Fiber Fortification

Optimizing Network Redundancy

Tuesday 20th August, 2024

Jesse Ruiter

Royal KPN N.V. | Delft University of Technology

kpn

TU Delft

Fiber Fortification

Optimizing Network Redundancy

by

Jesse Ruiter

to obtain the degree of Master of Science
at the Faculty of Electrical Engineering, Mathematics & Computer Science
of the Delft University of Technology,
to be defended publicly on Tuesday August 27, 2024 at 2:00 PM.

Project duration:	November 15, 2023 – August 27, 2024	
Thesis committee:	Dr. ir. E. Smeitink,	TU Delft, thesis advisor
	Dr. E.F.M. van Boven,	TU Delft, daily supervisor
	Dr. ir. R. van de Bovenkamp,	KPN, company supervisor
	Dr. C.E. Groenland,	TU Delft, external member

Cover: Network of the Netherlands by KPN (Modified)

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

At the beginning of this report, I would like to express my gratitude to everyone who contributed in any way to the completion of my thesis, and thereby to my years at the university. First and foremost (in roughly chronological order), I would like to thank Edgar van Boven for his enthusiasm during the lectures of the course Telecom Business Architectures and Models, which sparked my interest in pursuing my graduation project at KPN. I am grateful to Ruud van de Bovenkamp for defining the assignment, introducing me to the KPN network, and providing weekly guidance throughout the process. Special thanks to Eric Smeitink for chairing the thesis committee and offering valuable input on the project. I would also like to thank the Advanced Analytics TDO team for making me feel welcome as part of their team, as well as all the other KPN employees who took the time to familiarize me with the network.

Last but not least, I want to thank my parents, friends, and housemates for their support over the past years. Without them, my time as a student would not have been as enjoyable as it has been.

Jesse Ruiter
Delft, August 2024

Abstract

This thesis investigates methods to enhance the robustness of KPN's network. As reliance on telecom networks grows, improving network resilience is paramount. Two primary strategies exist: minimizing failure occurrence and mitigating failure impact. This research focuses on the latter, specifically increasing network redundancy by transforming existing ring structures into strand structures.

Current ring structures in the network avoid single points of failure (SPOFs) at the link level but fail to address node-level SPOFs, particularly at Metro Core (MC) locations. MC locations are pivotal, as they manage customer sessions and route all household network traffic. Node failures at these locations can disrupt connections for up to 100,000 households. Thus, ensuring that a single node-level failure can be compensated by another node is crucial. This requires considering both equipment and building failures, with geographically distant backup nodes necessary for the latter.

Transforming rings into strands, which inherently include two geographically separated MC locations, addresses these issues but incurs significant costs. While an exact cost estimation is beyond the scope of this research, the study approximates the length of additional cables required, totaling 206 kilometers. Including cost multipliers for tunnels, bridges, and highways, the adjusted length is equivalent to 222 'kilometers' of cable. Despite the substantial cost, node failures at the equipment level, which are more common, can be mitigated with redundant equipment at MC locations. Therefore, a cost-effective alternative might be forming strands only where rings from different access areas are already in proximity.

Contents

Acronyms

AON Active Optical Network. 11
APT Administration of Post and Telegraphy. 12

IoT Internet of Things. 9

KPN Koninklijke PTT Nederland (Royal PTT Netherlands). 12

MA Metro Access. 11
MC Metro Core. 10

NAT Network Address Translation. 14
NBTM Nederlandsche Bell-Telephoon Maatschappij. 12
NTU Network Termination Unit. 11

OAP Optical Aggregation Point. 11
ODF Optical Distribution Frame. 11
OLT Optical Line Terminal. 11
ONT Optical Network Terminal. 11

P&T State Enterprise of Post and Telegraphy. 12
PON Passive Optical Network. 11
PoP Point of Presence. 11
PTT State Enterprise of Post, Telegraphy and Telephony. 12

SPOF Single Point of Failure. 14

ZARA Zwolle, Arnhem, Rotterdam en Amsterdam. 10

Glossary

access area A segment of the KPN network comprising one MC location, multiple MA locations, and all associated buildings and cables that extend to households. 21

access network The part of the network that is responsible for connecting households to local KPN nodes. 21

backhaul network The part of the network that connects the core network to the access network. 21

core network The part of the network that consists of the four ZARA locations and connects to the datacenters of KPN and the services hosted there. Additionally, it is responsible for the connecting the network of KPN to the rest of the world. 21

georedundancy The practice of distributing systems, data, or infrastructure across multiple geographic locations to ensure service continuity in case of local failures or disasters. 25

MA location A local (Metro Access) node in the KPN network that is responsible for distributing network connections within a district. 21, 33

MC location A regional (Metro Core) node in the KPN network that is responsible for creating client sessions and routing network traffic. 10, 21, 22, 33

redundancy The capability of a network to decrease the impact of failures. This involves incorporating additional or duplicate components, paths, or systems into the network, so that if one component fails, others can take over its function, ensuring uninterrupted service. 9, 10

resilience The ability of a network to reduce the occurrence of failures. This involves designing and implementing measures that prevent faults and disturbances from happening, thereby maintaining continuous and reliable network performance. 9

ring A circular network topology that is a subpart of the entire network. One of its nodes, the main node, is responsible for the connection to the rest of the network. This research focuses on the case where the main node is an MC location and the other nodes in the ring are MA locations. 13

robustness The capability of a network to maintain its performance and continue operating under varying conditions, including failures and unexpected events. It encompasses both the reduction in the occurrence of failures and the mitigation of their impact. 10

strand A network topology where all intermediate nodes are connected to a source and a sink node. The source and the sink are distinct nodes, that are responsible for the connection to the rest of the network. 10, 15

ZARA location A national node in the KPN network that serves as a central point for managing network traffic, houses the services of KPN and connects to the rest of the world. 10

1 Introduction

1.1. Dependency on Digital Infrastructure

In recent years, the dependency on digital infrastructure has grown significantly, driven by various socio-economic changes and technological advancements. The COVID-19 outbreak in early 2020 played a pivotal role in accelerating this trend, as many individuals transitioned from traditional office environments to home offices. This sudden shift necessitated a rapid adaptation to new ways of working and communicating. Consequently, social activities such as cinema visits and in-person gatherings were largely replaced by digital alternatives like streaming services (e.g., Netflix) and virtual meetings (e.g., Zoom).

Even as the immediate necessity for online meetings has diminished in the post-pandemic era, the shift towards digital interactions has become deeply ingrained in our daily lives. Hybrid working models have emerged as the new norm, enabling people to replace lengthy commutes and business trips with efficient online meetings. This shift has not only transformed workplace dynamics but also impacted other sectors such as education, healthcare, and entertainment. For instance, remote learning and telehealth services have seen widespread adoption, further increasing the demand for reliable network services.

These changes highlight the need for a robust digital infrastructure that can handle the large amount of network traffic these activities create. The rapid increase in Internet of Things (IoT) devices, cloud computing and data-heavy applications has made this need even more urgent. As data traffic keeps growing, having a reliable network has become more important than ever, requiring ongoing improvements in network infrastructure to meet these rising demands.

1.2. Redundancy

The increasing dependency on digital infrastructure underscores the critical importance of network reliability. Network reliability is a multifaceted concept that involves ensuring uninterrupted service and maintaining optimal performance levels despite potential failures. To enhance the reliability of KPN's network, there are essentially two primary approaches: (1) reducing the occurrence of failures and (2) mitigating the impact of failures.

The first approach, known as improving resilience, involves measures such as strengthening cable durability, implementing rigorous maintenance protocols, and avoiding areas prone to cable breaks. Resiliency can also encompass proactive monitoring and predictive maintenance strategies to identify and address potential issues before they lead to failures. By enhancing the physical and operational aspects of the network, resiliency aims to minimize the likelihood of disruptions.

In contrast, the second approach, referred to as improving redundancy, entails incorporating backup servers, alternative paths, and redundant systems to ensure continuous operation despite failures. Redundancy strategies can include deploying multiple data centers, creating diverse routing paths, and utilizing failover mechanisms that automatically switch to backup systems in the event of a failure. By

providing multiple layers of redundancy, the network can maintain service availability and performance even when individual components fail.

Collectively, these strategies fall under the broader concept of enhancing the robustness of the network. Robustness refers to the network's ability to withstand and recover from adverse conditions, ensuring reliable service delivery under varying circumstances. Enhancing robustness involves a combination of resiliency and redundancy measures to create a stable network infrastructure.

This study will focus on the second approach: increasing the redundancy of the network to ensure its proper functioning. By exploring various redundancy strategies and their implementation, this research aims to provide insights into effective methods for boosting network reliability and supporting the continuous growth of digital infrastructure dependency.

1.3. KPN network

1.3.1. Overview

All the central offices and cabinets located throughout the Netherlands are ultimately connected to each other, forming the KPN network. The core of this network consists of four data centers in the cities of Zwolle, Arnhem, Rotterdam, and Amsterdam. These so-called ZARA locations are the points where internet traffic transitions from the KPN network to the global internet. Additionally, services such as television are distributed from these locations across the rest of the network. Each of these locations is interconnected with the other three, like a fully connected graph, ensuring an uninterrupted network connection even if one of the connections or one of the ZARA locations itself fails. [7]

These ZARA locations are connected to 160 Metro Core (MC) locations. The Netherlands is divided into an equal number of access areas, with these regional nodes serving as central points. The MC locations are responsible for assigning IP addresses to customers (see section 1.4) and caching popular content, reducing the need for requests to travel higher up the network. From a ZARA location, several MC locations are connected to another ZARA location. This structure is referred to as a strand in the rest of this report and shown in Figure 1.1.

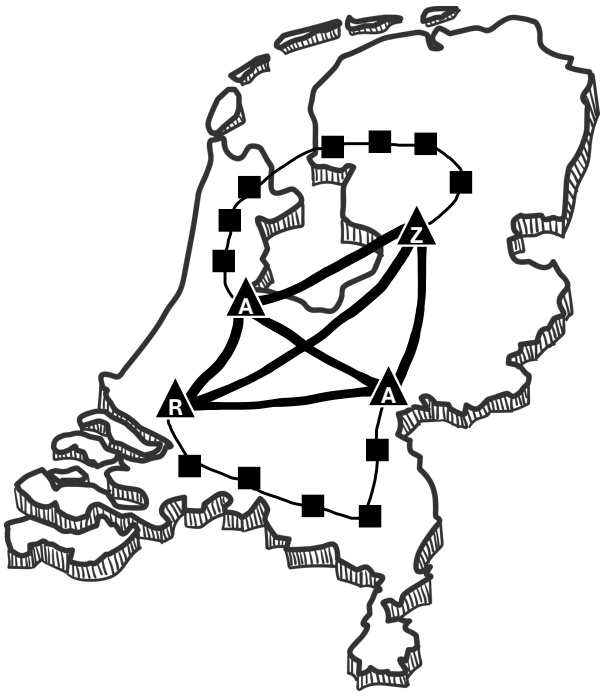


Figure 1.1: A visual representation of the interconnections between ZARA locations (triangles) and the connection of some MC locations (squares) to two ZARA locations using a strand topology.

In these access areas, both fiber and copper connections are currently active. Since the future of the network is focused on fiber, this report also focuses on these connections. For simplicity, copper connections are omitted from this description. KPN currently uses two types of fiber technologies, each with its own network structure: Active Optical Network (AON) and Passive Optical Network (PON). [15]

In Active Optical Networks, each connection/address has a separate fiber running to the central office. This central office is also known as a Point of Presence (PoP). Thus, for each connection, there is a dedicated active optical port in the PoP (i.e., each connection has its own laser that generates the optical signal). It is essentially the 'simple standard way' to set up a network, where the optical signals at the households are converted one-to-one into an electrical signal by the Network Termination Unit (NTU). In the case of AON, City PoPs and Area PoPs are present in the access area. City PoPs are connected in a ring to the Metro Core location and are also connected to about twenty Area PoPs. In an Area PoP, the fibers are organized in patch panels, called Optical Distribution Frame (ODF). These are often organized in ODF trays. The fibers terminate in the ODF trays, where they are spliced onto a connector, allowing an optical patch cable to be inserted. This patch cable is connected to the equipment at this location, which includes the laser that illuminates the fibers and the receiver that converts optical signals to electrical signals.

