Universidade de Aveiro 2020

Departamento de Eletrónica,
Telecomunicações e Informática

Pedro José Jorge Barata Nunes

Métodos de Melhoria da Disponibilidade e da Resiliência a Desastres em Redes de Telecomunicações

Upgrade Methods for Improving Availability and Disaster Resilience in Telecommunication Networks

Universidade de Aveiro
2020

Departamento de Eletrónica, Telecomunicações e Informática

Pedro José Jorge Barata Nunes

Métodos de Melhoria da Disponibilidade e da Resiliência a Desastres em Redes de Telecomunicações

Upgrade Methods for Improving Availability and Disaster Resilience of Telecommunication Networks

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica do Doutor Amaro Fernandes de Sousa, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Apoio financeiro da FCT no âmbito do projeto CENTRO-01-0145-FEDER-029312 ResNeD.

o júri

presidente

Prof. Doutor Luís Filipe de Seabra Lopes
professor associado da Universidade de Aveiro

Prof. Doutor Lúcia Maria Reis Albuquerque Martins
professora auxiliar da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Prof. Doutor Amaro Fernandes de Sousa
professor auxiliar da Universidade de Aveiro

agradecimentos

Agradeço a ambos os meus pais, à minha avó, e à minha madrinha, por terem sempre ajudado com os estudos e pelo seu apoio motivacional e financial, durante o meu percurso académico.

Agradeço ao meu orientador, o Professor Amaro de Sousa, por me ter apresentado este projeto, e pelo apoio e disponibilidade prestada ao longo desta tese.

Agradeço ao Instituto de Telecomunicações, polo de Aveiro, pelas condições proporcionadas para a realização desta dissertação que foi financiada pela Fundação para a Ciência e a Tecnologia (FCT), no âmbito do projeto CENTRO-01-0145-FEDER-029312 ResNeD.

palavras-chave

disponibilidade, encaminhamento com geodiversidade, resiliencia a desastres.

resumo

As redes de telecomunicações são um dos componentes essenciais na atual sociedade, no qual vários serviços dependem da sua funcionalidade para operarem eficientemente. O suporte de serviços críticos exige que as redes ofereçam altos níveis de disponibilidade entre os seus nós e sejam altamente resilientes a desastres de larga escala, tais como os provocados por fenómenos naturais (tremores de terra, tsunamis, etc.). Algumas técnicas podem ser implementadas para atingir estes objetivos. Nesta dissertação, considera-se o uso de encaminhamento com geodiversidade para reduzir o impacto de desastres de larga escala, com a desvantagem de exigir percursos de encaminhamento mais longos, reduzindo a disponibilidade resultante entre os nós origem-destino do encaminhamento. Assim, para obter simultaneamente alta disponibilidade e alta resiliência a desastres, é necessário melhorar a disponibilidade em alguns elementos da rede. Nesta dissertação são introduzidas diferentes estratégias para identificar eficazmente os elementos da rede que precisam de ser melhorados em termos de disponibilidade, para que a rede suporte os requisitos de disponibilidade e resiliência a desastres requeridos por serviços críticos.

keywords availability, geodiversity routing, disaster resilience.

abstract

In current societies, telecommunication networks are one of its essential components, in which different services depend on. Critical service requires these networks to provide high levels of availability between their nodes and high levels of resilient to large-scale natural disasters, either by avoiding them or quickly recover from them. Different techniques can be used to reach these goals. In this dissertation, it is considered the use of geodiversity routing to reduce the impact of large-scale disasters, with the downside of utilizing longer paths which, in turn, reduces the resulting end-to-end availability. This downside can be corrected if the availability of some network elements are upgraded so that the availability required by critical services is met, while maintaining the geodiversity required to prevent the impact of disasters. In this dissertation, different upgrade strategies are implemented to efficiently identify the network elements required to be upgraded, so that the network can provide critical services with high availability and high resilience to natural disasters.

Index

Figure Index

Table Index

1. Introduction

1.1 Motivation

Telecommunication networks are a key component in the current society's lifestyle, supporting a great range of services, from trivial ones to some that are critical for the safety and well-being of the society. Failures, in the latter case, can create severe consequences [1], due to, as an example, the unavailability to contact emergency services, putting lives at risk. So, it is required that the networks in which these services are dependent upon, be constantly maintained [2]. In addition, due to possible large-scale disasters, as natural disasters, these services can be severely degraded, requiring the usage of techniques that prevent or reduce the impact of such events. This dissertation addresses these two scenarios by focusing on the geodiversity routing through the network and on upgrading the necessary network components to ensure the availability required by critical services.

Geodiversity routing in a network takes into consideration the geographical diversity of the network topology when making routing decisions, using two different routing paths between each end-to-end node pair, with one being the backup for the second. The usage of two routing paths increases the availability of the network [3] since when one of the routing paths suffers a failure, the second one can still be available to support the services. To reduce the impact of disasters, these routing paths must be geographically apart so that if a disaster occurs, the probability of hitting both paths is reduced, allowing a higher disaster resilience [4].

Considering a network topology, in which the geographical distance between any two network components is known, and considering that the routing paths, between two end-to-end nodes, is geographically apart following a minimum distance *D*, then as long the diameter of a disaster is not higher than *D*, the network continues to be available since only one of the routing paths can be affected by the disaster. Note that the higher the value of the distance *D* is, the more resilient the network becomes, reducing the probability of a disaster affecting both routing paths simultaneously. Due to the geographical proximity of network elements (nodes and links), a desired distance of *D* cannot be always guaranteed. So, for each pair of network nodes, it is required to compute the maximum distance that is possible for any pair of geodiverse routing paths and if this maximum distance is lower than the desired value of *D*, then the maximum distance is considered instead.

The downside of implementing geodiverse routing is the utilization of longer paths between end-to-end nodes, possibly not allowing the availability required by critical services. In order to reach both the desired availability and disaster resilience, it is necessary to select key components in the network to be upgraded.

This can be done by analysing the routes taken between the end-to-end nodes and selecting the network components required to be improved to ensure that the resulting availability reaches the expected value for critical services [5].

In this dissertation, the network components can be improved in 2 different ways. In the first way, the availability of each link can be individually improved with a given cost, making it more resilient to exterior factors. In the second way, the Mean Time to Repair (MTTR) of the links is reduced by placing Maintenance Teams (MTs) on network nodes with a given cost. In both ways, the aim is to compute a minimum cost solution such that the availability improvement of the network links lets the final configuration of the network to support critical services with some desired values of end-to-end availability and disaster resilience.

1.2 Objectives

The main objective of this dissertation is to develop efficient heuristics able to compute the network components to be upgraded so that all node pairs of a given network can provide some desired values of end-to-end availability and disaster resilience.

In the case when the availability of links can be individually upgraded, the objective is to obtain methods to select the links to be upgraded. Since this problem has been recently addressed in Ref. [6], the aim is to reach algorithms which are more efficient than the ones proposed in that work and, for this purpose, the developed algorithms will be tested on the same problem instances.

In the case when the availability of links is improved by placing maintenance teams on nodes, the objective is to obtain methods to select the nodes where to place the maintenance teams. Since this problem shares many common characteristics with the previous problem, the aim is to apply the developed algorithms also to this case and to test them in the same problem instances.

1.3 Organization

This dissertation is organized in six chapters, with the current chapter introducing the objectives of this work.

Chapter 2 describes the background of this work with a description of the state of the art on the addressed subjects and an introduction of the components used for solving the addressed problems.

Chapter 3 details the problems under investigation illustrating them with some examples.

Chapter 4 describes the implemented heuristic strategies and details each developed method variant.

Chapter 5 presents the computational results obtained by the different methods introduced in the previous chapter and discusses their efficiency.

Finally, the conclusions of the work are presented in Chapter 6 together with some topics for further research.

2. Background

Several services in the current society's lifestyle are severely dependent on telecommunication networks, varying from basic social communication services, to critical services, such as emergency or smart grid communications. Due to their importance, they must be supported by telecommunication networks with high availability, which has been addressed in several research works.

Ref. [7] proposes a framework that allows a higher efficiency of resources management, as to increase the availability of the services, while Ref. [8] considers multilayer routing strategies as to provide routes with the highest availability between nodes. Ref. [9] and Ref. [10] introduce a strategy, termed spine, to provide end-to-end availability by embedding a high availability set of links and nodes at the physical layer.

Since large-scale disasters have become increasingly more frequent in time and wider in geographical coverage, it is deemed necessary the existence of methods that can handle the protection of the network and be able to minimize the impact of those events, making the network more resilient to them [2].

2.1 Disaster Resilient Methods

Critical services have an increased importance upon the occurrence of disasters, such as the example of emergency services in rescuing operations. This indicates that the networks are also required to have a high level of resilience to large scale disasters. Different techniques have been proposed to prepare the networks against the impact of these disasters [11].

A method introduced in Ref. [12] proposes the usage of backup routes to handle broken paths, to improve the restoration of the services. In Ref. [13], topology models are discussed to prevent all links that share a common resource to be affected by the failure of an individual one. The issue with these methods is that they address the cases where the failure happens on a single network element and do not handle the cases of large-scale disasters that simultaneously affect different network elements.

To handle this issue, Ref. [3] introduces the concept of geodiversity in path routing planning, proposing algorithms to calculate multiple paths that are geographically apart, avoiding regions with an higher probability of disasters occurrence. This would generate a lower correlation between the components of the network, allowing for a more robust flow between the end-to-end nodes. This concept was adapted to telecommunication networks in Ref. [14]. Works, such as Ref. [15] and Ref. [4], implemented and tested it on optical networks, introducing heuristics to calculate a geodiverse routing protocol and presenting improved models for network planning that can increase the resilience of a network.

Geodiversity in a network allows to select a pair of paths between each node pair, with enough geographical distance between themselves, so that in case of a disaster, it would not affect simultaneously both routing paths. Similarly, papers such as Ref. [16], developed heuristic algorithms to find the critical regions, as to plan the network to avoid those areas.

Geodiversity does come with its own downside. Due to the need to ensure that the routing is geodiverse enough to avoid large-scale disasters, the selected paths are longer, on average, reducing the resulting availability between the end-to-end node pairs. So, in general, the resulting end-to-end availability can become lower than the availability required by critical services. Ref. [17] and Ref. [6] address this issue by computing the set of links whose availability must be improved so that the network can provide geodiverse routing between each end-to-end pair of nodes and a desired end-to-end availability required by the critical services. Key links are selected for upgrade, assuming that the availability of the links, before and after the upgrade, as their upgrading cost are known.

In Ref. [6], which is very recent, the computation time required to run the proposed methods is still very high. For example, for the Germany50 network, to reach the availability of 0.99999, with a geodiversity of 160 Km between each pair of geodiverse routes, the method that has provided the lowest cost required approximately four hours to calculate the links to be upgraded. In this dissertation, one aim is to provide alternative methods that can provide better cost solutions with lower running times.

Alternatively, the resilience of a network can be improved by the quick repair of link failures. The monitoring and localization of failures in optical networks has been addressed by several works in the past, one of the oldest ones being Ref. [18]. Meanwhile, other works introduced schemes to reduce the hardware cost and the number of redundant alarms [19], or to improve the precision of the failure location [20]. Other works on this issue presented solutions to allow the location of multiple links failures in the network, as in Ref. [21] and Ref. [22].

Using these techniques improves partially the availability of the network, as it considers the reduction of the downtime of a link. The total downtime, though, is the sum of the failure localization (including its detection) time with the failure repair time. The latter time is not only the time to repair the failure but must also include the travelling time of a maintenance team from its base location to the location of the failure, an issue addressed in Ref. [23]. In this dissertation, one aim is to develop upgrade methods considering the addition of maintenance teams in nodes of the network that can quickly repair failure occurrences on the links connected to the nodes, improving in this way the availability of the links by reducing their mean time to repair.

2.2 Networks

Similar to the two previous papers [6][17], this dissertation will use, for testing purposes, the Germany50 and Coronet networks. Germany50 network [24], presented in Fig. 1, has 50 nodes and 88 links. Each node represents a location in Germany, Europe. Coronet network [25], shown in Fig. 2, is located in United States of America and has 75 nodes and 99 links. In both networks, the geographical coordinates of the nodes are publicly available. By assuming that the links follow the shortest path over the Earth surface, it is possible to determine the distances between components of the networks, i.e. links and nodes.

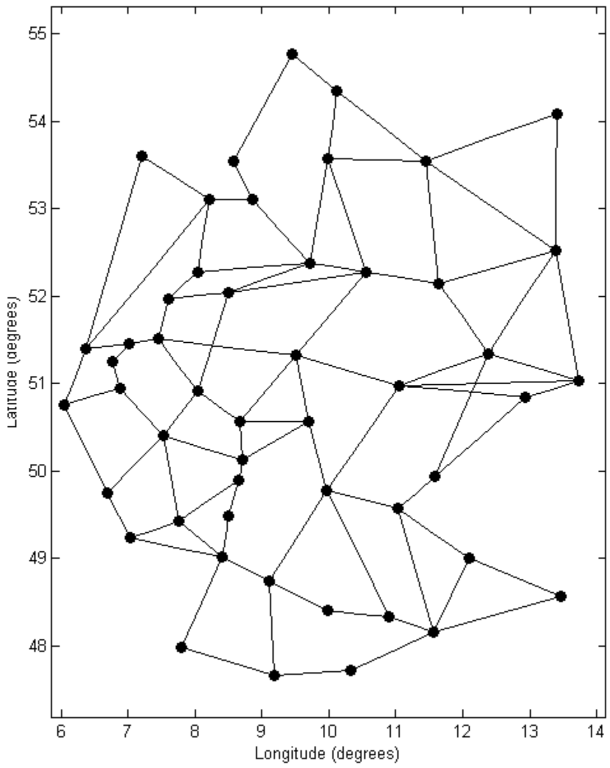

Figure 1 – Germany50 network [24]

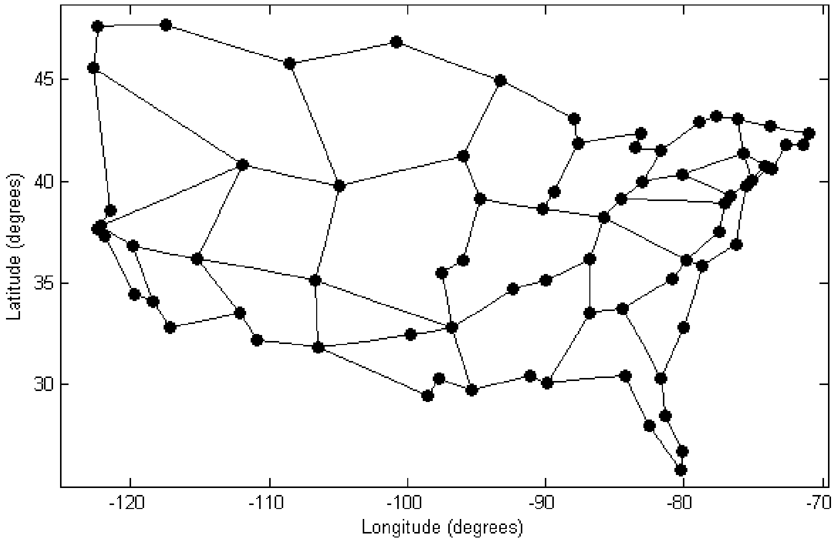

Figure 2 – Coronet network [25]

Table 1 shows some details of each network, including the number of nodes and links, the total number of node pairs, and the maximum and average length of the links (in Km). Note that the maximum geodiversity value that can be provided to each pair of nodes can be obtained by the optimal solution of the Maximum Distance *D* of Geodiverse Path optimization problem, as proposed in Ref. [26]. The value of DMax shown in the last column of the table is the highest of the maximum geodiversity values among all node pairs of each network.

Network	**#Nodes**	**#Links**	**#Node Pairs**	**Max Length**	**Ave. Length**	**DMax**
Germany50	50	88	1225	252	100.67	166
Coronet	75	99	2775	1017	329.72	707

Table 1 – Network Details

For different values of availability *A* and geodiversity *D*, the following tables, Table 2, and Table 3, show the number of node pairs of each network that cannot reach simultaneously the availability value *A* and the geodiversity value *D* without any upgrade of the network.

