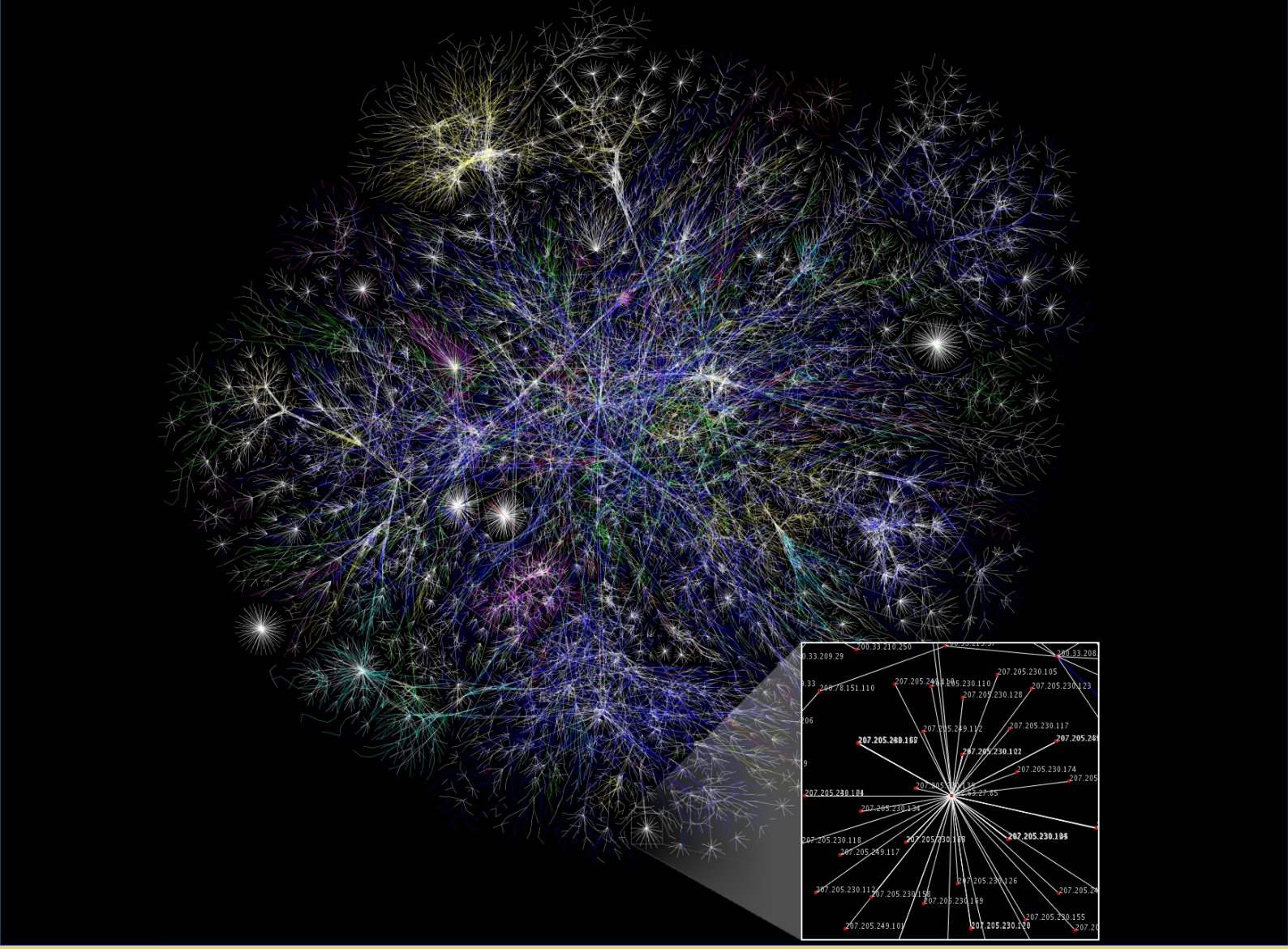


Ricardo da Silva Carvalho Mendes

Impairment-aware minimization of the number of regenerators in optical transport networks

Setembro, 2016

· U C ·

Universidade de Coimbra

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Impairment-aware minimization of the number of regenerators in optical transport networks

Ricardo da Silva Carvalho Mendes

Coimbra, Setembro 2016

Impairment-aware minimization of the number of regenerators in optical transport networks

Supervisor:

Doutora Teresa Martinez dos Santos Gomes

Co-Supervisor:

Doutor João Miguel Lopes dos Santos

Jury:

Doutora Lúcia Maria Reis Albuquerque Martins

Doutora Rita Cristina Girão Coelho da Silva

Doutora Teresa Martinez dos Santos Gomes

Dissertation submitted in partial fulfillment for the degree of Master of Science in Electrical and Computer Engineering.

Coimbra, Setembro 2016

Acknowledgements

Queria agradecer à incansável Professora Teresa, que como orientadora fez de tudo para que este trabalho tivesse sucesso.

Agradeço também ao Dr. João pelas sugestões e esclarecimentos providenciados que me permitiram enriquecer tanto o trabalho como os meus conhecimentos na área, e também à Coriant Portugal pela bolsa de estudo providenciada.

Um obrigado à Professora Lúcia, por me ter convidado a participar neste projecto e por me ter auxiliado quando necessitado.

Queria agradecer à minha família o suporte com que sempre pude contar em todo o meu percurso académico e à Adelina que me deu guarida nos últimos dias (semanas!) de trabalho.

Um muito obrigado a todos os meus amigos que mantiveram confiante e inspirado, mais nomeadamente aos que tiraram um pouco do seu tempo para me ajudar na revisão da escrita: Ana Cláudia, José Marques, Luís Henriques, Pedro Duarte, Pedro Girão e Ricardo Simões.

Agradeço por fim à Bruna Nogueira, que percorreu todo este caminho a meu lado.

Resumo

Com o aumento do consumo de largura de banda por parte das aplicações atuais, as redes de transporte precisam de suportar grandes volumes de tráfego, o que é conseguido usando *Wavelength Division Multiplexing (WDM)* em redes de transporte ótico.

A degradação da qualidade do sinal, restringe a distância máxima que um sinal pode atravessar sem sofrer regeneração. A qualidade de um sinal em redes óticas que utilizem WDM pode ser recuperado usando regeneração *Optical-Electrical-Optical (OEO)* para poder chegar ao seu destino. Tendo em conta que os regeneradores OEO são dispositivos caros, a chave para reduzir os custos de uma rede é a colocação esparsa destes dispositivos, conjugado com otimização do encaminhamento do tráfego. Nesta dissertação é abordado o problema do *Routing and Wavelength Assignment and Regenerator Placement (RWARP)* levando em conta a degradação do sinal, considerando as restrições de capacidade, com foco na minimização do número de regeneradores necessários. Este problema é $\mathcal{NP}$-Completo. Será proposta uma extensão de uma formulação existente de Programação Linear Inteira (PLI) para o problema RWARP, a qual obtem, quando exequível, uma solução ótima. É ainda proposta uma heurística eficiente para a resolução do problema anterior, a qual é seguidamente extendida, aplicando um algoritmo de distribuição de tráfego, para melhorar a sua eficiência. A heurística melhorada é capaz de fornecer resultados próximos do ótimo, na grande maioria dos testes considerados, numa fração do tempo de execução da resolução exata obtida com a formulação PLI.

Quatro variantes da formulação PLI para o problema RWARP serão comparadas, para avaliar o impacto no tempo de execução da introdução de certas restrições. A comparação destas variantes permite concluir que a introdução das capacidades de conversão do comprimento de onda nos regeneradores reduz o tempo de resolução do otimizador da formulação PLI.

Um único comprimento de onda é capaz de transmitir uma quantidade significativa de informação, e, actualmente, estão a ser colocadas fibras capazes de transportar vários Tbps. Isto torna a questão da resiliência das conexões óticas fundamental, onde a recuperação das conexões deve ser feita num tempo muito curto. Por conseguinte, mecanismos de proteção, por exemplo, a utilização, por conexão, de dois caminhos disjuntos nos riscos de falha (caminho ativo e caminho de recuperação) estão presentes nas redes de transporte. Uma vez que os cortes nas fibras são a forma mais frequente de falha em redes óticas, caminhos disjuntos nas fibras são normalmente utilizados.

Uma vez que a proteção dedicada (ao caminho) requer uma dispendiosa alocação de recursos, será utilizada recuperação pré-planeada contra falhas de uma única fibra, com a partilha de regeneradores entre caminhos de recuperação e partilha de regeneradores entre caminhos ativos e os respetivos caminhos de recuperação. Este problema pode ser identi-

ficado como o problema *Survivable Routing and Wavelength Assignment and Regenerator Placement (SRWARP)* o qual tem uma complexidade $\mathcal{NP}$-Hard.

Três variantes deste problema são definidos, os quais se distinguem pelos recursos que são (ou não) partilhados: *dedicado-dedicado*, se não há partilha; *dedicado-partilhado*, se os regeneradores de um caminho ativo podem ser reutilizados pelo caminho de recuperação correspondente; e *partilhado-partilhado*, onde para além da partilha do caso anterior há ainda a partilha da largura de banda e dos regeneradores entre os caminhos de recuperação cujos caminhos ativos são disjuntos nas fibras. Para as primeiras duas variantes, uma formulação PLI é apresentada, e para todas as variantes é proposta uma heurística eficiente. Os resultados para o problema SRWARP mostraram que a formulação PLI não é utilizável (na prática) para redes com tamanho real. Contudo, os resultados obtidos pela heurística desenvolvida são aceitáveis e obtidos num tempo relativamente curto. Como esperado, a variante *partilhado-partilhado* reduz drasticamente a capacidade necessária para suportar os caminhos de recuperação.

Os resultados deste trabalho foram ainda integrados numa heurística para a resolução de um problema de encaminhamento multi-camada, a qual foi desenvolvida no âmbito de uma outra dissertação de mestrado. Nesse contexto foi ainda implementado um um mecanismo de recuperação na camada ótica.

Palavras Chave: Colocação de Regeneradores, Sobrevivência do tráfego, encaminhamento tendo em conta a degradação do sinal, redes óticas, heurísticas.

Abstract

With the increase in bandwidth consumption of today's applications, backbone networks are required to carry great amounts of traffic, which is achieved using Wavelength Division Multiplexing (WDM) in transport optical networks.

