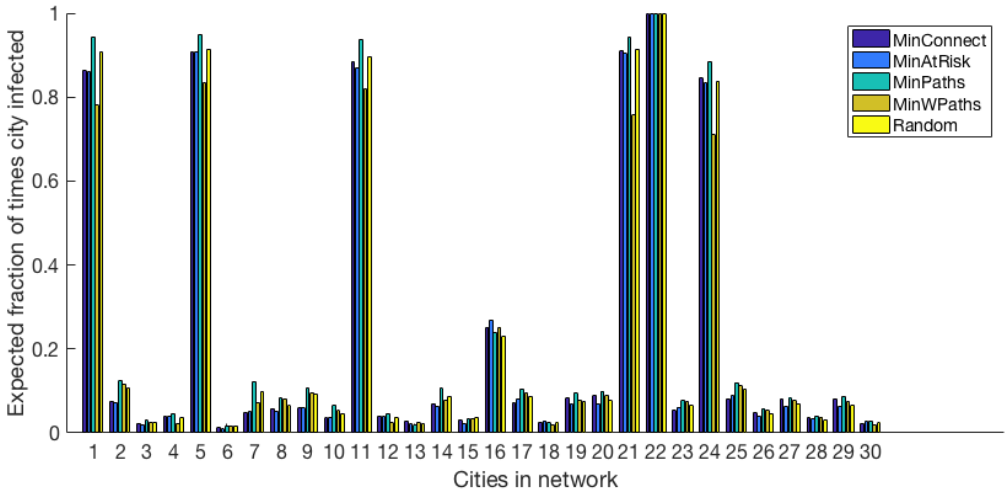


(a) Fraction of removed links $r = 0.1$.

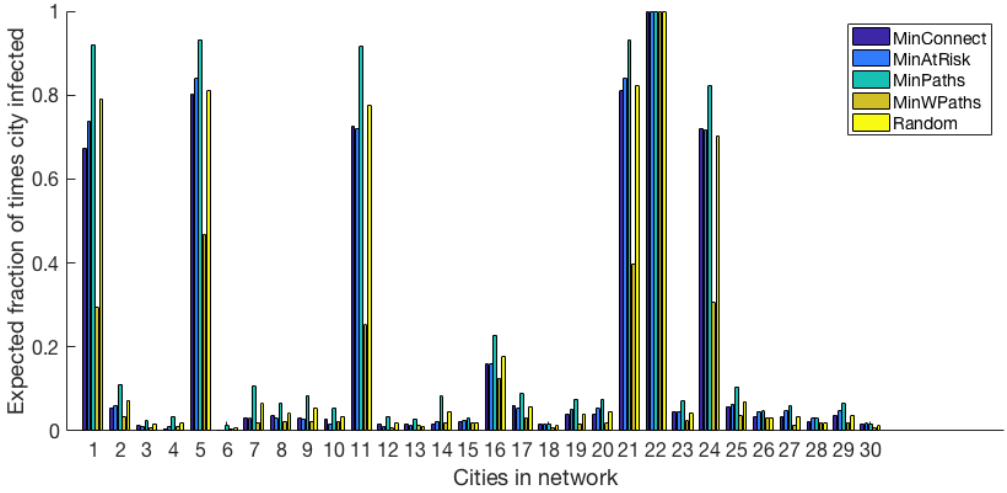


(b) Fraction of removed links $r = 0.3$.

Figure 25: Performance link removal methods in terms of fraction of times a certain city is infected for different fractions of removed links r. Scenario I: transmission probability $\tau_I = 0.15$, recovery probability $\rho = 0.10$. Initial infected city: Madrid.



(a) Fraction of removed links $r = 0.1$.



(b) Fraction of removed links $r = 0.3$.

Figure 26: Performance link removal methods in terms of fraction of times a certain city is infected for different fractions of removed links r. Scenario II: transmission probability $\tau_I = 0.9$, recovery probability $\rho = 0.15$. Initial infected city: Madrid.

9 References

Addis, B., Di Summa, M., & Grosso, A. (2013). Identifying critical nodes in undirected graphs: complexity results and polynomial algorithms for the case of bounded treewidth. *Discrete Applied Mathematics*, *161*(16), 2349–2360.

Airports Council International. (2013). *Total passenger traffic*. Retrieved from http://www.aci.aero/Data-Centre/Annual-Traffic-Data/Passengers/2013-final.

Albert, R. & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*(1), 1–47.

Bansal, S., Grenfell, B. T., & Meyers, L. A. (2007). When individual behaviour matters: homogeneous and network models in epidemiology. *Journal of the Royal Society Interface*, *4*(16), 879–891.

Barabási, A.-L. (2016). *Network science: the Barabási-Albert model*. Cambridge University Press.