Availability *A*	**Geodiversity *D* (in Km)**				
	0	**40**	**80**	**120**	**160**
0.99995	0	0	0	0	0
0.99998	75	85	227	257	261
0.99999	393	446	665	700	704

Table 2 – Amount of node pairs requiring upgrades, Germany50

Availability *A*	**Geodiversity *D* (in Km)**				
	0	**100**	**200**	**300**	**400**
0.9999	2022	2061	2149	2181	2184
0.99995	2379	2422	2475	2486	2487
0.99999	2724	2734	2737	2737	2737

Table 3 – Amount of node pairs requiring upgrades, Coronet

As expected, these tables show that the number of node pairs is higher both for higher values of required availability and higher values of required geodiversity. Moreover, it can be seen in Table 2 that all node pairs can reach the availability of 0.99995 for any required geodiversity value. So, when testing the algorithms in Germany50 network (in Chapter 5), the availability values of 0.99998 and 0.99999 will be considered.

3. Problem Description

3.1 General description

Consider a geographical network defined by a set of nodes and a set of links. Each link has an associated availability, before and after being upgraded. The geographical distance between all pairs of network components, either links or nodes, is known. The aim is to compute a network upgrade configuration so that the network is able to provide a pair of disjoint routing paths between each pair of network nodes, with a minimum availability *A* between the nodes and a minimum geodiversity *D* between the paths.

For illustration purposes, consider the network example in Fig. 3 with 6 nodes and 9 links. Assume in this example that a critical service is running only between the source node *s* and the target node *t* (the other nodes are labelled by the numbers 1 to 4). The lines between the nodes represent the geographical routes of the connecting links, with their individual availability values indicated in the figure. In this example, assume that between the node *s* and node *t*, the network must provide the critical service with an availability value of *A* = 0.9999, while ensuring that there is a minimal geodiversity between the two routing paths of *D* = 120Km.

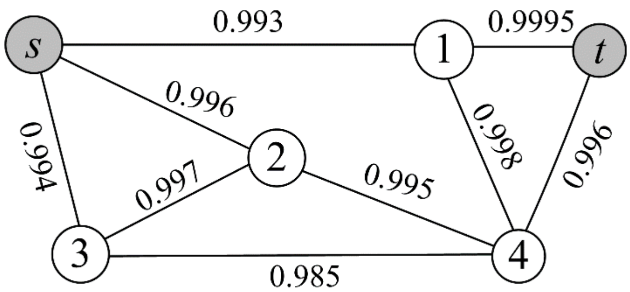

Figure 3 – Network Example

The availability of 2 disjoint paths is given by:

$$\Delta = 1 - \left(1 - \prod_{(i,j)\,\in\,P_1} a_{i,j}\right) \times \left(1 - \prod_{(i,j)\,\in\,P_2} a_{i,j}\right) \qquad (1)$$

where sets P_1 and P_2 are the sets of links of each routing path, a link identified by (i,j) represents the link between node i and node j, and $a_{i,j}$ represents the individual availability of link (i,j).

Fig. 4 and Fig. 5 shows the two possible pairs of disjoint paths between the nodes *s* and *t* highlighting in red the minimum geographical distance between the paths of each pair. In Fig. 4, a disaster with a geographical coverage of less than 70 Km affecting any intermediate element of the bottom path – node 2, link (2,4) or node 4 – cannot affect the upper path. Obviously, any disaster covering either one of the end-nodes or all links of one of them disrupts both paths and these cases cannot be protected. In Fig. 5, the minimum geographical distance becomes 130 Km, and this is the maximum value of *D* that can be provided to node pair *s* – *t* in this network example.

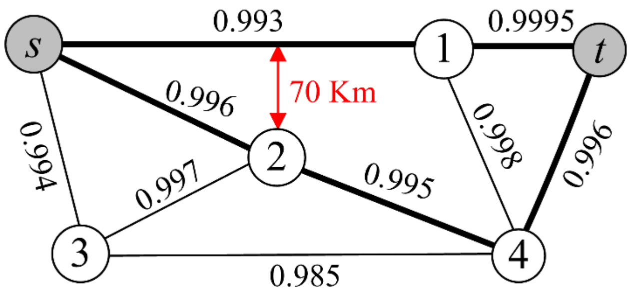

Figure 4 – Network Example Path 1

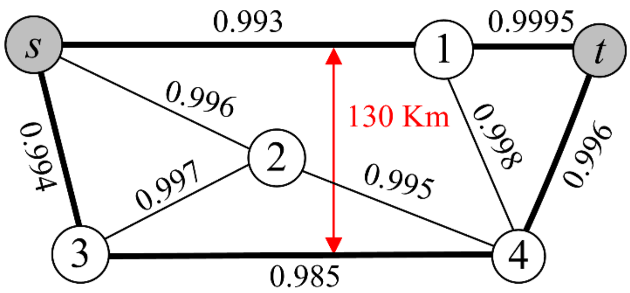

Figure 5 – Network Example Path 2

In Fig. 4, the paths reach an availability of 0.999903 which is higher than the required availability *A* (satisfying one of the conditions), but, as can be seen in the figure, the minimum geographical distance between the routes is of 70 Km which is lower than *D*. In Fig. 5, on the other hand, the geodiversity requirement *D* is fulfilled but due to the longer bottom path, the availability of the paths is 0.999814, which is lower than the required availability *A*.

To ensure that both the availability and the geodiversity are reached, the availability of some components in the network must be upgraded, as to increase the availability in key areas. In this dissertation, this issue is addressed in two scenarios. One scenario is to upgrade the links between the nodes, as to increase their individual availability. The second scenario is to upgrade nodes, by adding a dedicated maintenance team to each node, which increases the availability of the connecting links.

3.2 Geodiversity

To ensure the geodiversity in the network, it is required to set the rules to calculate it. Following Ref. [26], the geodiversity of a pair of routing paths is defined as the minimum distance between any intermediate element of one path and any element of the other path (elements are nodes and links). The intermediate elements of a path are all its elements excluding the end nodes and the links connected to the end nodes.

The methods used to calculate the distances between elements assume that the geographical path of links and the geographical distance between points is over a sphere, as to take into consideration the curvature of planet Earth. The algorithms implemented were based on the algorithms introduced in Ref. [27], which presents several formulas to calculate the distance between objects in a spherical Earth, through their geographical coordinates.

Ref. [26] also shows that only links need to be considered to compute the minimum geographical distance between two paths. The formula used to calculate the geographical distance between any two paths is:

$$D = \min_{c_i \in S_1, c_j \in S_2} \delta\{c_i, c_j\}$$

where S_1 and S_2 are the set of links of each path and $\delta\{c_i, c_j\}$ represents the geographical distance between link c_i and link c_j. In accordance to [26], the geographical distance between 2 links is defined as follows:

- If the links do not share neither the starting node nor the ending node of the paths, it is the minimum distance between any point of one link and any point of the other link. Therefore, in the case that two links intersect each other or share a common node, the geographical distance between them is zero.

- If the two links share the starting node or the ending node, it is the minimum distance between the distances of the non-common node of one link and the other link.

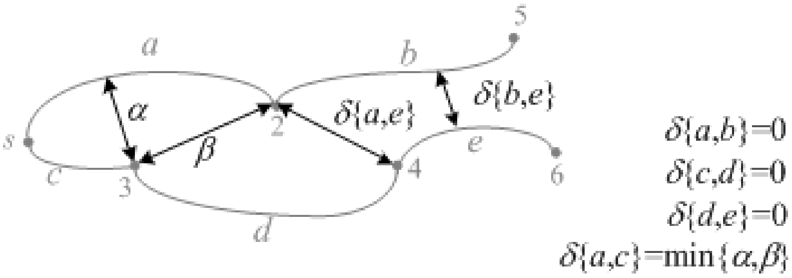

Figure 6 – Geodiversity calculation [17]

Fig. 6, taken from Ref. [17], illustrates the definition of the geographical distances between links. It shows a partial network with a starting node *s*, some intermediate nodes identified from 2 to 6 and the links identified with the letters from *a* to *e*.

As can be seen in Fig. 6, since link *a* and link *b* share node 2, which is neither the starting nor target node, the distance between these two links is $\delta\{a, b\} = 0$. Another example is the distance between link *a* and link *e*, which, as can be seen in the figure, is the distance between node 2 and node 4, their closest points. Finally, the distance between link *a* and link *c*, since they share the starting node *s*, is determined by the minimum between the distance *α* (the distance between node 3 and the closest point of link *a*) and the distance *β* (the distance between node 2 and the closest point of link *c*).

With the above as a basis, it is possible to generate a pair of disjoint paths for each end-to-end node pair and ensure that a geographical distance *D* is kept between both paths. The choice of the geodiversity value *D* to be supported by the network is up to the operator of the network.

There are both advantages and downsides to consider either low or high geodiversity values *D*. If the geodiversity of the network is low, its resilience to natural disasters is reduced as more network components (nodes or links) can be hit by large-scale disasters, increasing the time that the network will be down. On the other hand, having a high value *D* of geodiversity, while allowing for a higher resilience to natural disasters, is more expensive to guarantee as the paths are longer and require more components to be upgraded, to ensure that the availability matches what is expected for critical services.

Depending on the required geodiversity *D*, it is possible that due to network geographical topology, there are no possible pairs of paths that have the expected minimum distance *D* between at least some network nodes. In Ref. [26], a Mixed Integer Linear Programming model is proposed whose optimal solution allows to calculate the Maximum Distance *D* of Geodiverse Paths that can be guaranteed for each pair of network nodes. For the node pairs such that the expected geodiversity

D is higher than the maximum value that can be provided, the maximum value is used instead.

3.3 Availability

The availability of 2 disjoint paths is calculated by formula (1), as introduced in Section 3.1. The availability of a link is a value between 0 and 1 which measures the portion of time that the link is working, on average. Following Ref. [28], the availability of a link can be obtained from its length, according to the following formula:

$$A = 1 - \frac{MTTR}{MTBF} \tag{2}$$

The variables identified represent Mean Time To Repair (*MTTR*), which is set as 24h, as defined in Ref. [9], and Mean Time Between Failures (*MTBF*), which can be calculated from the formula:

$$MTBF = \frac{CC \times 365 \times 24}{l}$$

In the shown formula, *CC* represents the Cable Cut metric, standardized at 450 Km [9], and l, representing the length of the link (in Km). From the above formulas, it is noticeable that the longer the length of the link, the lesser its availability is.

With the calculation of the base availability of each link, it is possible to obtain the most available pair of node disjoint paths between a given pair of nodes [17], while ensuring that the geodiversity between them is not lower than what is expected (or the maximum possible value, whichever is lower). As shown previously, a pair of paths does not guarantee that the expected availability is reached, requiring some network components to be upgraded.

In this dissertation, two cases are considered. In the first case, links can be individually upgraded (as in Ref. [17] and Ref. [6]). In the second case, nodes can be upgraded with maintenance teams which increases the availability of the connecting links. In the next subsection, each case is described individually.

3.3.1 Upgrading Links

In the network, since the availability of a link represents its resilience to outside factors and degradation, it is assumed that to upgrade a link is to add an equivalent parallel link of the same length, to reduce the chance that the corresponding connection is broken (this is also the approach in Ref. [17] and Ref. [6]). In this case, when a link with availability *a* is upgraded, its upgraded availability a^u can be calculated by the following formula:

$$a^u = a * (2 - a)$$

Furthermore, since the cost of upgrading a link cannot be completely known, as it depends on many factors that can be different between each country, it is considered that the cost is equivalent to the links length.

For illustrative purposes, let us consider again the example presented in Fig. 4 (in section 3.1) where none of the two disjoint paths (shown in Fig. 5) can reach simultaneously an availability *A* of 0.9999 and geodiversity *D* of 120 Km. With the Link Upgrade (LU) of the link connecting node *s* and node 1 to the availability value of 0.9998 (illustrated in Fig. 7), now, there is one pair of routing paths (shown in Fig. 8) whose availability is 0.99998 (higher than *A*) and whose geodiversity is130 Km (also higher than *D*).

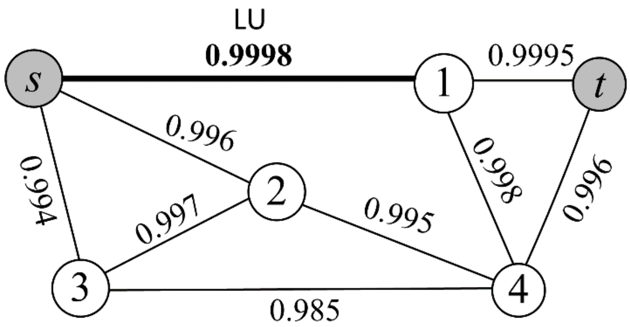

Figure 7 – Network Example, Link Upgrade

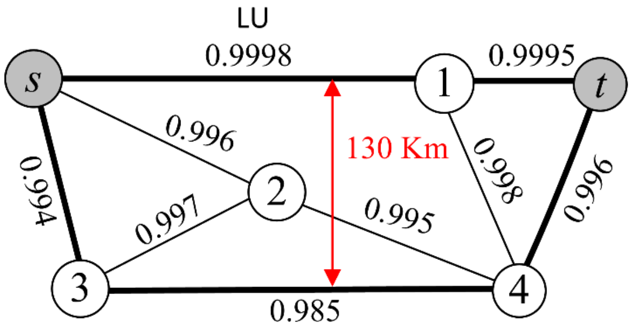

Figure 8 – Network Example, Link Upgrade Result

3.3.2 Upgrading Nodes

The second strategy is the upgrade of nodes by adding a dedicated maintenance team, with the focus of reducing the Mean Time to Repair in the availability of its links. According to the formula (2) presented in section 3.3, by reducing the value of MTTR, the availability of a link would be increased, possibly reaching the expected values.

A team placed in a node of the network is able to quickly move to a broken point in a link and perform the required repair procedures. Due to the geographical coverage of each network, a case by case analysis is required as to ensure that the average moving speed of the maintenance team makes possible to reach both the expected availability and geodiversity values. Also, the Mean Time to Repair of each link depends on how many of its end nodes have dedicated teams. If both nodes are upgraded (i.e., are assigned with a maintenance team each), each team only needs to travel at most half of the length of the link (assuming that the broken point is repaired by the closest maintenance team).

With the above assumptions, the following formula is used to calculate the value of the Mean Time to Repair of each link when at least one of its end nodes is assigned with one maintenance team:

$$MTTR^u = \frac{l}{2 \times V \times N_u} + T_r$$

The variables represent the length of the link l, the average moving speed of the maintenance teams V, the number of end nodes with dedicated teams N_u (which can be either 1 or 2) and the time to repair the broken link T_r, set as 1 hour.

For the networks used for testing the solving methods, the average moving speed for the Germany50 network is considered as 40 Km/h (assuming the teams travel by terrestrial means, as vans), and for the Coronet Network is considered as 180 Km/h (in this case, the geographical extension of the network is very high and it was assumed that teams travel by aerial means, e.g., helicopters, as in many cases, traveling by terrestrial means does not reduce significantly the overall MTTR values).

Again for illustrative purposes, let us consider the example presented in Fig. 4 (in section 3.1) where none of the two disjoint paths (shown in Fig. 5) can meet simultaneously an availability *A* of 0.9999 and geodiversity *D* of 120 Km. Consider the following 2 examples.

In the first example, a dedicated maintenance team (MT) is added to node 1, improving the availability of node 1 connecting links, as highlighted in bold in Fig. 9. With this solution, in Fig. 10, between the nodes *s* and *t*, there is now a pair of disjoint paths, with the required geodiversity and an availability of, approximately, 0.99992, higher than what is required, solving the problem.

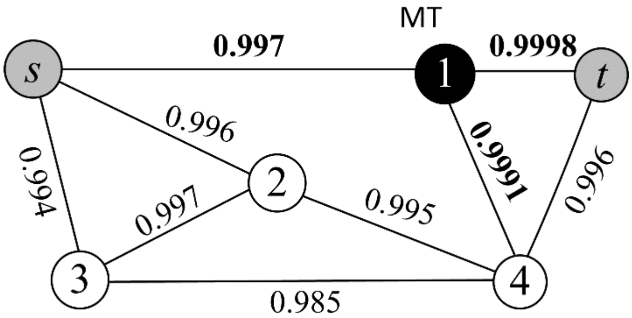

Figure 9 – Network Example, Node Upgrade 1

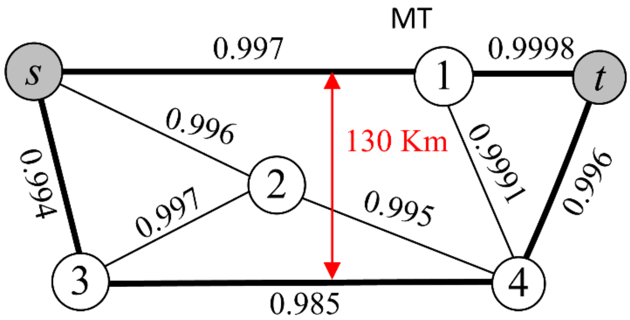

Figure 10 – Network Example, Node Upgrade 1 Result

A second example to solve the problem is illustrated in Fig. 11. First, node 3 is selected to be added a maintenance team, improving the availability of the node connecting links (highlighted in bold in the left figure). Nevertheless, since the availability of the disjoint paths would only reach, approximately, 0.99988, it is not enough to reach the availability requirements of the network. Then, choosing to add another maintenance team on node 4, notice that the availability of the link between nodes 3 and 4 is further improved due to having two end nodes with maintenance teams. With these two node upgrades, as can be seen in Fig. 12, the availability between the starting and target nodes is of, approximately, 0.99992, and the geodiversity is of 130 Km, reaching the availability and geodiversity requirements of the network.