Physical impairments limit the maximum length that a signal can travel without regeneration. The quality of a signal in optical WDM networks must thus be restored with Optical-Electrical-Optical (OEO) regeneration in order to reach its destination. As OEO regenerators are costly devices, their sparse deployment using routing optimization is the key to reduce the network cost. In this thesis, the problem of impairment aware Routing and Wavelength Assignment and Regenerator Placement (RWARP), considering capacity constraints, while focusing on minimizing the number of regenerators, is tackled. This problem is $\mathcal{NP}$-Complete. An extension to an existing Integer Linear Programing (ILP) formulation for the RWARP problem which provides, when feasible, an optimal solution, will be proposed. An efficient heuristic to solve the same problem is put forward, which is then enhanced with a traffic distribution method, to improve its effectiveness. Results show that the improved heuristic provides close to optimal results, for most of the tested cases, in a fraction of the ILP execution time.

Four variants of the RWARP ILP formulation will be compared to evaluate the impact on the execution time, of the introduction of certain constraints. The comparison of these variants led to the conclusion that the introduction of wavelength assignment capabilities in regenerators speeds up the execution time of the optimizer running the ILP formulation.

A single wavelength can carry a significant amount of information and fibres capable of carrying several Tbps are being deployed. This makes the resilience of the optical connections an important issue, where recovery should take place in a very short time. Therefore, protection mechanisms, for example, the use of two risk disjoint paths (the active and backup path) for the same connection, are present in transport networks. Since fibre cuts are the most frequent form of failure in Optical Network (ON), fibre disjointedness is usually required between paths.

Since dedicated (path) protection requires expensive resource allocation, pre-planned path recovery against single fibre failure, with regenerator sharing between backup paths and between an active path and its corresponding backup path, will be the explored recovery approach. This problem may be called the Survivable Routing and Wavelength Assignment and Regenerator Placement (SRWARP) problem and has been shown to be $\mathcal{NP}$-Hard. Three variants of this problem are defined, differing on the shared resources: *dedicated-dedicated*, if nothing is shared; *dedicated-shared*, if regenerators from an active path may be re-used for the corresponding backup path; and *shared-shared*, in which besides regenerator sharing between an active path and its backup, sharing of both bandwidth and regenerators is pos-

sible between backup paths whose active paths are fibre disjoint. For the two first variants, an ILP formulation is given, and for all variants, an efficient heuristic is proposed. Results for the ILP formulation of the SRWARP problem showed that it is impractical for real size networks. Nevertheless, the heuristics showed acceptable results in a relatively short amount of time. The *shared-shared* variant greatly reduces the amount of capacity required by the backup paths, as expected.

The results of this work were integrated with a multi-layer grooming heuristic developed in the context of another master thesis. A recovery mechanism at the optical layer was also considered.

Keywords: Regenerator placement, traffic survivability, impairment aware routing, optical networks, heuristics.

“There is nothing noble in being superior to your fellow man;

true nobility is being superior to your former self."

— Ernest Hemingway

Contents

List of Acronyms

ON	Optical Network
OADM	Optical Add/Drop Multiplexer
ROADM	Reconfigurable Optical Add/Drop Multiplexer
WDM	Wavelength Division Multiplexing
CWDM	Coarse Wavelength Division Multiplexing
DWDM	Dense Wavelength Division Multiplexing
OXC	Optical Cross Connect
ROADM	Reconfigurable Optical Add-Drop Multiplexer
BER	Bit-Error Rate
OEO	Optical-Electrical-Optical
CAPEX	*Capital Expenditure*
OPEX	*Operational Expenditure*
RWA	Routing and Wavelength Assignment
SLE	Static Lightpath Establishment
ILP	Integer Linear Programing
MILP	Mixed Integer Linear Programing
BAP	Branch And Price
RPP	Regeneration Placement Problem
RRP	Routing with Regeneration Problem

FF	First-Fit
RWARP	Routing and Wavelength Assignment and Regenerator Placement
ESRRP	Exact Single Request Regenerator Placement
REPARE	**RE**generator **P**lacement **A**nd **R**outing **E**stablishment
SRWARP	Survivable Routing and Wavelength Assignment and Regenerator Placement
GRWAR	Grooming and Routing and Wavelength Assignment with Regeneration

List of Figures

List of Tables

List of Algorithms

1 Introduction

1.1 Contextualization

With the increase in bandwidth consumption of today's applications, backbone networks are required to carry great amounts of traffic which is achieved using Wavelength Division Multiplexing (WDM) in the transport optical network.

In a transparent optical network, the signal is transmitted in the optical domain, without any electrical conversion, from a source to a destination. However, the maximum reach of a signal is limited due to physical impairments (e.g. noise and distortion) along the fibre requiring regeneration of the signal, which results in translucent networks. In these networks, an optical path is a sequence of lightpaths (transparent unregenerated segments of the path).

In WDM networks, an optical fibre is able to transport multiple wavelengths. Therefore, routing in these networks involves the calculation of end-to-end optical paths and the assignment of the same wavelength, in all the arcs of the path, when no wavelength conversion is available. This is the Routing and Wavelength Assignment (RWA) problem which is known to be $\mathcal{NP}$-Complete [2].

There are several types of impairments and different metrics that may be considered for regenerator placement. As distance is one of the most relevant parameters for determining the quality of a signal [3], in this work, the impairment threshold will be the maximum allowed length (optical reach) for a transparent lightpath.

The main cost of a network is associated with the electronic and opto-electronic devices such as regenerators [4] which also contribute to the network power consumption. This raises the problem of designing a network such that all demands are routed with adequate quality, while minimizing the number of required regenerators. Earlier works addressed the problem of minimizing the number of nodes with regeneration. More recently, the problem has been to minimize the number of per-wavelength regenerators [24].

In a single fibre, with WDM, Tbps of data may be transmitted as each wavelength can

carry a significant amount of information (100Gbps). This makes resilience of the optical connections an important issue, and recovery should take place in a very short time. Therefore, protection mechanisms, for example, the use of two risk disjoint paths (an active and a backup path), for the same connection, are present in transport networks. Fibre cuts are the most frequent form of failure in Optical Network (ON) [26], therefore fibre disjointedness is usually required between paths.

Considering dedicated (path) protection consumes too much resources, planned path recovery against single fibre failure, with regenerator sharing between backup paths and between an active path and its corresponding backup path will be the explored recovery approach. This problem has been shown to be $\mathcal{NP}$-Hard [3].

This work considers a static set of demands, that is, the complete set of demands requiring an optical path is known beforehand. The RWA problem will be solved considering wavelength conversion will only take place at the regenerators and the objective function is the minimization of the total number of deployed per-wavelength regenerators. It is also assumed that the total network capacity should be sufficient to transport all demands, i.e. an optical path for each request is possible such that no demand is blocked.

1.2 Objectives

Three problems will be addressed in this work:

- RWARP – Placement and minimization of the number of regenerators deployed in a WDM network to optically route a given static set of demands;
- SRWARP – An extension to the RWARP problem, considering path-based recovery, where the minimization of the number of regenerators is still the objective function;
- Integration of RWARP in a multi-layer grooming heuristic proposed in [20] and subsequent protection of the obtained optical routing solution.

The first objective is to obtain efficient heuristics for solving the RWARP and the SRWARP problems, and its evaluation in small problems using optimal solutions obtained solving adequate ILP formulations. The second objective is the integration of the results of this work with a multi-layer grooming heuristic developed in the context of another MSc. Thesis [20], with the subsequent addition of survivability.

1.3 Main Contributions

The contributions of this work are:

- Firstly, an efficient heuristic for the RWARP problem was proposed in [18][1] and is presented in subsection 3.2.4;
- Secondly, a more effective heuristic was developed for the RWARP problem and can be found in subsection 3.2.5;
- Thirdly, the ILP formulation [3] for solving the SRWARP problem (with unlimited fibre capacity and unidirectional traffic) was adapted to solve the RWARP and the SRWARP problem with capacity constraints, wavelength conversion capability and bidirectional traffic;
- Fourthly, an efficient heuristic for solving three variants of the SRWARP problem was proposed in subsection 3.3.3;
- Finally, the results of this work were integrated with a multi-layer grooming heuristic developed in the context of another MSc. Thesis [20], as described in section 3.4, where a segment-based recovery scheme was used.

In the context of the SRWARP problem, three variants are considered: *dedicated-dedicated*, *dedicated-shared* and *shared-shared* variants. In the first two variants, no bandwidth can be shared between backup paths, however, in *dedicated-shared* a backup path may re-use its primary path regenerators. The third variant extends the second variant allowing backup paths whose primary paths are fibre disjoint, to share bandwidth and regenerators. The first two variants were solved exactly using an ILP formulation and also heuristically. The third variant was solved using only a heuristic approach.

1.4 Document Structure

This document is organized as follows. In chapter 2, the context of application of the proposed heuristics and a brief literature review can be found. In chapter 3, notation is introduced followed by the descriptions and approaches to the three addressed problems: RWARP, SRWARP and the addition of survivability to a multi-layer heuristic. The results

[1]The corresponding accepted paper can be found in appendix F.

are presented and discussed in chapter 4. Finally, conclusions and further work are put forward in chapter 5.

2 State of the Art

2.1 Backbone Networks

Core networks interconnect multiple sub-networks by providing high capacity and high throughput communications. These networks, also known as backbone networks, may span an entire nation or even interconnect nations and are generally arranged in a mesh topology.

Optical WDM networks are established as the core of optical transport networks to face the increasing demand of today's applications. In this type of networks, the signals travel through lightpaths which may span multiple fibres on a given wavelength. Each fibre may carry multiple signals by multiplexing each into a different wavelength. The capacity of a network arc is defined by the wavelength bandwidth and the number of wavelengths that a fibre carries. Coarse Wavelength Division Multiplexing (CWDM) and Dense Wavelength Division Multiplexing (DWDM) have been defined as WDM wavelength frequency grids, where the former provides up to 16 wavelengths per fibre while the latter allows for over 160 channels [6].

Mesh topologies in optical transport networks became possible due to the advances in the optical nodes architecture. Optical Cross Connects (OXCs) are devices which allow high speed switching in the optical domain to route the lightpaths. Upon reaching a node, a propagating signal may either pass through it or be extracted. Optical Add/Drop Multiplexers (OADMs) allow wavelengths to be either *added* or *dropped* at the node or even to bypass the node i.e. passing signals may be optically switched to follow their respective routes while other signals are inserted or removed at the OADM.

Since the introduction of the OADM, improvements have been made to the architecture of these components [12]. State of the art OADMs are simultaneously reconfigurable, colourless, directionless and contentionless. Reconfigurability allows for software defined add-and-drop switching which is a major contribution for the mesh topology by allowing optical channels to be switched to any adjacent node instead of following a predefined linear or ring con-

figuration. Directionless is the property which allows any wavelength to be added/dropped into/from any port of the OADM. A colourless OADM allows any colour (wavelength) to be added/dropped in any of the OADM ports. Contentionless allows multiple copies of the same wavelength on a single add/drop structure.