In Passive Optical Networks, multiple connections are served per active optical port in the central office. A single active signal, which starts in the Optical Line Terminal (OLT) of a Metro Access (MA) location, is split/duplicated with a passive optical splitter in a street cabinet. Such an optical splitter, called an Optical Aggregation Point (OAP), can split the active optical signal into 64 fibers. All connected households receive a portion of the time to send and receive signals, as shown in Figure 1.2. Consequently, the theoretical bandwidth is lower for these types of connections and they have higher technical complexity. However, the installation and operational costs are lower, as the laser is shared among multiple households. Therefore, fewer lasers are needed, which also saves energy. The OLT encrypts each household's signal with a different key, so the Optical Network Terminal (ONT) in the homes can only decrypt the signal intended for them.

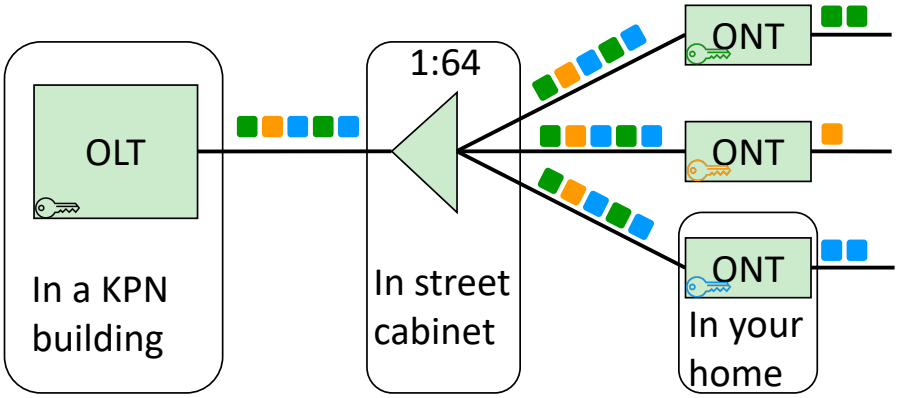


Figure 1.2: Packets from the Optical Line Terminal to the Optical Network Terminal.

Figure 1.3 shows a slightly outdated overview of the logical network structure, created by dr. ir. R. van de Bovenkamp [4]. The four red points represent the four ZARA locations. The pink points are disregarded for simplicity. The blue points represent Metro Core locations, which are connected with two lines to the rest of the network (i.e., two ZARA locations). The green points are the central offices in the access areas. This figure clearly shows that all internet traffic from an access area is routed to a Metro Core location. To limit the impact of failures, it has been agreed that no more than 100,000 households will be connected per access area.

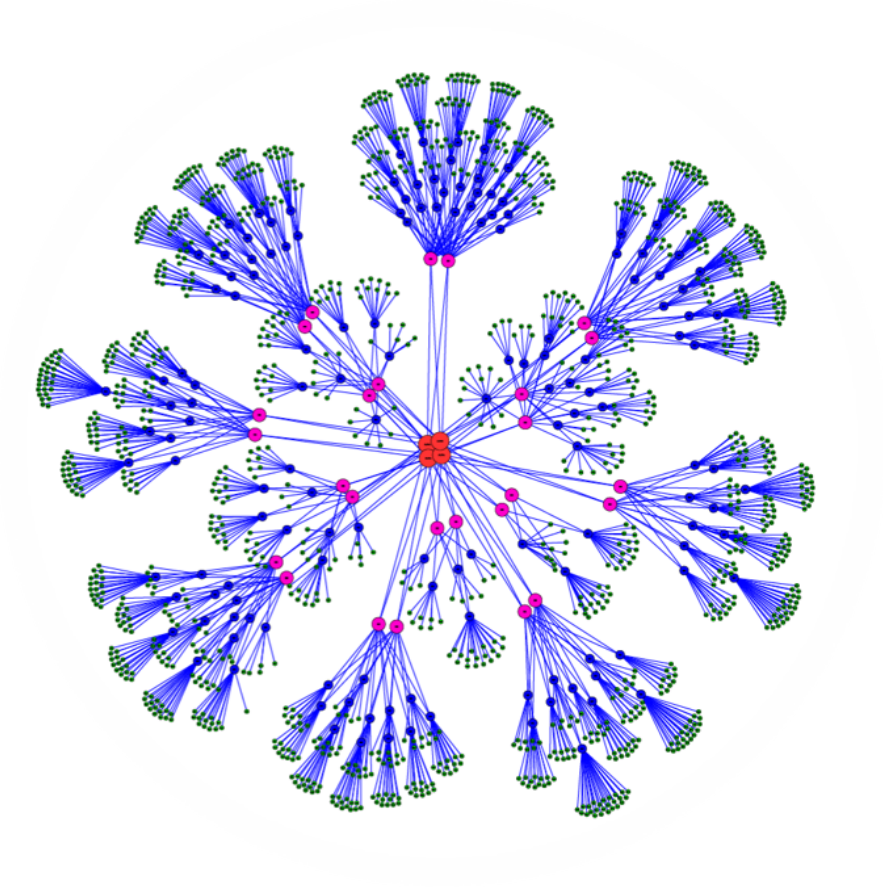


Figure 1.3: An overview of KPN's logical network structure.

1.3.2. History

As one of the leading telecommunications and IT providers in the Netherlands, KPN has been instrumental in establishing a robust network infrastructure crucial for the nation's connectivity. Its rich history dates back to the late 19th century when telecommunications were still in their infancy. The first public telephone network in the Netherlands became operational in Amsterdam, where a telephone operator from the Nederlandsche Bell-Telephoon Maatschappij (NBTM) connected 49 subscribers. In 1893, the Administration of Post and Telegraphy (APT) was established as an independent organization funded by the Ministry of Water Management, Trade & Industry. The introduction of the Telegraph and Telephone Law in 1904 led to a significant increase in telephone connections, reaching 75,000 subscribers by 1915. In that year, the organization became a state enterprise named State Enterprise of Post and Telegraphy (P&T). Only in 1928 was the name changed to State Enterprise of Post, Telegraphy and Telephony (PTT) to include the element 'telephone'.

In subsequent years, telephony became increasingly important, with the number of subscribers reaching two million by the 1960s. Until this time, calls were routed through a manually operated exchange, which was replaced by an automatic system in 1962. By the end of this decade, the first internet (ARPANET) was established, connecting two universities in the United States. It was not until 1988 that the National Research Institute for Mathematics and Computer Science got access to the internet and the Netherlands became the second country in the world to get connected to this digital highway.

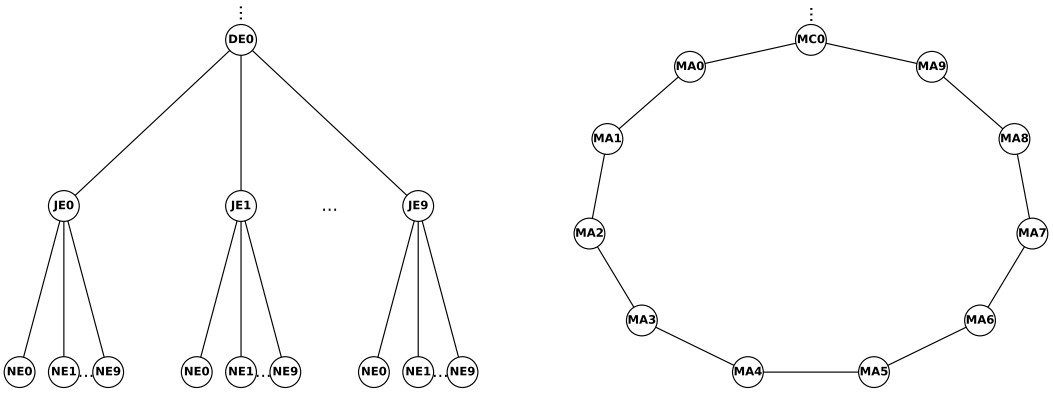
In 1989, the government restructured PTT into an independent entity, granting it the designation royal and renaming it Koninklijke PTT Nederland NV (KPN), with its primary operating companies being PTT Post BV and PTT Telecom BV. The government remained the sole shareholder until the company's public offering in 1994. PTT Post and PTT Telecom go their separate ways in 1998. PTT Telecom then becomes known as KPN NV.

As technology advanced, the usage of the network also evolved. The network was no longer solely used for telephony; internet traffic became an increasingly significant component of telecommunications. To handle the growing volume of network traffic, many copper cables were replaced with fiber optics. This transition started at the core of the network and extended in the direction of the households over time. KPN aims to have 80% of all households connected to fiber by 2026. [11] The advantages of fiber over copper include significantly higher bandwidth, allowing for faster data transmission, and immunity to electromagnetic interference, resulting in less signal degradation over long distances. [1, 2]

1.3.3. Hierarchy

The type of cable is not the only change the network has undergone; the network topology has also evolved over time. Initially, the network was constructed as a telephone network using copper cables, employing a tree topology. In this configuration, households were connected to a local exchange in their neighborhood. Several of these local exchanges were connected to a junction exchange in their municipality, which in turn were connected to a district exchange in their region. This hierarchical structure ensured that cable lengths were kept manageable, maintaining a strong signal quality. Moreover, in a tree topology there is only one path from one node to another, which simplifies both the routing and the maintenance on the network.

With the advent of fiber optics, the distance a cable can cover has significantly increased. While having a single route to other locations simplifies routing and maintenance, it also makes the network more vulnerable for failing connections. This has led to a revision of the network architecture. The network remains hierarchical, but the cables are now configured as rings. This difference is depicted in Figure 1.4. On the left, a district exchange connects to multiple junction exchanges, each of which is connected to several number exchanges. On the right, MA locations are connected in a ring structure to an MC location, which in turn is connected to the rest of the KPN network. An advantage of this latter configuration is that network traffic can be routed in both directions around the ring, so a cable break is no longer fatal for connectivity.



(a) Visualization of a tree network architecture.

(b) Visualization of a ring network architecture.

Figure 1.4: The graph topology in the previous and current KPN network.

1.4. Sessions

Just as it is necessary to have the recipient's address when sending mail, a subscriber of KPN needs to have an IP address. KPN uses two versions of IP addresses: IPv4 and IPv6. The former consists of a 32-bit address space, which allows for approximately 4.3 billion unique addresses. Due to the growth of internet-connected devices, IPv4 addresses have become scarce. To address this limitation, IPv6 was introduced, offering a vastly larger 128-bit address space, which can accommodate approximately 3.4×10^{38} unique addresses.

When a device connects to the KPN network, the MC location assigns it an IP address, either IPv4 or IPv6, depending on the network configuration and device capabilities. This IP address is essential for the establishment of sessions between the device and other entities on the internet. When the neces-

sary routing and forwarding rules are set up, data can be exchanged. Once a session is established, the IP address serves as the identifier for the device throughout the duration of the session. Session management involves monitoring and maintaining the connection, ensuring that data packets are correctly routed between the device and other internet hosts. This includes handling any changes in the network topology, such as the device moving between different access points within the network.

IPv4 Addressing

IPv4 addresses are typically represented in decimal format, separated by periods (e.g., `172.16.254.1`). This addressing method is well established and supported by most network infrastructure. However, the limited number of available addresses has led to the implementation of techniques such as Network Address Translation (NAT), which allows multiple devices on a local network to share a single public IPv4 address. While effective, NAT can introduce complexity and potential performance issues.

IPv6 Addressing

IPv6 addresses are represented in hexadecimal format, separated by colons (e.g., `2234:0000:0000:6904:0019:d2ff:feb3:5e4f`). The expanded address space not only resolves the issue of address exhaustion but also simplifies network configuration and improves routing efficiency.

