

Greening the Internet by putting networks to sleep

Bachelor Thesis

Author(s):

Furrer, Andri

Publication date:

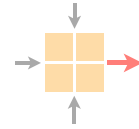
2022-12

Permanent link:

<https://doi.org/https://doi.org/10.3929/ethz-b-000601681>

Rights / license:

[Creative Commons Attribution 4.0 International](#)



Greening the Internet by putting networks to sleep

Bachelor Thesis

Author: Andri Furrer

Tutor: Dr. Romain Jacob

Supervisor: Prof. Dr. Laurent Vanbever

September to December 2022

Acknowledgements

I would like to express my deepest gratitude to my tutor Dr. Romain Jacob for his guidance, important advice and helpful notes for my work. I appreciated learning from him and am thankful for his insights on how things should get done.

Special thanks to Prof. Dr. Laurent Vanbever for granting me the opportunity to work on this project.

I thank Pascal Bruder and Sabine Denant for their valuable feedback and constructive criticism of my report. Finally, I thank my family and friends for their much-valued opinions and everlasting support.

Abstract

Today's network topologies are designed to withstand peak traffic and network failures. That results in a more extensive redundancy than needed for operating the network safely during the regular or underused operation. This project aims to determine to what extent redundancy can be reduced without compromising the functionality of the network.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Background and related work	2
1.3	Task and Goals	3
1.4	Overview	3
2	Dataset	4
2.1	Topology Zoo	4
2.1.1	Overview	4
2.1.2	GraphML	5
2.1.3	Dataset preparation	7
2.2	NetworkX	7
3	Methods	8
3.1	Link removal	8
3.1.1	Minimum cut	8
3.1.2	Minimum cut with specific values	8
3.1.3	Minimum cut with specific values and random seeds	8
3.2	Node removal	9
3.3	Node-link removal	9
3.4	Energy analysis	10
4	Results	11
4.1	Link removal	11
4.2	Node removal	14
4.3	Node-link removal	18
4.4	Energy analysis	20
4.4.1	Energy value pair 1/1	20
4.4.2	Energy value pair 10/1	22
5	Discussion	24
	References	25

List of Figures

1.1	A graph that shows the typical daily input and output traffic of the SwissIX infrastructure. Peak traffic of the output is at 431.95 Gigabits per second (Gbps), while the lowest point is at roughly 120 Gbps with an average of 306.78 Gbps. Input traffic peaks at 432.14 Gbps, and the lowest point is around 120 Gbps with an average of 306.79 Gbps [1].	2
2.1	The Switch backbone fibre network of Switzerland from the year 2010 [2].	4
4.1	The CDF for the greedy link removal algorithm. One can see that there are roughly 20 graphs where no edge could be removed and the graph with the highest ratio is still connected when over 80 % of its edges were removed.	12
4.2	The CDFs for the link removal algorithm with minimum cut values of 1 to 5. One can clearly see the shift of the curves to the left when the values are increased as well as the rising number of graphs where no edge could be removed.	13
4.3	The plotted CDF for the node removal algorithm with a core-to-edge ratio of 50 % based on the seeds' average. The curve is quite steep showing that the ratio of removed nodes is somehow nearby for all graphs of the dataset.	16
4.4	The plotted CDFs for the node removal algorithm based on the seeds' average. The curves are less steep for higher ratios meaning a much larger discrepancy between the graphs.	17
4.5	The scatter plot for the saved energy for each graph and core-to-edge ratio and energy units of 1/1 for node/edge. The outliers for a core-to-edge ratio of 0 % are graphs with many edges that can be removed by only applying the link removal algorithm.	20
4.6	The box plot for the saved energy for each core-to-edge ratio and energy units of 1/1 for node/edge. Within the box lay 50 % of all the points, that is the interquartile range (IQR) which is defined as the difference between the 75th and 25th percentiles of the data. The whiskers have in general a length of 1.5*IQR unless no data point is at this location, in which case, they are shortened until the next closest point. One can now observe, that for a majority of the graphs the saved energy lies much closer together, than the scatter plot 4.5 might have suggested.	21
4.7	The scatter plot for the saved energy for each graph and core-to-edge ratio and energy units of 10/1 for node/edge. For a core-to-edge ratio of 0 % much less energy can be saved compared to figure 4.5. In addition, the points are initially less spread out but this increases for higher core-to-edge ratios.	22

4.8 The box plot for the saved energy for each core-to-edge ratio and energy units of 10/1 for node/edge. Within the box lay 50 % of all the points, that is the interquartile range (IQR) which is defined as the difference between the 75th and 25th percentiles of the data. The whiskers have in general a length of $1.5 \cdot \text{IQR}$ unless no data point is at this location, in which case, they are shortened until the next closest point. Many graphs now have a small potential to save any energy by just turning links off. Only when more core nodes are removed do the values add up. 23

List of Tables

4.1	The table for the seed statistics of the link removal algorithm. It is interesting to see, that for a minimum cut value of 1, no difference between the ten seeds occurs. Larger minimum cut values have some difference, the maximum for one graph lies around 10 %, but the mean over the complete dataset is negligible.	11
4.2	The table for the link removal statistics is based on the seeds' average. As for each minimum cut value a graph exists, where no edge could be removed, at least one graph must exist which is already minimally connected when extracted from the Topology Zoo. On the other hand, there were graphs where a considerable amount of edges could be removed even for higher minimum cut values, meaning that they are well connected, possibly with many duplicated edges.	12
4.3	The table for the seed statistics of the node removal algorithm. The higher the core-to-edge ratio the higher the maximum occurred seed difference and the smaller the number of graphs where no difference between the seed occurred. This makes sense because, with more nodes to remove, more freedom of choosing a node exists which eventually can lead to a local optimum. These effects decrease for a core-to-edge ratio of 80 % and higher since almost all nodes are then removed. The same goes for the mean of the seed difference, though this number is overall quite small. . . .	14
4.4	The table for the node removal statistics is based on the seeds' average. Some graphs in the dataset have a small minimum of removed nodes, even for high core-to-edge ratios and some have a high maximum early on. The mean of removed nodes does not correlate to the core-to-edge ratio for small ratios, which means some potential core nodes cannot be removed without disconnecting the entire network.	15
4.5	The table for the seed differences for the link removal algorithm conducted on the obtained graphs after the node removal algorithm was applied. No different values for the mean of the removed edges between the seeds occurred.	18
4.6	The table for the statistics of the link removal based on one random seed. The mean value decreases consistently but much slower than the median value. An explanation is, that for higher core-to-edge ratios few large networks have many leftover edges where most smaller networks are already depleted.	19

Chapter 1

Introduction

1.1 Motivation

The technologies and products of the information and communication technology (ICT) industry are omnipresent in our daily life. This is represented in the total worldwide spending in the ICT sector. According to the International Data Corporation (IDC), spending grew to more than 5 trillion USD in the year 2021, and due to new upcoming technologies, the sector is still rapidly growing [3]. Unsurprisingly, the ICT industry is also responsible for a large amount of the global carbon footprint. According to Charlotte Freitag et al., the ICT sector estimated carbon footprint was 1.8 % -2.8 % of the worldwide greenhouse gas emissions in 2020 and, if supply chains are also included and the truncation error has been adjusted, the carbon footprint rises to 2.1 % -3.9 % [4]. Belkhir and Elmeligi calculated in 2018 a minimum of 3.06 % and a maximum value of 3.6 % of the total worldwide greenhouse gas emissions caused by the ICT sector for 2020 [5]. Charlotte Freitag et al. split this carbon footprint into three categories: networks, user devices, and data centres. Although these categories are intertwined, this project focuses primarily on networks. Each analysed report yields different percentages for the network category ranging from 22 % according to Belkhir and Elmeligi to 35 % according to Andrae and Elder [4, 5, 6].

While exact numbers are hard to estimate, it is clear that networks have a substantial carbon footprint. A contributing factor is the design of today's networks, which are built for peak usage and need redundancy in case of failures. This can be verified by observing the traffic these networks handle over time. An example is the daily, weekly, monthly, and yearly internet traffic graph of the Swiss Internet Exchange (SwissIX) [7]. Figure 1.1 shows that traffic varies significantly over time, and it can be seen that at night traffic is almost 75 % down. During the week, traffic is significantly higher than on the weekend, namely Saturday and Sunday. This raises the question if we could scale down the network when traffic is low and, if so, how much energy could we save.

One way to counter this overcapacity is turning network devices off, hence reducing the capacity and increasing the usage rate of the devices. Naturally, the turned-off devices do not consume any energy. With the traffic patterns described above, an enormous potential exists to save energy in times of low-use rate. However, it should be mentioned that a use rate of 100 % is not reachable nor desirable.