A network node containing Reconfigurable Optical Add-Drop Multiplexers (ROADMs), which are simultaneously directionless, contentionless, and colourless, present no limitation while routing lightpaths. However, due to the high cost of these devices, they are not present at every network node. Some of the network nodes may not have the ability to do wavelength conversion, in which case all the arcs of an optical path between those nodes must use the same wavelength. This is designated as the wavelength continuity constraint.

2.2 Routing in optical transport networks

Routing in WDM networks consists of finding an optical path (sequence of lightpaths) for a given set of demands (static routing) or for real-time incoming demands (dynamic routing). This problem is commonly referred to as RWA, where after finding a physical (topological) path to a demand, wavelength assignment must take place in order to define the composing lightpaths. The RWA problem is known to be $\mathcal{NP}$-Complete and may be decomposed into two sub-problems: the routing problem and the wavelength assignment problem [2, 36].

To route information through the network, a path (or multiple paths) must be found from a source node to a destination node, according to a certain criteria. Thus, the routing sub-problem may be formulated by assigning a cost (or weight) to each arc which may reflect the distance, the capacity or other useful metric. Hence, path selection is usually done minimizing the path cost, which is a function of the arcs weight. According to Azodolmolky *et al.* in [2], routing algorithms included in heuristics are generally based on a shortest path routing algorithm such as the Dijkstra's Shortest Path Algorithm [7]. Yen's K shortest path algorithm [35] where the K shortest paths from a source to a destination are obtained, is also frequently used in this context.

After routing the demands, lightpaths need to be formed, respecting the wavelength continuity constraint (when wavelength conversion is not a possibility). For this purpose several heuristics have been proposed, as for instance: Random assignment, First-Fit (FF) and Best-Fit (according to a criterion) [36]. Random assignment selects a random wavelength from the set of free wavelengths on a segment. FF on the other hand, selects the first free wavelength from the same set. The Best-Fit selects the wavelength according to some useful

metric. Results in [36] show that even though FF is not optimal, it is a good compromise between the simplicity of implementation and its effectiveness.

In ONs, physical impairments degrade the signal quality as the propagated distance increases. Multiple impairments have been defined, modelled and categorized as either linear or non-linear [23, 30]. Rai *et al.* in [25] stated that most impairments may be modelled as additive link metrics. Nonetheless Beshir *et al.* in [3] argued that it is sufficient to consider the worst impairment as a single metric. Moreover, they also consider, as an alternative, the use of a metric reflecting the distance that a signal can travel, because distance is strongly correlated with the signal quality.

The signal degradation that occurs at the wavelength level may be measured by the Bit-Error Rate (BER) which is defined as the number of bit errors per unit of time. A propagating signal may require regeneration before the impairments lead to an unacceptable BER. The optical reach has been defined as the maximum distance a signal can travel before its quality degrades to an unrecoverable level and as a result, regeneration is required before reaching this distance.

Per-wavelength 3R regenerators (selective regeneration) are commonly deployed in WDM ONs where "3R" stands for re-amplification, reshaping and re-timing which are the three operations required to fully restore the signal quality.

Since all-optical regenerators are under research, Optical-Electrical-Optical (OEO) regenerators are still the practical choice in WDM networks [17, 28]. In an OEO regenerator, the signal is first converted from the optical to the electrical domain, so that the regeneration is made at the electronic level, and then converted back to the optical domain to continue its path.

Besides selective regeneration, designated regeneration sites may be also deployed at the nodes of WDM networks. In these facilities, a node provides regeneration to all wavelengths, being equipped with numerous regenerators for that purpose.

Earlier studies on RWA considered ideal physical layer conditions [34, 36, 38], i.e. impairments were not taken into consideration. This assumption was acceptable for networks of reasonable length (e.g. metropolitan areas). Nowadays, core networks may span through large geographical areas and even though the optical reach may go up to 4000 km, it is still insufficient in certain networks [27]. Thus, regeneration is required when designing the network architecture. By definition, a lightpath is an unregenerated segment in the optical domain, nevertheless in the literature, two different types of lightpaths have been defined [3, 8]: lightpaths that traverse regenerators, and consequently undergo OEO conversions, are referred

to as translucent lightpaths, whereas unregenerated lightpaths are referred to as transparent lightpaths. As translucent lightpaths traverse OEO converters, different wavelengths may be assigned at the output of these devices. Converting an input wavelength into a different output wavelength is known as wavelength conversion. Thus, a translucent lightpath may have different wavelengths throughout its path. Transparent lightpaths on the other hand, have the same wavelength from end-to-end.

In more recent RWA studies, physical impairments are considered, where both multiple [8, 11, 15, 25, 37] and single [3, 23, 24] impairment metrics are taken into account. This type of routing is referred to as Impairment Aware RWA. Throughout the thesis, RWA will refer to the impairment aware variant.

2.3 RWA and Regeneration Placement

Beshir *et al.* in [4] state that the dominant cost in a network is associated with the electronic and opto-electronic devices such as regenerators. Besides being expensive, these OEO devices have a high power consumption, contributing thus for both the *Capital Expenditure* (CAPEX) and *Operational Expenditure* (OPEX) of the networks. Consequently, there is a practical interest in minimizing the number of nodes with regenerators or more recently in minimizing the number of regenerators deployed in the network.

Regenerator placement is considered to be a network design problem and thus, many approaches have been studied. These approaches differ on whether the objective should be to minimize the number of nodes with regeneration or to minimize the number of regenerators. Regeneration Placement Problem (RPP) is one of these approaches, where the objective is to minimize the number of nodes with regeneration such that a signal may travel from any source node to any destination node. In RPP, routing of demands may be a requirement [37] but some formulations of this problem, as already mentioned, simply seek to ensure that each demand has a feasible path from source to destination [22, 23]. Other approach, known as Routing with Regeneration Problem (RRP), considers that the network has regeneration already in place and the objective is to optimally route a set of demands using the minimum number of regenerators [15]. In [25] the authors rank nodes for the purpose of regenerator placement and then explore the impact on network performance of considering an increasing number of nodes with regenerators.

The combination of routing in WDM networks with regenerator placement (henceforth referred to as the RWARP problem) has as objective the minimization of either the number

of deployed regenerators or the number of regenerator nodes while optically routing a given set of demands [3, 15].

In [24], Rahman *et al.* demonstrate intuitively that the impairment aware routing in optical networks should focus on minimizing the number of regenerators rather than minimizing the number of nodes with regeneration as by doing so, the total number of regenerators can be minimal.

The RWARP problem contains the RWA problem, which is considered to be $\mathcal{NP}$-Complete. In fact, it is proven that the RRP approach is $\mathcal{NP}$-Hard (cf. [9]). Consequently, literature studies often tackle the problem by presenting an optimal (under certain assumptions) ILP formulation, followed by an efficient heuristic.

In [37], Zhang *et al.* propose a novel ILP formulation which seeks to minimize the number of regeneration nodes while routing the demands, considering two different impairment costs at each link. In order to simplify the formulation, they pre-compute an auxiliary graph $\mathcal{G}'$ from the original network graph $\mathcal{G}$. $\mathcal{G}'$ contains all nodes from $\mathcal{G}$ and an edge from every node u to every other node v, if a path from u to v exists and satisfies the signal quality constraints. In this conditions, nodes u and v are said to be neighbour nodes. If two nodes are not neighbour nodes, then regeneration is required to connect them in $\mathcal{G}$. Even though this graph transformation allows for a simpler ILP formulation, it is still not suitable for large networks. Thus, Zhang *et al.* also provide an efficient heuristic where good results were obtained.

Rahman *et al.* in [23] presented a RWARP Branch And Price (BAP) approach, where they introduce a new arc-chain approach, to solve a Mixed Integer Linear Programing (MILP)[1] formulation for the RPP variant. Results showed that the implemented BAP approach required less than 1% of the execution time of CPLEX solving a classical node-arc formulation for the same problem, in all cases. In fact, CPLEX results were only achievable in networks up to 14 nodes. Attempts to solve larger networks either timed out at 24 hours or ran out of memory.

Kuipers *et al.* in [15] sought to minimize the number of regenerators deployed, considering multiple impairments, with an algorithm called *Exact Single Request Regenerator Placement (ESRRP)*. They also assumed that there were enough wavelengths in any arc such that both the wavelength continuity constraint and the capacity constraint were relaxed i.e., a free wavelength always exist in any arc at any time. With this assumption, they may treat

[1]Note that a MILP is an ILP where some of the variables are indeed integers while other variables may not be integers.

each request independently and thus, prove that the ESRRP is exact/optimal. For the ESRRP, an auxiliary graph similar to the graph transformation presented by Zhang *et al.* in [37] is used. This *reachability* graph, which contains all nodes from the original graph, is created with segments (which in ESRRP are the shortest paths) from every node u to every reachable (w.r.t. impairments) node v. Each of this segments is a (logical) arc of cost 1 in the auxiliary graph $\mathcal{G}'$. A shortest path in $\mathcal{G}'$ corresponds to a path in $\mathcal{G}$ with minimal regeneration requirements between both ends. The cost of a path in $\mathcal{G}'$ subtracted by 1 is the number of intermediate nodes (in $\mathcal{G}'$) and therefore is the number of regenerators required by the path. In this thesis, the term regenerator node will be used to refer to a node with at least one regenerator, as seen in [3].

2.4 Traffic Survivability and the RWARP

Failures at the physical level (fibres and/or nodes) account for a considerable number of the networks failures and can cause large amounts of traffic to be lost. Since that at the optical level, channels are multiplexed in wavelengths, and even a wavelength may have multiple client connections (traffic grooming), recovery at this level is cheaper and easier to manage than recovering each affected connection individually [33].

In a survivable context, the path that carries traffic before failure is named active or primary path whereas the path that carries the traffic after failure is called backup path. Depending on how a backup path is computed, two distinct survival schemes are defined:

- *Protection scheme:* the backup path is precomputed and fully signalled before the failure. In this scheme, protection may be 1:1 or 1+1, where 1 active path is protected by 1 backup path; another possibility is M:N, where M (M≥N) active paths are protected by N backup paths. In 1:1 and M:N protection, backup paths are only used upon failure, whereas in 1+1 protection the data is duplicated and sent to the primary and backup path simultaneously. In all cases, the backup capacity is fully reserved, granting that recovery is possible after failure. Protection schemes allow for fast recovery, where sub-50ms reaction time must be achieved;

- *Restoration scheme:* the backup path may be either preplanned or dynamically allocated after failure. Nevertheless, when a failure occurs, additional signalling is required to establish the restoration path. In a preplanned restoration, the backup capacity is reserved, guaranteeing that foreseen failures are recovered. In dynamic restoration, re-

covery may not be possible at all times. Nevertheless, dynamic allocation may provide restoration to unforeseen failures. Since signalling is required, this scheme is generally slower than protection.