1.5. Network Failures

The KPN network consists of various components, each of which can potentially fail at some point. To minimize the impact of these failures, it is crucial to anticipate these scenarios in advance. D.R. Kuhn [16] classified the sources of failure into six categories: human error, acts of nature, hardware failures, software failures, overloads, and vandalism. In this report, we distinguish between three different types of components that are prone to failure: (1) cables, (2) equipment, and (3) buildings. We refer to the first type as link failures and the latter two as node failures. Both equipment and buildings are considered nodes within the KPN network.

Link failures are the most common type of failure. They can occur, for instance, when a cable is cut during excavation work or when a cable is improperly connected during maintenance. If these cables serve individual households, the impact is limited to a few customers being disconnected from the network. However, higher up in the network, a link failure can have more significant consequences. To mitigate this, such links are often installed redundantly, meaning they have a backup path. In such cases, network traffic can be rerouted through an alternative path if the original path becomes unavailable.

The second type of failure involves malfunctioning equipment. The cables leading to households pass through numerous switches on their way to national nodes. These switches are responsible for efficiently forwarding network packets to their destinations. As active devices, they are susceptible to failures. Preventive measures, such as connecting these devices with multiple power supplies, are implemented to reduce this risk.

The final type of failure we consider involves entire buildings becoming non-operational. Causes for such failures include fire, flooding (or other natural disasters), sabotage, or power outages. To address power outages, some locations are equipped with batteries, and generators can be brought to the site if necessary. In the case of other failures, having backup equipment at the same location is insufficient. Redundancy can only be achieved by having backup equipment at a geographically separate location. This scenario is the focus of this study.

1.6. Single Point of Failure

Several measures have already been implemented to prevent and mitigate network interruptions. These include the installation of redundant cables and the provision of backup batteries at various locations. The objective is to eliminate Single Points of Failure (SPOF), meaning the failure of a single component should not affect the overall functioning of the network. Only when multiple components fail simultaneously could a disruption occur.

This principle is evident in the KPN network through the way MC locations are connected. The connection begins at a ZARA location, passing through several MC locations before reaching another ZARA location. This structure ensures that network traffic from an MC location can route either clockwise or

counterclockwise to a ZARA location. A similar structure is used for MA locations. Here, the connection runs from an MC location through several MA locations back to the same MC location. In the event of a single link failure, traffic can still reach the MC location via the alternative route.

However, this structure introduces a single point of failure at the MC location itself. As this location forms the connection to the rest of the KPN network and the equipment at this location is responsible for handing out IP addresses to households, it is essential that households always can connect to an MC location. For that reason there should be a backup available in case of node failure. Node failure at equipment level, could be overcome by replicating the devices at the MC locations. However, there should also be preventive measures in case the building itself is affected by failure. Namely if there are issues at the MC location, all underlying households (up to 100,000) will be disconnected from the network. A way to prevent this, is making sure that every ring contains multiple MC locations.

1.7. Problem Statement

To enhance the redundancy of the network, this study aims to eliminate the single points of failure (SPOFs) at MC locations. These SPOFs exist because the ring both begins and ends at the same MC location. In the strand structure used for connecting MC locations, these SPOFs are not present. Therefore, the challenge is to transform the ring-based network architecture into a strand-based structure, which has proven to be effective at higher levels of the network.

The overarching goal is to ensure that the network's architecture can sustain the increasing demand for reliable and continuous digital connectivity, now and in the future. This transformation is critical for maintaining not just routine communications but also for supporting the nation's economic stability and growth in an increasingly digital world.

1.8. Research questions

The main research question is formulated as:

How to transform a network of rings into a network of strands as cost-effectively as possible?

To answer this question, the following subquestions will be addressed:

1. What are the advantages of the stand structure?
2. What factors play a role in the costs of this transformation?
3. What are the potential risks and challenges associated with the transformation?

1.9. Document structure

This thesis is structured as follows: Chapter 2, the Theoretical Framework, provides a deeper understanding of the foundational theories pertinent to this study. It explores concepts such as Graph Theory and Matching techniques, which are crucial for constructing redundant network infrastructures. Chapter 3 describes the methodologies employed throughout the research, offering an overview of the approach and detailing the algorithms used for network topology transformation, including matching, routing and strand algorithms. Chapter 4 presents the results and discusses the outcomes of the applied methodologies. The conclusion, Chapter 5, synthesizes the findings of the study, discussing key insights and offering recommendations for future enhancements of network infrastructures.

2

Theoretical Framework

2.1. Graph Theory

To effectively study complex systems such as telecommunications networks, it is crucial to abstract the actual physical layout into a more analytically tractable form. Graph theory provides a mathematical framework for this purpose, by reducing complex network components — such as buildings and cables — into simpler conceptual representations known as nodes and links. This abstraction is used to model pairwise relations between objects and allows for easier analysis and visualization of the network's structure. [14]

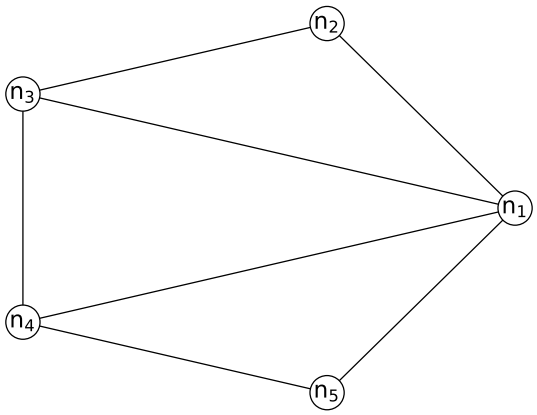


Figure 2.1: An example of a graph.

2.1.1. Definition

As pictures speak louder than words, Figure 2.1 shows an example of a graph. The simplest and least strict definition of a graph is the following: a graph is a set of points and lines connecting some pairs of the points. [21] In graph theory, the points are called nodes (or vertices) and the lines are called links (or edges). Thus, the graph depicted in Figure 2.1 comprises five nodes and seven links.

Nodes (N): These represent discrete entities such as routers, switches, or data centers within a telecommunications network. Each node can have attributes such as coordinates to denote its physical location or other relevant information. The set of nodes is written as $N = \{n_1, n_2, \ldots, n_i\}$, where n_x is the x-th node and i is the number of nodes. The number of elements in a set S is denoted by $|S|$. In a set no element is repeated and the order of elements in the list does not matter.

Links (L): In a telecommunications context, links represent the physical or logical connections, such as fiber optic cables or wireless links, that facilitate communication between nodes. Links can also have attributes such as distance, capacity, or bandwidth, which are essential for analyzing the quality and efficiency of the connections. These attributes are sometimes also referred to as the weights. Like nodes, the set of links is written as $L = \{l_1, l_2, \ldots, l_3\}$. Each link l_x is identified by the pair of nodes which are connected by l_x. An individual link is therefore written as $l_1 = (n_1, n_2)$.

A graph is typically denoted as $G = (N, L)$, capturing all the components of a network and their connections in a simplified model. By including attributes for nodes and links, such as location and distance, the graph model becomes more informative. This enriched model allows for a deeper analysis of the network's structure. For example, knowing the physical locations of nodes can assist in optimizing the network layout for better reliability.

2.1.2. Graph Types

Graph theory classifies different types of graphs that can be used to model a variety of network structures and scenarios in telecommunications. Each type of graph is suited to different aspects of network design and analysis. [17] Here, we focus on some fundamental types of graphs used in network theory: undirected graphs, directed graphs, multigraphs, bipartite graphs, and complete graphs.

Undirected Graphs: Undirected graphs are graphs where the relationship between two nodes is always mutual. That is, if n_1 and n_2 are nodes connected by a link in an undirected graph, then n_1 is related to n_2 and n_2 is related to n_1. In other words, the order of the nodes is not important. This mutual relationship is useful in scenarios such as modeling road networks. For example, in a road network represented as a undirected graph, a connection between junction A and B indicates the presence of a road segment that allows travel in both directions between A and B.

The formal definition can be expressed as follows:

$$L \subseteq \{(i, j) \mid i, j \in N, (i, j) = (j, i)\} \tag{2.1}$$

Directed Graphs: A directed graph is a type of graph in which the connection between two nodes is unidirectional. Unlike an undirected graph, in a directed graph, if node n_1 is connected to node n_2, there is a relationship from n_1 to n_2, but not necessarily from n_2 to n_1. For instance, in a network flow model, each connection has a source IP and a destination IP. This scenario can be represented by a directed graph, where the relationship begins at the source IP and ends at the destination IP. In visual representations of directed graphs, the links are depicted with arrows. These arrows indicate that the origin node has a relationship with the destination node, but the reverse relationship may not exist.

The formal definition can be expressed as follows:

$$L \subseteq \{(i, j) \mid i, j \in N, (i, j) \neq (j, i)\} \tag{2.2}$$

Multigraphs: In a typical undirected graph, each pair of nodes is connected by a single link. Likewise, in a directed graph, each node can connect to another node and vice versa, but each connection is singular. A multigraph, whether directed or undirected, allows for multiple links between the same pair of nodes. This is valuable when, for instance, modeling multiple connections between two IP addresses. In such a multigraph, there would be an edge between the two IP addresses for each individual connection observed.

The formal definition can be expressed as follows:

$$L \subseteq \{(i, j, n) \mid i, j \in N, n \in \mathbb{N}\} \tag{2.3}$$

Here, n indexes the multiple links between nodes i and j.

Bipartite Graph: A bipartite graph is a type of graph in which the nodes can be split into two distinct sets, with all links connecting a node from one set to a node from the other set. No links exist between nodes within the same set. This structure is useful in modeling scenarios where there are two distinct

types of entities that interact with each other, such as users and resources in a resource allocation problem. Bipartite graphs are therefore commonly used in matching problems.

The formal definition can be expressed as follows:

$$N = N_1 \cup N_2, \quad L \subseteq \{(i,j) \mid i \in N_1, j \in N_2\} \tag{2.4}$$

Complete Graph: A complete graph, also known as a fully connected graph, is a type of graph in which every pair of distinct nodes is connected by a unique edge. In an undirected complete graph, this means that there is a direct link between every pair of nodes, and the connections are bidirectional. Complete graphs are crucial in scenarios requiring direct communication between all nodes, such as in the design of certain network topologies where every device needs to communicate directly with every other device without any intermediaries.

The formal definition can be expressed as follows:

$$L = \{(i,j) \mid i,j \in N, i \neq j\} \tag{2.5}$$

2.1.3. Application in Telecommunications

This graph-based abstraction is not merely a theoretical construct but a practical tool that significantly enhances our ability to manage, analyze and optimize networks. In telecommunication networks, graph theory helps to simplify the complexity by modeling the network as a collection of nodes and their interconnecting links. This representation aids in various analyses, such as:

- **Pathfinding and Routing:** Determining the most efficient paths for data transmission between nodes, considering factors like distance and bandwidth.
- **Network Reliability:** Assessing the network's robustness by identifying critical nodes and links, and analyzing potential points of failure.
- **Optimization:** Enhancing network performance through strategic placement of nodes and links, and optimizing the use of available resources.

Understanding these graph structures helps in designing networks that are not only efficient but also robust against failures, ensuring continuous service availability even when individual network components fail.

2.2. Matching

Having established a foundational understanding of graph theory and its application in modeling telecommunications networks, it becomes pertinent to explore how these theoretical concepts can be pragmatically applied to solve real-world problems. One application of graph theory in network design and optimization is through the use of matching algorithms. These algorithms provide a method to pair elements within a network in a manner that optimizes certain criteria, such as minimizing the distance between connected nodes. Matching algorithms leverage the graph-based representation of networks to identify and form optimal pairs of nodes.

In the following sections, we will give a formal definition of a matching and delve into various matching algorithms.

2.2.1. Definition

In the context of graph theory, a matching in a graph is a set of links that do not share any nodes. [10] Formally, a matching M is a subset of L such that for every two links l_1 and l_2 in M, the endpoints of l_1 are distinct from the endpoints of l_2. That is, no node is incident to more than one link from the matching. Mathematically, this can be expressed as:

$$M \subseteq L \quad \text{such that} \quad \forall l_1, l_2 \in M, \quad l_1 \cap l_2 = \emptyset$$

2.2.2. Weights

A weighted graph is the basis for applying matching algorithms. The objective is to maximize or minimize a certain attribute of the pairings, such as distance, cost or capacity. The choice of weight metric significantly influences the matching outcome and must be chosen based on the specific requirements of the network design. Distance between nodes and the associated costs will be the main weight in this research.