Barabási, A.-L. & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Boginski, V. & Commander, C. W. (2009). Identifying critical nodes in protein-protein interaction networks. *Clustering Challenges in Biological Networks*, 153–167.

Brown, G., Carlyle, M., Salmerón, J., & Wood, K. (2006). Defending critical infrastructure. *Interfaces*, *36*(6), 530–544.

Callaway, D. S., Newman, M. E., Strogatz, S. H., & Watts, D. J. (2000). Network robustness and fragility: percolation on random graphs. *Physical Review Letters*, *85*(25), 1–4.

Capar, I. & Kuby, M. (2012). An efficient formulation of the flow refueling location model for alternative-fuel stations. *IIE Transactions*, *44*(8), 622–636.

Chen, W., Wang, Y., & Yang, S. (2009). Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 199–208). ACM.

Chung, N. N., Chew, L. Y., Zhou, J., & Lai, C. H. (2012). Impact of edge removal on the centrality betweenness of the best spreaders. *EPL (Europhysics Letters)*, *98*(5), 1–11.

Di Summa, M., Grosso, A., & Locatelli, M. (2012). Branch and cut algorithms for detecting critical nodes in undirected graphs. *Computational Optimization and Applications*, *53*(3), 649–680.

Domingos, P. & Richardson, M. (2001). Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 57–66). ACM.

Enns, E. A., Mounzer, J. J., & Brandeau, M. L. (2012). Optimal link removal for epidemic mitigation: a two-way partitioning approach. *Mathematical Biosciences*, *235*(2), 138–147.

Grimaldi, R. P. (2004). An introduction to graph theory. In T. Dollevoet (Ed.), *Discrete mathematics* (Chap. 3, pp. 109–175). Harlow, England: Pearson.

He, J., Liang, H., & Yuan, H. (2011). Controlling infection by blocking nodes and links simultaneously. In *International workshop on internet and network economics* (pp. 206–217). Springer.

Kang, C. & Goldman, A. (2016, December). In Washington pizzeria attack, fake news brought real guns. *The New York Times*.

Kempe, D., Kleinberg, J., & Tardos, É. (2003). Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 137–146). ACM.

Kermack, W. O. & McKendrick, A. G. (1927). A contribution to the mathematical theory of epidemics. In *Proceedings of the Royal Society of London A: mathematical, physical and engineering sciences* (Vol. 115, *772*, pp. 700–721). The Royal Society.

Kimura, M., Saito, K., & Motoda, H. (2009). Blocking links to minimize contamination spread in a social network. *Transactions on Knowledge Discovery from Data (TKDD)*, *3*(2), 1–22.

Koch, D., Illner, R., & Ma, J. (2013). Edge removal in random contact networks and the basic reproduction number. *Journal of Mathematical Biology*, 1–22.

Kuhlman, C. J., Tuli, G., Swarup, S., Marathe, M. V., & Ravi, S. (2013). Blocking simple and complex contagion by edge removal. In *Proceedings of the 13th international conference on data mining (ICDM)* (pp. 399–408). IEEE.

Latora, V. & Marchiori, M. (2004). How the science of complex networks can help developing strategies against terrorism. *Chaos, Solitons & Fractals*, *20*(1), 69–75.

Marcelino, J. & Kaiser, M. (2012). Critical paths in a metapopulation model of H1N1: efficiently delaying influenza spreading through flight cancellation.

Nandi, A. K. & Medal, H. R. (2016). Methods for removing links in a network to minimize the spread of infections. *Computers & Operations Research*, *69*, 10–24.

Newman, M. E., Forrest, S., & Balthrop, J. (2002). Email networks and the spread of computer viruses. *Physical Review E*, *66*(3), 1–4.

Tong, H., Prakash, B. A., Eliassi-Rad, T., Faloutsos, M., & Faloutsos, C. (2012). Gelling, and melting, large graphs by edge manipulation. In *Proceedings of the 21st ACM international conference on information and knowledge management* (pp. 245–254). ACM.

Veremyev, A., Prokopyev, O. A., & Pasiliao, E. L. (2014). An integer programming framework for critical elements detection in graphs. *Journal of Combinatorial Optimization*, *28*(1), 233–273.

Wang, Y., Chakrabarti, D., Wang, C., & Faloutsos, C. (2003). Epidemic spreading in real networks: an eigenvalue viewpoint. In *Proceedings of the 22nd international symposium on reliable distributed systems* (pp. 25–34). IEEE.

World Health Organization. (2016). *Progress report 2016: prevent HIV, test and treat all*. World Health Organization.

Yang, H.-X., Wu, Z.-X., & Wang, B.-H. (2013). Suppressing traffic-driven epidemic spreading by edge-removal strategies. *Physical Review E*, *87*(6), 1–5.