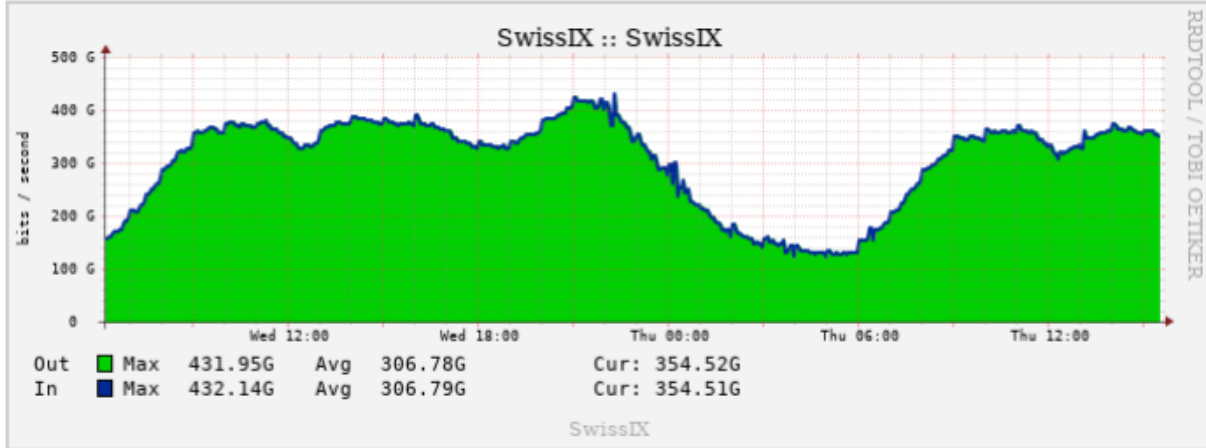


Figure 1.1: A graph that shows the typical daily input and output traffic of the SwissIX infrastructure. Peak traffic of the output is at 431.95 Gigabits per second (Gbps), while the lowest point is at roughly 120 Gbps with an average of 306.78 Gbps. Input traffic peaks at 432.14 Gbps, and the lowest point is around 120 Gbps with an average of 306.79 Gbps [1].

1.2 Background and related work

After decades of innovation, development, and implementation of new technologies in the ICT sector, a growing awareness of energy efficiency and sustainability has taken shape. It is supported by the energy transition and spareness of electricity and resources. Recent examples of steps in that direction for some areas of the ICT sector are the workshop “HotCarbon 2022: 1st Workshop on Sustainable Computer Systems Design and Implementation” that took place in San Diego, USA. According to the organizers, the goal was to bring researchers and practitioners in computer and network systems together to address the sustainability of computer systems. It is stated that the research community “has had a relentless focus on the implications of increased scale and raw performance of consumer devices, cloud systems and datacenters, as well as mobile and wireless networks. We believe we also need to address the side-effects of scale through innovative approaches to how we build, deploy, operate and retire our creation” [8].

Another example is the Schloss Dagstuhl seminar “Power and Energy-aware Computing on Heterogeneous Systems (PEACHES)”, taking place in Wadern, Germany. It is manifested that “There is an urgent need to understand how computing fits into the broader picture of our energy consumption, and what role there is for computing to reduce our carbon footprint and help to accelerate the transition to renewables” [9].

Finally, the “IAB workshop on Environmental Impact of Internet Applications and Systems, 2022”, organised as an online meeting, discussed the Internet’s environmental impact with a focus on technical aspects ranging from the Internet ecosystem to devices, data centres, etc., among similar topics [10].

These are all examples of the current focus on network devices’ environmental aspects, energy efficiency, and sustainability. These efforts will only increase in importance in the future as the demand grows in the ICT sector.

This project aims to roughly estimate how many devices and links could be turned off in real network topologies when some assumptions are given. Hence, the project contributes to the present discussion on sustainability and opens the path for similar projects in a related direction.

1.3 Task and Goals

The task is to investigate how many devices of networks from the Topology Zoo described in chapter 2 can be turned off [11]. As explained in the motivation, turning off devices saves energy. The potential energy savings can be calculated when a specific energy value is assigned to the devices.

The dataset contains 261 networks of all shapes and sizes around the globe, providing a simplified view of some real networks. The turning off of devices is achieved by removing nodes and edges from these topologies. To simplify, we do not consider actual traffic volume and network capacity. We only look at the connectivity of the networks to maintain basic functionality by reaching all devices. The goal is to see how many nodes and, in a second step, how many links can be removed without losing connectivity.

After estimating the number of devices we can turn off, an arbitrary energy value gets assigned to nodes and edges. One then receives the potential energy savings and can conclude the energy savings behaviour regarding the removed nodes and edges and, thus, the networks.

1.4 Overview

Chapter 2 describes the dataset and the used NetworkX package. The next chapter 3 presents the methods used to reach the goals defined above and is followed by the results and a discussion of these and the project in general in chapter 4 and 5.

Chapter 2

Dataset

2.1 Topology Zoo

2.1.1 Overview

The University of Adelaide provides the Internet Topology Zoo, a collection of network topologies from all over the world [12]. The dataset contains 261 networks that are all undirected graphs. Some are multigraphs, meaning they have multiple edges between their nodes. A typical example is shown below.

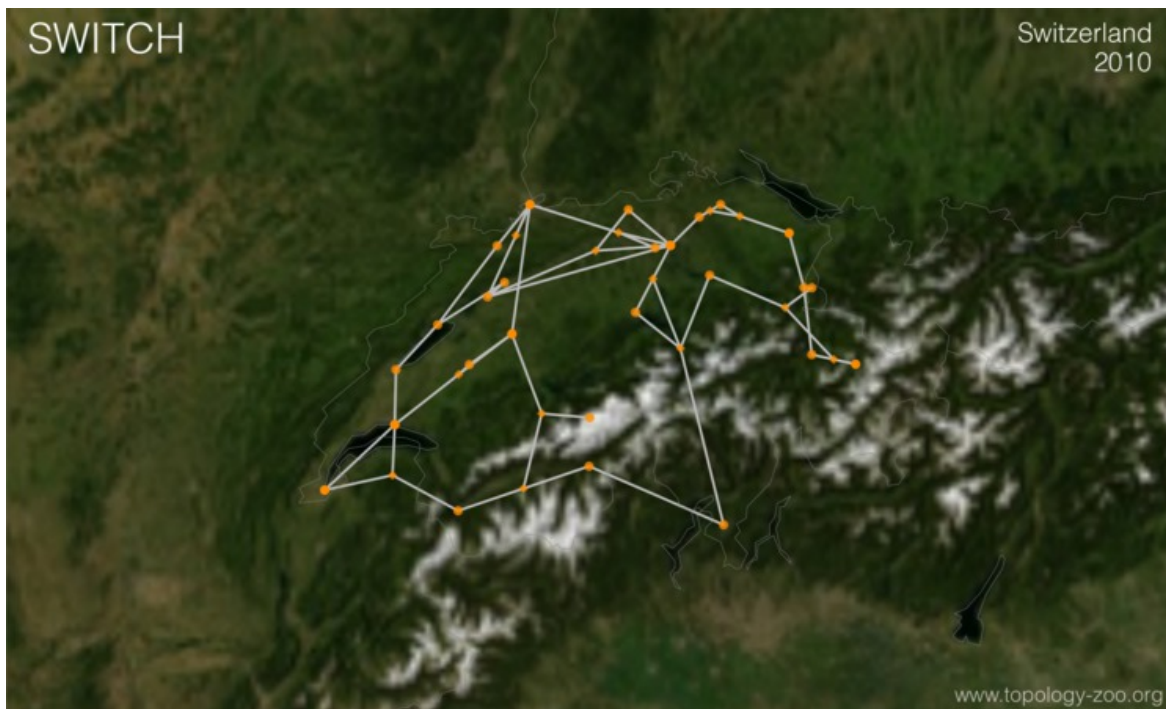


Figure 2.1: The Switch backbone fibre network of Switzerland from the year 2010 [2].

2.1.2 GraphML

The networks are stored in GraphML file format. GraphML is an XML-based file format for graphs. Each network is saved in an individual file. It builds upon an XML file containing the definition of the general graph attributes and the ones of node and edge followed by a graph tag. Among others, the general attributes include name, obtained date, location, and type of network. The graph tag part lists first the data of the general attributes. The tag is then filled up with all nodes and edges of the graph. Each node has a unique id attribute. Every edge has a source and target set to a node id, thus defining the edge unambiguously. At this point, it should be indicated once more that there are only undirected graphs in the Topology Zoo dataset. In an undirected graph, one edge's source and target node are strongly connected. Additional attributes for the edges and nodes, such as labels, longitude, and latitude, are listed at each node or edge. A snippet of the structure of the GraphML type is displayed below.