2.2.3. Algorithms

Greedy Approach

The greedy matching algorithm is a straightforward approach where pairs are formed by iteratively choosing the two closest unpaired nodes based on the link weights. While not always yielding the absolute optimal solution, this algorithm is efficient and often provides a sufficiently good solution for large graphs.

Algorithm Steps:

1. Sort all links in the graph by weight (distance).
2. Initialize an empty set of matches.
3. Iterate through the sorted link list and add an link to the match set if neither of the nodes it connects is already matched.

Perfect Matching

For more precise requirements, the Minimum Weight Perfect Matching algorithm finds a set of pairs such that every node is matched exactly once, and the total weight (sum of distances in this case) is minimized. This problem is efficiently solvable using algorithms like the Blossom algorithm for general graphs.

Selecting the right algorithm depends on the network's specific requirements: whether speed, precision, or stability is prioritized. In some cases, adaptations or combinations of the above algorithms may be necessary to meet specific operational requirements.

2.3. Shortest Paths

Another application of graph theory is finding the shortest routes between locations. This topic is particularly relevant to this study, as the goal is to minimize the costs associated with transforming rings into strands. These costs are strongly related to the length of the cables that need to be added to the network of KPN. If these cables were laid in a straight line, determining the route would be trivial. However, the cables follow the road network of the Netherlands, so a method must be found to calculate the best route. This section first provides a definition of the shortest path, followed by the methods to find it.

2.3.1. Definition

The shortest path problem involves determining the path with the minimum total distance or cost between two nodes in a weighted graph. Based on the earlier definition of a graph, we represent the graph as $G = (N, L)$, where N is the set of nodes and L is the set of links. Each link $(u, v) \in L$ has an associated weight $w(u, v)$, which indicates the cost, distance, or time required to traverse from node u to node v. [9]

Given a source node $s \in N$ and a destination node $t \in N$, the objective is to identify a path P from s to t such that the sum of the weights of the links in P is minimized. Mathematically, the shortest path P_{st} is represented as a sequence of nodes $\{n_1, n_2, \ldots, n_k\}$ where $n_1 = s$ and $n_k = t$, and the total weight $\sum_{i=1}^{k-1} w(n_i, n_{i+1})$ is minimized.

2.3.2. Dijkstra's Algorithm

Dijkstra's algorithm is one of the most widely used algorithms for finding the shortest path in graphs. It operates by maintaining a set of nodes whose shortest distance from the source is known and iteratively expands this set. [8] Dijkstra's algorithm does not handle negative link weights because it assumes

that once a node's shortest path is determined, it will not change. This assumption holds only if all link weights are non-negative. In the presence of negative weights, a previously determined shortest path could be invalidated by a path found later that includes a negative weight, leading to incorrect results. In the scope of this research this is not a problem, because the link weights are related to the distances between nodes and these are always positive.

The algorithm consists of the following three steps:

1. **Initialization:**
    - Set the distance to the source node s to 0 and to all other nodes to infinity.
    - Create a priority queue (min-heap) and insert the source node with distance 0.
2. **Relaxation:**
    - While the priority queue is not empty:
        - Extract the node u with the minimum distance from the priority queue.
        - For each neighbor v of u:
            - If the distance to v through u is less than the current known distance to v:
                - Update the distance to v.
                - Insert v into the priority queue with the updated distance.
3. **Termination:**
    - When the priority queue is empty, the shortest distances from the source to all nodes are known.

3

Approach

The objective of this study is to transform the ring structures within the KPN network into strands. This transformation aims to optimize the redundancy of the network while minimizing costs.

This chapter is structured to first give an overview of the algorithm used in this transformation process, followed by a detailed explanation of its various components.

3.1. Overview

The hierarchical structure of KPN's network allows for various divisions based on its function and geographical reach. Firstly, the network can be divided into layers that are closer to the internet exchange point in the Netherlands, and layers that are closer to individual households. Within this segmentation, the KPN network comprises three primary layers: the core network, the backhaul network, and the access network. Comparing the KPN network to a road system, think of the core network as the equivalent of major highways: they handle a lot of traffic, but there aren't many of them. In this analogy, the backhaul network functions like regional roads, linking local nodes to regional nodes. Lastly, the access network operates akin to residential streets, directly connecting individual homes and businesses to the broader network infrastructure. There are many of these connections, but they handle less traffic.

Another method of dividing the KPN network is based on its geographic distribution within the Netherlands. Here, the country is divided into 160 distinct regions, each of which is served by a MC location. These regions also contain multiple MA locations, along with various infrastructure components such as splitters, distribution points, and ultimately, connections to households. This division based on geographic regions is called an 'access area', denoting the specific area served by a particular set of network resources. It encompasses all the necessary components to deliver telecommunications services efficiently to the residents and businesses within that region.

An understanding of those divisions is required to comprehend the proposed multi-step algorithm, that aims to convert the rings in the current backhaul network into strands. The strand topology has already proven successful in the core network, where a chain of regional nodes are connected at both ends to a national node. Implementing a similar structure to connect MA locations to MC locations is expected to yield comparable benefits. The transformation is divided into three main components:

1. **The matching algorithm** is the initial step in this transformation process. It focuses on pairing access areas within the existing ring topology. These pairings are essential as they form the basis for the connections in the new strand topology. By carefully creating these pairs, the costs of the network transformation can be kept low.
2. **The routing algorithm** takes over when the access areas are paired. This component is responsible for determining the optimal paths between the paired access areas. It takes the dutch road network as input and outputs the route for the new cable connection.
3. **The strand algorithm** is the final step, which implements the actual reconfiguration of the network. Once the access areas are paired and connected, this step involves restructuring the network to

form the new strand topology.

The output of Algorithm 1 serves as the input for Algorithm 2, which in turn outputs to Algorithm 3. Consequently, the number of choices available in Algorithm $i+1$ is constrained by the decisions made in Algorithm i. To ensure that a wide range of options is considered, all three algorithms must run with minimal runtime complexity. This approach allows for multiple iterations through all the steps in search of the optimal outcome. The following sections will provide detailed explanations of each component of the algorithm.

The reason for an algorithm with multiple steps is the complexity of estimating the costs of the conversion to strands. In the following section, the large number of possible pairs is mentioned, and each of these pairs can be connected in numerous ways. Therefore, the pairing and the creation of connections between them are initially separated. After several optimization iterations of both algorithms, it became possible to combine these steps, merging the first two steps into one. This is described in section 4.3 regarding the results.

3.2. Matching Algorithm

A straightforward solution to the problem addressed in this research involves pairing each ring in the current backhaul network with the nearest ring from a different access area. This approach would connect two geographically redundant rings and reconfigure them into two strands (as further detailed in section 3.4 on the strand algorithm). However, a constraint exists at the MC locations that impedes this solution. Specifically, the equipment at MC locations is designed to operate in tandem, limiting communication to only two endpoints. Hence, if a access area X comprises two rings, it is not possible to pair ring X_a to access area Y and pair ring X_b to access area Z.

To overcome this limitation, the first step in the conversion process is the matching algorithm. The goal is to make pairs of access areas. This is used when creating strands from the rings in the paired access areas. There are multiple ways to create these pairs, but as the goal is to minimize the costs of the stand conversion, we also focus on that in this first step.

3.2.1. Fully Connected Graph

The matching algorithm uses graph $G = (N, L)$ to determine the optimal pairings of access areas. The set of nodes N in this graph represents the access areas, which number 160 in the KPN network. Since any access area can be paired with any other, the set of links consists of all possible node pairs. This type of graph is known as a complete graph (K_N) and has the following properties:

- Number of nodes[1]: $|N| = 160$
- Number of links: $|L| = \frac{N(N-1)}{2} = \frac{160 \times 159}{2} = 12,720$

3.2.2. Link Weights

Determining the costs of connecting access areas with cables is not straightforward. Therefore, we need an alternative measure to express these costs. A metric closely related to the costs of laying cables is the length of the cables to be added. Hence, in the matching algorithm, we use the distance between access areas as the weight to be minimized.

However, the exact distance to be minimized is also complex, as it involves the distance between the two access areas where the cables are closest. To approximate this distance, we use the coordinates of the MC locations as a starting point.

Version 1: Distance as the Crow Flies

In the first version of this matching algorithm, the direct (straight-line) distance between MC locations was used. This provides a reasonable approximation of the costs of forming strands between these access areas and is easy to calculate. However, analysis of the resulting pairs revealed that the costs in some cases were much higher than expected based on the direct distance. This discrepancy arises because KPN's cables predominantly follow the road network, where costs vary by road type. For

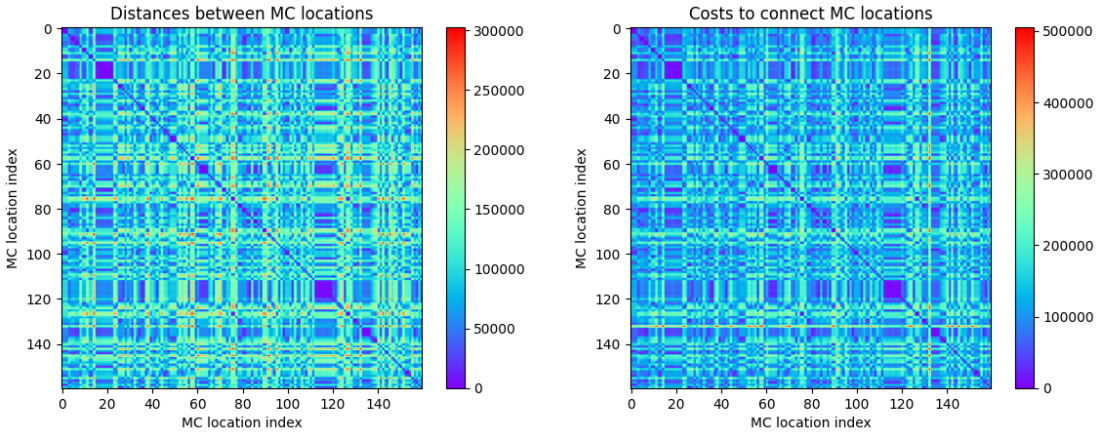
[1] In the remainder of this report, context will indicate whether the set or its cardinality is meant, without using the vertical bars.

instance, it is significantly more expensive to dig along highways, and laying cables over bridges and through tunnels also incurs higher costs.

Version 2: Distance by Road

Due to these shortcomings in version 1, a revised approach was adopted in version 2. Instead of using straight-line distances, link weights are determined by the distance along roads. To account for different road types, the distance over bridges and through tunnels is multiplied by a factor of 10, while the distance along highways is multiplied by a factor of 60. This adjustment aims to ideally avoid these roads or, at the very least, reflect realistic costs.

Both sets of link weights are plotted in Figure 3.1. It shows that both weights yield similar results. However, the 'costs' over the road are always higher than the straight-line distances, especially when the road is a highway or the road crosses bridges and tunnels.



(a) Weight matrix using the distance as the crow flies.

(b) Weight matrix using an approximation of costs by road.

Figure 3.1: The distance matrices used to form a perfect matching.

3.2.3. Runtime complexity

Ideally, each access area is paired with another access area, meaning we seek a perfect matching that covers every node in the graph. The number of possible perfect matchings in a graph can be calculated based on the number of nodes.

Number of perfect matchings: $N_{pm} = (N-1)!! = 159!! = 5.449 \times 10^{141}$

Using a brute-force approach, this is the number of options that must be considered. However, Zvi Galil [10] describes an algorithm with a runtime complexity of $O(N^3)$. This significantly reduces the number of options to be considered when the number of nodes is large, as is the case here. This comparison is shown in Figure 3.2.