Table 2.1 illustrates the common variations of the restoration and protection schemes. The "Cross-Connection on Backup Route" column refers to the required configuration at the OXCs to successfully re-route the traffic through the backup path.

Table 2.1: Comparison of some typical types of recovery. Table transcribed from [33]

	Backup Route Calculation	**Wavelength Assignment on Backup Route**	**Cross-Connection on Backup Route**
Restoration	Preplanned	Preplanned	After Failure
	Preplanned	Dynamic	After Failure
	Dynamic	Dynamic	After Failure
Protection	Preplanned	Preplanned	Before Failure

Depending on the recovery scope, three different survivable categories exist: *path-based*, *link-based* or *segment-based*. In the path-based survivable category, the active path is fully protected by a backup path which may be either link disjoint or node disjoint. Link disjoint paths ensure link failure recovery, whereas node disjoint paths protect from both link and node failures. In case a backup path cannot be found, the request is blocked. Link-based survivability protects every link by assigning a local route between the two ends. Finally, in the segment-based category, segments of the active path are protected with alternative (backup) segments. Path protection requires less capacity than link protection but it is also slower since signalling is required between both ends of the path instead of both ends of the link. The segment-based scheme is a compromise between the first two.

Protection schemes may be further categorized based on whether backup paths may or may not share resources [13]. In a *shared protection scheme*, backups whose active paths are link disjoint may share resources (assuming nodes do not fail). On the other hand, in a *dedicated protection scheme*, resources are exclusively reserved for a path request[2], i.e. no sharing is allowed. In preplanned restoration, protection resources are also pre-allocated to ensure recovery [19]. These resources may also be dedicated or shared (as long as the corresponding active paths are link disjoint) among the backup paths. Note that shared protection, as can be found in [13], could also be designated as shared preplanned restoration due to the fact that the backup paths can only be fully established after fault detection.

[2]Note that the 1+1 scheme is always dedicated since both paths are continuously in use, making it impossible to share any resources.

Since single link failures are the predominant form of failure in ON (cf. [26]), recovery models for this case are often studied [3, 13]. In order to ensure traffic survivability on link failure, link-disjointedness is required between the primary and the backup path/segment. Kuipers in [14] identifies several disjoint paths problems, whose objective is to find two disjoint paths:

- *Min-Sum* – The sum of the disjoint paths' weight (cost) is minimized;
- *Min-Max* – The maximum weight path between the two paths is minimized;
- *Min-Min* – The minimum weight path between the two paths is minimized;
- *Bounded* – Primary path weight must be less or equal to a value Δ_1 and backup path weight must be less or equal to a value Δ_2;
- The conjunction of two of these variations, where one is the main objective and the other resolves the ties (secondary objective). Example: The Min-Sum Min-Max link disjoint paths problem seeks to minimize the total cost of the pair of paths and in case of a tie, the cost of the longest path is minimized.

The Min-Sum approach is suitable for the 1+1 protection scheme, where traffic travels through both paths. The Min-Min variant, on the other hand, is useful for the 1:1 protection scheme, where the backup path is only used in case of failure in the primary path. The Min-Max approach may be used if it is desirable to have both paths of similar cost. The Min-Sum problem is the only one which is easily solvable (see [5, 31, 32]).

Suurballe in [31] presented an algorithm for the Min-Sum disjoint paths problem which computes k link (or node) disjoint paths (of minimal additive cost) between a source and a destination node in a given network using k shortest path computations. This algorithm was later extended by Suurballe and Tarjan to allow to compute two disjoint paths from a source to every other node in the network using two shortest path computations [32]. Both these algorithms make use of (a variation of) the Dijkstra's shortest path combined with graph transformations.

Bhandari considers Suurballe's algorithm to be of complex implementation for practicing engineers. In fact, in [5] he presents a practical algorithm for the Min-Sum disjoint paths problem for finding $k \geq 2$ disjoint paths, also using a modified version of Dijkstra's algorithm and simple graph transformations. He also notes that computing the shortest path, removing this first path arcs from the graph, followed by the computation of the second shortest path in the modified graph (which is in fact a Min-Min heuristic) is not optimal for the Min-Sum

problem. He also illustrates that this heuristic may fail to generate pairs of disjoint paths when such paths do exist (the trap problem). Nevertheless, in some cases, this Min-Min approach provides a better solution for the regeneration placement problem in a survivable context. Also in [29], for solving a survivable all-optical routing in WDM networks with physical impairments problem, the authors prefer to use a Min-Min based approach, due to its flexibility and low computational cost, instead of using a Min-Sum approach.

Survivability has also been studied as an extension to the RWARP problem [3,11]. In this particular problem (which shall henceforth be referred to as SRWARP), the objective is to find a pair of disjoint optical paths (link or node disjoint depending on what type of failure the recovery scheme aims to restore) such that the number of regenerators (or regeneration nodes) is minimized. Bandwidth and regenerators between backup paths may be shared if certain conditions are met - *shared* variant - or not shared - *dedicated* variant. If the backup paths are only used after the failure occurs (such as in the restoration and in 1:1 protection schemes), regenerators from an active path may also be used by the corresponding backup path.

Beshir *et al.* in [3] presented an SRWARP approach for both *dedicated* and *shared* protection schemes. To optimally minimize the number of regenerators, an ILP formulation is presented for the dedicated scheme, under two variants. The first variant is completely dedicated – no bandwidth nor regenerators sharing – and the second variant (which is labelled as *dedicated-shared*) allows regenerator sharing between the backup path and its respective active path. Besides being a dedicated scheme, the assumption that enough wavelengths always exist in any arc of the network (no capacity constraints), allowed the formulation to be run independently for each of the requests. Thus, the optimizer can efficiently compute a solution for each demand sequentially. Nevertheless, for larger networks, the number of variables and constraints may still require a lot of computational time and memory. Thus, Beshir *et al.* [3] also present an efficient heuristic adequate for both dedicated variants; additionally they also propose a heuristic for sharing regenerators between backup paths. In these heuristics, obtaining a link disjoint pair of paths takes a similar approach to Bhandari's and Suurballe's algorithms. Similarly to the ILP formulation, the heuristics also assume enough capacity at all times.

Another approach which does not consider routing and wavelength assignment is presented by Rahman *et al.* in [22]. In this approach, a survivable RPP problem is studied where all faulty scenarios (single or multiple node/link failures) can be handled. Recall that the objective of RPP is to identify the minimum number of regenerator nodes so that every

pair of nodes can establish a lightpath. In a survivable RPP, a connection between every pair of surviving nodes must exist after a failure occurs. Since no routing is involved, no wavelength assignment is taken into account and consequently no capacity is considered. An ILP formulation is presented to solve this problem, nevertheless its complexity invalidates the direct use in an optimizer as CPLEX. Thus they present an efficient *branch-and-cut* algorithm that, using the ILP formulation, provides an optimal solution.

3 Minimizing the number of regenerators in transport WDM networks

3.1 Notation

Throughout the thesis the following notation will be used. A directed graph is defined as $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of n nodes composing the graph and $\mathcal{A}$ the set of m arcs connecting the nodes. An arc l can be referred to as an ordered pair $l = (u, v)$ where u and v are the nodes connected by the arc, with $u, v \in \mathcal{N}$ and $(u, v) \in \mathcal{A}$. In $\mathcal{G}$, (u, v) is an *outgoing* arc of node u and an *incoming* arc of node v and thus node v is *adjacent* to node u. Node u and v are the tail and head of arc (u, v), respectively. Consequently the set of outgoing arcs, known as *arc adjacency list*, from node u is defined as $\mathcal{A}^+(u) = \{(u, v) \in \mathcal{A} : v \in \mathcal{N}\}$. $\mathcal{N}^+(u)$ refers to the set of nodes adjacent to node u (*node adjacency list*) defined as $\mathcal{N}^+(u) = \{v \in \mathcal{N} : (u, v) \in \mathcal{A}\}$. Similarly, $\mathcal{A}^-(u)$ is the set of incoming arcs to node u and is defined as $\mathcal{A}^-(u) = \{(v, u) \in \mathcal{A} : v \in \mathcal{N}\}$. Furthermore, and considering WDM networks, each arc l has a capacity of $W_l = W$ wavelengths and an occupied capacity, in terms of wavelengths, of $\Phi_l \leq W$. The cost of using a wavelength of an arc $l = (u, v)$ is c_{uv} or $r(l)$.

In an undirected graph (u, v) is equivalent to (v, u). Thus, instead of referencing (u, v) as an arc, on undirected graphs, the term edge or link will be used.

In $\mathcal{G}$, a topological path p (or simply a path, when there is no ambiguity[1]) from a source node s to a destination node d is defined as $p_{sd} = \langle s, u_1, \ldots, u_w, d \rangle$ where $s, d, u_k \in \mathcal{N}$ for $k = 1, \ldots, w$. The set of nodes in a path p_{sd} shall be referred to as $\mathcal{N}_{p_{sd}}$ and the set of arcs $\mathcal{A}_{p_{sd}}$. A path p_{sd} may also be defined as a concatenation of sub-paths such that $p_{sd} = p_{su} \diamondsuit p_{ud}$, that is it coincides with path p_{su} from s to u and with p_{ud} from u to d. The number of arcs in p_{sd} shall be called $|\mathcal{A}_{p_{sd}}|$ and the cost of the path $c(p_{sd}) = \sum_{(u,v)\in p_{sd}} c_{uv}$.

[1]The designation path will be used loosely to refer to topological lightpaths or optical paths when its precise meaning can be easily deducted from context.

A set of paths is referred to as P where, more specifically, P_{sd} is a set of paths from a node $s \in \mathcal{N}$ to a node $d \in \mathcal{N}$.

Let i designate an end-to-end demand (or request); $s_i \in \mathcal{N}$ and $d_i \in \mathcal{N}$ will be often used as the source node and the destination node of the demand i, respectively. The end-to-end path of a demand i is referred to as $p_{i,sd}$.

A translucent lightpath will be used to refer to a node-to-node path wherein traversed nodes may apply electrical conversion for regeneration of the signal quality and wavelength conversion. Transparent lightpaths, on the other hand, are fully optical paths that may be also referred to as the unregenerated segments of the translucent lightpaths. Unless stated otherwise, a lightpath will refer to the transparent lightpath and will be represented by $\xi_{sd} = (p_{sd}, \lambda_{sd})$, that is, the topological path p_{sd} from a source s to a destination d and a wavelength λ_{sd} representing, respectively, the physical path of the lightpath and the assigned wavelength (common to all arcs of the transparent lightpath). Note that a demand (demand i for instance) will be served by a lightpath or by a sequence of consecutive transparent lightpaths or, in other words, an optical path $p^{\lambda}_{i,sd} = \langle \xi_{sv_1}, \xi_{v_1v_2}, \ldots, \xi_{v_{w'}d} \rangle$. The topological path of $p^{\lambda}_{i,sd}$ is simply the concatenation of the topological paths of the lightpaths belonging to $p^{\lambda}_{i,sd}$, that is $p_{i,sd} = p_{sv_1} \diamondsuit p_{v_1v_2} \diamondsuit \ldots \diamondsuit p_{v_{w'}d}$. An optical path from s to d, not explicitly associated with a demand, will be designated by p^{λ}_{sd}.