Listing 2.1: ARPANET from 1969, GraphML file

```
1 <?xml version="1.0" encoding="utf-8" ?><graphml xmlns="http://graphml.graphdrawing
  .org/xmlns" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns http://graphml.
  graphdrawing.org/xmlns/1.0/graphml.xsd">
2 <key attr.name="key" attr.type="int" for="edge" id="d35" />
3 <key attr.name="id" attr.type="string" for="edge" id="d34" />
4 <key attr.name="label" attr.type="string" for="node" id="d33" />
5 <key attr.name="Longitude" attr.type="double" for="node" id="d32" />
6 <key attr.name="id" attr.type="int" for="node" id="d31" />
7 <key attr.name="Country" attr.type="string" for="node" id="d30" />
8 <key attr.name="Latitude" attr.type="double" for="node" id="d29" />
9 <key attr.name="Internal" attr.type="int" for="node" id="d28" />
10 <key attr.name="Testbed" attr.type="int" for="graph" id="d27" />
11 <key attr.name="LastProcessed" attr.type="string" for="graph" id="d26" />
12 <key attr.name="DateYear" attr.type="string" for="graph" id="d25" />
13 <key attr.name="NetworkDate" attr.type="string" for="graph" id="d24" />
14 <key attr.name="Transit" attr.type="int" for="graph" id="d23" />
15 <key attr.name="Developed" attr.type="int" for="graph" id="d22" />
16 <key attr.name="Creator" attr.type="string" for="graph" id="d21" />
17 <key attr.name="Layer" attr.type="string" for="graph" id="d20" />
18 <key attr.name="LastAccess" attr.type="string" for="graph" id="d19" />
19 <key attr.name="DateMonth" attr.type="string" for="graph" id="d18" />
20 <key attr.name="DateModifier" attr.type="string" for="graph" id="d17" />
21 <key attr.name="SourceGitVersion" attr.type="string" for="graph" id="d16" />
22 <key attr.name="IX" attr.type="int" for="graph" id="d15" />
23 <key attr.name="Customer" attr.type="int" for="graph" id="d14" />
24 <key attr.name="ToolsetVersion" attr.type="string" for="graph" id="d13" />
25 <key attr.name="label" attr.type="string" for="graph" id="d12" />
26 <key attr.name="Commercial" attr.type="int" for="graph" id="d11" />
27 <key attr.name="Backbone" attr.type="int" for="graph" id="d10" />
28 <key attr.name="DateType" attr.type="string" for="graph" id="d9" />
29 <key attr.name="Type" attr.type="string" for="graph" id="d8" />
30 <key attr.name="Version" attr.type="string" for="graph" id="d7" />
31 <key attr.name="Source" attr.type="string" for="graph" id="d6" />
32 <key attr.name="Access" attr.type="int" for="graph" id="d5" />
33 <key attr.name="Provenance" attr.type="string" for="graph" id="d4" />
34 <key attr.name="Network" attr.type="string" for="graph" id="d3" />
35 <key attr.name="GeoExtent" attr.type="string" for="graph" id="d2" />
36 <key attr.name="GeoLocation" attr.type="string" for="graph" id="d1" />
37 <key attr.name="DateObtained" attr.type="string" for="graph" id="d0" />
38 <graph edgedefault="undirected">
```

```

39 <data key="d0">25/01/11</data>
40 <data key="d1">United States</data>
41 <data key="d2">Country</data>
42 <data key="d3">ARPANET</data>
43 <data key="d4">Secondary</data>
44 <data key="d5">0</data>
45 <data key="d6">http://som.csudh.edu/cis/lpress/history/arpamaps/</data>
46 <data key="d7">1.0</data>
47 <data key="d8">REN</data>
48 <data key="d9">Historic</data>
49 <data key="d10">1</data>
50 <data key="d11">0</data>
51 <data key="d12">Arpanet196912</data>
52 <data key="d13">0.3.34 dev-20120328</data>
53 <data key="d14">0</data>
54 <data key="d15">0</data>
55 <data key="d16">e278b1b</data>
56 <data key="d17"><</data>
57 <data key="d18">12</data>
58 <data key="d19">25/01/11</data>
59 <data key="d20">ARPA</data>
60 <data key="d21">Topology Zoo Toolset</data>
61 <data key="d22">1</data>
62 <data key="d23">0</data>
63 <data key="d24">1969_12</data>
64 <data key="d25">1969</data>
65 <data key="d26">2011_09_01</data>
66 <data key="d27">0</data>
67 <node id="0">
68   <data key="d28">1</data>
69   <data key="d29">37.45383</data>
70   <data key="d30">United States</data>
71   <data key="d31">0</data>
72   <data key="d32">-122.18219</data>
73   <data key="d33">SRI</data>
74 </node>
75 <node id="1">
76   <data key="d28">1</data>
77   <data key="d29">34.42083</data>
78   <data key="d30">United States</data>
79   <data key="d31">1</data>
80   <data key="d32">-119.69819</data>
81   <data key="d33">USCB</data>
82 </node>
83 <node id="2">
84   <data key="d28">1</data>
85   <data key="d29">34.05223</data>
86   <data key="d30">United States</data>
87   <data key="d31">2</data>
88   <data key="d32">-118.24368</data>
89   <data key="d33">UCLA</data>
90 </node>
91 <node id="3">
92   <data key="d28">1</data>
93   <data key="d29">40.76078</data>
94   <data key="d30">United States</data>
95   <data key="d31">3</data>
96   <data key="d32">-111.89105</data>
97   <data key="d33">UTAH</data>

```

```

98     </node>
99     <edge source="0" target="1">
100       <data key="d34">e0</data>
101       <data key="d35">0</data>
102     </edge>
103     <edge source="0" target="2">
104       <data key="d34">e1</data>
105       <data key="d35">0</data>
106     </edge>
107     <edge source="0" target="3">
108       <data key="d34">e2</data>
109       <data key="d35">0</data>
110     </edge>
111     <edge source="1" target="2">
112       <data key="d34">e3</data>
113       <data key="d35">0</data>
114     </edge>
115   </graph>
116 </graphml>

```

2.1.3 Dataset preparation

Aside from the first visualisation to get a feeling for the different topologies, filtering unconnected or unhelpful networks was necessary. Some graphs of the dataset were initially not connected due to some isolated nodes or smaller components. To not lose them for further application, all but the single largest component were removed and stored as a new graph. This step is justifiable because the removed isolated nodes or small components are independent of the other nodes. After all, the connectivity of the remaining graph remains unchanged. Hence the removed nodes have no significant effect on the possible results.

Further, the geographical location of all the nodes was reviewed to find possible duplications. These are redundant nodes and represent a reliability feature. The key idea is that these nodes can be safely removed without further investigation as long as we do not lose connectivity. This idea was skipped out later because the node removal algorithm handles these duplicated nodes. However, by searching for these nodes at the same location, others without longitude and latitude values were discovered. Most of them were nodes on the edge of a graph providing some functionality independent of their respective network locations. Some networks were then found consisting only of these nodes without location and possibly even without connectivity. This handful of graphs was removed from the dataset as part of the preparation process. This step is also justifiable because only a few graphs that consisted of nodes without a well-defined geographical location were lost.

2.2 NetworkX

An essential tool for this project was the NetworkX open-source Python library. This package provides many basic operations on graphs, including the handling of GraphML files. Graphs with nodes, edges, and attributes can easily be retrieved, visualised, and altered. These functionalities make the exploration of the Topology Zoo dataset possible and allow the implementation of desired algorithms. That is the link, node, and link-node removal to get the potential energy savings. NetworkX, for example, grants functions to get the number of nodes and edges, remove them, or add new ones. The provided functionality includes different forms of connectivity and centrality or ways to visualise graphs [13, 14].

Chapter 3

Methods

The first steps of the project involved getting familiar with the tools. This included the dataset as described in chapter 2 as well as NetworkX and some graph theory. The tasks were divided into link and node removal, then combined and finally, a conducted analysis of potential energy savings. The first part was to remove possible links from the original graphs of the dataset, meaning no nodes have been removed. This algorithm can then later be applied to any other graph, which is why it was an excellent point to start achieving the project's goals.

3.1 Link removal

3.1.1 Minimum cut

The first idea to remove links was to iterate over all links of a given graph and remove them if they did not disconnect the remaining graph when doing so ¹. This is essentially the minimum cut problem on any given subset of nodes from a graph with a value of 1. The minimum cut problem states how many cuts, i.e., removed links are necessary until one ends up with any two subsets of nodes. A subset can be an arbitrary partition of nodes of a given graph. The difference to the implemented algorithm is, that we want to remove as many links as possible until we reach the minimum cut value of 1 between any pair of nodes. The first version is the greedy approach with no redundancy being left. If any additional edge is removed, the graph gets disconnected. This algorithm yields minimally connected graphs where all dispensable edges are removed.

3.1.2 Minimum cut with specific values

The same principle was used above, but the algorithm was extended with a specific minimum cut value. A range of numbers from 1 to 5 allowed some redundancy. After the greedy minimum cut algorithm was applied by turning off any additional links, the entire network would disconnect immediately. When a larger value is used, the network or certain parts of it remain connected. These differences were examined to see what redundancy costs in terms of removed links.