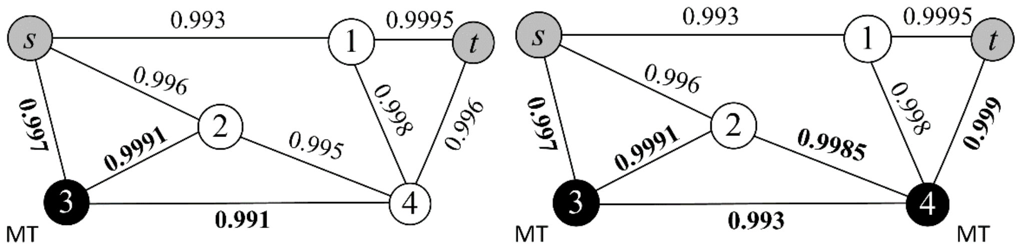

Figure 11 – Network Example, Node Upgrade 2

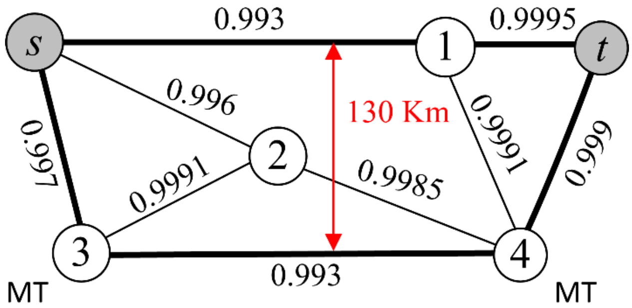

Figure 12 – Network Example, Node Upgrade 2 Result

While the second example did choose more nodes to be upgraded than the first example, it is not necessarily the worst solution, as the addition of maintenance teams in nodes 3 and 4 can have a lower cost than the addition of a single maintenance team in node 1.

The cost of placing maintenance team on nodes is very much dependent on each particular case (network operator, country, etc.) and no realistic values are available. So, instead of generating random cost values, we have assigned cost values in a way that can be easily known by readers interested in replicating this work. The node ID integer value (starting from 1 up to the number of nodes of the network) was used to artificially generate cost values using the following formula:

$$c_i = mod(i, v) + c_b$$

The base cost, represented as c_b, was set as 1 and the v was set as 4. In the formula, $mod(i, v)$ represents the remaining of the integer division of the node ID i by v, indicating that the cost for each node is an integer value between 1 and 4. By this formula, node 1 has a cost of 2, node 2 has a cost of 3, node 3 has a cost of 4, node 4 has a cost of 1, etc.

In Fig. 13, the Germany50 network is shown where each node is identified by its ID value and colour coded depending on its cost (white indicates a cost equal to 1, yellow indicates a cost equal to 2, orange indicates a cost equal to 3 and red indicates a cost equal to 4). The same colour code is repeated in Fig. 14 showing the Coronet network.

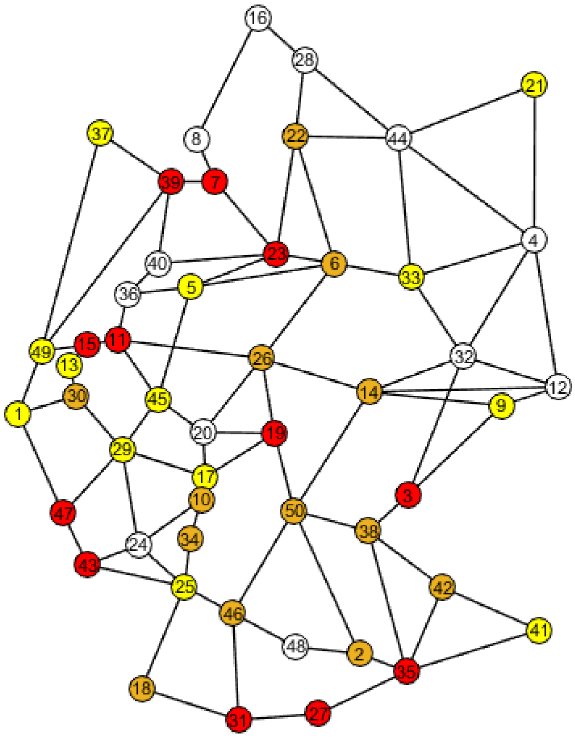

Figure 13 – Nodes and their costs in Germany50 network

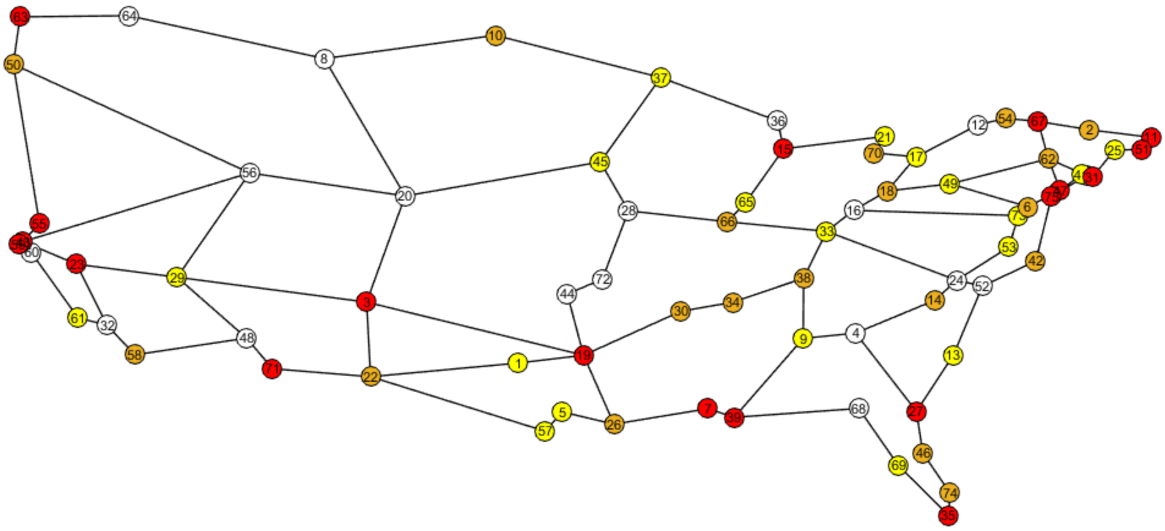

Figure 14 – Nodes and their costs in Coronet network

4. Solving Algorithms

4.1 High Level Description

In general, a set of components (links in the Link Selection problem or nodes in the Node selection problem) needs to be selected (to be upgraded) to increase the value of the availability in key areas of the network, as to be able to reach a required geodiversity *D* and availability *A* in the network for all pairs of network nodes. As such, the problem stands in calculating the set of components with the lowest upgrading cost, as efficiently as possible.

To solve the problem of obtaining the required geodiversity and availability in the network for all pairs of network nodes, several algorithms were implemented and tested. These algorithms are heuristic approaches based on greedy strategies to obtain an upgrade solution with a last step to remove the selected components that are redundant at the end of the greedy phase. All algorithms use three basic processing steps:

1. Filtering process: computes a pair of geodiverse paths between each pair of network nodes with a given upgrade configuration, and filter (i.e., select) the node pairs that do not have the expected availability *A*.

2. Component selection process: based on the results of the filtering process, selects one or more components to be upgraded.

3. Removal process: based on an upgrade solution, removes the redundant upgraded components.

Depending on the implemented method, these steps can be implemented in different ways and the overall methods can loop the two first steps in different ways.

The filtering process is implemented running the Guaranteed Available Pair of Geodiverse Paths (GAPGP) algorithm for all pairs of nodes of interest, as introduced in Ref. [17]. For each pair of nodes in the network, the GAPGP algorithm computes the pair of geodiverse paths (i.e., that ensures the required geodiversity *D*) with the highest availability in the network when its availability is below the required availability *A*, or the first pair of geodiverse paths (not necessarily the one with the highest availability) that is found providing the required geodiversity *A*.

The component selection process selects some (still not upgraded) components based on the filtering process results. The component selection process considers different variants in the number of components to be selected and the selection criterion. Depending on the method, the filtering process can be run again or not: when the filtering process is run again, the method loops the two processes (the filtering process and the component selection process) until the selected components are enough to ensure the required availability *A* and geodiversity *D* to all node pairs of the network.

Different algorithm variants were implemented and tested, and they can be classified in 3 main methods:

- The Minimum Upgrade Cost with Availability and Geodiversity (MUCAG) method, similar to the proposed in Ref. [17]: the component selection process selects one single component and the filtering process is run again with the result of the upgraded component.

- The MUCAG with Multiple Component Selection: the component selection process selects multiple components until the required availability is provided to the node pair with the worst availability and the filtering process is run again with the result of the upgraded components.

- The MUCAG with Single Filtering Process: the component selection is looped for all node pairs by increasing order of their availability (as provided by the filtering process) and, for each node pair, the component selection selects multiple components until the required availability is provided to the node pair.

Finally, the removal process is run in the final step of each method when the set of selected components already provides the required availability *A* and geodiversity *D* to all node pairs. The removal process removes the selected components that are redundant (i.e., that can be not upgraded and still let the required availability *A* and geodiversity *D* be guaranteed to all node pairs) in order to reduce the cost of the final upgrade solution. This method also has some variants, as the order of processing of the upgraded components list can affect its result.

Next, the filtering process will be detailed in section 4.2. Then, the algorithm variants are detailed separately for the 3 main methods in sections 4.3, 4.4 and 4.5. Finally, the different methods implemented for the removal process are detailed in section 4.6.

4.2 Filtering Process

For a given geographical network, a required availability *A* and geodiversity *D* and a given set of node pairs of the network, the filtering process runs the GAPGP algorithm for each pair of nodes.

The GAPGP algorithm was proposed in Ref. [17] and is used to obtain a pair of paths, in a network, between two nodes, with the highest possible availability while ensuring the required geodiversity *D* is kept between the paths. The calculation of each path is implemented through the usage of a pathfinding method, obtaining thus the path with the highest possible availability.

To obtain the pair of paths, GAPGP follows a looping two-step algorithm. The first step of each loop constantly generates one path per loop between the pair of nodes, in a decreasing value of availability order, guaranteeing that in the first loop the first generated path has the highest availability possible, in the second loop the second generated path has the second highest availability possible, and so on.

In each loop, the first step generates a path and the second step attempts to calculate a second path with maximum availability ensuring the geodiversity *D* is kept between the two paths. This second path is computed in an auxiliary graph where the components of the first path and the components whose distance from any element of the first path is below *D* are eliminated. If such second path exists, the availability of both paths is computed by formula (1) introduced in section 3.1.

The looping steps are stopped upon one of these criteria is met:

- The availability of the two paths reaches the required availability *A* for the node pair, returning then these paths.

- The second path has an availability higher than (or equal to) the availability of the first path, indicating that the availability of the best path pair found so far cannot be further improved, returning it then.

- Either the first step cannot generate any more paths or a limit *K* on the number of loops was reached, returning then the best path pair found so far.

Concerning the second criterion, since the first step generates paths in a decreasing availability order, if the availability of the second path is not lower than the availability of the first path, it means that the second path was already obtained in the first step in a previous loop and, therefore, the same will be repeated from that point onward. So, no better pair of paths can be found subsequently.

The *K* limit considered in the third criterion prevents the algorithm from running an exaggerate amount of time, reducing the final computation time of the solving methods. In these computational results, this value is set as 500, meaning that the generator in the first step can only generate up to five hundred paths. This value was required to be high enough to allow a higher variance of path pairs, possibly finding better paths, but low enough to not significantly increase the computation time.

The pathfinding method implemented for both of the steps is the A* algorithm [29], but, in the case of the first step (i.e., to compute the first path), its stopping criteria was modified to allow several paths be obtained in decreasing order, instead of stopping in the first path found.

To make the filtering process more computational efficient, the following two improvements were implemented.

To reduce computation times, a link elimination is conducted for each node pair before running the GAPGP algorithm. Since the objective is to compute a pair of paths with a given geodiversity *A*, in general, some links connected to the source node (and/or to the target node) can be eliminated: a link connected to the source node whose geographical distance to all other links also connected to the source node is lower than the required geodiversity *A* (see the definition of the geographical distance between links in section 3.2) cannot be in any of the 2 paths (the same argument in the target node). So, such links are eliminated in front before running the GAPGP algorithm. Removing these links significantly reduces the number of available paths and, consequently, reduces the average number of loops run by GAPGP.

Since the GAGPG algorithm is run to multiple pairs of nodes and the different runs are independent of each other, the second improvement is to implement the filtering process with multiprocessing. This allows the highest possible usage of the available computational platform, further reducing the computation time for computing all pairs of nodes [30] as each process handles the calculation of the pair of paths for a different pair of nodes.

After all processes (in a multiprocessing algorithm) finish, a final task is run so that the filtering process returns:

- a list of pairs of nodes that did not reach the expected availability (the result obtained from the GAPGP method had an availability lower than the required

A) ordered increasingly by the availability of the paths obtained for each pair of nodes.

- a counter per component indicating the frequency of each component in the pairs of paths of the outputted pairs of nodes, i.e., the number of pairs of paths that contain the component.

- a list per component of the outputted node pairs whose pair of paths contain each component.

Note that, depending on the case, components are either links in the Link Upgrade problem or nodes in the Node Upgrade problem. The information returned by the filtering process allows to obtain the information required to run the component selection process afterwards in all implemented methods.

4.3 Minimum Upgrade Cost with Availability and Geodiversity (MUCAG)

The Minimum Upgrade Cost with Availability and Geodiversity (MUCAG) algorithm, which was proposed in Ref. [17] for the link upgrade problem, has the purpose of identifying the components that should be upgraded as to ensure the availability of the network that is appropriate for the services deemed critical.

This algorithm uses a looping greedy heuristic, selecting the best component to be upgraded in each loop. Each loop runs two steps:

- In the first step, the filtering process is run considering the already upgraded components.

- In the second step, based on the output result of the filtering process (run in the first step), if the list of node pairs is not empty, the best yet not upgraded component is selected to be upgraded; if the list is empty, stop and return the list of all selected components ordered by their insertion order.

In the first loop, all pairs of nodes are analysed by the filtering process in the first step. Then, in the subsequent loops, the filtering process analyses only the pairs of nodes that were outputted in the previous loop (as the others are already guaranteed that can be provided with the required availability and geodiversity).

In the second step, a single component is selected, by analysing the frequency counter values of the not yet selected components outputted by the filtering process and their costs. Due to the possible combinations between these two factors, resulting in different algorithm efficiencies, 3 algorithm variants were implemented: MinCost-MaxCount, MaxCount-MinCost and MinCostOverCount. Next, each of these 3 variants is further detailed.

4.3.1 MinCost-MaxCount

The MinCost-MaxCount variant focuses mainly on the cost of each component to select the best component, in an attempt of selecting the component with the lowest cost. In case of a tie between different components (i.e., components with the same cost), their frequency counter is then taken into consideration, by selecting the component with the highest frequency counter.

The aim of this variant, by selecting the components of lower cost, is to attempt to reduce the overall cost of the solution, with the downside of, on average, having a higher number of components to be upgraded.

4.3.2 MaxCount-MinCost

The MaxCount-MinCost variant is the opposite of the MinCost-MaxCount and considers primarily the frequency counter of the components, by choosing the component with the highest frequency counter. Then, if different components have the same frequency counter, the variant considers the selection of the component with the lowest cost.

This variant has the goal of trying to maximize the number of pairs of nodes that meet the requirements (by upgrading the most frequent component in each loop, a higher number of pair of nodes can be affected, possibly making them meet the availability requirement). The downside of this variant is that in each loop, there is a possibility of the higher cost components being selected for upgrading.

4.3.3 MinCostOverCount

The MinCostOverCount variant attempts to maximize the amount of node pairs affected, while selecting nodes of a lower cost. This is reached by selecting the component which has the lowest ratio of its cost over its frequency counter value. The objective of this variant is to reduce the downsides of the previous 2 variants.