Although only directed lightpaths have been mentioned, in this work the traffic demands are all bidirectional symmetric and the corresponding paths will be made of a pair of symmetrical lightpaths. Therefore, each link in the optical network will be made of two fibres, each supporting a directed arc (in opposite directions) in the optical network.

In the survivability context, a demand requires a primary/active optical path $ap^{\lambda}_{i,sd}$ and a backup optical path $bp^{\lambda}_{i,sd}$. $DP^{\lambda}_{i,sd} = (ap^{\lambda}_{i,sd}, bp^{\lambda}_{i,sd})$ will be used to refer to the pair of primary and backup paths of demand i.

If the regenerators in the intermediate nodes of a path do not have wavelength conversion capability, the end-to-end optical path must satisfy the wavelength continuity constraint, that is all links in the path must use the same wavelength. In this work, wavelength conversion will only be performed by the regenerators.

3.2 RWARP Problem

3.2.1 Problem Definition

Throughout this section we will focus on the WDM RWARP problem. Given a single fibre[2] network and a static set of demands, the objective is to route each request through the network by allocating feasible lightpaths while minimizing the number of required regenerators. We will assume that each demand fully occupies a single wavelength on each link. That is, a lightpath will be a segment of a single end-to-end path for a given request. Of course, if no regeneration is required the end-to-end path will coincide with one lightpath. We will also assume that every fibre has the same number of wavelengths. Furthermore, the metric used for the impairment of each link is the length that an unregenerated segment can have, which is bounded by the optical reach of Δ kilometres. The regeneration will be per wavelength – selective regeneration – and each regenerator is capable of converting the input wavelength to any output wavelength.

3.2.2 ILP Formulation

Beshir and Kuipers *et al.* [3] provided an exact ILP formulation for the RWARP problem with arc-disjoint dedicated path protection without wavelength conversion. Below is presented the formulation based on that work with the following modifications:

a) the need for path protection was suppressed;

b) capacity and bidirectional traffic restrictions were added;

c) replacement of a constraint to ensure that regenerators can introduce wavelength conversion capability in the network.

Indices:

$i = 1, \ldots, D$	Request ID, where D is the total number of requests
$u, v = 1, \ldots, n$	Node ID.
$l = 1, \ldots, m$	Arc ID.
$\lambda = 1, \ldots, W$	Wavelength ID.
$\mathcal{A}^-(u)/\mathcal{A}^+(u)$	Incoming/outgoing arcs of node u

[2]In fact, a fibre pair in each link to support traffic in both directions.

Binary Variables:

$x_{i,l,u,\lambda}$ Is 1 if wavelength λ on arc l is used by demand i and the last used regenerator on the path before getting to arc l is at node u. Node u can also be the source node.

$\tau_{i,u,v,\lambda}$ Is 1 if the path of demand i has a regenerator at node u immediately followed by a regenerator at node v on wavelength λ. Node u can also be the source node.

Objective:

Minimize the total number of regenerators needed on the network:

$$\sum_{i}\sum_{\lambda}\sum_{u\in\mathcal{N}}\sum_{v\in\mathcal{N}\setminus\{u\}}(\tau_{i,u,v,\lambda}) \tag{3.1}$$

Constraints:

Flow Conservation constraints:

At the source node of each demand only a single flow (for that request) can leave the node:

$$\sum_{l\in\mathcal{A}^{+}(s_i)}\sum_{\lambda}x_{i,l,s_i,\lambda}=1 \qquad \forall i \tag{3.2}$$

where s_i is the source node of the demand i.

For the intermediate nodes of each demand i, i.e. nodes that are neither the source nor the destination, the incoming flow has to match the outgoing flow regardless of that node having (or not) a regenerator for the given demand:

$$\sum_{l\in\mathcal{A}^{-}(v)}x_{i,l,u,\lambda}-\sum_{l\in\mathcal{A}^{+}(v)}x_{i,l,u,\lambda}=\tau_{i,u,v,\lambda} \quad \forall i;\forall v\in\mathcal{N}\setminus\{s_i,d_i\};\forall u\in\mathcal{N}\setminus\{v\};\forall\lambda \tag{3.3}$$

where v is the intermediate node and d_i is the destination node of demand i. Note that $\tau_{i,u,v,\lambda}$ is equal to 1 when wavelength λ is used in the segment between u and v (nodes with regeneration or u as source and v with regeneration) which is a transparent lightpath associated with demand i.

If a node v has a regenerator used by demand i, the last node with a regenerator in

the next segment should be node v:

$$\sum_{l \in \mathcal{A}^+(v)} \sum_{\lambda} x_{i,l,v,\lambda} - \sum_{u \in \mathcal{N} \setminus \{v\}} \sum_{\lambda} \tau_{i,u,v,\lambda} = 0 \qquad \forall i; \forall v \in \mathcal{N} \setminus \{s_i, d_i\} \tag{3.4}$$

v is thus the *tail* of the next lightpath and the *head* of the previous lightpath (from u to v). Note that the sum for all λ's is needed as the outgoing λ may be different from the regenerated λ, as we are assuming that regenerators may be simultaneously wavelength converters. This new constraint (w.r.t. [3]) may also be stated as: if v has a regenerator for wavelength λ required by demand i (previously regenerated/originated at an upstream node u), then an outgoing flow for demand i must exist on node v, using one of the emergent arcs from v and any available wavelength on that arc.

Wavelength constraints:

At a given arc l one wavelength λ may be used, at most, by a single lightpath:

$$\sum_{i} \sum_{u \in \mathcal{N}} x_{i,l,u,\lambda} \leq 1 \qquad \forall l \in \mathcal{A}; \forall \lambda \tag{3.5}$$

Simple Path constraints:

The end-to-end path should be a simple path, i.e. without cycles. Thus the source node of a given request should not have any incoming flow relative to that request:

$$\sum_{l \in \mathcal{A}^-(s_i)} \sum_{u \in \mathcal{N}} \sum_{\lambda} x_{i,l,u,\lambda} = 0 \qquad \forall i \tag{3.6}$$

Also at the source node and for each demand, the outgoing flow that is not originated on this node should be 0:

$$\sum_{l \in \mathcal{A}^+(s_i)} \sum_{u \in \mathcal{N} \setminus \{s_i\}} \sum_{\lambda} x_{i,l,u,\lambda} = 0 \qquad \forall i \tag{3.7}$$

For the intermediate nodes case, the incoming flow for a given request i should be either 1 or 0 (1 for the destination), that is only one lightpath of a given demand may be incident to the node:

$$\sum_{l \in \mathcal{A}^-(v)} \sum_{u \in \mathcal{N}} \sum_{\lambda} x_{i,l,u,\lambda} \leq 1 \qquad \forall v \in \mathcal{N} \setminus \{s_i\}; \forall i \tag{3.8}$$

Note that this constraint also applies to the destination node where the value of this sum will always be 1.

Impairment constraints:

Any lightpath must satisfy an impairment threshold of Δ:

$$\sum_{l \in \mathcal{A}} \sum_{\lambda} r(l) \cdot x_{i,l,u,\lambda} \leq \Delta \qquad \forall u \in \mathcal{N}; \forall i \tag{3.9}$$

with $r(l)$ the distance cost of arc l.

Capacity constraints:

Each arc has a limited number of wavelengths which can be used for the lightpaths. Therefore, the total number of lightpaths on each arc should be less or equal to the capacity of each arc:

$$\sum_{i} \sum_{u} \sum_{\lambda} x_{i,l,u,\lambda} \leq W_l \qquad \forall l \tag{3.10}$$

The previous formulation assumes that the graph is directed. If an undirected graph is to be used, it is first necessary to transform the edges into two symmetrical arcs so that the graph becomes directed. That is, each edge (u, v) is replaced by the arcs $l = (u, v)$ and $l' = (v, u)$ of equal capacity ($W_l = W_{l'}$). Note that as this study focuses on the core of the network, the assumptions that l and l' have equal capacity and that all demands are bidirectional and symmetric are realistic. Furthermore, the following *Bidirectional Traffic constraints* must be added:

$$\sum_{i} \sum_{u \in \mathcal{N}} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda}) \leq 1 \qquad \forall l; \forall \lambda \tag{3.11}$$

this constraint assures that if a given wavelength is in-use on a given arc, the symmetrical arc on the same wavelength cannot be used, as it is needed (already occupied) by the bidirectional traffic. Finally, the capacity constraint must take into account the bidirectional traffic and thus, equation (3.10) changes to:

$$\sum_{i} \sum_{u} \sum_{\lambda} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda}) \leq W_l \qquad \forall l \tag{3.12}$$

which states that the traffic that goes through l and l' is at most W_l as the remaining capacity (recall that a pair of fibres is to be considered) will be needed for the bidirectional traffic.

The wavelength tunability property of the regenerator nodes relaxes the RWARP problem

as the wavelength continuity constraint is relaxed by placing a regenerator. Nonetheless, placing a regenerator degrades the solution as the objective is to minimize the number of deployed regenerators. If no wavelength tunability is to be considered one would only have to replace equation (3.4) by equation (5) presented in [3]:

$$\sum_{l \in \mathcal{A}^+(v)} x_{i,l,v,\lambda} - \sum_{u \in \mathcal{N} \setminus \{v\}} \tau_{i,u,v,\lambda} = 0 \qquad \forall i; \forall v \in \mathcal{N} \setminus \{s_i, d_i\}; \forall \lambda \tag{3.13}$$

stating that if demand i uses a regenerator at node v for wavelength λ, then v is the last regenerator node of the outgoing flow of v on the same wavelength λ.

3.2.3 ILP Variants

In order to evaluate the ILP formulation performance, in terms of execution time, four variants were implemented and experimentally compared, with or without considering wavelength conversion capabilities and/or capacity constraints:

ILP-1 has wavelength conversion capabilities at regenerators and capacity constraints – ILP formulation presented in previous subsection;

ILP-2 has capacity constraints but no wavelength conversion capabilities – equation (3.4) is replaced by equation (3.13) from ILP-1;

ILP-3 does not have capacity constraints but has wavelength conversion capabilities at regenerators – this corresponds to suppressing equation (3.10) or (3.12) from ILP-1;

ILP-4 is the ILP formulation presented by Beshir *et al.* in [3] without: path protection; wavelength conversion; capacity and bidirectional traffic constraints. This last formulation will be used as reference.