3.1.3 Minimum cut with specific values and random seeds

In the last step, the link selection was randomised by using ten random seeds. The seeds randomised the process and allowed reproducibility as well. If links are tested and removed in a specific order,

¹NetworkX function *is_connected*: https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.components.is_connected.html

it may lead to some local optimum. We used ten random seeds to change the link removal order, which does not guarantee finding the maximum number of links one can remove from the graph but does make the search more robust.

3.2 Node removal

The next task was removing nodes from the graphs to reduce the number of devices turned on. Generally, networks consist of well-connected core nodes and sparsely connected edge nodes, that is, nodes on the edge of a network. The idea was to remove some of these core nodes to lower redundancy. The decision to label a node as a core was randomised and based on the degree centrality². The higher that value, the higher the chance that a node got the label core.

It is essential to know that edge nodes cannot be removed without losing access to them. On the other hand, the core nodes can be removed without changing the connectivity of edge nodes. That is why it seemed like a good idea to merge the identical nodes as described in chapter 2. These identical nodes were almost always at the core of a network. Hence if a device, indicated by a node, is present multiple times in one location, one could turn off the spare ones to reduce the number of active devices. These redundant nodes were merged to maintain an active link connection to all edge nodes. When n nodes were merged, then $n - 1$ nodes were removed from a graph.

Later we abandoned this merging of the nodes since the implementation of the algorithm handles it anyway. This ensures evading possible poor assumptions and preserves the universal aspects of the algorithm.

In the next phase, it was crucial to know how many and which of the remaining nodes could be removed without disconnecting the leftover graph. The key idea was to assign the labels *core* and *edge* to the nodes of a graph with a specific ratio between these two labels, the core-to-edge ratio. Eleven ratios from 0 to 1 with an increment of 0.1 were chosen to cover the whole spectrum. Then only the nodes assigned as core could be removed, and only if this removal would not disconnect the remaining graph. To check the latter condition, the attribute of articulation points comes in handy. An articulation point, also known as a cut vertex, is a node whose removal would increase the number of components of a graph. In other words, removing any of its cut vertices would disconnect a graph. After each successful node removal, all leftover nodes were checked if they are a cut vertex³. From these, a new node to remove was picked. To evade best or worst picks, the selection of cut vertices was randomised using a random seed with size ten, as was the case with the link removal.

3.3 Node-link removal

After successfully implementing the link and node removal, both algorithms were combined. First, all possible nodes through the node removal algorithm were removed for each core-to-edge ratio. This procedure yields $k_{node} = 11 * 10 = 110$ graphs, and on each sample, the link removal algorithm with ten seeds was run. This yields $k_{node-link} = 10 * k_{node} = 1100$ samples containing 11 ratios, ten node removal, and ten link removal seeds.

²NetworkX function *degree_centrality*: https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.degree_centrality.html

³NetworkX function *articulation_points*: https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.components.articulation_points.html

3.4 Energy analysis

In the last task, an analysis of potential energy savings was conducted. A specific energy value was assigned to each node and edge of a graph. This gives a particular energy value for each graph. Removing nodes and links from a graph effectively turns the devices off and, therefore, saves energy. If a device is turned off, we assumed the total energy assigned to it can be saved. With a core-to-edge ratio of 0, only links will be removed and no nodes. Hence we essentially have the only link removal part. When the core-to-edge ratio is incremented, removing nodes with the label core becomes possible. By steadily increasing it to a ratio of 1, an even more significant amount of nodes can be removed, and hence we save more and more energy.

Chapter 4

Results

4.1 Link removal

The project's first results were obtained after applying the greedy link removal algorithm on the graphs of the Topology Zoo. The greedy approach corresponds to a minimum cut value of 1 as explained in chapter 3. The algorithm returned the graph's total number of removed links, receiving a value for each seed and graph of the dataset. The graphs have a different total number of edges, so comparing the absolute value of removed edges is a bad idea. For a better comparison, the ratio of the removed edges to the total edges of a graph was calculated. To observe the influence of the seeds on the outcome, these obtained ratios were then compared for each seed.

Table 4.1 displays the percentage of graphs where all seeds remove the same amount of edges, the largest observed difference within the ten seeds of one graph and the overall mean. One can see that no difference for the minimum cut value of 1 occurs and only small differences for the mean for larger values. Hence, it is reasonable to take the average of the ten seeds for the edge removal ratio. The numbers are shown in table 4.2, where a fast decrease can be observed for increasing minimum cut values. This is no surprise, as fewer edges can be removed with the stronger conditions of a larger minimum cut value.

Minimum cut value	Graphs with no seed difference in %	Largest seed difference in %	Mean of seed difference in %
1	100	0	0
2	63.49	13	1.33
3	73.41	9	0.71
4	83.73	6	0.37
5	87.7	5	0.25

Table 4.1: The table for the seed statistics of the link removal algorithm. It is interesting to see, that for a minimum cut value of 1, no difference between the ten seeds occurs. Larger minimum cut values have some difference, the maximum for one graph lies around 10 %, but the mean over the complete dataset is negligible.

Minimum cut value	Minimum of removed links in %	Maximum of removed links in %	Median of removed links in %	Mean of removed links in %
1	0	86	21.5	22.27
2	0	81	16	16.24
3	0	76	7	9.61
4	0	70	3	6
5	0	66	0	4.14

Table 4.2: The table for the link removal statistics is based on the seeds' average. As for each minimum cut value a graph exists, where no edge could be removed, at least one graph must exist which is already minimally connected when extracted from the Topology Zoo. On the other hand, there were graphs where a considerable amount of edges could be removed even for higher minimum cut values, meaning that they are well connected, possibly with many duplicated edges.

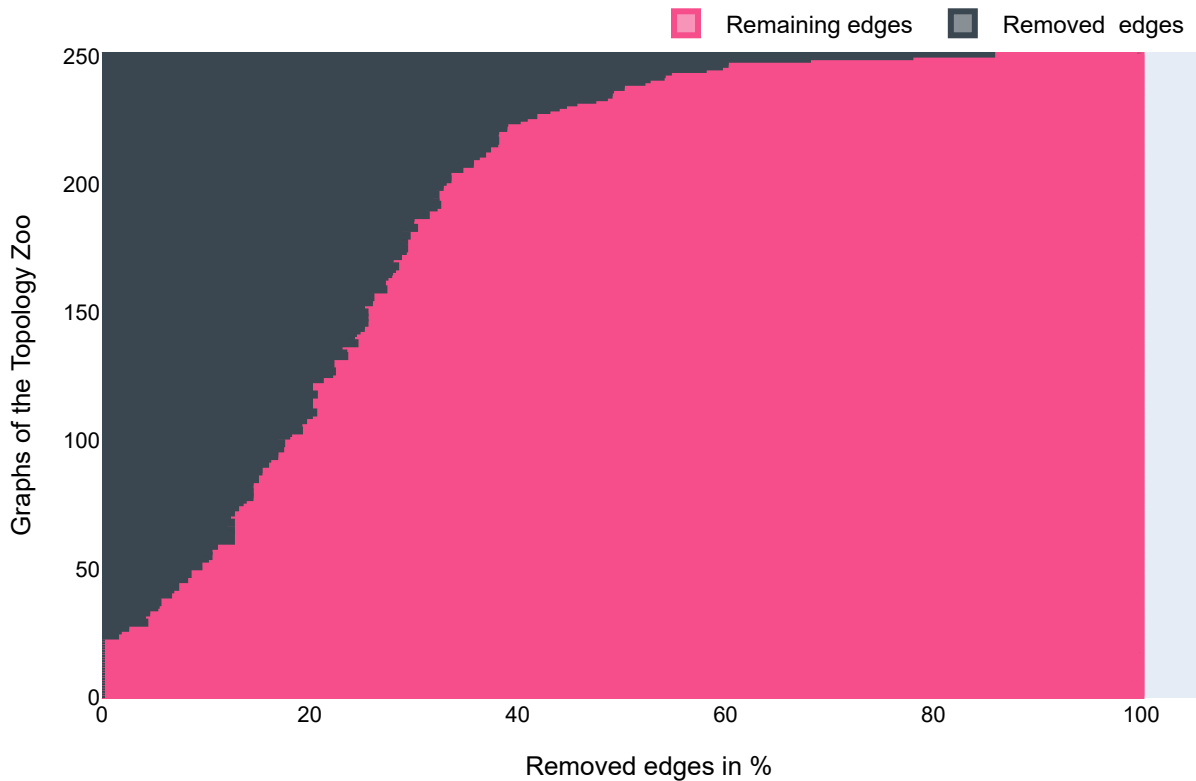


Figure 4.1: The CDF for the greedy link removal algorithm. One can see that there are roughly 20 graphs where no edge could be removed and the graph with the highest ratio is still connected when over 80 % of its edges were removed.