For notation simplicity, the name of these three variants is shortened by removing the Min and Max prefixes (for example, MinCost-MaxCount is referred to just as Cost-Count). In the case of the CostOverCount, it is shortened to Cost/Count. This notation is added to the name of the method (for example, MUCAG-Cost-Count refers to the MUCAG method with the MinCost-MaxCount variant).

4.4 MUCAG with Multiple Component Selection (MS)

This method is based on the previous MUCAG method but now it allows several components to be selected for upgrade in each loop of the method. Similar to the MUCAG method, each loop has two steps where the first step runs the filtering process in the same way as described to the MUCAG method.

The second step is significantly different. Instead of selecting a single component, it can select multiple components. Also, instead of considering the selection of components among all not yet upgraded ones, it restricts the selection to the components belonging only to the pair of paths of the node pair with the worst availability (outputted by the filtering process). In this step, the algorithm selects iteratively the best component, one at the time. When a component is selected to be upgraded, the availability of the node pair is recalculated. If the node pair has the required availability, the step terminates. Otherwise, a new component is selected.

The aim of this method is to reduce the computation time that occurs in the first step due to the filtering process: by selecting more components on each loop, the number of node pairs meeting the requirements becomes higher, thus reducing the total number of loops and, consequently, the overall computation time.

The node pair with the lowest availability is selected since there is a higher probability of requiring more components to be upgraded (to reach the availability requirements) and, therefore, improving the availability of more subsequent node pairs. Otherwise (i.e., selecting the pair of nodes with highest availability, but still lower than the required value), there is a high probability of only one component being selected, which would make the method to have a computation time similar to the previous MUCAG method, or even worse, due to the required extra runs.

Like in the previous method, in the selection of the best component at each iteration of the second step, the 3 algorithm variants described in the previous section (Cost-Count, Count-Cost and Cost/Count) were implemented and tested. Moreover, since the selection of components focuses on improving the availability of a given pair of paths, it might happen that the best component, under each of the

3 criteria (Cost-Count, Count-Cost and Cost/Count), might be not enough to improve the availability of the path pair to the required value while another component might be able to do so.

Therefore, for each of the 3 criteria, another variant was implemented. When processing each not yet upgraded component (using one of the 3 variants), there is a distinction between the components that, if upgraded, guarantee that the path pair in analysis reaches the required availability and the other components. Then, the new variant grants a higher rank of importance to the components that by being upgraded, guarantee that the path pair in analysis reaches the required availability. Moreover, in the case of several components being able to ensure the required availability, the component is selected accordingly to the basic criterion.

In these variants, a component guaranteeing that a path pair reaches the required availability is always selected over a component that does not guarantee the required availability even if the first is worse than the second under the variant criterion. The aim is to reduce the number of selected components that might become redundant at the end of the algorithm.

With the introduction of this variant, the total number of algorithm variants is 6 as each of the 3 variants considered in the previous method (Cost-Count, Count-Cost and Cost/Count) has now 2 variants (with or without distinction). To distinguish between the different variants, the suffix *ND* (No Distinction) or *WD* (With Distinction) is used to identify the variants. So, the name MS-Cost-Count-WD represents the MUCAG with Multiple Component Selection using the Cost-Count with Component Distinction variant.

4.5 MUCAG with Single Filtering Process (SF)

This method is based on the previous method (the MUCAG with Multiple Component Selection) but now the filtering process is run only once at the beginning of the method. Then, the method runs a loop for each node pair outputted by the filtering process by increasing order of their availability. Each loop runs two steps:

- In the first step, the GAGPG algorithm is run to compute a pair of geodiverse paths for the current node pair considering the already upgraded components; the list of node pairs (and the frequency counter) per component is updated.

- In the second step, if the pair of paths (computed in the first step) does not have the required availability, it selects iteratively the best component, one at the time, among the not yet selected ones belonging to the pair of paths until they reach the required availability.

In the first step, the aim is to update the most available pair of paths that results from the upgrades done in the previous loops. Moreover, since at the end of each loop, the current node pair reaches the required availability, it is removed from the list of node pairs of the components that contain it and all frequency counters of such components are decremented by 1 (recall the output information of the filtering process described in section 4.2).

Concerning the second step, note that if the pair of paths computed in the first step has the required availability, then, the second step does nothing, and the method goes directly to the next loop. At the end (i.e., when all node pairs have been processed) the method return the list of all selected components ordered by their insertion order.

By running the filtering process only once in this method, the aim is to reduce the computation time even further when compared with the previous method (in the first step, the GAGPG algorithm is run instead of the filtering process). In the downside, because the frequency counters information is based on the filtering process run only once at the beginning, its usage in the definition of the best components might be less efficient (in terms of the quality of the final solutions) than the previous methods where this information is updated in the first step of every loop by the filtering process.

Like in the previous method, in the selection of the best component at each iteration of the second step, the 6 algorithm variants described in the previous section (Cost-Count, Count-Cost and Cost/Count, each one with WD or ND) were also implemented and tested. To distinguish between this method and the previous one (the MUCAG with Multiple Selection), the prefix SF (Single Filter) is used to the identification of this method. For example, SF-Cost-Count-ND indicates the MUCAG with Single Filtering Process method with the variant of MinCost-MaxCount while making no distinction between the components.

Besides the initial run of the filtering process (which is implemented using multiprocessing, see section 4.2), the loop part of this method is single process (unlike the previous methods that by using the filtering process in the loop part are multiprocessing in their first steps). Moreover, it is well known that in any greedy algorithm, the order by which the node pairs are considered can affect the obtained solution at the end of the algorithm.

So, in order to take the maximum advantage of the processing power of our computational platform, a multiprocessing version of this method was also implemented and tested. The method is run in parallel for different node pair orderings (each one on a single process) and the best among all solutions is computed at the end.

For this multiprocessing variant, based on the initial order of the node pairs outputted by the filtering process, all permutations of the first k node pairs were considered and all swaps up to a range r were also considered. This means that for the first k node pairs, all possible combinations of their order are used and all orderings that result from swapping one of the first k node pairs with one node pair between the orders $k+1$ and r are also used. So, the total number of orderings is:

$$c = k! + (r - k) \times k, \qquad r > k$$

The different orderings were used to maximize the number of processors of the available computational platform, which has a total of 16 cores. Note that a total of 16 processes is not advisable because the running times can be significantly affected by background operating system processes.

So, k was set to 3 (running all permutations of the first 3 node pairs) and r was set to 5 (running all swaps between each of the first 3 node pairs and the node pairs from the 4th and the 5th), meaning that a total of 12 different node pair orderings were run in parallel, increasing the probability of obtaining a better final solution.

4.6 Removal Process

Since the previous algorithms follow a greedy approach, it is expected that some of the components selected for upgrade during the process may not be necessary to provide the required availability to all pairs of nodes in the network. So, to reduce the final cost of the solution, the removal process is run at the final step to find and remove the redundant components.

Recall that all previous methods start by running the filtering process considering the network without any upgrade component. So, the first outputted list of node pairs (ordered increasingly by the availability of the paths obtained for each pair of nodes) excludes the ones that are provided with the required availability and geodiversity values even if there is no upgraded component. So, the removal process only considers this list of node pairs.

The removal process follows a greedy approach. In each iteration, an upgraded component of the solution is individually analysed. This analysis consists in running the GAPGP algorithm in a loop for each node pair in the above list considering the network configuration without the upgraded component. As soon as the result of the GAPGP does not reach the required availability for one node pair, the loop is stopped, and the component is not removed. Otherwise (i.e., if the GAPGP result always reaches the required availability for all node pairs), the component is removed from the upgrade solution.

The removal process can be used at the end of either one of the previous algorithm variants (as described in sections 4.3, 4.4 and 4.5). In this case, the input to the removal process is a single upgrade solution.

Moreover, the removal process can be also used at the end of a sub-set of the previous variants. In this case, the input is a set of upgrade solutions and the removal process is implemented with multiprocessing: each process analyses each upgrade solution and, at the end, it returns the solution with the lowest cost, and in case of a tie, the solution with the smallest number of upgraded components.

The removal process is independent of the method variant(s) previously run. The removal process analyses the upgraded components of an input upgrade solution for a given order. So, different orders might result in removals of different sets of components. The implemented orders are detailed separately in the next subsections.

4.6.1 Removal by Insertion Order

One of the variants for the removal of components is to have them ordered by the order that they were selected for upgrade (recall that all methods return an upgrade solution as a list of upgraded components ordered by their insertion order).

In this variant, it is considered that the components selected in the first iterations of the selection method have a higher probability of being redundant, since components selected afterwards in the network can possibly guarantee the availability in areas affected by the first selected components. The aim is to increase the number of removed components, since the last components in the order were necessary to provide the availability and geodiversity values to the last node pairs.

4.6.2 Removal by Cost

Another variant for the removal of components is to have them sorted by cost, starting from the component with the highest cost, to the lowest cost, and in case of a tie in cost, using the insertion order. This variant focus is on reducing the final cost of the solution by attempting to remove the highest cost components first.

4.6.3 Removal by Frequency

This variant is only used when more than one upgrade solution is given as input to the removal process. It considers the frequency (i.e., number of times) that each component is included in the different input solutions. The frequency of each component is computed among all input solutions, but each solution is handled individually, with the frequency information given as input to all parallel processes.

This variant was implemented with 2 versions: to consider the order of the components from the lowest frequency to the highest frequency or vice-versa. In both versions, in case of a tie in the frequency value, the cost from the highest to the lowest is used.

The variant of ordering the components from lowest to highest frequency assumes that a higher frequency of a component represents a component which has a higher probability of belonging to good solutions. So, this variant focuses on trying to remove first the components with a lower frequency value (i.e., the components that are in fewer solutions).

The second variant of ordering the components from highest to lowest frequency assumes that if a higher frequency component can be removed, the remaining components of the different upgrade solutions are more different between them. So, the final upgrade solutions will be also more different between them, potentially, reaching a better final solution.

5. Computational Results

To test the efficiency of the different methods described in Chapter 4, computational tests were conducted considering the Germany50 and Coronet networks which were introduced in Chapter 2.

All methods were implemented in Python and the program was run in a server platform with an Intel Xeon CPU E5-2650v2, with 2.60GHz, 64GB of RAM, dual processor, and a total of 16 cores, in which 14 processes were used for multiprocessing in the filtering process of the different algorithm variants (2 processes were left so that the running times are not significantly affected by background operating system processes).

Each method variant was tested in both mentioned networks, with different required values of availability A and geodiversity D. Due to the different geographical scales and average link lengths, the availability and geodiversity required values considered for each network are different.

The Germany50 network has the geographical coverage of the Germany country and can provide an availability of 0.99995 between all its node pairs even for the maximum geodiversity values that can be provided by the network. So, two higher values of required availability A were considered: an availability of 0.99998 and 0.99999. The values of geodiversity considered for the Germany50 network are values from 0 Km (representing no required geodiversity between the two node disjoint routing paths for each node pair) up to the value of 160 Km (a value close to the maximum possible geodiversity of this network) in multiples of 40 Km.

For the Coronet network, which is defined over the USA country, since it has a much wider geographical coverage, several pairs of nodes cannot reach the availability of 0.9999, even without imposing any geodiversity requirement between the pair of routing paths. So, the values of 0.9999, 0.99995 and 0.99999 were considered. For its geodiversity values, the maximum value used for testing was of 400 Km, with the remaining values varying in decrements of 100, down to 0 Km.

In the remaining of this chapter, the results obtained by the different algorithms are presented and discussed is 2 separated sections: section 5.1 is dedicated to the Node Upgrade problem, while section 5.2 is dedicated to the Link Upgrade problem.

5.1 Node Upgrade Results

The node upgrade problem considers that the availability of links is improved by placing maintenance teams on nodes and the objective is to select the nodes where to place the maintenance teams so that the cost is minimized and the availability and geodiversity values are guaranteed for all node pairs of the network. Next, the results of each method are presented in separated subsections.

5.1.1 MUCAG

The first set of results aims to compare the efficiency of the 3 variants of the MUCAG method (Cost-Count, Count-Cost and Cost/Count). In all cases, the removal process run at the end of the algorithm is based on the components ordered in decreasing order of their cost.

In the following Table 4 and Table 5, **C** is the cost of the final solution, **#N** is the number of upgraded nodes and **T** is the total runtime of the algorithm, in seconds. The solutions with the lowest cost, for each set of availability and geodiversity values, are highlighted in green (and the cost values highlighted in bold).

Avail.	Method	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	MUCAG-Cost-Count	4	**4**	16	4	**4**	17	7	**8**	37	7	**9**	98	7	**9**	72
	MUCAG-Count-Cost	2	6	13	3	8	15	4	10	25	7	14	31	7	14	33
	MUCAG-Cost/Count	4	**4**	16	3	**4**	15	7	**8**	29	7	**9**	67	7	**9**	69
0.99999	MUCAG-Cost-Count	9	10	36	10	**11**	44	14	20	413	14	**20**	299	14	**20**	294
	MUCAG-Count-Cost	6	14	22	7	20	29	10	26	57	10	24	85	10	24	78
	MUCAG-Cost/Count	7	**9**	30	9	**11**	35	13	**19**	131	13	**20**	66	14	**20**	95

Table 4 – Results of MUCAG method, for Node problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	MUCAG-Cost-Count	21	**27**	439	23	**30**	452	24	**31**	532	28	49	760	33	56	1174
	MUCAG-Count-Cost	19	36	348	22	42	371	24	53	513	23	**46**	581	26	63	506
	MUCAG-Cost/Count	23	32	391	22	32	471	26	37	515	30	51	650	27	**49**	945
0.99995	MUCAG-Cost-Count	27	**38**	667	27	**40**	762	33	**54**	840	33	**58**	968	33	58	1241
	MUCAG-Count-Cost	29	66	594	33	76	669	34	73	665	42	92	734	35	77	741
	MUCAG-Cost/Count	27	41	656	31	54	697	36	60	791	32	**58**	1142	31	**53**	1078
0.99999	MUCAG-Cost-Count	44	89	1308	47	102	1412	60	129	1544	56	127	1915	56	127	1781
	MUCAG-Count-Cost	40	103	1265	39	93	1423	46	113	1653	56	131	1675	56	131	1824
	MUCAG-Cost/Count	39	**85**	1247	44	**89**	1440	48	**106**	1842	53	**117**	1723	53	**117**	1711

Table 5 – Results of MUCAG method, for Node problem, Coronet

As can be seen from the tables, the variants Cost-Count and Cost/Count find in almost all cases the best solutions. In the overall, none of them is better, on average, over the other. Nevertheless, the results show that, on average, the Cost-Count variant is more efficient for lower values of availability and geodiversity while the Cost/Count variant becomes the most efficient method when these two values are higher.

On the other hand, the variant Count-Cost is less efficient in finding the best solutions as, among all cases, it has found the best solution only in one case (the case of the Coronet network, with an availability of 0.9999 and a geodiversity of 300 Km). Note, though, that in many cases the less efficient Count-Cost variant is the one whose solutions include less upgraded nodes. This fact was expected as by using the frequency cost value as the main selection property, it tends to select fewer number of upgraded nodes.

5.1.2 MUCAG with Multiple Component Selection

The next set of results aims to compare the efficiency of the 6 variants of the MUCAG with Multiple Component Selection method (Cost-Count, Count-Cost and Cost/Count, each one with WD or ND). Moreover, the aim is also to compare these results with the results presented in the previous section. Again, the removal process run at the end of each algorithm variant is based on the components ordered in decreasing order of their cost.

Table 6 and Table 7 present the obtained results. The meaning of each column is the same as in the previous tables. Moreover, these tables include an extra line for each set of availability and geodiversity values with the cost value of the best solution found in the previous MUCAG method. Note that in these tables the solutions highlighted in green (and the cost values highlighted in bold) are the lowest cost solutions (for each set of availability and geodiversity values) among all results (i.e., these results and the results of the previous section).