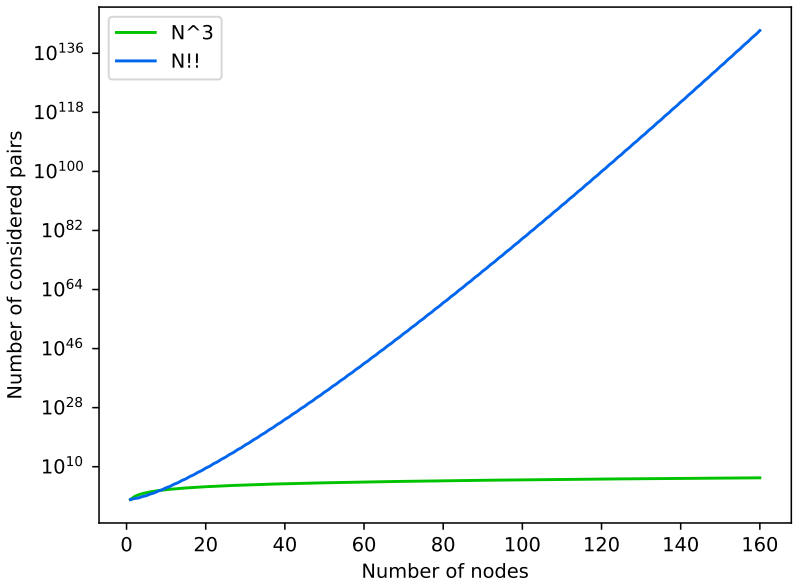


Figure 3.2: A comparison between the brute-force ($N!!$) and Galil (N^3) approach.

3.3. Routing Algorithm

With the access areas identified for connection, the next step is to determine the optimal route for these connections. A shortest path algorithm naturally brings to mind Dijkstra's algorithm, but modifications are necessary to adapt it for this specific case. Dijkstra's algorithm is designed to find the cheapest path between node a and node b in a graph. However, an access area consists of multiple rings, and each ring comprises multiple coordinates. Therefore, Dijkstra's algorithm must be rewritten to handle multiple sources and multiple destinations.

3.3.1. Input graph

As Dijkstra's algorithm is designed to work on a graph, the question arises regarding which graph should be used. Thanks to OpenStreetMap[2], data on the road network in the Netherlands is publicly available. However, this data is not initially in graph format. Fortunately, the Python package OSMnx [3] can convert OpenStreetMap data into graph format. The links in this graph represent the roads in the Netherlands, including information such as name, type, and length. The nodes correspond to the intersections of these roads.

In addition to the road network, the graph must also include the paths of the rings. Since the coordinates of the cables are known, we can combine these with the coordinates of the nodes. All nodes where the rings of access area a pass are marked as sources, and all such nodes of access area b are marked as destinations. This combined data forms the input for the routing algorithm.

3.3.2. Multi-sources multi-destinations Dijkstra

In the extended version of Dijkstra's shortest path algorithm, each source node is initialized with a distance of zero and inserted into a priority queue. This priority queue is used to manage the nodes to be explored based on their tentative distances. The core of the algorithm involves extracting the node with the smallest distance from the priority queue, updating the distances to its neighbors if a shorter path is found, and pushing the neighbors into the queue. This process repeats until the queue is empty or a destination node is reached. When a destination node is found, the algorithm reconstructs the path by tracing back from the destination node to the source node using a predecessor dictionary. If no destination is reachable from the sources, the algorithm returns an indication of failure. The complete

[2] `https://www.openstreetmap.org/about`

algorithm can be found in Appendix A.

3.3.3. Runtime complexity

The runtime complexity of the algorithm is mostly determined by the operations involving the priority queue and graph traversal.

Firstly the source and destination nodes are identified based on a specific attribute. Suppose there are S source nodes and D destination nodes. The nodes are filtered and classified, which takes $O(N)$ time, where N is the total number of nodes in the graph.

For each source node, the algorithm sets its distance to zero and inserts it into the priority queue. Inserting each node into the priority queue takes $O(\log S)$ time. Thus, initializing all source nodes into the priority queue takes $O(S \log S)$ time.

The core of the algorithm involves processing nodes extracted from the priority queue. Each node is processed exactly once, and for each node, all its neighbors are examined. Extracting a node from the priority queue takes $O(\log N)$ time. Checking and updating the distances to each neighbor involves a priority queue insertion, which is also $O(\log N)$. Consequently, processing all nodes and their links results in a time complexity of $O((N + L) \log N)$, where L is the total number of links.

Path reconstruction, which occurs after finding a destination node, involves tracing back through the predecessor dictionary from the destination node to a source node. This operation takes $O(N)$ time in the worst case but is generally negligible compared to the main loop's complexity.

Therefore, the overall runtime complexity of the algorithm is $O((N+L) \log N)$. This makes the algorithm efficient and suitable for large graphs, as it scales logarithmically with the number of nodes and links in the graph.

3.4. Strand Algorithm

The strand algorithm is the final step in the conversion process, where the network is reconfigured based on the identified paths and connections. The input of the algorithm consists of a graph with the original ring structure and the added paths between those rings.

3.4.1. Georedundancy

As the objective of the algorithm is to add redundancy to the network, the strands created in this step must follow the georedundancy guidelines that also applied to the rings. This means that the left and right sides of any node in a strand must not be positioned close to each other, in order to prevent a single cable cut from disconnecting nodes. In practice this requirement handled by default, because the strand structure reuses the cables that were already used in the ring structure. The next sections will verify this claim by illustrating three possible scenarios.

Rings separated

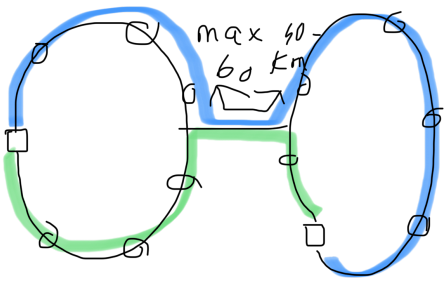
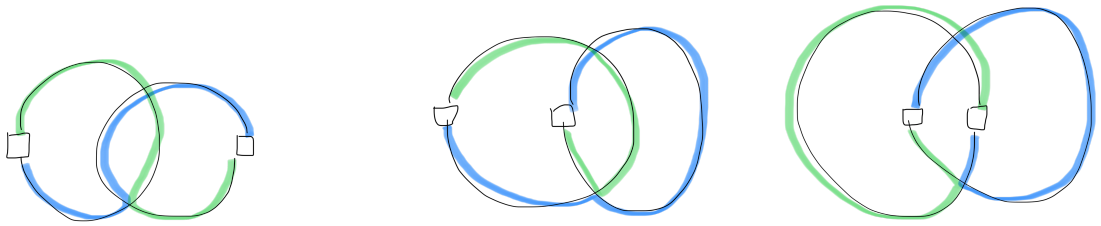


Figure 3.3: Scenario 1 where the rings are separated.

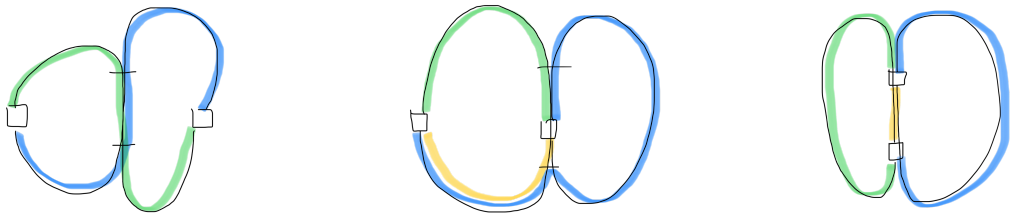
Rings cross



(a) Both MC locations outside intersection. **(b)** One MC location within intersection. **(c)** Both MC locations within intersection.

Figure 3.4: Scenario 2 where the rings cross.

Rings share duct



(a) Both MC locations outside shared duct. **(b)** One MC location within shared duct. **(c)** Both MC locations within shared duct.

Figure 3.5: Scenario 3 where the rings share a duct.

3.4.2. Implementation of Paths

The strand algorithm comprises two core processes designed to operate on a graph structure.

The first process is a recursive depth-first search algorithm that identifies all possible paths between a specified source and destination node within a graph (the MC locations). It maintains a list of all discovered paths while traversing the graph. Each neighbor of the current node is explored, ensuring that each link is visited only once during the traversal of a single path. Upon reaching the destination node, the current path is added to the list of all paths.

The second process aims to find a subset of paths from the previously identified set that covers all nodes in the graph with the minimum total weight. It begins by calculating all possible paths between the start and end nodes. For each subset of these paths, it checks if the subset covers all nodes in the graph. If so, it computes the total weight of the subset. The weight of a path is determined by summing the weights of the links in the path, where link weights correspond to the distances between nodes. This process iterates through all possible subsets of the paths, starting from subsets containing the smallest number of paths, and keeps track of the subset with the minimal total weight that covers all nodes. The minimal subset, once identified, is returned, ensuring the most efficient coverage of the graph in terms of total path weight.

Together, these processes enable the identification of not just all potential paths between two nodes, but also the optimal combination of these paths that spans the entire graph with the least cumulative weight. This is crucial for achieving network redundancy at low costs in the new strand structure.

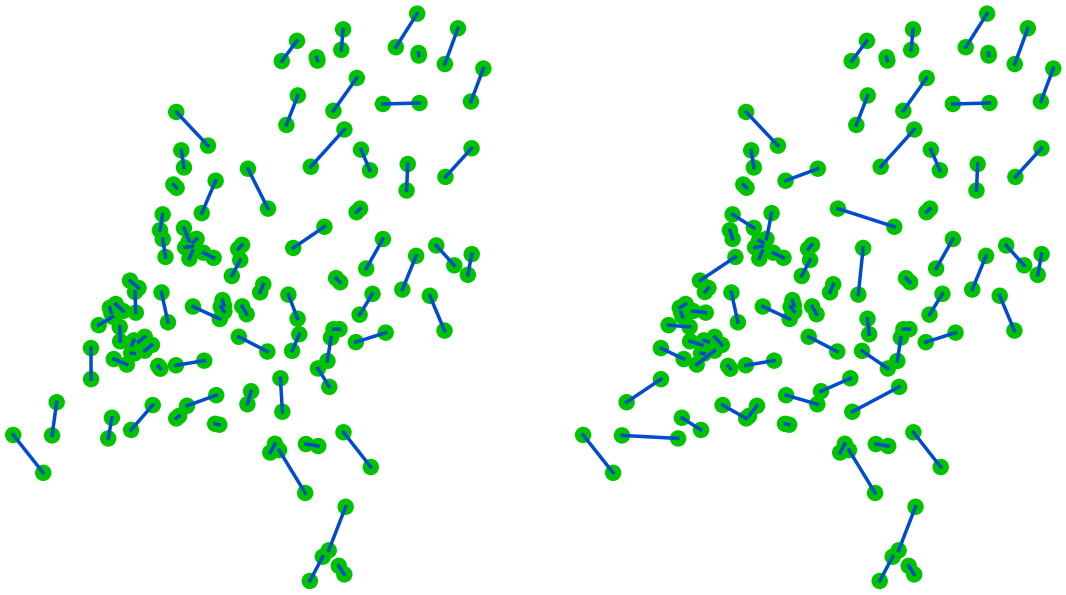
4

Results

This chapter presents the results generated by the algorithms described in the previous chapter. During the analysis of the results it turned out that in contrast to expectation the matching and routing algorithms can be combined to create more cost-effective pairs. Therefore section 4.3 is added to discuss the results that follow from this.

4.1. Matching

In section 3.2 two versions of the matching algorithm were presented. Version 1 used distance as the crow flies and version 2 improved upon this using distance by road. Additionally version 2 algorithm introduced a cost multiplier for highways and roads that go through tunnels or over bridges. There are 160 MC locations, so perfect matching results in 80 pairs, which are visualized in Figure 4.1.



(a) Pairs based on distance as the crow flies.

(b) Pairs based on distance by road.

Figure 4.1: The pairs created by the matching algorithm.