When the ratio of removed edges is sorted in ascending order, the cumulative distribution function (CDF) as shown in figure 4.1 can be retrieved, where the different edge removal ratios are displayed.

The plot shows, consistent with the numbers from table 4.2, that roughly half of the graphs are still connected when 20 % of the edges are removed. Around 80 % of the graphs have a ratio of 33 % or below. When more conservative minimum cut values are used, it could be expected that fewer edges are removed. Figure 4.2 shows the distinctions between the different values. For larger minimum cut values, the curve is shifting to the left, where fewer percentages of edges are removed. So there are fewer graphs where many edges could be removed. Also, it can be seen that for smaller minimum cut values the differences are more profound whereas for larger values differences weaken. This results in stronger effects on the number of removed edges at a change in lower minimum cut values.

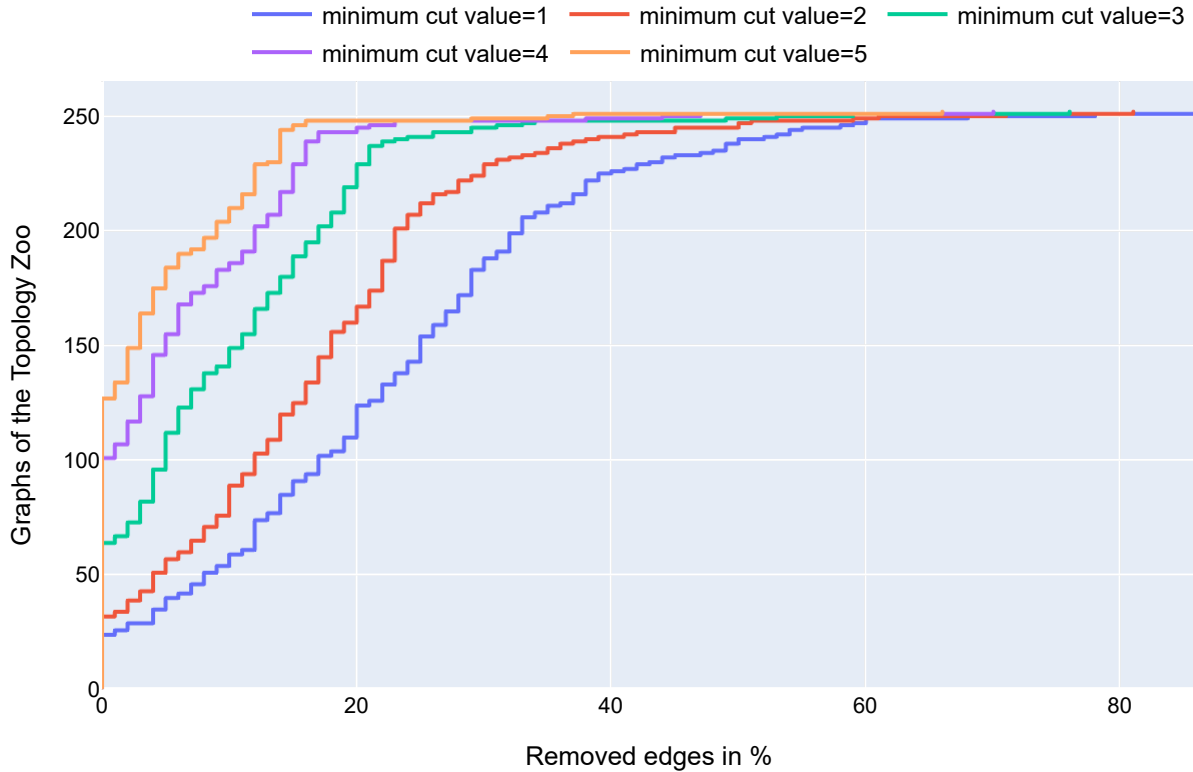


Figure 4.2: The CDFs for the link removal algorithm with minimum cut values of 1 to 5. One can clearly see the shift of the curves to the left when the values are increased as well as the rising number of graphs where no edge could be removed.

4.2 Node removal

The node removal algorithm applied to the original graphs of the Topology Zoo yielded the number of removed nodes for each seed and core-to-edge ratio. This number strongly depends on this ratio, determining how many nodes of the graphs are labelled as core and could possibly be removed.

To analyse the different results of the seeds, table 4.3 displays the mean value of the ten seeds of a graph, the maximum difference within ten seeds occurring in the dataset as well as the percentage of graphs where no seed difference occurs. In general, it can be said that the random picking of nodes does not significantly influence the outcome of how many nodes can be removed. No difference can be observed for most of the graphs across all core-to-edge ratios. That is why the average of the ten seeds was taken for the general statistics of the node removal algorithm.

Core-to-edge ratio in %	Graphs with no seed difference in %	Largest seed difference in %	Mean of seed difference in %
10	95.24	5	0.13
20	86.51	9	0.52
30	73.02	19	1.38
40	64.29	28	2.41
50	58.73	31	2.74
60	58.33	43	3.85
70	54.76	52	4.73
80	55.95	47	4.85
90	63.49	56	3.62
100	100	0	0

Table 4.3: The table for the seed statistics of the node removal algorithm. The higher the core-to-edge ratio the higher the maximum occurred seed difference and the smaller the number of graphs where no difference between the seed occurred. This makes sense because, with more nodes to remove, more freedom of choosing a node exists which eventually can lead to a local optimum. These effects decrease for a core-to-edge ratio of 80 % and higher since almost all nodes are then removed. The same goes for the mean of the seed difference, though this number is overall quite small.

The statistics for the node removal are shown in table 4.4. As expected, it can be seen that the mean of removed nodes over all graphs is steadily increasing. That being said, one might also expect that the mean value and the core-to-edge ratio would approximately correlate. But as it is, considerably fewer nodes could be removed for smaller core-to-edge ratios. This can be seen for a mean value of 31.56 % removed nodes and 50 % core-to-edge ratio.

Core-to-edge ratio in %	Minimum of removed nodes in %	Maximum of removed nodes in %	Median of removed nodes in %	Mean of removed nodes in %
10	0	25	7	7.32
20	0	33	12	12.28
30	6	50	17	18.15
40	8	50	25	25.05
50	12	57	31	31.56
60	15	75	40	40.4
70	20	83	51	50.52
80	28	100	65	62.79
90	35	100	82	80.76
100	100	100	100	100

Table 4.4: The table for the node removal statistics is based on the seeds' average. Some graphs in the dataset have a small minimum of removed nodes, even for high core-to-edge ratios and some have a high maximum early on. The mean of removed nodes does not correlate to the core-to-edge ratio for small ratios, which means some potential core nodes cannot be removed without disconnecting the entire network.

Figure 4.3 shows the CDF plot for all graphs with a core-to-edge ratio of 50 % and where the average of the seeds was taken. It can be seen that from all graphs at least some nodes were removed and around 80 % of the graphs have a node removal ratio of 40 % or lower, indicating some of the nodes labelled as core are essential for the network connectivity.

Figure 4.4 shows the percentage of removed nodes for all core-to-edge ratios. One can see that the curves of lower ratios are less steep and the differences between the graphs increase for higher ratios.