Avail.	Method	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	MS-Cost-Count-ND	3	**4**	14	3	**4**	14	7	8	38	8	**9**	37	8	**9**	37
	MS-Cost-Count-WD	3	**4**	14	3	**4**	15	5	**7**	24	8	**9**	28	8	10	24
	MS-Count-Cost-ND	2	6	13	3	10	14	4	11	23	6	15	22	6	15	22
	MS-Count-Cost-WD	2	6	14	3	10	15	5	10	25	6	13	35	6	13	37
	MS-Cost/Count-ND	3	**4**	15	3	**4**	15	7	8	35	7	14	21	7	14	22
	MS-Cost/Count-WD	3	**4**	15	3	**4**	15	7	10	32	8	11	22	7	10	22
	Prev. Best Sol. Cost	4			4			8			9			9		
0.99999	MS-Cost-Count-ND	6	**9**	22	7	11	24	16	22	256	15	20	111	15	20	108
	MS-Cost-Count-WD	6	10	18	6	**10**	19	10	**16**	57	12	19	535	12	19	540
	MS-Count-Cost-ND	6	17	18	8	20	19	10	27	100	10	27	42	11	30	33
	MS-Count-Cost-WD	6	16	25	7	21	28	8	23	59	10	26	60	10	26	91
	MS-Cost/Count-ND	6	**9**	21	8	13	22	11	17	79	11	19	68	12	19	113
	MS-Cost/Count-WD	6	10	18	6	**10**	19	10	17	56	11	**18**	44	11	**18**	46
	Prev. Best Sol. Cost	9			11			19			20			20		

Table 6 – Results of MUCAG with Mult. Select., for Node problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	MS-Cost-Count-ND	22	31	156	21	33	163	25	34	213	25	49	272	25	50	265
	MS-Cost-Count-WD	19	**27**	151	18	**29**	151	25	34	212	23	**39**	197	24	41	200
	MS-Count-Cost-ND	18	43	142	20	43	182	20	39	206	24	54	244	24	54	245
	MS-Count-Cost-WD	17	37	472	18	40	598	24	58	733	21	50	720	21	50	730
	MS-Cost/Count-ND	20	33	129	19	33	154	30	51	168	26	47	252	24	**40**	288
	MS-Cost/Count-WD	20	31	127	20	35	150	23	38	195	25	44	183	23	**40**	245
	Prev. Best Sol. Cost	27			30			31			46			49		
0.99995	MS-Cost-Count-ND	23	**36**	220	27	48	197	30	47	391	36	68	278	36	68	279
	MS-Cost-Count-WD	23	**36**	206	25	42	203	28	**45**	251	27	**48**	310	32	60	253
	MS-Count-Cost-ND	26	52	197	25	57	181	34	68	282	29	63	499	29	63	484
	MS-Count-Cost-WD	23	51	158	26	60	226	28	60	183	28	64	327	28	64	340
	MS-Cost/Count-ND	31	59	177	24	43	238	30	48	295	28	**48**	427	32	55	423
	MS-Cost/Count-WD	24	42	194	25	43	182	29	51	239	33	65	317	31	59	411
	Prev. Best Sol. Cost	38			40			54			58			53		
0.99999	MS-Cost-Count-ND	44	93	256	52	105	228	45	**99**	529	56	127	287	56	127	286
	MS-Cost-Count-WD	41	94	248	50	101	225	52	118	342	56	130	247	56	130	249
	MS-Count-Cost-ND	41	93	297	45	99	345	49	115	836	57	132	481	57	132	476
	MS-Count-Cost-WD	43	104	467	56	143	229	60	150	259	53	124	233	53	124	228
	MS-Cost/Count-ND	41	90	250	47	108	324	46	105	927	56	130	257	56	130	260
	MS-Cost/Count-WD	50	113	323	53	123	229	46	**99**	516	54	126	254	54	126	255
	Prev. Best Sol. Cost	85			89			106			117			117		

Table 7 – Results of MUCAG with Mult. Select., for Node problem, Coronet

These results show that the computation runtime of these method variants is reduced to approximately half of the runtime of the previous method variants, which was one of the aims of their proposal.

Moreover, for the Germany50 network, the cost of the best solutions of these method variants are equal or, in many cases, better than the best cost values of the previous method variants. However, the same cannot be said for the Coronet network, which in approximately half of the cases, these method variants obtained a best cost value which is worse than the best cost value obtained by the previous method variants. This happens especially for the cases of the availability of 0.99999, in which the MUCAG-Cost/Count method still guarantees the best solution for four of the five geodiversity values.

Similar to the conclusion in the previous section, the results show that, on average, the Cost-Count variants are more efficient for lower values of availability and geodiversity while the Cost/Count variants become more efficient when these two values are higher.

Note also that the Count-Cost-WD and Count-Cost-ND variants do not obtain any lowest cost solution. In the case where in the previous section, the MUCAG-Count-Cost obtained the best solution with a cost of 46 (the case of the Coronet network, with an availability of 0.9999 and a geodiversity of 300 Km), now a better cost value (of 39) was obtained by the MS-Cost-Count-WD (see Table 7). So, up to this point, the Count-Cost variants are the only variants that have not obtained any lowest cost solution.

For the ND (No Distinction) and WD (With Distinction) variants, although no major differences are observed concerning their average efficiency, it can be observed that the methods with distinction obtain the best results slightly more frequently than the ones without distinction.

5.1.3 MUCAG with Single Filtering Process

The next set of results aims to compare the efficiency of the 6 variants of the MUCAG with Single Filtering Process method (Cost-Count, Count-Cost and Cost/Count, each one with WD or ND). Moreover, the aim is also to compare these results with the results presented in the previous sections. As before, the removal process run at the end of each algorithm variant is based on the components ordered in decreasing order of their cost.

Table 8 and Table 9 show the results obtained with the single processing variants of this method, with the lowest cost solutions (and the cost of the best solutions) obtained up to this point highlighted as in the previous tables.

Avail.	**Method**	**D = 0 Km**			**D = 40 Km**			**D = 80 Km**			**D = 120 Km**			**D = 160 Km**		
		#N	**C**	**T**	**#N**	**C**	**T**	**#N**	**C**	**T**	**#N**	**C**	**T**	**#N**	**C**	**T**
0.99998	SF-Cost-Count-ND	4	5	14	4	5	15	7	8	48	9	11	38	9	11	38
	SF-Cost-Count-WD	3	**4**	13	3	**4**	14	7	8	34	8	**9**	30	7	**9**	28
	SF-Count-Cost-ND	2	6	12	3	10	14	4	10	23	6	15	23	6	15	23
	SF-Count-Cost-WD	2	6	13	3	10	14	4	11	19	6	16	24	6	16	24
	SF-Cost/Count-ND	3	5	14	4	5	16	8	10	29	8	10	31	8	10	30
	SF-Cost/Count-WD	3	5	13	3	5	13	5	9	41	7	10	28	6	10	28
	Prev. Best Sol. Cost	4			4			7			9			9		
0.99999	SF-Cost-Count-ND	8	11	17	10	15	17	14	20	272	15	20	586	15	20	581
	SF-Cost-Count-WD	6	**9**	15	6	**9**	16	9	**16**	102	12	19	605	12	19	605
	SF-Count-Cost-ND	6	17	15	7	18	20	10	27	439	10	24	398	10	25	404
	SF-Count-Cost-WD	6	17	16	6	18	20	12	32	119	10	26	47	10	26	48
	SF-Cost/Count-ND	8	11	17	10	15	17	11	21	349	13	22	1573	13	22	1552
	SF-Cost/Count-WD	6	11	16	7	12	17	10	**16**	96	14	23	246	14	23	240
	Prev. Best Sol. Cost	9			10			16			18			18		

Table 8 – Results MUCAG with Single Filter, for Node problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	SF-Cost-Count-ND	20	27	76	24	39	74	25	33	123	27	51	157	27	51	171
	SF-Cost-Count-WD	20	28	91	24	35	123	25	33	115	28	49	158	25	44	143
	SF-Count-Cost-ND	17	38	71	18	38	160	24	46	527	26	51	986	26	51	987
	SF-Count-Cost-WD	18	43	72	18	38	160	22	43	255	35	88	269	29	69	319
	SF-Cost/Count-ND	15	**24**	81	19	34	98	25	38	266	27	46	261	24	**39**	722
	SF-Cost/Count-WD	17	29	73	19	34	96	24	43	80	25	44	192	25	44	189
	Prev. Best Sol. Cost	27			29			31			39			40		
0.99995	SF-Cost-Count-ND	29	50	104	25	43	96	31	48	283	33	68	332	35	64	230
	SF-Cost-Count-WD	26	44	111	24	42	108	29	48	121	31	54	284	27	**49**	240
	SF-Count-Cost-ND	25	54	96	28	67	129	32	66	360	41	95	3878	41	95	3868
	SF-Count-Cost-WD	26	63	58	26	64	130	27	56	383	29	68	545	37	87	538
	SF-Cost/Count-ND	25	51	140	24	43	122	33	59	1046	40	81	1285	40	81	1280
	SF-Cost/Count-WD	32	64	74	28	54	77	30	57	187	29	50	233	27	**49**	243
	Prev. Best Sol. Cost	36			40			45			48			53		
0.99999	SF-Cost-Count-ND	44	99	72	54	124	66	46	**95**	155	56	130	340	56	130	337
	SF-Cost-Count-WD	39	90	272	48	102	55	55	119	70	57	133	111	56	129	131
	SF-Count-Cost-ND	41	96	92	59	152	70	52	125	344	56	132	393	57	136	421
	SF-Count-Cost-WD	38	89	154	49	122	61	54	131	206	53	133	435	57	140	502
	SF-Cost/Count-ND	44	99	73	52	118	61	48	104	530	56	130	338	56	130	338
	SF-Cost/Count-WD	41	89	80	50	117	63	48	104	246	55	126	304	56	128	283
	Prev. Best Sol. Cost	85			89			99			117			117		

Table 9 – Results of MUCAG with Single Filter, for Node problem, Coronet

From the obtained results, it can be concluded that this method introduces only marginal gains to the best results obtained up to this point as these method variants only improve the previous best cost value in one case in the Germany50 network (Table 8) and four cases in the Coronet network (Table 9). The computation runtimes are in average larger than the previous Multiple Selection methods, taking approximately three quarters of the runtimes of the MUCAG method variants.

Confirming the conclusions obtained before, the Count-Cost and Count/Cost variants provide better cost solutions, on average, and the Count-Cost variant returns the worse results, never obtaining a solution with the lowest cost. Moreover, the comparison between the WD and the ND variants show again that WD variants tend to obtain better results, on average, although there are counter examples.

To test the multiprocessing variants of the method, the variants SF-Count-Cost-WD and SF-Count/Cost-WD were used as, on average, were the most efficient ones in the single process variants. Two different multiprocessing variants were tested: one where only all permutations of the first *k* node pairs were used and the other where all swaps between each of the first *k* node pairs and the node pairs from *k*+1 until *r* were also used. The aim is to check if the swap permutations let the method obtain better solutions or if the permutations of the first k node pairs are enough to obtain good solutions. In the tests, *k* was set to 3 and *r* was set to 5.

Table 10 and Table 11 shows the results obtained by the different multiprocessing variants of the method (the results of the single processing SF-

Count-Cost-WD and SF-Count/Cost-WD variants are repeated in these tables for comparison reasons).

Avail.	Method	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	SF-Cost-Count-WD	3	**4**	13	3	**4**	14	7	8	34	8	**9**	30	7	**9**	28
	“”, k=3	3	**4**	14	3	**4**	14	5	8	37	7	**9**	68	7	**9**	53
	“”, k=3, r=5	3	**4**	15	3	**4**	16	5	8	44	7	**9**	69	7	**9**	55
	SF-Cost/Count-WD	3	5	13	3	5	13	5	9	41	7	10	28	6	10	28
	“”, k=3	3	5	14	3	5	14	5	8	43	7	**9**	52	7	**9**	40
	“”, k=3, r=5	3	5	14	3	5	15	5	8	46	7	**9**	53	7	**9**	41
	Prev. Best Sol. Cost	4			4			7			9			9		
0.99999	SF-Cost-Count-WD	6	**9**	15	6	**9**	16	9	**16**	102	12	19	605	12	19	605
	“”, k=3	6	**9**	17	6	**9**	22	9	**16**	325	11	**18**	641	12	19	618
	“”, k=3, r=5	6	**9**	17	6	**9**	22	9	**16**	391	10	**18**	678	10	**18**	666
	SF-Cost/Count-WD	6	11	16	7	12	17	10	**16**	96	14	23	246	14	23	240
	“”, k=3	6	10	17	6	**9**	19	10	**16**	204	12	19	291	12	19	297
	“”, k=3, r=5	6	10	24	6	**9**	19	9	**16**	214	11	**18**	307	11	**18**	308
	Prev. Best Sol. Cost	9			9			16			18			18		

Table 10 – Results of Variance in Order, for Node problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	SF-Cost-Count-WD	20	28	91	24	35	123	25	33	115	28	49	158	25	44	143
	“”, k=3	20	28	91	24	35	120	22	32	113	22	36	196	25	41	270
	“”, k=3, r=5	20	28	94	23	35	158	22	32	116	22	36	397	25	41	301
	SF-Cost/Count-WD	17	29	73	19	34	96	24	43	80	25	44	192	25	44	189
	“”, k=3	17	29	72	19	34	97	22	32	87	19	**34**	269	22	38	305
	“”, k=3, r=5	17	29	75	19	34	98	22	32	174	19	**34**	279	23	**37**	313
	Prev. Best Sol. Cost	24			29			31			39			49		
0.99995	SF-Cost-Count-WD	26	44	111	24	42	108	29	48	121	31	54	284	27	49	240
	“”, k=3	26	43	113	24	42	106	29	48	121	26	**46**	296	26	**46**	254
	“”, k=3, r=5	26	43	113	24	42	110	26	**44**	173	26	**46**	390	26	**46**	275
	SF-Cost/Count-WD	32	64	74	28	54	77	30	57	187	29	50	233	27	49	243
	“”, k=3	26	48	91	28	54	107	27	46	189	26	48	241	27	**46**	970
	“”, k=3, r=5	26	48	91	28	54	109	27	46	193	26	48	656	27	**46**	979
	Prev. Best Sol. Cost	36			40			45			48			49		
0.99999	SF-Cost-Count-WD	39	90	272	48	102	55	55	119	70	57	133	111	56	129	131
	“”, k=3	39	90	272	48	102	55	52	110	72	54	122	179	53	122	260
	“”, k=3, r=5	39	90	277	45	93	56	50	110	83	54	122	213	53	122	627
	SF-Cost/Count-WD	41	89	80	50	117	63	48	104	246	55	126	304	56	128	283
	“”, k=3	41	89	79	50	117	62	48	104	443	52	119	454	52	119	461
	“”, k=3, r=5	41	89	80	50	117	90	45	98	458	52	119	485	52	119	642
	Prev. Best Sol. Cost	85			89			95			117			117		

Table 11 – Results in Variance in Order, for Node problem, Coronet

The first conclusion is that the addition of swap permutations is recommended, as in several cases, these variants obtained better results than the variants without the swap permutations. Secondly, the multiprocessing variants allow the MUCAG with Single Filtering Process method to obtain equal or better results in approximately half of the cases.

In the Germany50 network, it has obtained the same previous best results for 9 out of 10 cases with the multiprocessing SF-Cost-Count-WD (with *k*=3 and *r*=5).

Moreover, in the Coronet network, the multiprocessing variants have obtained better solutions than the previous methods in 5 different cases.

Among all results presented so far (in this and all previous sections), an obvious conclusion is that there is no single algorithm variant that obtains the best results for all problem instances. However, there are 4 algorithm variants that have obtained most of the best solutions found among all methods:

- The first variant is the multiprocessing SF-Cost-Count-WD, which as noticed before, obtained the best results for 9 of the 10 cases of the Germany50 network. Moreover, this method has also obtained the best solution in 3 cases of the Coronet network.

- The second variant is the multiprocessing SF-Cost/Count-WD variant since, despite not being as effective (in finding the best solutions) as the previous variant, still managed to obtain good results, on average, including the best ones in 3 cases of the Coronet network.

- Another variant of value is the MUCAG-Cost/Count, as it managed to obtain good results for the higher availability values and has obtained the best results for 4 cases of the Coronet network.

- The last variant is the MS-Cost-Count-WD, as it has obtained the best cost values for many of the remaining cases (i.e., the cases such that the 3 previous variants did not obtain the best cost values).

Among these four method variants, the best solutions were obtained in 84% of the total possible cases (i.e., 21 out of 25 cases).

5.1.4 Removal process

With the group of 4 method variants selected in the previous section, a new program was implemented where these four methods are run in sequence and, at the end, all 26 upgrade solutions (12 generated by multiprocessing SF-Cost-Count-WD variant, 12 generated by the multiprocessing SF-Cost/Count-WD variant, 1 generated by the MUCAG-Cost/Count variant and 1 generated by the MS-Cost-Count-WD variant) are given as input solutions to the removal process. Then, the four different removal orders (as described in section 4.6) were tested. The removal process is run separately for each removal order, returning among all obtained solutions the one with the lowest cost.