Recall from subsection 2.4 that ILP-3 and ILP-4 could be ran independently for each demand, as no capacity constraints are considered, allowing for a substantially reduced execution time. Since the objective is to benchmark all variants, to compare possible improvements that may arise from relaxing certain constraints, equal execution conditions must be met, i.e. all demands are jointly solved.

3.2.4 RWARP Heuristic

The ILP presented in subsection 3.2.2 provides an exact solution to the RWARP problem but it takes considerable amount of time to compute it. This was the motivation to develop

Algorithm 1 RWARP heuristic – *solveRwarp*($\mathcal{G}(\mathcal{N}, \mathcal{A})$, Δ, $\mathcal{I}$)

Input: A Graph $\mathcal{G}(\mathcal{N}, \mathcal{A})$, an impairment threshold Δ and an array of demands $\mathcal{I}$

Output: $P_{\mathcal{I}}^{\lambda} = \{p_{i,sd}^{\lambda} : i \in \mathcal{I}\}$, the set of optical paths $p_{i,sd}^{\lambda}$ where i is an index of a demand from $\mathcal{I}$, and s and d the source and destination of i, respectively. A given $p_{i,sd}^{\lambda}$ may be *null*, due to the lack of residual capacity in the network

1: $P_{\mathcal{I}}^{\lambda} \leftarrow \emptyset$
2: Sort $\mathcal{I}$ ▷ By non-increasing distance of the corresponding shortest paths
3: **for** $i \in \mathcal{I}$ **do**
4: For each pair of nodes $u, v \in \mathcal{N}$, compute the shortest path p_{uv}^{*} such that $c(p_{uv}^{*}) \leq \Delta$ and each $arc \in A_{p_{uv}^{*}}$ has at least a free wavelength
5: Create a graph $\mathcal{G}'(\mathcal{N}, \mathcal{A}')$ where $\mathcal{A}'$ contains the arcs (u, v) where each arc of cost $1 + (|\mathcal{A}_{p_{uv}^{*}}|/m)$ logically represents a path p_{uv}^{*}
6: Find all paths of minimum cost $p_{sd}^{'k}$ in $\mathcal{G}'$, where k is used as an index for these paths. If no path was found, **go to** step 4 for the following demand
7: Expand the sequence of arcs of $p_{sd}^{'k}$ to the corresponding sequence of paths p_{uv}^{*} in $\mathcal{G}$ to obtain p_{sd}^{k}
8: Remove possible existing loops in all p_{sd}^{k} paths
9: Remove identical paths, if any, from the paths obtained in step 8 to obtain $\hat{P}_{sd}^{k} = \{p_{sd}^{k}, k = 1, \ldots, K_{sd}\}$ where K_{sd} is the number of the remaining paths
10: $p_{i,sd}^{\lambda} \leftarrow tryWavelengthAssignment(\hat{P}_{sd}^{k})$
11: **if** $p_{i,sd}^{\lambda} = null$ **then** ▷ Wavelength assignment attempt failed
12: $p_{i,sd}^{\lambda} \leftarrow assignWavelength(\hat{P}_{sd}^{k})$
13: **end if**
14: $P_{\mathcal{I}}^{\lambda} \leftarrow P_{\mathcal{I}}^{\lambda} \cup \{p_{i,sd}^{\lambda}\}$ ▷ Add new optical path to the solution set
15: **end for**
16: **return** $P_{\mathcal{I}}^{\lambda}$

a heuristic able to provide close to optimal results in a fraction of the time. The heuristic in Algorithm 1 is based on the ESRRP algorithm provided in [15]. *Kuipers and al.* proved that the ESRRP is exact for the routing and regeneration problem when link capacity is not taken into account (assumption that there are always enough free wavelengths) and consequently the wavelength assignment and continuity is relaxed.

Algorithm 1 illustrates the RWARP heuristic in pseudo code where conceptually three main parts may be observed: demand selection (steps 2 and 3), routing for the selected demand (steps 4 to 9) and finally the wavelength assignment and regenerator placement for the final path (steps 10 to 13). Starting at the demand selection, step 2 sorts the array of demands according to a predefined criterion. In [24], *Rahman et al.* observed that for the routing and wavelength assignment the deployment of lightpaths in a longest-route-

Algorithm 2 *tryWavelengthAssignment*$(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,\hat{P}^k_{sd})$

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and a set of candidate paths $\hat{P}^k_{sd}$

Output: An optical path p^λ_{sd} such that its corresponding topological path is an element of $\hat{P}^k_{sd}$ for which a successful wavelength assignment was possible; if the wavelength assignment fails for every path in $\hat{P}^k_{sd}$ then the output is a *null* path

1: Sort $\hat{P}^k_{sd}$ ▷ Sorted according to non-decreasing distance
2: **for each** $p_{sd} \in \hat{P}^k_{sd}$ **do**
3: Create consecutive maximum distance segments from s to d $(p_{su_1},\ p_{u_1u_2},\dots p_{u_kd})$ respecting Δ and try to assign a wavelength to each of these segments. Let $\xi_w = (p_w,\lambda_w),\ \forall w \in \{su_1,u_1u_2,\dots,u_kd\}$ if a wavelength was successfully assigned to a p_w or *null* otherwise
4: **if** $\xi_w \neq null,\ \forall w$ **then**
5: $p^\lambda_{sd} \leftarrow \langle \xi_{su_1},\xi_{u_1u_2},\dots,\xi_{u_kd}\rangle$
6: Place regenerators at the last node of each lightpath $\xi_w\ \forall w \neq u_kd$, for the assigned wavelength λ_w
7: **return** p^λ_{sd}
8: **end if**
9: **end for**
10: **return** *null* ▷ No wavelength assignment was possible

first provides, in general, better performance when comparing to the shortest-route-first and random selection. As each demand will be satisfied with a set of consecutive lightpaths (and their symmetric), sorting by the longest-route-first (w.r.t. number of arcs needed from source to destination) will be advantageous to ensure that this higher demanding requests have a free wavelength on each arc. As at step 2 the routes are still unknown, the shortest path from source to destination of each demand is used as an approximation to the distance of the final demand path. Throughout this work, the shortest path from a source to a destination in a given network was computed with Dijkstra shortest path algorithm – Algorithm 10 presented in Appendix A.

After sorting the demands in $\mathcal{I}$, the heuristic proceeds to find a feasible path from source to destination for each demand sequentially (steps 4 to 15). In step 4, a set of shortest paths from each node u to every node v that have at least a free wavelength (enough capacity for i) and with a cost (impairment value) below or equal to the threshold is obtained. This set of paths is used in step 5 to create an auxiliary logical graph $\mathcal{G}'(\mathcal{N},\mathcal{A}')$ where $\mathcal{A}' = \{(u,v) : c(p^*_{uv}) < \Delta, p^*_{uv} \in \mathcal{G}\}$, that is, $\mathcal{A}'$ contains logical arcs that connect the source u to the destination v of each path obtained in step 4. These logical arcs are set to have a cost of 1 (a hop) plus the division of the number of arcs in the path p^*_{uv}, $|\mathcal{A}_{p^*_{uv}}|$, with m (the

Algorithm 3 $assignWavelength(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,\hat{P}^k_{sd})$

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and a set of candidate paths $\hat{P}^k_{sd}$

Output: An optical path p^λ_{sd} such that its corresponding topological path is the first element of $\hat{P}^k_{sd}$ for which a successful wavelength assignment was possible; if the wavelength assignment fails for every path in $\hat{P}^k_{sd}$ then the output is a *null* path

1: Set $P'_{sd} \leftarrow \emptyset$ where P'_{sd} is an auxiliary set of logic paths from s to d
2: **for each** $p_{sd} \in \hat{P}^k_{sd}$ **do**
3: Create a reachability graph $\mathcal{G}''(\mathcal{N}_{p_{sd}},\mathcal{A}'')$ where $\mathcal{A}''$ is a set of logical arcs with a cost of 1 (hop) from every $u \in \mathcal{N}_{p_{sd}}$ to every $v \in \mathcal{N}_{p_{ud}} \setminus \{u\}$, with p_{ud} the remaining segment from u to d (that is $p_{sd} = p_{su} \diamondsuit p_{ud}$), as long as the segment from u to v has a distance lower or equal to Δ and there is at least a free common wavelength along the segment
4: Compute the shortest path from s to d, p'_{sd} on $\mathcal{G}''$ and add it to P'_{sd}
5: **end for**
6: **if** $P'_{sd} \neq \emptyset$ **then** ▷ A solution must exist if called from Algorithm 1
7: Select the minimum cost (hop count) path p'^*_{sd} from all paths in P'_{sd}
8: Expand the logic arcs of p'^*_{sd} into the segments assigning the corresponding free wavelength obtained in step 3 to obtain the lightpaths $\xi_w,\ \forall w \in \{su_1, u_1u_2, \ldots, u_kd\}$. Let $p^\lambda_{sd} = \langle \xi_{su_1}, \xi_{u_1u_2}, \ldots, \xi_{u_kd} \rangle$ be the final optical path
9: Place regenerators at the last node of each lightpath $\xi_w,\ \forall w \neq u_kd$ for the assigned wavelength λ_w
10: **return** p^λ_{sd}
11: **end if**
12: **return** *null* ▷ If called from the RWARP heuristic in Algorithm 1 it won't reach here.

number of arcs in $\mathcal{G}$). This cost value will always be greater than 1 by an increment that depends on the number of physical arcs in the path. This added increment (which is not present in ESRRP) discriminates higher hop count physical segments with the same number of logical hops which consequently may lead to a better overall network capacity usage when lower cost logic arcs are selected instead. Note that the floor of the sum of the cost of any path from s to d in $\mathcal{G}'$ will always equal the number of segments in the path and subtracting 1 from this number will equal the number of regenerators needed if no regenerator is added only to ensure wavelength assignment to create an admissible solution.

Using $\mathcal{G}'$, all minimum cost paths from s to d are computed in step 6 of Algorithm 1 to form the paths p'^k_{sd} where k is an index to these paths. In this work, the K shortest paths were computed with Ernesto Martins's and Marta Pascoal's implementation[3] of the Yen's K shortest path algorithm [35], as presented in [16]. If no path was found at this step, then

[3]The code implementation was made by Yan-Qi and can be found in https://github.com/yan-qi/k-shortest-paths-java-version.

there's a capacity shortage and no path will be assigned to the demand, the heuristic will thus return to the beginning of the cycle at step 4 for the next demand. Step 7 transforms/maps the logical arcs in p'^{k}_{sd} to the corresponding p^{*}_{uv} in $\mathcal{G}$ producing the real end-to-end paths p^{k}_{sd}. At this step loops may arise. Step 8 removes these loops but may consequently create identical paths which are then removed at step 9, to form $\hat{P}^{k}_{sd}$ – the set of candidate paths.