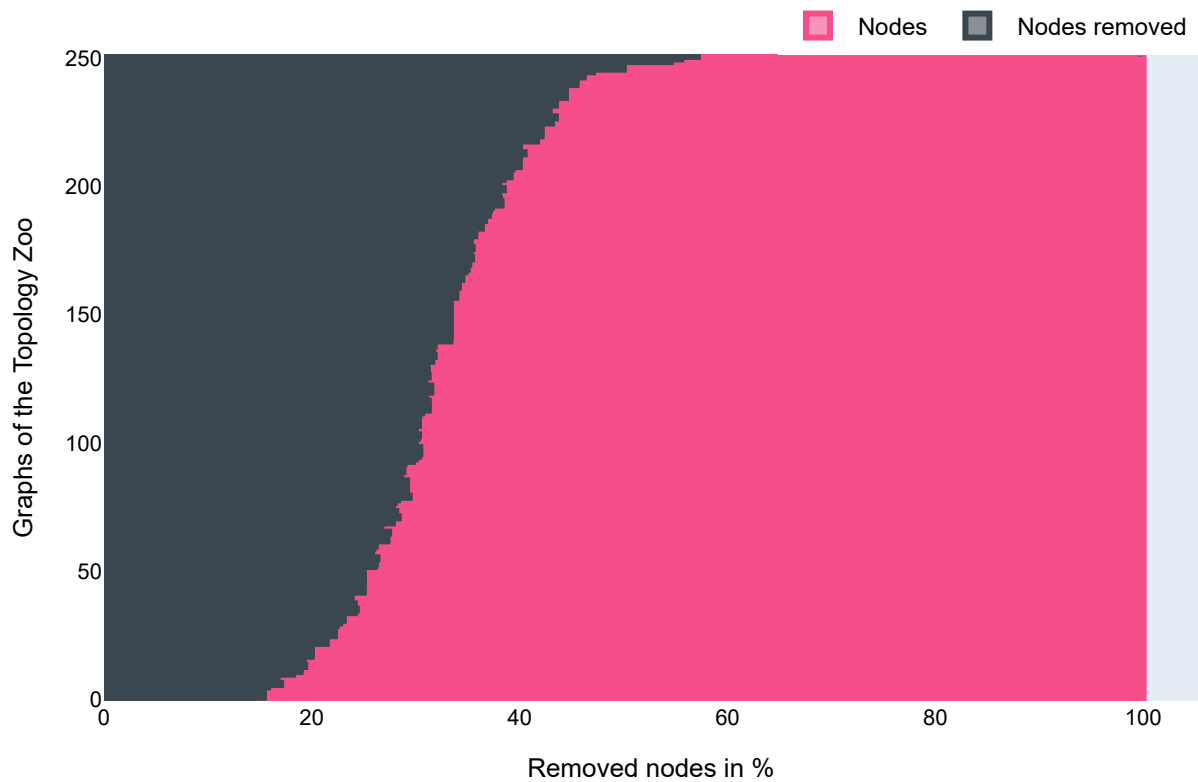


Figure 4.3: The plotted CDF for the node removal algorithm with a core-to-edge ratio of 50 % based on the seeds' average. The curve is quite steep showing that the ratio of removed nodes is somehow nearby for all graphs of the dataset.

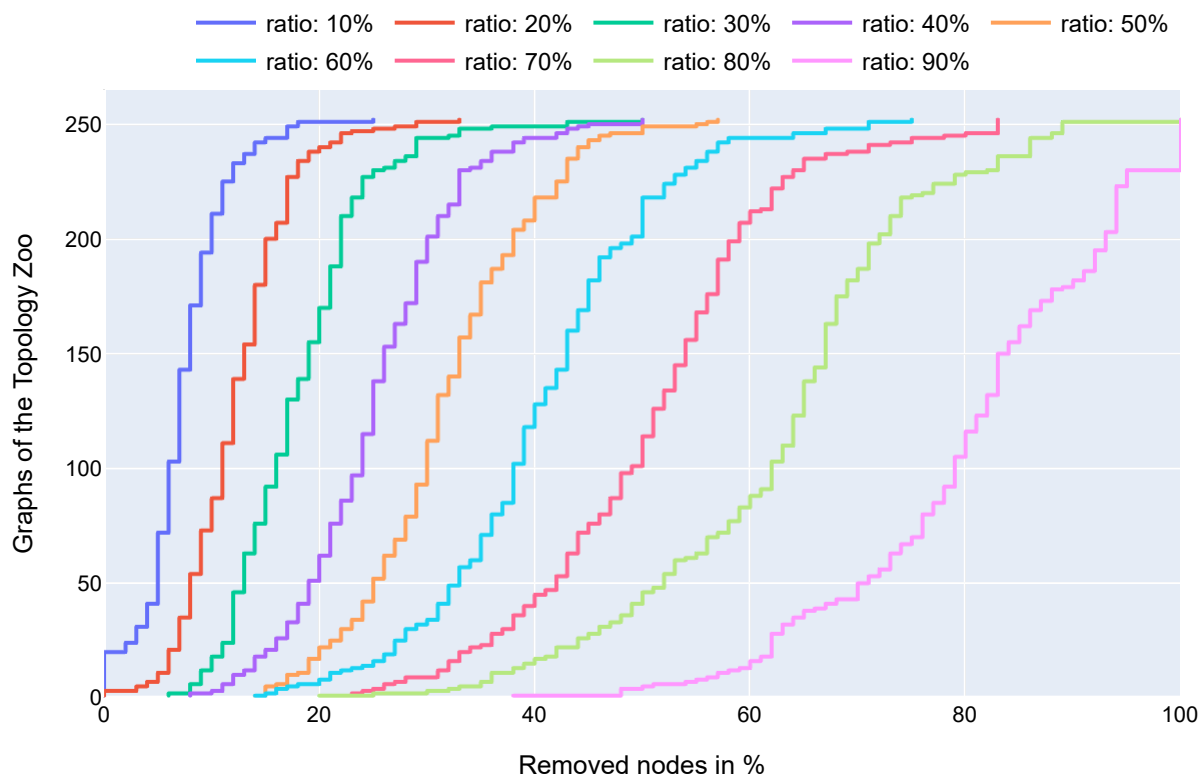


Figure 4.4: The plotted CDFs for the node removal algorithm based on the seeds' average. The curves are less steep for higher ratios meaning a much larger discrepancy between the graphs.

4.3 Node-link removal

The following section describes the results from the combination of the two algorithms. First, the possible nodes were removed and then the remaining removable links. In table 4.5, the different seeds of the link removal algorithm are displayed. The table shows the percentages of the additional nodes that could be removed by applying the link removal algorithm. It is essential to notice that this is not the absolute value because, in the end, we have 100 % of the edges of the graph removed by design. The most important result here is that all seeds again lead to an equal number of removed edges, as was the case for the greedy link removal algorithm as described in section 4.1. However, not all seeds removed the exact same links to maintain connectivity but for the following table, as well as the next section of energy analysis, only the absolute value is essential, hence a single seed from the link removal algorithm is sufficient.

Core-to-edge ratio in %	Seed 1 - Mean of removed edges in %	Seed 2 - Mean of removed edges in %	Seed 3 - Mean of removed edges in %	Seed differences in %
10	17.21	17.21	17.21	0
20	15.17	15.17	15.17	0
30	13.08	13.08	13.08	0
40	11.35	11.35	11.35	0
50	9.99	9.99	9.99	0
60	8.37	8.37	8.37	0
70	6.76	6.76	6.76	0
80	5.35	5.35	5.35	0
90	2.9	2.9	2.9	0
100	0	0	0	0

Table 4.5: The table for the seed differences for the link removal algorithm conducted on the obtained graphs after the node removal algorithm was applied. No different values for the mean of the removed edges between the seeds occurred.

Table 4.6 shows some statistics for the combined node and link removal for one seed. The mean of removed nodes over all graphs is, of course, identical to the one in table 4.4. Additionally, the values of the removed edges from the remaining graphs after the node removal algorithm are displayed. The mean of these removed edges over all core-to-edge ratios steadily decreases and finally reaches zero. This is expected as more nodes, and their connected edges were removed from the graphs. Another observation is, that in some cases, a substantial variation of the maximum and minimum values from the mean exist. An explanation is given by analysing the graphs in the Topology Zoo, noting that some have many more edges than nodes, and hence numerous nodes can be removed when the node removal algorithm is applied for the first time. On the other hand, there exist graphs that are at the beginning minimally connected. That is, by removing any additional edges, these graphs are no longer connected.

Core-to-edge ratio in %	Mean of removed nodes in %	Minimum of removed edges in %	Maximum of removed edges in %	Median of removed edges in %	Mean of removed edges in %
10	7.32	0	83	16	17.21
20	12.28	0	83	12	15.17
30	18.15	0	80	11	13.08
40	25.05	0	71	9	11.35
50	31.56	0	75	7	9.99
60	40.4	0	68	5	8.37
70	50.52	0	62	1.5	6.76
80	62.79	0	62	0	5.35
90	80.76	0	64	0	2.9
100	100	0	0	0	0

Table 4.6: The table for the statistics of the link removal based on one random seed. The mean value decreases consistently but much slower than the median value. An explanation is, that for higher core-to-edge ratios few large networks have many leftover edges where most smaller networks are already depleted.

4.4 Energy analysis

In the last part, the results of the energy analysis based on the obtained graphs after the node-link removal are presented. By assigning two energy value pairs to the nodes and edges of the graphs, the potentially consumed energy of a graph can be determined as well as the possible savings by turning off nodes and links. The two value pairs 1/1 and 10/1 were chosen for node/edge. This gives any graph a predefined energy potential where the corresponding values can be subtracted when nodes or edges are removed from the graph. Likewise, the saved energy can be determined by adding the energy values of the removed nodes and edges.

4.4.1 Energy value pair 1/1

Figure 4.5 shows the scatter plot for the saved energy with an energy value pair 1/1 for node/edge. Each point represents the saved energy value for one graph and occurs on each core-to-edge ratio. The ten seeds for the node removal algorithm were accounted for by calculating the mean values of these seeds, and a random link seed was picked as it does not change the total number of removed edges as seen in table 4.5. One can observe that the points are well spread out. Therefore, some graphs have a significant difference at a specific core-to-edge ratio.