In the following Table 12 and Table 13, the results of these computational tests are presented.

Avail.	Order	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	Cost	3	**4**	22	3	**4**	22	5	**7**	74	7	**9**	137	7	**9**	112
	Insertion	3	**4**	22	3	**4**	22	5	**7**	74	7	**9**	108	7	**9**	109
	Frequency Des.	3	**4**	22	3	**4**	22	5	**7**	64	7	**9**	106	7	**9**	112
	Frequency Asc.	3	**4**	22	3	**4**	22	5	**7**	76	7	**9**	112	7	**9**	138
	Prev. Best Sol. Cost	4			4			7			9			9		
0.99999	Cost	6	**9**	48	6	**9**	53	9	**16**	623	10	**18**	1090	10	**18**	1086
	Insertion	6	**9**	49	6	**9**	53	9	**16**	513	10	**18**	1007	10	**18**	1027
	Frequency Des.	6	**9**	46	6	**9**	53	9	**16**	440	10	**18**	1059	10	**18**	1024
	Frequency Asc.	6	**9**	47	6	**9**	53	9	**16**	566	10	**18**	1160	10	**18**	1133
	Prev. Best Sol. Cost	9			9			16			18			18		

Table 12 – Results of Removal methods, for Node problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	Cost	19	27	709	18	**29**	963	22	32	1078	19	**34**	1425	23	**37**	1578
	Insertion	19	27	709	18	**29**	976	22	32	1076	19	**34**	1441	23	38	1481
	Frequency Des.	19	27	712	18	**29**	955	22	32	1081	20	35	1465	23	39	1465
	Frequency Asc.	19	27	704	18	**29**	983	22	32	1066	19	**34**	1449	23	38	1564
	Prev. Best Sol. Cost	24			29			31			34			37		
0.99995	Cost	23	**36**	1445	24	42	1610	26	**44**	1815	62	**46**	2417	63	**46**	2745
	Insertion	22	**36**	1452	24	42	1612	27	46	1816	26	47	2360	27	47	2292
	Frequency Des.	24	42	1435	26	44	1628	28	46	1796	28	54	2293	28	49	2339
	Frequency Asc.	23	**36**	1438	24	41	1607	25	**44**	1803	27	50	2339	27	50	2720
	Prev. Best Sol. Cost	36			40			44			46			46		
0.99999	Cost	39	**85**	3068	44	**89**	3508	45	98	4198	53	**117**	4054	53	**117**	4632
	Insertion	40	86	3136	43	89	3506	44	98	4207	52	121	4059	52	118	4519
	Frequency Des.	39	88	3063	45	93	3499	45	101	4322	53	122	4015	52	119	4487
	Frequency Asc.	41	88	3236	45	92	3500	44	**95**	4142	52	**117**	4109	53	118	4496
	Prev. Best Sol. Cost	85			89			95			117			117		

Table 13 – Results of Removal methods, for Node problem, Coronet

As can be seen in Table 12, for Germany50 network, all removal orders have obtained the best cost values found so far for all cases (possibly meaning that the obtained solutions are optimal ones, although not guaranteed). The main reason for the same efficiency among the different removal orders might be due to the small number of possible nodes to be removed in the input upgrade solutions.

For the Coronet Network, since each input upgrade solution has a higher number of nodes, the order of removal does affect the obtained results. The results of Table 13 show that the removing process by cost (used in all previous sections) is the best among all alternatives as it has obtained the best cost values for 11 out of 15 cases. The second best is the removing process by frequency, ordered ascendingly, which has obtained the best cost values for 6 out of 15 cases and it was better than the removing by cost in one case (availability of 0.999999 and geodiversity of 200 Km). The other two removing order alternatives have much worse efficiency performance.

For illustrative reasons, Fig. 15 presents in the right the best solution obtained for the Germany50 network for an availability of 0.99999 and for the geodiversity value *D* = 40 Km (the thickness of the links is proportional to the availability improvement due to the upgrade of its end nodes). Moreover, in the left of Fig. 15, node costs are coded in different colours where white indicates a cost of 1, yellow indicates a cost of 2, orange indicates a cost of 3 and red indicates a cost of 4.

It is interesting to check that most of the upgraded nodes have simultaneously a low cost and a higher degree, i.e., a higher number of directly connected links. This is the case of nodes 20, 32, 44 and 49. The other two upgraded nodes are node 16 that has the minimum cost of 1 and node 38 that has a high node degree of 4.

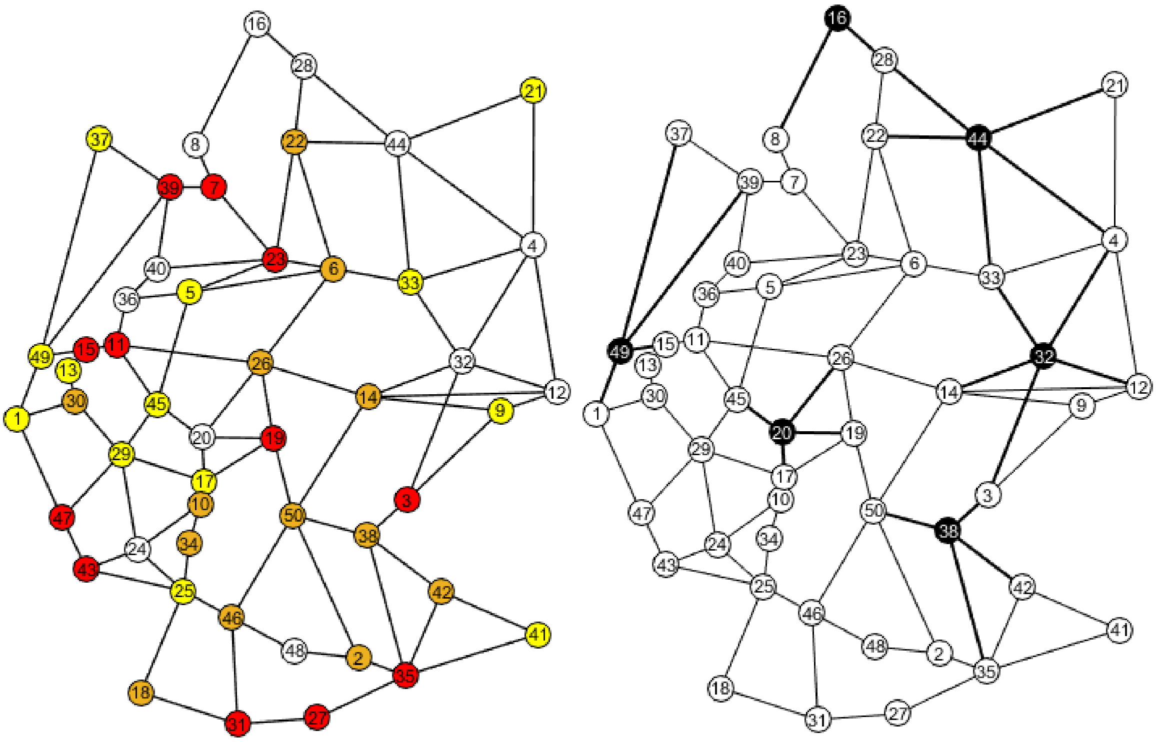

Figure 15 – Left: node costs of Germany50 coded by different colours.
Right: upgraded nodes for *D* = 40 Km.

Fig. 16 presents the best solutions obtained for the Germany50 network for an availability of 0.99999 and for the geodiversity values *D* = 80 and 120 Km. As the requirement for the geodiversity increases, the required distance between the 2 routing paths of each node pair also increases, more nodes need to be upgraded since more links are needed to have a higher availability. Again, most of the upgraded nodes in these two solutions are either nodes with low cost or nodes with high node degree or node with both properties.

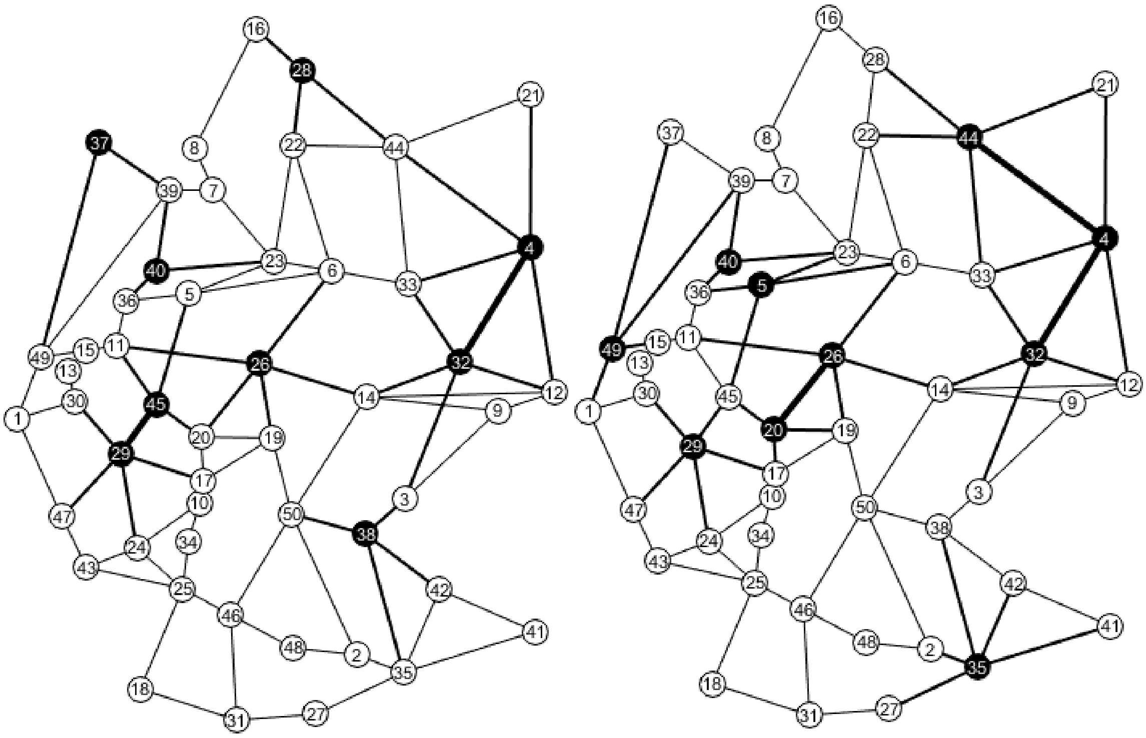

Figure 16 – Upgraded nodes for *D* = 80 Km (left) and D = 120 Km (right) in Germany50 network

5.2 Link Upgrade Results

The link upgrade problem considers that the availability of links can be individually improved and the objective is to select the links to be upgraded so that the total upgrade cost is minimized and the availability and geodiversity values are guaranteed for all node pairs of the network. Next, the results of each method are presented in separated subsections.

5.2.1 MUCAG

The first set of results aims to compare the efficiency of the 3 variants of the MUCAG method (Cost-Count, Count-Cost and Cost/Count). In all cases, the removal process run at the end of the algorithm is based on the components ordered in decreasing order of their cost.

In the following Table 14 and Table 15, **C** is the cost of the final solution, #**N** is the number of upgraded link and **T** is the total runtime of the algorithm, in seconds. The solutions with the lowest cost, for each set of availability and geodiversity values, are highlighted in green (and the cost values highlighted in bold).

Avail.	Method	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	MUCAG-Cost-Count	13	724	45	14	767	43	24	1423	243	30	1807	280	27	1706	296
	MUCAG-Count-Cost	7	**594**	16	6	**605**	19	15	**1276**	46	18	**1502**	61	18	1608	63
	MUCAG-Cost/Count	13	732	32	13	690	31	25	1521	114	27	1764	167	22	**1559**	164
0.99999	MUCAG-Cost-Count	30	1914	159	29	1807	173	36	2577	1047	41	3192	1266	42	3410	1851
	MUCAG-Count-Cost	16	**1319**	44	18	**1507**	52	23	**2192**	115	30	**2597**	175	31	**2805**	233
	MUCAG-Cost/Count	21	1339	84	24	1573	96	37	2794	236	42	3084	387	36	2871	379

Table 14 – Results of MUCAG method for Link problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	MUCAG-Cost-Count	66	15326	7213	60	14620	7062	66	16845	9438	70	19294	9549	70	19294	9307
	MUCAG-Count-Cost	33	**11940**	1085	44	**13197**	1339	44	**14240**	1597	51	**16214**	1777	52	**16586**	1764
	MUCAG-Cost/Count	56	13205	2418	56	13248	2830	62	15581	3384	61	16892	3495	61	17222	3514
0.99995	MUCAG-Cost-Count	68	17077	7199	65	17382	7856	70	19887	16411	73	21583	8991	72	21025	8344
	MUCAG-Count-Cost	46	**14498**	1878	58	16009	2443	54	**17616**	2604	59	**19363**	2821	63	21346	2869
	MUCAG-Cost/Count	54	15096	3572	61	**15914**	3951	67	17938	4558	71	20151	4302	70	**20200**	4194
0.99999	MUCAG-Cost-Count	73	**18746**	8202	74	**20212**	9346	77	23257	11212	77	**24233**	10817	80	**25558**	11901
	MUCAG-Count-Cost	68	19310	3885	72	21952	4246	79	23418	5259	79	25382	6773	81	26176	6308
	MUCAG-Cost/Count	70	19161	5304	76	21157	5985	77	**22778**	6392	80	25356	7697	80	25693	9567

Table 15 – Results of MUCAG method for Link problem, Coronet

Contrary to the node upgrade problem, in the case of the link upgrade problem, the MUCAG-Count-Cost is the most efficient between the three variants, managing to obtain the best result in 17 out of 25 cases (among both networks).

Moreover, the efficiency between the other two variants is very similar, with the MUCAG-Cost-Count obtaining the best result of one case more than the MUCAG-Cost/Count, with the former only reaching these results for the Coronet network with the availability of 0.99999.

5.2.2 MUCAG with Multiple Component Selection

The next set of results aims to compare the efficiency of the 6 variants of the MUCAG with Multiple Component Selection method (Cost-Count, Count-Cost and Cost/Count, each one with WD or ND). Moreover, the aim is also to compare these results with the results presented in the previous section. Again, the removal process run at the end of each algorithm variant is based on the components ordered in decreasing order of their cost.

Table 16 and Table 17 present the obtained results. These tables include an extra line with the cost value of the best solution found in the previous MUCAG method and the highlighted solutions are the lowest cost solutions among these results and the results of the previous section.