The final path – optical path – is selected from the set of candidate paths either in Algorithm 2 ($tryWavelengthAssignment()$) or in Algorithm 3 ($assignWavelengthWith-Conversion()$) where the wavelength assignment and regenerator placement take place. The former algorithm is faster but may fail often when the network starts to have low spare capacity whereas the latter algorithm will always provide a solution when called from the RWARP heuristic and will thus be called in case of failure of the former. Both algorithms are responsible for creating the lightpaths, assigning the wavelength and placing the respective regenerators and thus, at the end of step 13 an optical path $p^{\lambda}_{i,sd}$ was successfully computed. Having $p^{\lambda}_{i,sd}$, computing the symmetrical optical path $p^{\lambda}_{i,ds}$ is as simple as using the same wavelengths in the symmetric arcs of $p^{\lambda}_{i,sd}$. The regenerators are considered bidirectional and are thus already computed.

The $tryWavelengthAssignment()$ routine starts by sorting the input set of paths $\hat{P}^{k}_{sd}$ accordingly to the paths distances. The shortest path will be the path requiring less regeneration (distance is the impairment metric taken into account) so that every lightpath has a distance less or equal than and as close to Δ as possible. Starting with the most favourable path, in step 3 of Algorithm 2, ligthpaths are attempted to be formed by creating maximum distance (unregenerated) segments with respect to the impairment threshold and then assigned a wavelength to each of these segments. Wavelength assignment was implemented using a FF approach [36]. For efficiency purposes, each arc contains an array of bits where each index represent a wavelength and the value of 1 represents a free wavelength. To obtain the FF free common wavelength on a segment, consecutive *and* operations through the arcs of the segment are executed and at the end, the index of the first bit with a value of 1 is returned. Note that as the regenerators have wavelength conversion capabilities and at the end of the segments a regenerator is needed, each segment may have a different wavelength but a lightpath in the segment has to comply to the wavelength continuity constraint in all of its composing arcs. The arcs belonging to a given segment may have different free wavelengths and thus finding a common free wavelength to assign to the segment may fail. If wavelength assignment fails in step 3 of Algorithm 2, the for loop proceeds to the next favourable path and if all paths fail, the method shall return a *null* path – step 10. If all

segments of a path are successfully assigned a wavelength – forming the respective lightpaths – the optical path p_{sd}^{λ} is thus the sequence of these lightpaths from s to d in step 5. The routine continues at step 6 by placing regenerators for the assigned wavelength at the head of each lightpath except at the last head as it will be d, the destination node. Step 7 returns the formed path.

As one can see, the $tryWavelengthAssignment()$ routine prioritizes the distance when creating the segments without taking into account if there's a free common wavelength at each arc of a segment (albeit having at least a free wavelength on each arc). This may fail often when there's a shortage of free capacity on the arcs as the demands start to occupy some arcs in the network. The routine $assignWavelength()$ (see Algorithm 3) on the other hand tries to create the longest segments on a free common wavelength, respecting the impairment distance threshold. The $tryWavelengthAssignment()$ routine provides the same number of regenerators as the $assignWavelength()$ when there's enough capacity for the maximum distance segments to be formed. When this is not the case, the former will fail to retrieve a path but the latter will successfully compute one as Algorithm 1 ensures that after step 6 a path with enough capacity does exist. In short, Algorithm 2 is faster and will work while it is possible to respect the wavelength continuity constraint on maximum distance unregenerated segments, whereas Algorithm 3 is slower but will always provide a solution.

Algorithm 3 is the $assignWavelength()$ routine. For each path p_{sd} in the set of input paths $\hat{P}_{sd}^{k}$, a corresponding path composed of logical arcs will be created and added to the set of logical paths P'_{sd} in steps 1 to 5. A logical path p'_{sd} is created using a reachability graph $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ where $\mathcal{N}_{p_{sd}}$ is the set of nodes in the corresponding p_{sd} and $\mathcal{A}''$ is a set of logical arcs of cost 1 from every $u \in \mathcal{N}_{p_{sd}}$ to every downstream node $v \in \mathcal{N}_{p_{ud}} \setminus \{u\}$, where p_{ud} is the segment defined as $p_{sd} = p_{su} \diamondsuit p_{ud}$, where the segment from u to v in $\mathcal{G}$ have a length less than or equal to the impairment threshold and at least a free common wavelength. That is, the $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ graph connects every node u in path p_{sd} to all reachable (w.r.t. impairment threshold and common wavelength of the connecting arcs) downstream nodes. Recall that by assigning a cost of 1 (a hop) to a segment from u to v, the cost of a logic path from s to d in $\mathcal{G}''$ minus 1 will be the number of regenerators needed to traverse the path. After generating this auxiliary graph in step 3, the shortest path p'_{sd} is computed at step 4 as a set of consecutive logic arcs and added to the set of logical paths P'_{sd}. This logical path creation is repeated for each input path which will then be used to create a final path in steps 7 to 9. The final logical path p'^{*}_{sd} is selected as the minimum hop count path in P'_{sd} (see step 7

in Algorithm 3). As the objective of the RWARP heuristic is to minimize the number of regenerators, the minimum hop count path will be the most suitable path. In step 8, the logical arcs in p'^{*}_{sd} are mapped into the real corresponding segments obtained in step 3 and assigned the respective wavelength to form the lightpaths of the optical path p^{λ}_{sd}. Step 9 places regeneration as in step 6 of Algorithm 2 and the optical path is finally returned at step 10.

3.2.5 Improving the RWARP Heuristic

After analysing the results of the presented heuristic (see section 4.2), it was clear that when the shortest paths start to lack capacity for the demands, a higher number of regenerators was being deployed with Algorithm 3. That result suggested that the need for wavelength conversion often comes from the constant selection of the shortest path, requiring common segments to be part of a greater number of demand paths and therefore incapacitating the composing arcs.

In order to prevent capacity shortages, the demands must thus be routed in a manner that the traffic is distributed through the network using alternative segments/paths, that are equivalent in terms of the number of regenerators needed to the shortest path, but have lower in-use capacity. Bernard Fortz and Mikkel Thorup in [10] proposed a function for the arc cost where the arc load and the arc capacity were taken into account. Their function allowed to distribute the traffic throughout the network, increasing the possibility of supporting a greater number of demands with the same network capacity. Fortz and Thorup stated that their exact definition of the cost function is not crucial, as long as it is piece-wise linear increasing and convex. Thus, in this work, the following arc cost function, which will be referred to as $C_F(l)$ or capacity cost function, was implemented [10]:

$$C_F(l) = \begin{cases} \Phi_l & , \quad 0 \leq \dfrac{\Phi_l}{W} < \dfrac{1}{3} \\ 3\Phi_l - \dfrac{2}{3}W & , \quad \dfrac{1}{3} \leq \dfrac{\Phi_l}{W} < \dfrac{2}{3} \\ 10\Phi_l - \dfrac{16}{3}W & , \quad \dfrac{2}{3} \leq \dfrac{\Phi_l}{W} < \dfrac{9}{10} \\ 70\Phi_l - \dfrac{178}{3}W & , \quad \dfrac{9}{10} \leq \dfrac{\Phi_l}{W} < 1 \\ \infty & , \quad \dfrac{\Phi_l}{W} = 1 \end{cases} \tag{3.14}$$

The capacity cost of a path, say path p_{sd}, is simply the sum of the costs of the arcs

composing the path, that is:

$$C_F(p_{sd}) = \sum_{l \in \mathcal{A}_{p_{sd}}} C_F(l) \tag{3.15}$$

In the proposed heuristic, the distance of an arc is still essential as it bounds the unregenerated segments distance, defining thus, the regenerator placement. Therefore, $C_F(l)$ can not replace the distance as the single cost metric but it can be used to select between candidate segments or paths. In fact, the traffic distribution in this work is only applied to segments or paths which supposedly provide the same number of regenerators by selecting from equivalent candidates (w.r.t. the number of regenerators) the ones having the smallest capacity cost. This may be implemented in two different parts of the heuristic: in step 4 of Algorithm 1 and in step 1 of Algorithm 2.

In step 4 of Algorithm 1, a shortest segment is being formed between every node u to every node v as long as the segment has a free wavelength in each composing arc and a cost lower than the impairment threshold. This results in the overuse of these shortest segments causing potential arc capacity shortages. This situation may be improved with the following modification of step 4 of Algorithm 1:

- For each pair of nodes $u, v \in \mathcal{N}$, compute all paths p_{uv}^k, where $k = 1 \dots K_{uv}$ is an index to these paths, such that $c(p_{uv}^k) \leq \Delta$, $\forall k$ and each $arc \in \mathcal{A}_{p_{uv}^k}$ has at least a free wavelength. Select, from the K_{uv} paths, the final u to v path as the path whose cost, according to the Fortz capacity cost function, is minimal. That is, let p_{uv}^* be the selected path, $C_F(p_{uv}^*) \leq C_F(p_{uv}^k)$, $\forall k$.

This modification will introduce a variation in the segments created to support lightpaths. Note that between a pair of nodes $u, v \in \mathcal{N}$, all the considered paths p_{uv}^k will turn into a single lightpath (if used for the demand optical path), and thus the distribution of traffic between any of these paths won't result in deterioration of the final objective value – that is, the number of deployed regenerators.

Finally, the second change to the proposed algorithm, is implemented at step 1 of Algorithm 2, where the input candidate end-to-end paths were sorted according to a non-decreasing distance. As aforementioned, the shortest path is not the optimal choice when capacity is taken into consideration. Thus, the sorting of the paths is now done in a non-decreasing order of the capacity cost $C_F(p_{sd}^k)$, $\forall p_{sd}^k \in \hat{P}_{sd}^k$. Recall that each path of the input set of paths $\hat{P}_{sd}^k$ requires the same number of regenerators if wavelength assignment is achieved. Therefore, selecting the path with least in-use arcs capacity will contribute to

the traffic distribution between demands, due to reducing the probability of arc capacity shortages, and will increase the probability of the first path being the feasible solution.

This extension to the original heuristic considerably increases the execution time. Nonetheless, the number of regenerators is significantly reduced for networks which have a greater number of demands. There is a trade-off between the execution time and the quality of the final result. In fact, in this extension, the number of alternative paths in both updated steps may be limited to a given K as the implemented algorithm to find these paths is the Yen's K shortest paths.

Since the improved RWARP heuristic follows similar steps to Algorithm 1, the pseudo code to the former heuristic may be found in Algorithms 11 and 12 in appendix B. The improved heuristic will be referred to as *solveRwarpImproved*.