It is clear that the pairs created by both versions of the algorithm are very similar; 52 pairs remain the same in version 2. In many cases, an optimization for straight-line distance also serves as an optimization for road distance. However, there are some pairs in version 1 that are separated by bodies

of water, which causes the road distance to be much longer than the straight-line distance. Since cables are laid along roads, it is better to work with these distances. This results in different pairs, particularly evident in the western part of the Netherlands.

A good example of a case where version 2 is a better cost estimator is depicted in Figure 4.2. In this instance, version 1 created a pair that crossed the North Sea Canal near IJmuiden, because in a straight line the MC locations are close together. However, in reality the cables need to go through a tunnel in this case, which would make this pair very expensive. In version 2, this was recognized, and the access area north of the canal got paired with the other access area visible on the right side of the image. Although the distance between these access areas is greater, the costs to connect these will be lower, as there is no need to cross a body of water.

Figure 4.2: An example of MC areas that are close together but expensive to connect.

Version 1 of the algorithm, optimized for straight-line distance, results in MC location pairs that have a minimal Euclidean distance. However, this is not necessarily the best estimator of the actual costs to connect the rings of these MC areas. The fiber optic cables will not be laid in a straight line but along roads. Moreover, the costs of new cables along some roads are higher than along others. This is accounted for in the second version of the matching algorithm, and the different outcomes are shown in Table 4.1. Since version 1 of the algorithm uses straight-line distance as a metric and version 2 uses a cost approximation, both metrics are displayed in the table for comparison. This is done by first determining which pairs would created by V1 and then looking up the values for these pairs in the cost matrix used for V2. Similarly the distances between the pairs created by V2 can be looked up in the distance matrix used by V1.

Table 4.1: A comparison between the results of both versions of the matching algorithm.

(a) Pairs created by version 1.

Statistic	Distance	Cost
count	80	80
mean	10,848	17,450
std	6,927	17,223
min	1,252	1,864
25%	4,427	5,955
50%	10,367	16,149
75%	16,459	21,490
max	24,982	131,987
sum	867,845	1,396,007

(b) Pairs created by version 2.

Statistic	Distance	Cost
count	80	80
mean	11,772	16,368
std	7,699	16,236
min	1,252	1,864
25%	4,427	5,719
50%	11,970	15,488
75%	17,246	21,881
max	29,500	131,987
sum	941,796	1,309,465

In Figure 4.3, the same results are presented in a line graph, where the distance and cost of each MC pair are sorted in descending order and then accumulated. It is clear that version 2 of the algorithm yields worse results for straight-line distance, as it is not optimized for this metric. However, version 2 does achieve lower costs compared to version 1. Since the cost metric is more effective at creating cheap strands, the distance metric will be excluded from further consideration.

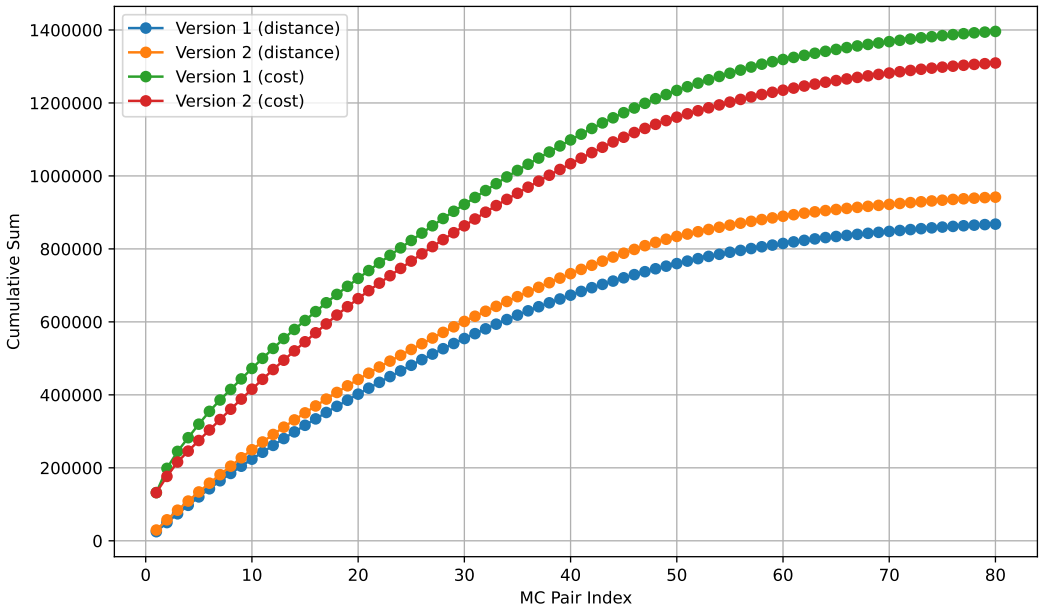


Figure 4.3: A comparison between the two versions of the matching algorithm for both distance and cost.

Note that the distances between the MC areas in this matching algorithm are determined by the coordinates of the MC locations. In reality, the costs are primarily determined by the distance between the nearest rings of different MC areas. However, the matching algorithm is also intended to deliver results quickly, allowing multiple configurations to be tested. Using multiple coordinates per access area would not contribute to this desired speed. Therefore, the coordinates of the MC location were chosen to approximate the distances.

4.2. Routing

Now that the matching algorithm has determined which MC areas will be connected, the next step is to find the actual shortest route between them. For this, the coordinates of the MC locations are no longer used as an approximation of the nearest pairs. Instead, it is the cables in the rings themselves that need to be connected. Generally, these will be closer to each other than the MC locations, which form the center of the access area.

The routing algorithm described in section 3.3 works with a graph containing the road network of the Netherlands. This graph is created using OSMnx [3] and initially consists of 18,885,173 nodes and 39,253,828 links. This is not ideal because the runtime of the routing algorithm scales with the number of nodes and links, which is quite large in this case. Therefore, some optimization steps are required. The first step is simplifying the graph by removing all nodes that are not intersections or dead-ends, by creating a link directly between the end points that encapsulate them. This already reduces the number of nodes to 4,808,392 and the number of links to 12,846,075. Secondly, the map of the entire Netherlands is not needed to calculate the shortest route between two access areas. Thus, all nodes and links falling outside the extreme coordinates of the two respective access areas are removed. As the third optimization step, duplicate links between the same nodes are removed. Originally, the

graph is a multigraph, where multiple roads are possible between the same nodes, but for the routing algorithm, only the shortest link needs to be retained. The fourth and final optimization is the conversion from a directed graph to an undirected graph. While roads often allow traffic in only one direction, this does not matter for the cables to be laid. Thus, two directional links between the same nodes can be replaced by a bidirectional link. After these optimizations, the graph per MC pair consists on average of 61,253 nodes and 88,469 links, which significantly reduces the runtime of the algorithm.

An example of such an optimized graph is shown in Figure 4.4. It consists of 24,145 nodes and 31,989 undirected links. The red lines represent the rings from two paired access areas that need to be connected, and the green and blue nodes indicate intersections close to those rings. The algorithm is supposed to find the closest pair of a green and a blue dot following the road network. In this case, it has basically two choices: the left dam or the right dam. Because the green and blue dots (i.e., the source and destination nodes) are closer to each other near the right dam, the routing algorithm picks the road over the right dam as the new cable connection. The distance over the road between the nearest green and blue dots is 8,482 meters. The scaled distance (including the multiplier for highways, bridges, and tunnels) is higher, namely 9,429, because 105 meters is marked as a bridge and thus multiplied by 10.

Figure 4.4: Two access areas plotted on the road network.

A summary of the results of the routing algorithm is shown in Table 4.2. This clearly shows that the pairs created by V2 of the matching algorithm contain rings that have a smaller scaled distance than the pairs created by V1. This means that the V2 pairs are likely to result in smaller costs to connect, than the pairs created by V1. Another noticeable difference is that only 79 of the V1 pairs could be connected by the routing algorithm. This is because version 1 of the matching algorithm created a pair where no road exists between the access areas in the cropped graph of the road network. Therefore the distance by road could not be computed.

Table 4.2: The scaled distance between pairs created by V1 and V2 of the matching algorithm.

Statistic	Version 1	Version 2
count	79	80
mean	11,725	9,179
std	52,365	50,882
min	2	2
25%	7	6
50%	3,551	3,250
75%	5,833	5,197
max	457,149	457,149
sum	926,240	734,344

With an average scaled distance of 9,179, the rings are, as predicted, closer to each other than the cost estimation of 16,368 given by the matching algorithm as output. This can be well explained by the routing algorithm using all coordinates of the rings instead of only the coordinates of the (centrally located) MC location. Three-quarters of the pairs even have a scaled distance within 5,197, compared to the estimated value of 21,881 from the matching algorithm. However, the scaled distance of the most expensive pair is much higher than the costs estimated by the matching algorithm. The reason for this can again be explained by the fact that the routing algorithm does not use the complete road network but only a cropped version using the extreme coordinates of the two access areas. This is depicted in Figure 4.5. As a result, only one route remains which goes through the Westerschelde tunnel and is also a highway. In reality, there is also a much cheaper route that avoids tunnels and highways, namely via a part of Belgium.

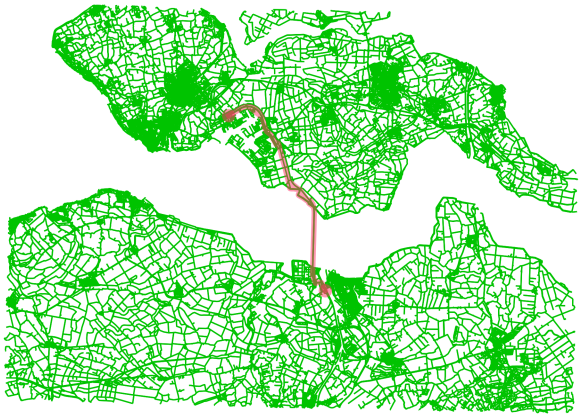


Figure 4.5: The shortest route between two rings.

4.3. Matching and Routing Combined

To reduce the number of nodes and links in the graph used for the routing algorithm, it initially seemed beneficial to limit the graph to the outer boundaries of the paired access areas. This prevented the graph from exceeding memory limits during loading. However, this 'fix' simultaneously introduced the problem that the best route was not always found. Therefore, in a second attempt to solve this problem, the graph was stored in a more efficient format. Although this significantly increases the number of nodes (4,808,392) and links (6,775,849) in the graph, it has only a limited impact on the speed of the routing algorithm. This is due to the way the shortest path algorithm operates. It spreads like an oil slick over the road network in search of the nearest target. Therefore, the routing algorithm will not suddenly start exploring roads in Groningen when it is looking for the nearest points in South Holland.

With this optimization, the possibility arose to improve the matching algorithm as well. Whereas initially only one point was used to pair the access areas, it became possible to use the actual coordinates

of the rings. As a result, instead of the existing distance (V1) and cost (V2) matrices, a new matrix can be filled with the scaled distance between the nearest rings of all possible pairs of access areas. To accelerate the calculation of these values, the fact that certain pairs are not possible due to the restriction that the cable length between MA locations must not exceed 40 kilometers is taken into account. This is due to the light intensity of the lasers used. Therefore, if the routing algorithm has not found a ring of the relevant access area within a radius of 40 kilometers, a very high value is entered in the matrix. This is intended to prevent these access areas from being matched and ensures that no time is wasted calculating the shortest route between these rings.

While calculating the shortest distances, the shortest route can also be stored. When the matching algorithm has determined which access areas can be best combined, the route between them can also be output immediately. The (one-time) filling of the matrix may take considerably longer (around 12 hours on a powerful laptop), but for creating other pairs, the routing algorithm does not need to run again.

The pairs and distances resulting from this run can be found in Appendix B. One pair has been created that is at the limit of the maximum distance between MA locations. For this access area, it is therefore recommended to add redundancy in another way, as described in section 1.5. Additionally, the maximum desired distance in the matching algorithm can still be tuned. For example, by removing all links between nodes with a distance greater than 6 kilometers, not all access areas will be able to be matched anymore. However, the sum of the remaining pairs will then decrease significantly and may even be lower than the sum of the first 75 pairs of the current run.