Avail.	**Method**	**D = 0 Km**			**D = 40 Km**			**D = 80 Km**			**D = 120 Km**			**D = 160 Km**		
		#N	**C**	**T**	**#N**	**C**	**T**	**#N**	**C**	**T**	**#N**	**C**	**T**	**#N**	**C**	**T**
0.99998	MS-Cost-Count-ND	12	846	15	13	892	16	22	1368	32	23	1551	53	24	1612	51
	MS-Cost-Count-WD	9	947	17	10	1021	18	17	1344	44	20	1674	39	21	1772	39
	MS-Count-Cost-ND	8	673	15	6	548	15	12	**1080**	45	17	1617	30	19	1668	43
	MS-Count-Cost-WD	6	809	16	6	**542**	17	9	1190	39	14	**1422**	61	16	**1468**	61
	MS-Cost/Count-ND	8	687	15	7	565	15	20	1395	44	20	1452	41	23	1615	56
	MS-Cost/Count-WD	7	642	15	6	548	16	17	1473	38	16	1540	41	16	1658	48
	Prev. Best Sol. Cost	594			605			1276			1502			1559		
0.99999	MS-Cost-Count-ND	28	2123	37	25	1806	38	32	2439	75	39	3178	305	39	3113	268
	MS-Cost-Count-WD	19	1583	28	21	1687	34	35	2750	136	30	2702	92	35	2990	166
	MS-Count-Cost-ND	13	**1313**	20	16	1511	23	31	2552	93	28	2710	77	31	2783	137
	MS-Count-Cost-WD	13	**1313**	41	16	1552	51	21	**2188**	127	23	2697	157	22	**2375**	161
	MS-Cost/Count-ND	16	1392	24	17	**1482**	24	29	2414	71	34	2693	58	34	2842	157
	MS-Cost/Count-WD	16	1439	24	18	1611	26	25	2341	62	31	2837	61	30	2960	111
	Prev. Best Sol. Cost	1319			1507			2192			2597			2805		

Table 16 – Results of MUCAG with Mult. Select. for Link problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	MS-Cost-Count-ND	54	13763	315	59	15147	374	60	16090	359	75	19853	444	75	19853	445
	MS-Cost-Count-WD	43	13306	235	50	13905	361	53	16197	410	52	16520	247	53	17361	299
	MS-Count-Cost-ND	34	12122	164	40	**12417**	219	49	15440	254	47	15779	200	47	16490	203
	MS-Count-Cost-WD	29	**11547**	1324	42	14699	1953	43	15577	2017	46	**15163**	2498	47	16529	2517
	MS-Cost/Count-ND	42	12718	247	53	13163	196	53	14832	292	55	15985	246	55	16504	215
	MS-Cost/Count-WD	43	12208	251	52	14032	266	51	14983	235	52	16478	263	52	**16478**	257
	Prev. Best Sol. Cost	11940			13197			14240			16214			18586		
0.99995	MS-Cost-Count-ND	63	16957	544	62	17984	346	63	18631	594	69	20462	497	69	20462	592
	MS-Cost-Count-WD	55	15689	274	62	17043	460	60	18527	511	67	20655	344	67	20768	333
	MS-Count-Cost-ND	43	15229	250	55	16091	347	58	17746	303	60	19110	259	60	19110	263
	MS-Count-Cost-WD	41	15214	2418	49	17625	3289	55	20562	4337	64	20266	4281	63	21038	4181
	MS-Cost/Count-ND	57	15749	411	57	16096	370	62	18533	435	63	**18943**	405	61	**18923**	364
	MS-Cost/Count-WD	47	15014	273	57	17165	351	54	**16761**	297	63	19497	382	62	19895	331
	Prev. Best Sol. Cost	14498			15914			17616			19363			20200		
0.99999	MS-Cost-Count-ND	71	19393	445	74	20787	629	75	**22603**	627	77	24331	959	77	25566	1912
	MS-Cost-Count-WD	72	19959	503	72	20592	688	73	23556	709	78	24881	934	77	25805	1201
	MS-Count-Cost-ND	69	20553	512	71	21181	463	73	23040	750	77	24764	831	77	**24808**	660
	MS-Count-Cost-WD	66	20172	6070	80	24635	7744	72	23731	8071	84	27424	8751	75	25442	9481
	MS-Cost/Count-ND	73	20154	442	73	20288	538	75	23116	792	75	**23991**	1129	79	25571	1556
	MS-Cost/Count-WD	70	20648	539	73	21046	542	72	23333	532	78	26035	1102	78	26260	953
	Prev. Best Sol. Cost	18746			20212			22778			24233			25558		

Table 17 – Results of MUCAG with Mult. Select. for Link problem, Coronet

Comparatively to the previous MUCAG method variants, the Multiple Selection method variants have an overall reduction of the runtime to, approximately, one fourth of the runtime of the former method. The obtained results tend to reach the values shown in the previous method for approximately 75% of cases (among both networks).

Again, the Count-Cost variants are more efficient among all variants. Between the other two variants, the Cost/Count variants now obtain more lowest cost solutions than the Cost-Count variants making these last variants the worst of the three in terms of efficiency.

For the *ND* (No Distinction) and *WD* (With Distinction) variants, both obtain the same number of lowest cost solutions, with the variant *WD* being more efficient for the Germany50 network and the variant *ND* more efficient for the Coronet network. So, in overall, both the MS-Count-Cost-ND and the MS-Count-Cost-WD methods are the most efficient methods, on average, in terms of the number of lower cost solutions.

5.2.3 MUCAG with Single Filtering Process

The next set of results aims to compare the efficiency of the 6 variants of the MUCAG with Single Filtering Process method (Cost-Count, Count-Cost and Cost/Count, each one with WD or ND). Again, the aim is also to compare these results with the results presented in the previous section. Also, the removal process run at the end of each algorithm variant is based on the components ordered in decreasing order of their cost.

Table 18 and Table 19 show the results obtained with the single processing variants of this method. As before, these tables include an extra line with the cost value of the best solution found in the previous methods and the highlighted solutions are the lowest cost solutions among these results and the results of the 2 previous sections.

Avail.	Method	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	SF-Cost-Count-ND	12	846	15	13	879	16	22	1326	47	24	1631	46	24	1612	47
	SF-Cost-Count-WD	10	654	17	13	1105	16	20	1496	45	21	1705	33	18	1585	33
	SF-Count-Cost-ND	6	556	14	7	588	13	17	1479	28	17	1697	26	19	1764	28
	SF-Count-Cost-WD	6	596	14	7	734	15	11	1367	38	13	**1318**	25	15	1737	27
	SF-Cost/Count-ND	6	**548**	15	6	548	14	19	1403	52	24	1743	39	23	1650	92
	SF-Cost/Count-WD	6	**548**	14	6	548	15	15	1370	34	16	1425	35	16	1534	34
	Prev. Best Sol. Cost	594			542			1080			1422			1468		
0.99999	SF-Cost-Count-ND	24	1909	20	25	1987	27	33	2427	130	36	2808	264	41	3564	574
	SF-Cost-Count-WD	17	1620	21	18	1633	21	29	2462	87	38	2982	103	40	3178	166
	SF-Count-Cost-ND	14	**1294**	18	14	**1306**	18	27	2429	112	29	2819	91	30	2972	123
	SF-Count-Cost-WD	13	1439	18	16	1769	19	22	2339	41	27	2873	46	29	3023	48
	SF-Cost/Count-ND	22	1592	22	16	1401	18	33	2582	83	34	2680	113	34	2742	151
	SF-Cost/Count-WD	15	1366	17	17	1638	19	30	2511	53	32	2896	64	33	3000	64
	Prev. Best Sol. Cost	1313			1482			2188			2597			2375		

Table 18 – Results of MUCAG with Single Filter for Link problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	SF-Cost-Count-ND	55	13251	216	65	14191	369	65	17356	794	60	17692	1729	64	18406	858
	SF-Cost-Count-WD	46	13027	136	53	13770	359	55	16473	303	51	16057	218	53	16974	295
	SF-Count-Cost-ND	36	12187	117	50	14523	142	61	18728	1801	71	22311	1086	71	22311	1065
	SF-Count-Cost-WD	31	11595	93	39	13177	187	38	14751	156	58	19976	98	62	21366	121
	SF-Cost/Count-ND	55	14101	368	63	15694	615	64	16371	369	64	18470	1334	64	18357	1121
	SF-Cost/Count-WD	44	13280	110	46	13003	169	50	15079	141	55	16207	246	54	17069	260
	Prev. Best Sol. Cost		11547			12417			14240			15163			16478	
0.99995	SF-Cost-Count-ND	62	17123	217	65	16938	348	66	20529	1174	71	21303	1177	72	21025	843
	SF-Cost-Count-WD	58	16036	156	62	17614	513	61	17258	199	65	20082	297	65	21091	229
	SF-Count-Cost-ND	48	16308	144	57	16733	251	61	19260	365	63	20896	1339	64	21563	2071
	SF-Count-Cost-WD	44	16086	124	51	17826	128	71	23722	249	68	23724	227	70	23624	175
	SF-Cost/Count-ND	50	14837	291	65	17082	378	66	19371	1105	69	21203	4353	70	21698	6034
	SF-Cost/Count-WD	57	17303	158	66	18980	423	59	18452	194	66	20900	723	61	19408	540
	Prev. Best Sol. Cost		14498			15914			16761			18943			18923	
0.99999	SF-Cost-Count-ND	72	19370	154	73	20798	251	77	23307	1091	77	24233	1366	80	25558	3153
	SF-Cost-Count-WD	71	20202	149	71	20813	310	76	22706	332	76	24380	1829	80	25558	2376
	SF-Count-Cost-ND	72	21070	166	69	20800	389	74	22837	340	76	24575	586	79	25239	1089
	SF-Count-Cost-WD	69	20705	150	70	21616	483	73	24170	624	86	28259	1109	84	28093	2838
	SF-Cost/Count-ND	73	19953	175	72	20860	223	77	23307	987	75	**23991**	1223	80	25799	3814
	SF-Cost/Count-WD	72	21727	214	72	20867	200	77	24578	548	79	26656	4299	77	25811	1233
	Prev. Best Sol. Cost		18746			20288			22603			23991			24808	

Table 19 – Results of MUCAG with Single Filter for Link problem, Coronet

Comparative to the method variants of the previous sections, there is an improvement only in four of the cases in the Germany50 network, with the other cases staying significantly far, on average, from the lowest cost values obtained by the previous method variants. In terms of runtime, the average runtime of these method variants is reduced by, approximately, one fifth of the MUCAG method and showing that these method variants are the fastest among all three methods.

On average, it can be concluded that the most efficient single process variants were SF-Count-Cost-ND, SF-Count-Cost-WD and SF-Count/Cost-ND. So, to test the multiprocessing variants of the method, these 3 variants were used. Like in the node upgrade problem (section 5.1), two different multiprocessing variants were tested: one where only all permutations of the first k=3 node pairs were used and the other where all swaps between each of the first k node pairs and the node pairs from k+1 until r=5 were also used. Again, the aim is to check if the swap permutations let the method obtain better solutions or if the permutations of the first k node pairs are enough to obtain good solutions.

Table 20 and Table 21 shows the results obtained by the different multiprocessing variants of the method (the results of the single processing variants are repeated in these tables for comparison reasons).

Avail.	Method	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	SF-Count-Cost-ND	6	556	14	7	588	13	17	1479	28	17	1697	26	19	1764	28
	“”, k=3	6	556	16	7	588	14	15	1295	49	16	1531	36	18	1556	39
	“”, k=3, r=5	6	556	15	7	588	16	15	1295	55	15	1382	46	16	1447	42
	SF-Count-Cost-WD	6	596	14	7	734	15	11	1367	38	13	**1318**	25	15	1737	27
	“”, k=3	6	575	15	6	657	16	11	1367	39	13	**1318**	31	14	1458	30
	“”, k=3, r=5	6	575	15	6	548	16	8	**1064**	40	13	**1318**	35	14	1458	36
	SF-Cost/Count-ND	6	**548**	15	6	548	14	19	1403	52	24	1743	39	23	1650	92
	“”, k=3	6	**548**	15	6	548	16	15	1241	67	20	1516	53	20	**1440**	90
	“”, k=3, r=5	6	**548**	17	6	548	19	16	1221	74	20	1516	57	20	**1440**	99
	Prev. Best Sol. Cost	548			542			1080			1318			1468		
0.99999	SF-Count-Cost-ND	14	**1294**	18	14	**1306**	18	27	2429	112	29	2819	91	30	2972	123
	“”, k=3	14	**1294**	19	14	**1306**	23	27	2429	126	29	2550	99	29	2853	132
	“”, k=3, r=5	14	**1294**	21	14	**1306**	24	25	2356	130	26	2549	106	29	2825	192
	SF-Count-Cost-WD	13	1439	18	16	1769	19	22	2339	41	27	2873	46	29	3023	48
	“”, k=3	13	1439	20	16	1769	19	22	2339	43	27	**2481**	73	29	3023	54
	“”, k=3, r=5	14	1404	20	15	1698	21	21	**2173**	80	27	**2481**	71	29	3023	59
	SF-Cost/Count-ND	22	1592	22	16	1401	18	33	2582	83	34	2680	113	34	2742	151
	“”, k=3	16	1328	24	16	1401	19	27	2342	130	33	2626	143	34	2742	201
	“”, k=3, r=5	16	1328	25	15	1356	26	28	2297	142	30	2576	230	32	2647	263
	Prev. Best Sol. Cost	1294			1306			2188			2597			2375		

Table 20 – Results of Variance in Order, for Link problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	SF-Count-Cost-ND	36	12187	117	50	14523	142	61	18728	1801	71	22311	1086	71	22311	1065
	“”, k=3	36	12187	124	50	14523	154	56	16444	1907	63	19607	6676	59	18549	6573
	“”, k=3, r=5	36	12187	129	50	14367	182	56	16444	2808	60	18501	6864	59	18549	6865
	SF-Count-Cost-WD	31	11595	93	39	13177	187	38	14751	156	58	19976	98	62	21366	121
	“”, k=3	31	11595	97	39	13177	199	38	14751	167	58	19976	102	62	21366	129
	“”, k=3, r=5	31	11595	101	39	13177	208	38	14751	172	52	17568	107	57	19850	133
	SF-Cost/Count-ND	55	14101	368	63	15694	615	64	16371	369	64	18470	1334	64	18357	1121
	“”, k=3	55	14101	402	63	15694	669	64	16371	410	58	17248	1398	60	17680	1680
	“”, k=3, r=5	55	14101	409	62	15310	697	57	15352	590	58	17248	1757	60	17680	1966
	Prev. Best Sol. Cost	11547			12417			14240			15163			16478		
0.99995	SF-Count-Cost-ND	48	16308	144	57	16733	251	61	19260	365	63	20896	1339	64	21563	2071
	“”, k=3	48	16308	156	57	16733	275	61	17927	443	63	20896	2702	67	20818	2616
	“”, k=3, r=5	48	16308	158	53	16034	277	61	17927	599	63	20896	2894	69	20794	2734
	SF-Count-Cost-WD	44	16086	124	51	17826	128	71	23722	249	68	23724	227	70	23624	175
	“”, k=3	44	16086	132	51	17826	116	71	23722	282	68	23724	247	70	23624	188
	“”, k=3, r=5	44	16086	137	51	17826	118	53	19651	296	68	22753	391	65	22717	312
	SF-Cost/Count-ND	50	14837	291	65	17082	378	66	19371	1105	69	21203	4353	70	21698	6034
	“”, k=3	50	14837	312	65	17082	412	66	19371	1971	69	21203	5149	68	21551	11200
	“”, k=3, r=5	50	14837	325	66	17011	421	64	18247	2594	69	20912	6982	68	21551	11444
	Prev. Best Sol. Cost	14498			15914			16761			18943			18923		
0.99999	SF-Count-Cost-ND	72	21070	166	69	20800	389	74	22837	340	76	24575	586	79	25239	1089
	“”, k=3	72	21070	178	69	20800	426	74	22837	706	76	24575	1157	79	25189	3393
	“”, k=3, r=5	72	21070	182	69	20800	437	74	22837	922	76	24575	1329	79	25189	3605
	SF-Count-Cost-WD	69	20705	150	70	21616	483	73	24170	624	86	28259	1109	84	28093	2838
	“”, k=3	69	20705	163	70	21536	514	73	24170	680	86	28259	1228	84	28093	3129
	“”, k=3, r=5	69	20705	166	70	21536	536	73	24170	703	74	25786	1285	84	28093	3320
	SF-Cost/Count-ND	73	19953	175	72	20860	223	77	23307	987	75	**23991**	1223	80	25799	3814
	“”, k=3	73	19953	186	72	20860	239	77	23307	1263	75	**23991**	1347	80	25799	4033
	“”, k=3, r=5	73	19953	191	72	20860	248	78	22965	1406	75	**23991**	2041	80	25799	4341
	Prev. Best Sol. Cost	18746			20288			22603			23991			24808		

Table 21 – Results in Variance in Order, for Link problem, Coronet

The first conclusion is that the addition of swap permutations is recommended but its impact is not as significant as it was in the node upgrade problem: there are a significant number of cases such that it didn’t improve the obtained cost values and in most cases the cost improvement is low in percentage.

For the Coronet network, the multiprocessing variants did not obtain any cost improvement in the obtained solutions. On the other hand, for the Germany50 network, there were cost improvements in four of the cases.

Among all results presented so far (in this and all previous sections), a conclusion which is similar to the one in the node upgraded problem is that there is no single algorithm variant that obtains the best results for all problem instances. Moreover, in the link upgrade problem (unlike the node upgrade problem), there is no short set of algorithm variants whose efficiency is much better than the other ones.

5.2.4 Removal process

To test the different component orders of the removal process, a set of 4 method variants were selected. The first two variants are the SF-Count-Cost-WD and MS-Count-Cost-WD, since these individually have obtained the highest number of lowest cost solutions (each variant has obtained the lowest cost solutions in 4 cases). The third variant is SF-Cost/Count-ND as it has obtained the best cost solutions in 3 cases. The last variant is MS-Count-Cost-ND that, despite obtaining the best cost solutions in only 3 cases, it has obtained these best results in the Coronet network for the higher considered geodiversity value.

With the 4 selected method variants, a new program was created where the 4 variants are run in sequence and, at the end, all maximum of 16 upgrade solutions are given as input solutions to the removal process. Then, the four different removal orders (as described in section 4.6) were tested. The removal process is run separately for each removal order, returning among all obtained solutions the one with the lowest cost.

In the following Table 22 and Table 23, the results of these computational tests are presented.