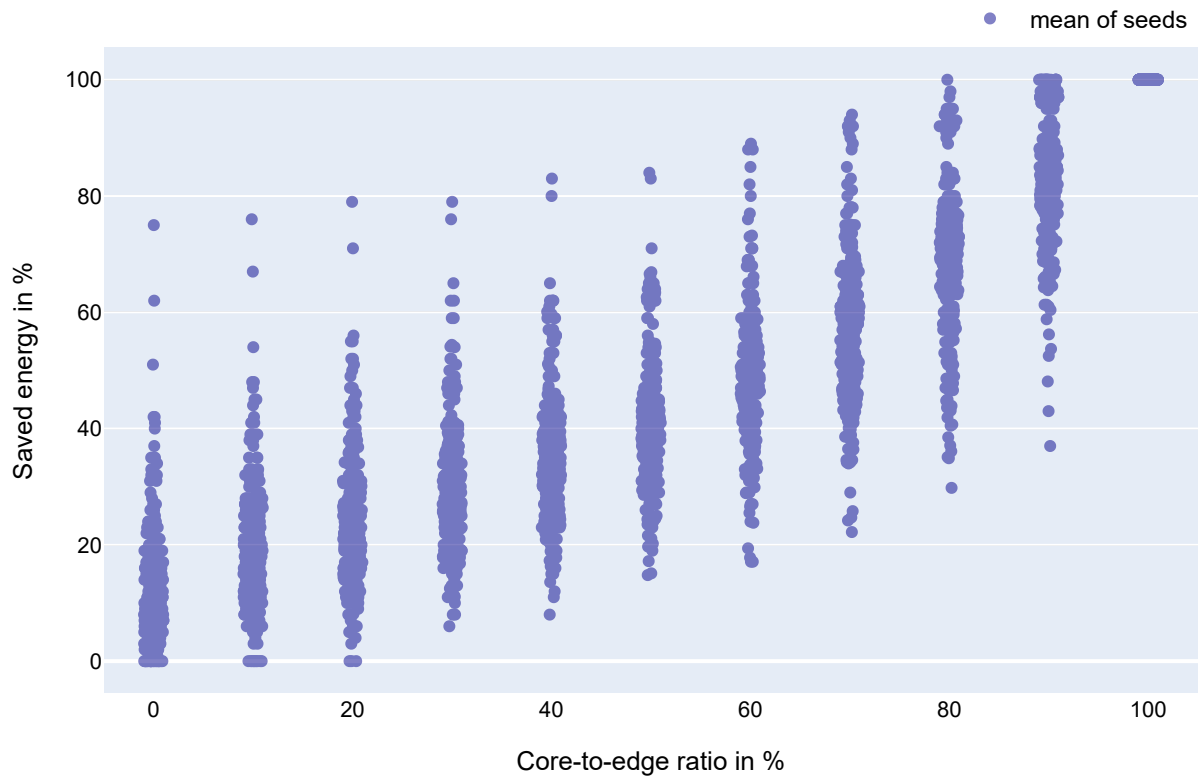


Figure 4.5: The scatter plot for the saved energy for each graph and core-to-edge ratio and energy units of 1/1 for node/edge. The outliers for a core-to-edge ratio of 0 % are graphs with many edges that can be removed by only applying the link removal algorithm.

On the other hand, figure 4.6 shows that for most core-to-edge ratios, half of the saved energy

values of the graphs are nearby. Observing the median values, one can see that they are consistently increasing and reaching roughly 40 % of saved energy at the core-to-edge ratio of 50 %, meaning that the saved energy increases even stronger for higher ratios.

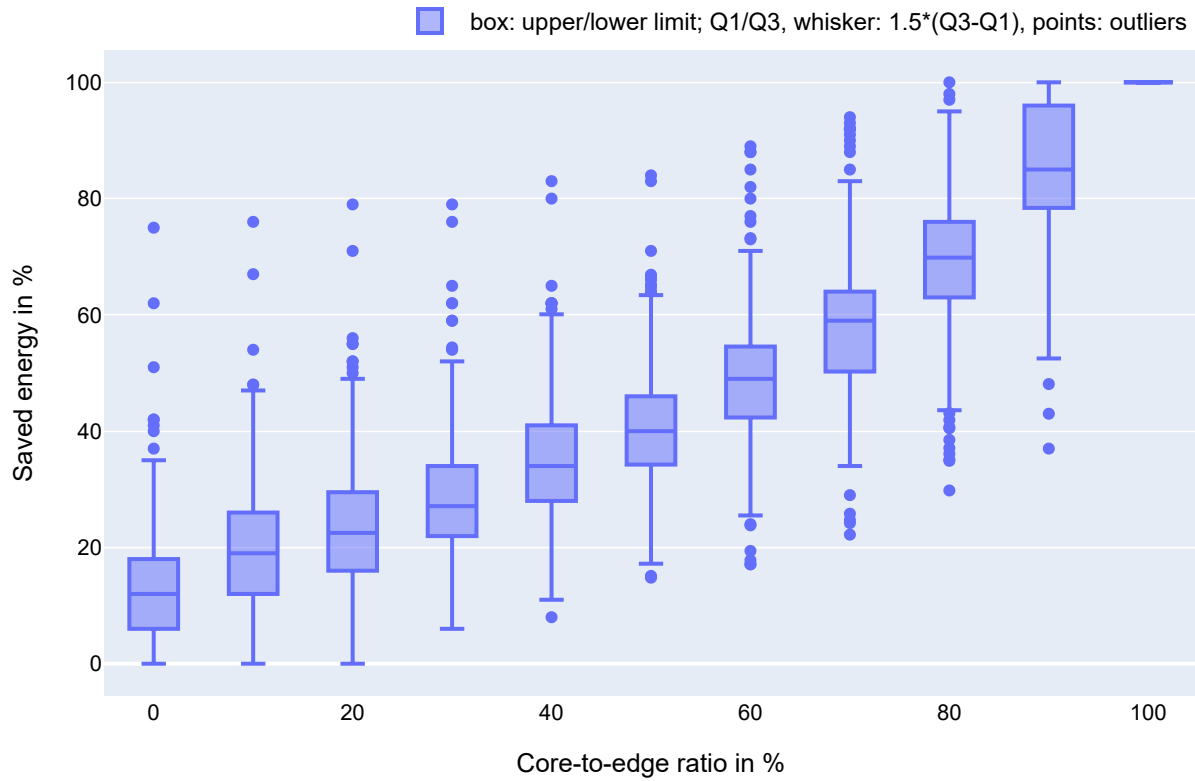


Figure 4.6: The box plot for the saved energy for each core-to-edge ratio and energy units of 1/1 for node/edge. Within the box lay 50 % of all the points, that is the interquartile range (IQR) which is defined as the difference between the 75th and 25th percentiles of the data. The whiskers have in general a length of $1.5 \cdot \text{IQR}$ unless no data point is at this location, in which case, they are shortened until the next closest point. One can now observe, that for a majority of the graphs the saved energy lies much closer together, than the scatter plot 4.5 might have suggested.

4.4.2 Energy value pair 10/1

When a higher energy value to the node is assigned, one can observe that only turning off links does not contribute much to saving energy. Increasing the difference between the core and edge value would intensify this effect. Comparing it to figure 4.6, one can find that less energy can be saved for lower core-to-edge ratios in general. Removing more nodes is the most significant contribution and the most substantial increase in saved energy.