Figure 4.6 shows the advantage in scaled distance when combining the algorithms. Especially the worst pair in terms of scaled distance is improved a lot. This is due to the fact that the combined algorithm is able to choose a route via Belgium, instead of having to go via a highway, as explained in section 4.2. Some other pairs also slightly improved, but it can be concluded that the cost metric of the matching V2 algorithm is a decent indicator of the distance between the rings of MC pairs.

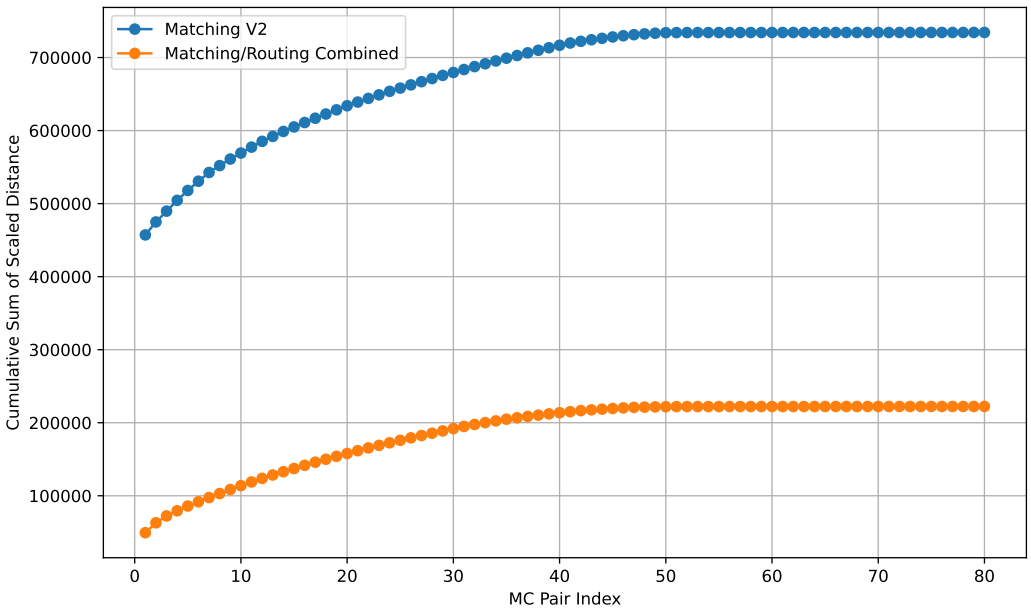
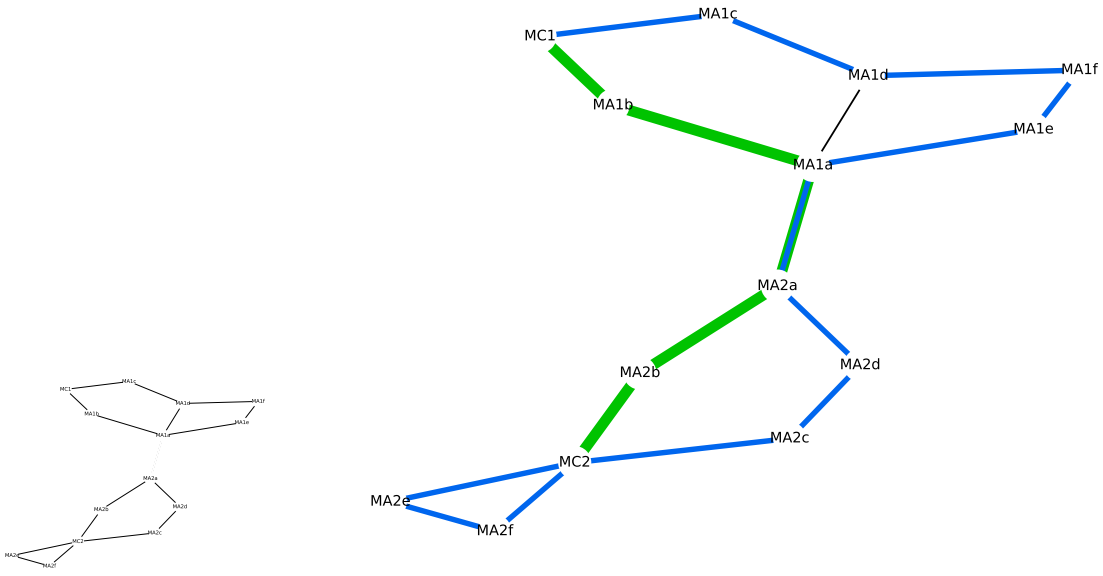


Figure 4.6: A comparison of scaled distance between rings of MC pairs using the matching V2 algorithm vs. the combined algorithms.

4.4. Strand

The creation of strands is the last step in the algorithm, as stated in section 3.4. It takes two connected rings as input graph, and outputs the cheapest paths from MC location 1 to MC location 2. The example output is depicted in Figure 4.7. The number indicates to which ring the location belonged before. Both rings included one MC location and six MA locations. The link (MA1a, MA2a) in this figure was created by the routing algorithm to connect the two rings. The green and blue lines show the new strand structure, starting at MC1, covering each MA location and finishing at MC2.



(a) The original rings. **(b)** The new strands.

Figure 4.7: An imaginary example of the new strand structure from two connected rings.

For the example route that was shown in section 4.2, the strands would follow the path as depicted in Figure 4.8.

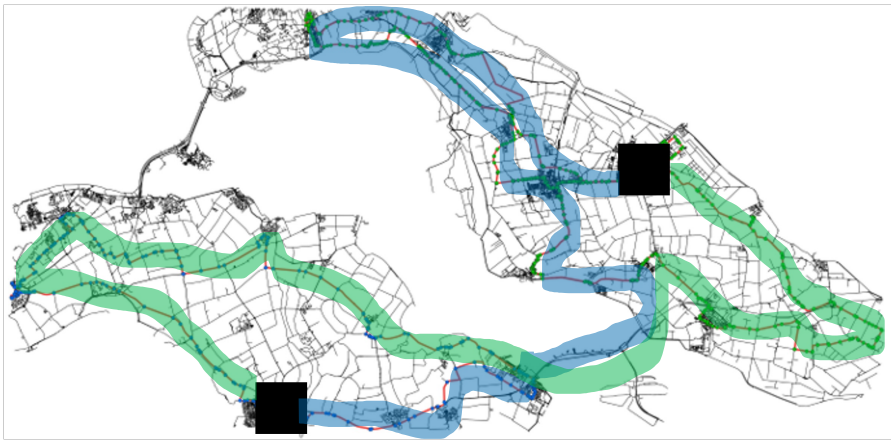


Figure 4.8: The new strand structure of a real MC pair.

5

Conclusion

This research explored ways to enhance the robustness of KPN's network. In recent years, society has increasingly relied on digital infrastructure, with KPN being a crucial component. There are various methods to improve the robustness of a telecom network. On one hand, the occurrence of failures can be minimized, and resilience can be enhanced by using stronger cables, for example. On the other hand, the impact of failures can be mitigated, and redundancy can be increased by adding extra cables to the network. The focus of this research was on increasing redundancy, specifically by transforming the existing rings in the network into strands. A three-step algorithm was used to select the rings to transform, compute the length of the cables to be added, and find the routes for the strand structure.

The current ring structure can avoid single points of failure (SPOFs) at the link level. If nodes cannot be reached clockwise due to a cable break, they can still be reached counterclockwise, and vice versa. However, SPOFs at the node level cannot be avoided with this structure. This is particularly problematic for MC locations, which are responsible for setting up customer sessions and all network traffic from households is routed to/through these nodes. Failures at these locations can be catastrophic for all connections in an MC area (up to 100,000 households).

Q1: Advantages of the strand structure Therefore, it is crucial that a single node-level failure can be compensated for by another node. Two types of nodes are distinguished: equipment and buildings. If a piece of equipment fails, it could be mitigated by having backup equipment ready at the same location. However, scenarios where an entire location becomes unavailable, such as due to natural disasters, must also be considered. In such cases, the activities of the original location must be transferred to a geographically distant location. Unlike the ring structure, a strand structure can achieve this. A strand always contains two MC locations that are geographically separated.

Q2: Costs of transformation into strands However, transforming rings into strands incurs high costs. It is necessary to map these costs before deciding if the benefits of this transformation outweigh them. An actual cost estimation is beyond the scope of this research. Instead, the length of the additional cables needed for the network is approximated. It involves 206 kilometers of cable. Costs also depend on the type of road the cables traverse. Thus, a multiplier of 10 is applied per meter of tunnel or bridge, and a multiplier of 60 per meter of highway. Including these multipliers, the scaled distance of the cables to be added amounts to 222.

Q3: Risks of transformation into strands Although the exact costs of transforming to strands are unknown, it is clear that adding 206 kilometers of cable to KPN's network comes with a significant price tag. Moreover, node failures at the equipment level (the most common type) can be prevented by placing redundant equipment at MC locations. Therefore, another option could be to form strands only when the rings of MC areas already run close to each other. This approach would reduce excavation work and hence costs.

5.1. Future Work

Since the matching algorithm forms the basis for further results, optimization in this area significantly impacts the overall outcome of the algorithm. This research mainly focussed on the option of a perfect matching, where all MC areas are paired. However, this results in some pairs being far apart, making the benefits of connecting them likely not worth the costs. Future work could investigate a threshold above which MC areas are not paired. Relaxing the requirement for a perfect match might yield different (cheaper) results.

Additionally, cables already run throughout the Netherlands to connect MC locations. It is cheaper to use existing fiber optic cables and ducts compared to laying new cables. Future work could therefore examine whether there is available capacity in the existing network. If this can prevent excavation work between some pairs, it could save money.

Finally, the current approach is to lay one new cable between MC areas. However, there are situations where strands become very long, and laying two new cables might not incur much additional cost. Future research could investigate whether adding a feedback loop after the strand algorithm results in better strands. This means that when long strands are identified and the original rings run close to each other at multiple locations, an additional cable is laid between them.

5.2. KPN Recommendations

The algorithms presented in this work and the resulting outcomes provide a foundation for making an informed decision about whether or not to transform rings into strands. In particular, Appendix B can be consulted to determine whether the costs outweigh the benefits based on the length of the cables to be added. Additionally, a decision can be made to opt for a hybrid solution, where only the rings of the most cost-effective MC pairs are transformed into strands.

References

[1] S Babani et al. “Comparative study between fiber optic and copper in communication link”. In: *Int. J. Tech. Res. Appl* 2.2 (2014), pp. 59–63.

[2] David Bailey and Edwin Wright. “1 - Introduction”. In: *Practical Fiber Optics*. Ed. by David Bailey and Edwin Wright. Oxford: Newnes, 2003, pp. 1–7. ISBN: 978-0-7506-5800-3. DOI: https://doi.org/10.1016/B978-075065800-3/50001-1. URL: https://www.sciencedirect.com/science/article/pii/B9780750658003500011.

[3] Geoff Boeing. *Modeling and Analyzing Urban Networks and Amenities with OSMnx*. 2024. URL: https://geoffboeing.com/publications/osmnx-paper/ (visited on 05/28/2024).

[4] R. van de Bovenkamp. “A simulation and capacity management tool for KPN’s Ethernet network”. MA thesis. Delft University of Technology, 2010.

[5] Yufei Cheng et al. “Analysing GeoPath diversity and improving routing performance in optical networks”. In: *Computer Networks* 82 (2015). Robust and Fault-Tolerant Communication Networks, pp. 50–67. ISSN: 1389-1286. DOI: 10.1016/j.comnet.2015.02.021. URL: https://www.sciencedirect.com/science/article/pii/S1389128615000699.

[6] Amaro de Sousa et al. “Minimization of the network availability upgrade cost with geodiverse routing for disaster resilience”. In: *Optical Switching and Networking* 31 (2019), pp. 127–143. ISSN: 1573-4277. DOI: 10.1016/j.osn.2018.10.003. URL: https://www.sciencedirect.com/science/article/pii/S1573427718300687.