Avail.	Order	D = 0 Km			D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	Cost	6	**548**	25	6	**542**	28	8	**1064**	115	13	**1318**	116	20	**1440**	167
	Insertion	6	**548**	25	6	**542**	29	8	**1064**	117	13	**1318**	142	14	1458	142
	Frequency Des.	6	**548**	24	6	**542**	29	8	**1064**	118	13	**1318**	117	14	1458	140
	Frequency Asc.	6	**548**	25	6	**542**	30	8	**1064**	116	13	**1318**	136	14	1458	144
	Prev. Best Sol. Cost	548			542			1064			1318			1440		
0.99999	Cost	13	1313	63	15	1356	72	21	**2173**	305	27	2481	457	22	**2375**	471
	Insertion	14	1280	63	15	1356	74	21	**2173**	307	25	**2425**	471	22	**2375**	518
	Frequency Des.	14	**1261**	63	15	1356	71	21	**2173**	315	27	2481	518	22	**2375**	466
	Frequency Asc.	12	1294	65	15	1356	74	21	**2173**	289	27	2481	442	22	**2375**	501
	Prev. Best Sol. Cost	1294			1306			2173			2481			2375		

Table 22 – Results of Removal methods for Link problem, Germany50

Avail.	Method	D = 0 Km			D = 100 Km			D = 200 Km			D = 300 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.9999	Cost	29	**11547**	1837	40	**12417**	2989	38	14751	2956	46	**15163**	4343	47	16490	4531
	Insertion	29	11644	1856	40	**12417**	3082	37	14810	2840	46	**15163**	5104	46	**16470**	4617
	Frequency Des.	29	11644	1810	40	**12417**	3128	38	14912	2874	46	**15163**	4883	47	16529	4896
	Frequency Asc.	29	**11547**	1863	40	**12417**	2999	38	14807	2830	46	**15163**	6606	47	16529	5458
	Prev. Best Sol. Cost	11547			12417			14240			15163			16478		
0.99995	Cost	50	14837	3014	55	16091	3894	58	17746	6495	60	19110	11545	60	19110	16368
	Insertion	44	14901	2975	52	16199	3842	56	17693	14546	61	19148	10763	60	19110	19419
	Frequency Des.	47	14774	2977	53	16147	3805	54	17901	7624	60	19110	10274	60	19110	14674
	Frequency Asc.	45	14854	2917	53	16187	3817	56	17975	9307	60	19339	11937	60	19339	22471
	Prev. Best Sol. Cost	14498			15914			16761			18943			18923		
0.99999	Cost	73	19953	6405	72	20860	7986	78	22965	10214	75	**23991**	11242	77	**24808**	17016
	Insertion	70	19915	6443	70	21435	7978	73	23440	10657	74	24807	12115	74	25096	21386
	Frequency Des.	65	20331	6483	71	21676	8045	72	23324	10858	77	25559	12922	76	24950	18390
	Frequency Asc.	69	20214	6436	70	21069	7962	72	23123	10706	72	24799	12315	74	25097	15791
	Prev. Best Sol. Cost	18746			20288			22603			23991			24808		

Table 23 – Results of Removal methods for Link problem, Coronet

From these results, it is shown that the efficiency of all methods is not very different. The components ordered by cost obtained the best cost solutions in 12 cases (out of the 25 cases). The components ordered by insertion order obtained the best cost solutions in 10 cases. The components ordered by frequency obtained the best cost solutions in 9 cases for the frequency descending order and also in 9 cases for the frequency ascending order. Moreover, the insertion order has improved the previous obtained cost values in 2 cases and the frequency descending order has improved the previous obtained cost values in 1 case.

5.2.5 Comparison with previous known results

The link upgrade problem has been recently addressed in Ref. [6] and, for that reason, the developed algorithms were tested on the same networks with the same geographical characteristics.

The cases used in this dissertation only differ from the cases in Ref. [6] in the Coronet network. This dissertation considers the availability values of 0.9999, 0.99995 and 0.99999 while Ref. [6] only considers the availability values of 0.9999 and 0.99999. Moreover, this dissertation considers the geodiversity values *D* = 100, 200, 300 and 400 Km while Ref. [6] considers the values *D* = 100, 200, 400 and 600 Km.

The next Table 24 and Table 25 present for all common cases the cost of the best solution (and the running time of the best variant that has obtained such cost) of the results of this dissertation and the results reported in Ref. [6]. In each case, the best between the two solutions is highlighted in the same way as in all previous tables.

Avail.	Work	D = 40 Km			D = 80 Km			D = 120 Km			D = 160 Km		
		#N	C	T	#N	C	T	#N	C	T	#N	C	T
0.99998	Ref. [6]	5	**530**	2909	11	1105	80	15	**1268**	2874	15	**1358**	8297
	This work	6	542	17	8	**1064**	40	13	1318	25	20	1440	90
0.99999	Ref. [6]	13	1342	24	23	2192	214	25	2501	7788	25	2501	14484
	This work	14	**1306**	18	21	**2173**	80	25	**2425**	471	22	**2375**	161

Table 24 – Comparison of results with Ref. [6], Germany50

Avail.	Work	D = 100 Km			D = 200 Km			D = 400 Km		
		#N	C	T	#N	C	T	#N	C	T
0.9999	Ref. [6]	42	12420	7968	40	**13940**	3423	44	**15989**	1876
	This work	40	**12417**	219	44	14240	1597	46	16470	4617
0.99999	Ref. [6]	72	21952	2397	79	23539	5332	80	25857	23024
	This work	73	**20288**	538	75	**22603**	627	77	**24808**	660

Table 25 – Comparison of results with Ref. [6], Coronet

In a total of 14 cases, the algorithms developed in this dissertation obtained lower cost solutions in 9 cases. Moreover, all these 9 best solutions were obtained with much shorter running times (in the Coronet cases, the runtime reduction is huge).

In the remaining 5 cases, the best solutions are the ones reported in Ref. [6] and all such cases were obtained by an heuristic based on an ILP (Integer Linear Programming model) which was shown in Ref. [6] that is computationally heavy. In these 5 cases, though, the best cost values obtained by the algorithms developed in this dissertation are always better than the best cost solutions of the greedy based methods proposed in Ref. [6].

6. Conclusion and Future Work

6.1 Conclusion of the Work

Telecommunication networks are a key infrastructure on which critical services are dependent upon. They must provide a high availability between end-to-end nodes and be resilient to large-scale disasters, in particular when supporting critical services.

Implementing geodiversity routing in the network allows the planning of two routing paths between the end-to-end nodes, with a high geographical distance between them. As such, it minimizes the risk of a large-scale natural disaster disrupting both paths simultaneously. Due to the possible longer paths required by geodiversity routing, the availability of the network might not reach the required values for critical services. To fulfil both the requirements of availability and geodiversity, some network components might need to be upgraded.

In this dissertation, it was considered that the network components can be improved in 2 different cases. In the first case, the link upgrade problem considers that the availability of each link can be individually improved with a given cost. In the second case, the node upgrade problem considers that the Mean Time to Repair of the links (and consequently their availability values) can be reduced by placing maintenance teams on network nodes with a given cost.

For both problems, different algorithm variants were implemented with the aim of computing a minimum cost solution such that the final network configuration supports critical services with some desired values of end-to-end availability and disaster resilience. All variants are heuristic approaches based on greedy strategies to obtain an upgrade solution with a last step to remove the selected components that are redundant at the end of the greedy phase.

The different variants can be divided into 3 main methods: (i) the Minimum Upgrade Cost with Availability and Geodiversity (MUCAG), inspired from previous works, (ii) the MUCAG with Multiple Component Selection, which is a modification of the previous method focusing on each iteration on the pair of nodes with the lowest availability and (iii) the MUCAG with Single Filtering Process, which is a modification of the previous method focusing on reducing the running times of the algorithm. The last method has the advantage of enabling a multiprocessing implementation.

As any greedy algorithm, these methods compute an upgrade solution by selecting iteratively one component at a time to be upgraded. To select the best component on each decision, different strategies were used combining the upgrade cost of the components and a frequency counter which measures the potential impact on its selection in the end-to-end availability of the node pairs. One interesting result of the computational results is that giving more importance to the cost in this decision provides much better results in the case of the node upgrade problem but does not provide the best results, on average, in the case of the link upgrade problem. This illustrates the fact that one algorithm that is good for one problem might be not good for another problem.

In the implementation of the removal process (in the last step to remove the redundant upgrade components of a greedy upgrade solution), different strategies were also implemented and tested and, overall, the computational results have shown that trying to remove components sorted by cost, from the component with the highest cost to the lowest cost, is the best removal strategy.

While the node upgrade problem has not been addressed before in the scientific literature (as far as the author in this dissertation is aware), the link upgrade problem has been recently addressed in Ref. [6] and most of the proposed methods in that work are also based on greedy strategies to obtain an upgrade solution with a last step to remove the redundant components.

Concerning the comparison of the best results reported in Ref. [6] and the best results obtained in this dissertation, the algorithms developed in this dissertation obtained simultaneously lower cost solutions and in shorter runtimes for 9 cases among a total of 14 common cases. In the remaining 5 cases, the best solutions are the ones reported in Ref. [6] which were obtained by an ILP based heuristic. In these 5 cases, though, the best cost values obtained in this dissertation are always better than the best cost solutions of the greedy based methods reported in Ref. [6].

6.2 Future Work

Due to the lack of information concerning the cost of placing dedicated teams in different network nodes, the cost values of the obtained solutions do not represent real life cases. If realistic values become known, it would be interesting to test all method variants with such information in order to review the conclusions taken between the efficiency of the different variants.

As already explained, the networks used for testing were the ones of Ref. [6]. Nevertheless, is would be also interesting to test all method variants for both the link upgrade and node upgrade problems for other known networks with different characteristics in terms of geographical coverage and average node degree (defined as the average number of links connected to each node).

Finally, the method variants implemented in this dissertation were based on greedy heuristics. Other heuristic approaches (like Hill Climbing, Simulated Annealing or Tabu search) exist which might be more efficient although more complex in terms of algorithm complexity. Such approaches might be interesting to be investigated to check if the resulting algorithms can provide better cost solutions (although maybe with longer running times).

References

[1] A. Alashaikh, D. Tipper, and T. Gomes, “Supporting differentiated resilience classes in multilayer networks,” in *Proceedings of the 2016 12th International Conference on the Design of Reliable Communication Networks, DRCN 2016*, 2016, pp. 31–38.

[2] J. Rak *et al.*, “RECODIS: Resilient Communication Services Protecting End-user Applications from Disaster-based Failures,” in *International Conference on Transparent Optical Networks*, 2016, vol. 2016-Augus, pp. 1–4.

[3] J. P. Rohrer, A. Jabbar, and J. P. G. Sterbenz, “Path diversification: A multipath resilience mechanism,” in *Proceedings of the 2009 7th International Workshop on the Design of Reliable Communication Networks, DRCN 2009*, 2009, pp. 343–351.

[4] Y. Cheng, M. T. Gardner, J. Li, R. May, D. Medhi, and J. P. G. Sterbenz, “Analysing GeoPath diversity and improving routing performance in optical networks,” *Comput. Networks*, vol. 82, pp. 50–67, May 2015.

[5] J. Zhang, E. Modiano, and D. Hay, “Enhancing Network Robustness via Shielding,” *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2209–2222, Aug. 2017.

[6] A. de Sousa, T. Gomes, R. Girão-Silva, and L. Martins, “Minimization of the network availability upgrade cost with geodiverse routing for disaster resilience,” *Opt. Switch. Netw.*, vol. 31, no. August 2018, pp. 127–143, Jan. 2019.

[7] M. Xia, M. Tornatore, S. Sevilla, L. Shi, C. U. Martel, and B. Mukherjee, “A novel SLA framework for time-differentiated resilience in optical mesh networks,” *J. Opt. Commun. Netw.*, 2011.

[8] M. Tornatore, D. Lucerna, B. Mukherjee, and A. Pattavina, “Multilayer protection with availability guarantees in optical WDM networks,” *J. Netw. Syst. Manag.*, 2012.

[9] A. Alashaikh, T. Gomes, and D. Tipper, “The Spine concept for improving network availability,” *Comput. Networks*, vol. 82, pp. 4–19, May 2015.

[10] A. Alashaikh, D. Tipper, and T. Gomes, “Embedded network design to support availability differentiation,” *Ann. des Telecommun. Telecommun.*, vol. 74, no. 9–10, pp. 605–623, 2019.

[11] T. Gomes *et al.*, “A survey of strategies for communication networks to protect against large-scale natural disasters,” in *Proceedings of 2016 8th International Workshop on Resilient Networks Design and Modeling, RNDM 2016*, 2016.

[12] D. S. Yadav, A. Chakraborty, and B. S. Manoj, “A Multi-Backup Path Protection scheme for survivability in Elastic Optical Networks,” *Opt. Fiber Technol.*, vol. 30, pp. 167–175, Jul. 2016.

[13] V. Miletic, D. Maniadakis, B. Mikac, and D. Varoutas, “On the influence of the underlying network topology on optical telecommunication network availability under shared risk link group failures,” in *DRCN 2014 - Proceedings, 10th International Conference on Design of Reliable Communication Networks*,

2014, pp. 1–8.

[14] J. P. Rohrer, A. Jabbar, and J. P. G. Sterbenz, “Path diversification for future internet end-to-end resilience and survivability,” *Telecommun. Syst.*, 2014.

[15] Y. Cheng, M. T. Gardner, J. Li, R. May, D. Medhi, and J. P. G. Sterbenz, “Optimised heuristics for a geodiverse routing protocol,” in *DRCN 2014 - Proceedings, 10th International Conference on Design of Reliable Communication Networks*, 2014, pp. 1–9.

[16] S. Trajanovski, F. A. Kuipers, A. Ilic, J. Crowcroft, and P. Van Mieghem, “Finding critical regions and region-disjoint paths in a network,” *IEEE/ACM Trans. Netw.*, 2015.

[17] A. De Sousa, T. Gomes, R. Girao-Silva, and L. Martins, “Minimizing the network availability upgrade cost with geodiversity guarantees,” *2017 9th Int. Work. Resilient Networks Des. Model.*, pp. 1–8, Sep. 2017.

[18] C. Mas, I. Tomkos, and O. K. Tonguz, “Failure location algorithm for transparent optical networks,” *IEEE J. Sel. Areas Commun.*, 2005.

[19] B. Wu, P. H. Ho, K. L. Yeung, J. Tapolcai, and H. T. Mouftah, “Optical layer monitoring schemes for fast link failure localization in all-optical networks,” *IEEE Commun. Surv. Tutorials*, 2011.

[20] J. Tapolcai, B. Wu, P. H. Ho, and L. Rónyai, “A novel approach for failure localization in all-optical mesh networks,” *IEEE/ACM Trans. Netw.*, 2011.

[21] Y. Xuan, Y. Shen, N. P. Nguyen, and M. T. Thai, “Efficient multi-link failure localization schemes in all-optical networks,” *IEEE Trans. Commun.*, 2013.

[22] M. L. Ali, P. H. Ho, J. Tapolcai, and S. Subramaniam, “Multi-link failure localization via monitoring bursts,” *J. Opt. Commun. Netw.*, 2014.

[23] H. Y. Chang and P. C. Wang, “Upgrading service availability of optical networks: A labor force perspective,” *Int. J. Commun. Syst.*, 2018.

[24] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, “SNDlib 1.0-Survivable Network Design Library,” in *Networks*, 2010, vol. 55, no. 3, pp. 276–286.

[25] “Monarch Network Architects.” [Online]. Available: http://www.monarchna.com/topology.html. [Accessed: 15-Dec-2019].

[26] A. De Sousa, D. Santos, and P. Monteiro, “Determination of the minimum cost pair of d-geodiverse paths,” *DRCN 2017 - 13th Int. Conf. Des. Reliab. Commun. Networks*, vol. 2017, pp. 101–108, 2017.

[27] C. Veness, “Calculate distance and bearing between two Latitude/Longitude points using haversine formula,” 2019. [Online]. Available: http://www.movable-type.co.uk/scripts/latlong.html. [Accessed: 15-Dec-2019].

[28] J.-P. Vassuer, M. Pickavet, and P. Demeester, *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS - Jean-Philippe Vasseur, Mario Pickavet, Piet Demeester - Google Books*. 2004.

[29] P. E. Hart, N. J. Nilsson, and B. Raphael, “A Formal Basis for the Heuristic

Determination of Minimum Cost Paths," *IEEE Trans. Syst. Sci. Cybern.*, 1968.

[30] P. Kadionik, "Introduction to Embedded Systems," *Communicating Embedded Systems*. 2013.