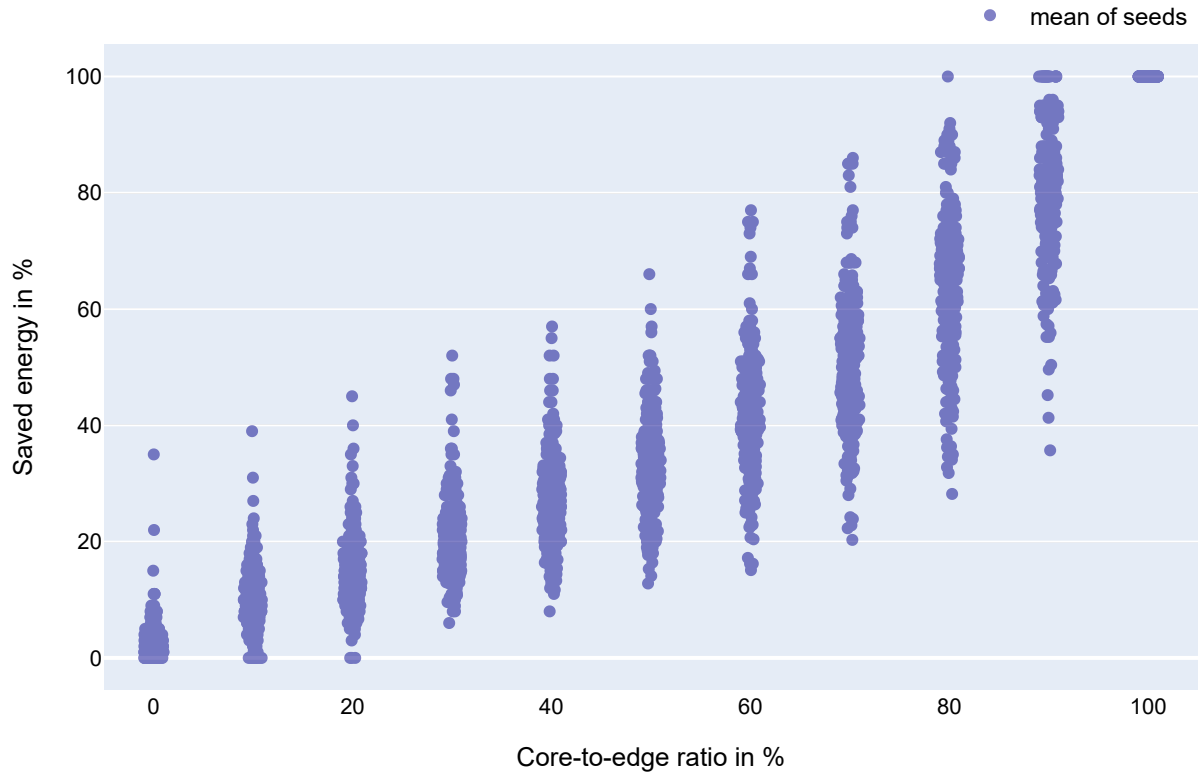


Figure 4.7: The scatter plot for the saved energy for each graph and core-to-edge ratio and energy units of 10/1 for node/edge. For a core-to-edge ratio of 0 % much less energy can be saved compared to figure 4.5. In addition, the points are initially less spread out but this increases for higher core-to-edge ratios.

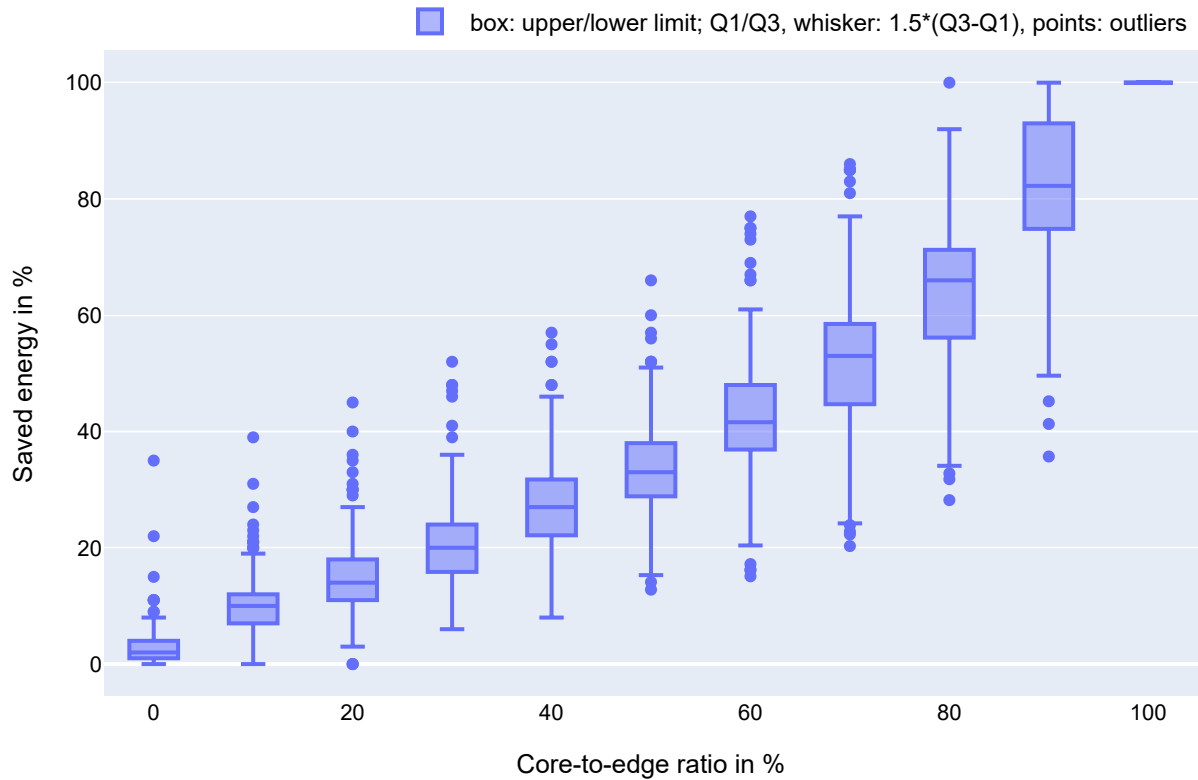


Figure 4.8: The box plot for the saved energy for each core-to-edge ratio and energy units of 10/1 for node/edge. Within the box lay 50 % of all the points, that is the interquartile range (IQR) which is defined as the difference between the 75th and 25th percentiles of the data. The whiskers have in general a length of $1.5 \cdot \text{IQR}$ unless no data point is at this location, in which case, they are shortened until the next closest point. Many graphs now have a small potential to save any energy by just turning links off. Only when more core nodes are removed do the values add up.

Chapter 5

Discussion

The results show that for a greedy link removal approach, 22.27 % of all the edges can be removed without compromising the connectivity of the networks of the Topology Zoo. Depending on the predefined core-to-edge ratio, the number of removed links decreases when nodes were removed before. A ratio of 50 % allows the removal of 31.56 % of all nodes and 9.99 % of all links. The energy analysis states that the results strongly depend on initial parameters, such as the energy assigned to a node and edge, how many spare links a graph has, etc. This is even more true for the network's number of nodes and edges because, with large networks, one can choose viable nodes as cores or select nodes and edges to remove. Nonetheless, the results provide approximate values of what can be expected if some network conditions are known or can be defined. A core-to-edge ratio of 50 % means possible energy savings of around 40 %. In general, many analysed networks seem to have quite some potential to save energy.

The results generally answer the defined goals and related questions at the beginning. The potential of the networks of the Topology Zoo can now be roughly estimated for a primary and greedy approach. But there is room for a more detailed and in-depth analysis to estimate the potential to save energy in a network even further. For example, by including more redundancy by allowing larger minimum cut values, an upper bound for the maximum ratio of removed cores or defining nodes that can never be removed.

Possibilities for future work are plentifully available. Options include the extensions described above or similar ones. Also, a forthcoming option is fusing both link and node removal algorithms and then finding the minimum for this optimisation problem to know the best relation between removed edges and nodes. Finally, more options for node labelling could be a viable approach to get more sophistication.

In any case, this project has shown that the potential to save energy in networks certainly exists. With plenty of additional work and many more projects in a similar direction and beyond, perhaps one day, the carbon emissions of the ICT sector can be reduced by exploiting this potential.

References

- [1] SwissIX Public Traffic Statistics. <https://ixpmanager.swissix.ch/statistics/ixp>. [Online; accessed 22-December-2022].
- [2] SWITCH, Switzerland, Backbone, Fibre, 2010. <http://www.topology-zoo.org/maps/Switch.jpg>. [Online; accessed 08-December-2022].
- [3] IDC - Global ICT spending. <https://www.idc.com/promo/global-ict-spending/forecast>. [Online; accessed 08-December-2022].
- [4] Charlotte Freitag, Mike Berners-Lee, Kelly Widdicks, Bran Knowles, Gordon S. Blair, Adrian Friday. *The real climate and transformative impact of ICT: A critique of estimates, trends, and regulations*. Cell Press, 2021.
- [5] Lotfi Belkhir, Ahmed Elmeli. *Assessing ICT global emissions footprint: Trends to 2040 recommendations*. Elsevier Ltd., 2018.
- [6] Anders S. G. Andrae, Tomas Edler. *On Global Electricity Usage of Communication Technology: Trends to 2030*. MDPI, 2018.
- [7] SwissIX - traffic graphs of Switzerland. <https://www.swissix.ch/infrastructure/traffic>. [Online; accessed 08-December-2022].
- [8] HotCarbon 2022: 1st Workshop on Sustainable Computer Systems Design and Implementation. <https://hotcarbon.org/>. [Online; accessed 09-December-2022].
- [9] Dagstuhl seminar - Power and Energy-aware Computing on Heterogeneous Systems (PEACHES). <https://www.dagstuhl.de/en/program/calendar/semhp/?semnr=22341>. [Online; accessed 09-December-2022].
- [10] IAB workshop on Environmental Impact of Internet Applications and Systems, 2022. <https://www.iab.org/activities/workshops/e-impact/>. [Online; accessed 09-December-2022].
- [11] S. Knight, H.X. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *Selected Areas in Communications, IEEE Journal on*, 29(9):1765–1775, october 2011.
- [12] Topology zoo, University of Adelaide. <http://www.topology-zoo.org/index.html>. [Online; accessed 08-December-2022].
- [13] NetworkX documentation, open-source Python library. <https://networkx.org/documentation/stable/index.html>. [Online; accessed 08-December-2022].
- [14] NetworkX code, open-source Python library. <https://github.com/networkx>. [Online; accessed 08-December-2022].