[7] *De techniek achter KPN koper en glasvezel*. URL: https://community.kpn.com/kennisbank-internet-132/de-techniek-achter-kpn-koper-en-glasvezel-456480 (visited on 07/12/2024).

[8] E. W. Dijkstra. “A note on two problems in connexion with graphs”. In: *Numerische Mathematik* 1.1 (Dec. 1959), pp. 269–271. ISSN: 0945-3245. DOI: 10.1007/BF01386390. URL: https://doi.org/10.1007/BF01386390.

[9] David Eppstein. “Finding the k Shortest Paths”. In: *SIAM Journal on Computing* 28.2 (1998), pp. 652–673. DOI: 10.1137/S0097539795290477. eprint: https://doi.org/10.1137/S0097539795290477. URL: 10.1137/S0097539795290477.

[10] Zvi Galil. “Efficient algorithms for finding maximum matching in graphs”. In: *ACM Comput. Surv.* 18.1 (Mar. 1986), pp. 23–38. ISSN: 0360-0300. DOI: 10.1145/6462.6502. URL: https://doi.org/10.1145/6462.6502.

[11] *Glas aan, koper uit*. URL: https://netwerk.kpn.com/glas-aan-koper-uit/ (visited on 07/10/2024).

[12] Teresa Gomes and José Craveirinha. “Efficient calculation of the most reliable pair of link disjoint paths in telecommunication networks”. In: *European Journal of Operational Research* 181.3 (2007), pp. 1055–1064. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2006.03.005. URL: https://www.sciencedirect.com/science/article/pii/S037722170600141X.

[13] M. Farhan Habib et al. “Disaster survivability in optical communication networks”. In: *Computer Communications* 36.6 (2013). Reliable Network-based Services, pp. 630–644. ISSN: 0140-3664. DOI: 10.1016/j.comcom.2013.01.004. URL: https://www.sciencedirect.com/science/article/pii/S0140366413000224.

[14] Adeel Iqbal et al. “Cognitive D2D communication: A comprehensive survey, research challenges, and future directions”. In: *Internet of Things* 24 (2023), p. 100961. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2023.100961. URL: https://www.sciencedirect.com/science/article/pii/S2542660523002846.

[15] *KPN Netwerk - Glasvezelnetwerk: Aanleg en onderhoud*. URL: https://gathering.tweakers.net/forum/list_messages/2052244 (visited on 07/15/2024).

[16] D.R. Kuhn. “Sources of failure in the public switched telephone network”. In: *Computer* 30.4 (1997), pp. 31–36. DOI: 10.1109/2.585151.

[17] Leigh Metcalf and William Casey. “Chapter 5 - Graph theory”. In: *Cybersecurity and Applied Mathematics*. Boston: Syngress, 2016, pp. 67–94. ISBN: 978-0-12-804452-0. DOI: https://doi.org/10.1016/B978-0-12-804452-0.00005-1. URL: https://www.sciencedirect.com/science/article/pii/B9780128044520000051.

[18] Justin P. Rohrer, Abdul Jabbar, and James P.G. Sterbenz. “Path diversification: A multipath resilience mechanism”. In: *2009 7th International Workshop on Design of Reliable Communication Networks*. Oct. 2009, pp. 343–351. DOI: 10.1109/DRCN.2009.5339988.

[19] Stojan Trajanovski et al. “Finding Critical Regions and Region-Disjoint Paths in a Network”. In: *IEEE/ACM Transactions on Networking* 23.3 (June 2015), pp. 908–921. ISSN: 1558-2566. DOI: 10.1109/TNET.2014.2309253.

[20] Bruno Vidalenc et al. “Dynamic risk-aware routing for OSPF networks”. In: *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. 2013, pp. 226–234.

[21] Vitaly I. Voloshin. *Introduction to Graph Theory*. New York: Nova Science Publishers, Inc., 2009. ISBN: 9781614701132. URL: http://ebookcentral.proquest.com/lib/delft/detail.action?docID=3019331.

[22] Jianan Zhang, Eytan Modiano, and David Hay. “Enhancing Network Robustness via Shielding”. In: *IEEE/ACM Transactions on Networking* 25.4 (Aug. 2017), pp. 2209–2222. ISSN: 1558-2566. DOI: 10.1109/TNET.2017.2689019.

A

Routing algorithm

```
def get_cheapest_source_destination(self, sources: List[int], targets: List[int]) -> Tuple[
    List[int], int, int, float]:
    """Get the cheapest path from one of a set of sources to one of a set of targets.

    Returns:
        tuple (the cheapest edge path, the source, the target, the costs)
    """
    discovered = []
    dist = defaultdict(lambda: 1e99)
    pred = dict()

    for source in sources:
        heapq.heappush(discovered, (0, source))
        dist[source] = 0
        pred[source] = -1

    found_target = -1
    visited = set()
    done = False
    while not done:
        if len(discovered) == 0:
            done = True
            continue
        d_u, u = heapq.heappop(discovered)

        if u in visited:
            # nodes can be discovered multiple times because we don't update them in the heap
            continue
        visited.add(u)

        if u in targets:
            found_target = u
            done = True
            continue

        # Update distances of unvisited neighbors
        for v in self.graph.neighbors(u):
            if v not in visited:
                length = self.graph.get_edge_data(u, v).get('length', 999999)
                d = dist[u] + length
                if d < dist[v]:
                    dist[v] = d
                    pred[v] = u
                heapq.heappush(discovered, (d, v))

    if found_target == -1:
        warnings.warn("None␣of␣the␣targets␣can␣be␣reached␣from␣the␣sources.")
        return [], -1, -1, -1

```

```
    p = [found_target]
    while pred[p[-1]] != -1:
        p.append(pred[p[-1]])

    p = p[::-1]  # Reverse the path to get it in source-to-target order

    # Get the edge list from the path nodes
    path_edges = [(p[i], p[i + 1]) for i in range(len(p) - 1)]

    return path_edges, p[0], found_target, dist[found_target]
```

B

MC pair results

Table B.1: Overview of the most cost-effective MC pairs, sorted by scaled distance. The values indicate the length of the cables to be added, expressed in meters.

	MC1	MC2	Highway	Bridge/Tunnel	Distance	Scaled Distance
1.	**Asd-Slod**	**Asd-Slov**	0	0	2	2
2.	**Zl**	**Zl-N**	0	0	2	2
3.	**Amf**	**Amf-Zlh**	0	0	3	3
4.	**Ht-Slg**	**Ht-Vt**	0	0	3	3
5.	**Asd-Spui**	**Asd-Z**	0	0	3	3
6.	**Hrl**	**Hrl-Hbk**	0	0	3	3
7.	**Asd-Asv**	**Asd-Drs**	0	0	3	3
8.	**Al**	**Ledn-Wdz**	0	0	3	3
9.	**Ut-C**	**Ut-Ovv**	0	0	3	3
10.	**Amr**	**Amr-C**	0	0	3	3
11.	**Bd**	**Bd-Dnbs**	0	0	4	4
12.	**Gv-Bnh**	**Ledn-Csl**	0	0	4	4
13.	**Alr-Hvn**	**Alr-Sfk**	0	0	4	4
14.	**Nm**	**Nm-Dkbg**	0	0	4	4
15.	**Ap-W**	**Ap-Z**	0	0	4	4
16.	**Ed-Klh**	**Wg**	0	0	5	5
17.	**Hlm-San**	**Hlm-Wdp**	0	0	5	5
18.	**Rt-Pdt**	**Rt-Wah**	0	0	5	5
19.	**Ven**	**Vl**	0	0	6	6
20.	**Hvs**	**Hz**	0	0	6	6
21.	**Rt-Klg**	**Rt-N**	0	0	6	6
22.	**Ble**	**Spij**	0	0	7	7
23.	**Ehv-Ton**	**Ehv-Vkh**	0	0	7	7
24.	**Gv-Mx**	**Lsdm-Lev**	0	0	8	8
25.	**Rt-Grod**	**Rt-W**	0	0	8	8
26.	**Niwg-N**	**Ut-Z**	0	0	9	9
27.	**Dt**	**Gv-Btij**	0	0	14	14
28.	**Ah-Rkw**	**Drt**	0	0	20	20
29.	**Dk**	**Vwd**	0	0	28	28
30.	**Ah-Psh**	**Ah-Pts**	0	0	127	127
31.	**Rat**	**Rsn**	0	0	284	284
32.	**Gn-C**	**Hgz**	0	32	40	325
33.	**Owd**	**Wv**	0	0	337	337
34.	**Hfd**	**Kawij-Aw**	0	0	705	705

Table B.1 continued from previous page

	MC1	MC2	Highway	Bridge/Tunnel	Distance	Scaled Distance
35.	**Asn**	**Gn-Z**	0	0	854	854
36.	**Anp**	**Lw**	0	0	901	901
37.	**Asd-Dim**	**Wp**	0	0	911	911
38.	**Cl-C**	**Tl**	0	0	1,092	1,092
39.	**Hn**	**Nsw**	0	0	1,439	1,439
40.	**Db**	**Zt-C**	0	0	1,460	1,460
41.	**Tb-Reit**	**Ww**	0	0	1,544	1,544
42.	**Rt-Trb**	**Ztm-Mzt**	0	0	1,723	1,723
43.	**Apg**	**Wf-old**	0	0	1,811	1,811
44.	**Dtn**	**Zh-old**	0	0	1,851	1,851
45.	**Ddt-Zdt**	**Rt-lJsm**	0	0	1,891	1,891
46.	**Skn**	**Ws**	0	0	2,315	2,315
47.	**Hrv**	**Lw-Aln**	0	0	2,398	2,398
48.	**Gv-Twv**	**Nawij**	0	0	2,503	2,503
49.	**Os**	**Veg**	0	0	2,705	2,705
50.	**Es**	**Hgl**	0	0	2,989	2,989
51.	**Mt-Amb**	**Std-Gln**	0	0	3,127	3,127
52.	**Cu**	**Hm-Rpb**	0	0	3,147	3,147
53.	**Gd**	**Kmpn-C**	0	0	3,179	3,179
54.	**Dtc**	**Wtw**	0	0	3,208	3,208
55.	**Gr-Hrwk**	**Sdt-C**	0	32	3,081	3,372
56.	**Ut-W**	**Wd-Ut**	0	0	3,400	3,400
57.	**Dv**	**Zp**	0	75	2,778	3,449
58.	**Aml**	**Odz**	0	0	3,523	3,523
59.	**Ehv-Strij**	**Tb**	0	0	3,818	3,818
60.	**Co**	**Emn-Agsl**	0	0	3,831	3,831
61.	**Mp**	**Stwk**	0	0	3,906	3,906
62.	**Rm**	**Std**	0	13	3,872	3,988
63.	**Bnv**	**Hd**	0	0	4,020	4,020
64.	**Rsd**	**Sb-old**	0	0	4,302	4,302
65.	**Drn**	**Zv**	0	0	4,355	4,355
66.	**Asd-N**	**Pm**	0	36	4,051	4,379
67.	**Lc**	**Nd-old**	0	42	4,108	4,486
68.	**Hedr-C**	**Sgn**	0	32	4,350	4,637
69.	**Dne**	**Wt**	0	44	4,513	4,912
70.	**Ddv**	**Hgv**	0	141	3,789	5,061
71.	**Gs**	**Mdb**	0	0	5,298	5,298
72.	**Bak**	**Eo**	0	41	5,078	5,446
73.	**Ddt-C**	**Obl**	0	0	5,592	5,592
74.	**Fn**	**Sk**	0	0	5,706	5,706
75.	**Bv**	**Zd-Odh**	0	28	5,629	5,880
76.	**Ehz**	**Mdmr**	0	37	6,141	6,473
77.	**Oth**	**Zvb**	0	0	7,073	7,073
78.	**Mdh**	**Zr**	0	105	8,482	9,429
79.	**Eb**	**Lls-N**	0	254	11,235	13,520
80.	**Boz**	**Tnz**	0	934	41,009	49,412
Total			**0**	**1,846**	**205,662**	**222,275**