3.3 Survivable-RWARP

This section addresses the survivability of the traffic within the context of the RWARP problem for single fibre failure scenarios. The following subsections will define the SRWARP problem and present an ILP formulation and an efficient heuristic to solve this problem.

3.3.1 Problem Definition

The SRWARP problem is an extension to the RWARP problem, where end-to-end global recovery is considered. Consequently, the assumptions made in subsection 3.2.1 still apply. Nevertheless, the objective is now to find, for each request, an active path (or primary path) and a backup path, such that in a case of a single fibre failure, all affected traffic may be re-routed through the backup path, while minimizing the number of required regenerators. As end-to-end path recovery is considered, backup paths may share the same wavelength (bandwidth) and regenerators if the respective active paths are arc disjoint. Furthermore, as backup paths are only used after the link failure occurs, the respective active path's regenerators may also be used.

Throughout this section, three variations of the SRWARP problem [3] will be addressed: *dedicated-dedicated*, *dedicated-shared* and *shared-shared* variants. In the first two variants, the recovery is considered to be dedicated that is, no bandwidth can be shared between backup paths. However, the backup paths in *dedicated-shared* may re-use the primary path's regenerators, while in the *dedicated-dedicated*, nothing is shared. By contrast, in the

last variant (*shared-shared*) backup paths whose primary paths are link disjoint, may share bandwidth – in this case the same wavelength – and even regenerators.

3.3.2 ILP Formulation

Similarly to subsection 3.2.2, the SRWARP ILP formulation is based on the ILP in [3], where the modifications considered in that sub-section still apply, but now path protection is addressed. To this extent, Beshir *et al.* have proposed two variants of the ILP which differ on wheter backup paths may use primary path's regenerators or not. Below the implemented ILP for both variants is presented, where indices have the same meaning as in subsection 3.2.2. Note that all the ILP RWARP formulation's constraints are either simply extended or replicated to cover for the backup path addition in this ILP formulation and therefore, the explanations in this subsection will be brief.

In the formulation below $x_{i,l,u,\lambda}$ and $\tau_{i,u,v,\lambda}$ have the same meaning as in subsection 3.2.2.

Binary Variables:

$y_{i,l,u,\lambda}$ Is 1 if wavelength λ on arc l is used by demand i backup path and the last used regenerator on the path before getting to arc l is at node u. Node u can also be the source node.

$\psi_{i,u,v,\lambda}$ Is 1 if the backup path of demand i has a regenerator at node u immediately followed by a regenerator at node v on wavelength λ. Node u can also be the source node.

$\alpha_{i,u}$ Is 1 if a regenerator in node u is used for demand i. The regenerator may be shared between primary and backup paths of demand i. This variable is only required for the *dedicated-shared* variant.

Objective:

Minimize the total number of regenerators needed on the network. For the *dedicated-dedicated* variant:

$$\sum_{i}\sum_{\lambda}\sum_{u\in\mathcal{N}}\sum_{v\in\mathcal{N}\setminus\{u\}}(\tau_{i,u,v,\lambda}+\psi_{i,u,v,\lambda}) \tag{3.16}$$

For the *dedicated-shared* variant:

$$\sum_{i}\sum_{u\in\mathcal{N}}\alpha_{i,u} \tag{3.17}$$

Constraints:

Flow Conservation constraints:

Two flows should now leave the source flow s_i, one for the primary path and the other for the backup path:

$$\sum_{l \in \mathcal{A}^+(s_i)} \sum_{\lambda} (x_{i,l,s_i,\lambda} + y_{i,l,s_i,\lambda}) = 2 \qquad \forall i \tag{3.18}$$

The intermediate nodes flow constraint remains unchanged for the primary path and it is simply replicated for the backup path:

$$\begin{aligned} \sum_{l \in \mathcal{A}^-(v)} x_{i,l,u,\lambda} - \sum_{l \in \mathcal{A}^+(v)} x_{i,l,u,\lambda} &= \tau_{i,u,v,\lambda} \quad \text{and} \\ \sum_{l \in \mathcal{A}^-(v)} y_{i,l,u,\lambda} - \sum_{l \in \mathcal{A}^+(v)} y_{i,l,u,\lambda} &= \phi_{i,u,v,\lambda} \\ & \forall i; \forall v \in \mathcal{N} \setminus \{s_i, d_i\}; \forall u \in \mathcal{N} \setminus \{v\}; \forall \lambda \end{aligned} \tag{3.19}$$

Similarly, for the regenerator node constraint as the head of the previous lightpath and the tail of the following lightpath, the constraint is simply replicated for the backup path:

$$\begin{aligned} \sum_{l \in \mathcal{A}^+(v)} \sum_{\lambda} x_{i,l,v,\lambda} - \sum_{u \in \mathcal{N} \setminus \{v\}} \sum_{\lambda} \tau_{i,u,v,\lambda} &= 0 \quad \text{and} \\ \sum_{l \in \mathcal{A}^+(v)} \sum_{\lambda} y_{i,l,v,\lambda} - \sum_{u \in \mathcal{N} \setminus \{v\}} \sum_{\lambda} \phi_{i,u,v,\lambda} &= 0 \\ & \forall i; \forall v \in \mathcal{N} \setminus \{s_i, d_i\} \end{aligned} \tag{3.20}$$

Wavelength constraints:

With the protection, an arc's wavelength can only be used by one path, the active or the backup path:

$$\sum_{i} \sum_{u \in \mathcal{N}} (x_{i,l,u,\lambda} + y_{i,l,u,\lambda}) \leq 1 \qquad \forall l \in \mathcal{A}; \forall \lambda \tag{3.21}$$

Simple Path constraints:

Both active and backup paths should be simple paths. Therefore, at the source node of each request, the incoming flow for both paths relative to the request should be null:

$$\sum_{l \in \mathcal{A}^-(s_i)} \sum_{u \in \mathcal{N}} \sum_{\lambda} (x_{i,l,u,\lambda} + y_{i,l,u,\lambda}) = 0 \qquad \forall i \tag{3.22}$$

For both primary and active paths, at the source node and for each demand, the outgoing flow that is not originated on this node should be 0:

$$\sum_{l \in \mathcal{A}^+(s_i)} \sum_{u \in \mathcal{N} \setminus \{s_i\}} \sum_{\lambda} (x_{i,l,u,\lambda} + y_{i,l,u,\lambda}) = 0 \qquad \forall i \tag{3.23}$$

The incoming flow on an intermediate node, for each demand, can be at most one for the primary or for the backup path:

$$\sum_{l \in \mathcal{A}^-(v)} \sum_{u \in \mathcal{N}} \sum_{\lambda} x_{i,l,u,\lambda} \leq 1 \text{ and } \sum_{l \in \mathcal{A}^-(v)} \sum_{u \in \mathcal{N}} \sum_{\lambda} y_{i,l,u,\lambda} \leq 1 \qquad \forall v \in \mathcal{N} \setminus \{s_i\}; \forall i \tag{3.24}$$

Impairment constraints:

The impairment constraint is duplicated for the backup path:

$$\sum_{l \in \mathcal{A}} \sum_{\lambda} r(l) \cdot x_{i,l,u,\lambda} \leq \Delta \text{ and } \sum_{l \in \mathcal{A}} \sum_{\lambda} r(l) \cdot y_{i,l,u,\lambda} \leq \Delta \qquad \forall u \in \mathcal{N}; \forall i \tag{3.25}$$

Recall that $r(l) = W_{uv}$, with $l = (u, v)$.

Capacity constraints:

The arc's capacity is now used by both active and backup paths:

$$\sum_{i} \sum_{u} \sum_{\lambda} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda} + y_{i,l,u,\lambda} + y_{i,l',u,\lambda}) \leq W_l \qquad \forall l \tag{3.26}$$

where bidirectional traffic is already considered.

Bidirectional Traffic constraints:

These constraints now ensure that a given wavelength on a given arc is now only used for either a path or for its symmetric of the active path or the backup path. That is, only one of the four paths may use the wavelength.

$$\sum_{i} \sum_{u \in \mathcal{N}} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda} + y_{i,l,u,\lambda} + y_{i,l',u,\lambda}) \leq 1 \qquad \forall l; \forall \lambda \tag{3.27}$$

Disjointedness constraints:

The disjointedness constraints are now required as the primary path has to be arc

disjoint with the corresponding backup path. Therefore, and considering bidirectional traffic:

$$\sum_{\lambda} \sum_{u \in \mathcal{N}} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda} + y_{i,l,u,\lambda} + y_{i,l',u,\lambda}) \leq 1 \qquad \forall i; \forall l \tag{3.28}$$

Dedicated-shared constraints:

The following constraints are solely for the *dedicated-shared* variant:

$$\sum_{\lambda} \sum_{u \in \mathcal{N}} (\tau_{i,u,v,\lambda} + \psi_{i,u,v,\lambda}) \leq 2 \cdot \alpha_{v,i} \qquad \forall i; \forall v \in \mathcal{N} \tag{3.29}$$

This equation allows $\alpha_{u,i}$ to signal if a regenerator (shared or not) is needed by demand i at node u.

3.3.3 SRWARP Heuristic

The SRWARP heuristic has many common elements with the RWARP heuristic. The main difference lies on the possibility of backup paths sharing either bandwidth and/or regenerators as defined previously in subsection 3.3.1. Algorithms 4, 5 and 6 present the complete SRWARP heuristic for the *shared-shared* variant. The remaining two variants may be obtained with slight changes in certain steps of the three methods. These changes will be identified and presented throughout the heuristic explanation.

Prior to starting describing the SRWARP heuristic, some definitions must be stated in order to simplify the explanations:

- an *usable wavelength* refers to either a free wavelength or a shareable wavelength which is a wavelength that can be shared between backup paths. For the *dedicated-dedicated* and *dedicated-shared* an usable wavelength is always a free wavelength;
- a *shareable* (or *re-usable*) *regenerator* is a regenerator which can be re-used for the backup path under consideration. Note that a regenerator assigned to an active path can only be shared by its backup path. In the *dedicated-dedicated* variant, no shareable regenerator exists;
- arcs incident in a re-usable regenerator, such that they have an usable wavelength equal to the regenerated input wavelength, are designated as *preferred arcs*;

In the definition of preferred arcs, for regenerators shared by an active path and its backup, the regenerated input wavelength is determined by the active path regenerated wavelength;

in the case of regenerators shared only by backup paths, the regenerated input wavelength is determined by the backup path which originated the deployment of this regenerator;

For each demand, an iterative two step approach inspired in [19] (for solving the Min-Min problem) is used in Algorithm 4. It sequentially tries to calculate a disjoint path for each element of a set of candidate active paths obtained using the routing approach of Algorithm 11 for a single demand, which is carried out by Algorithm 13 in appendix B. Sometimes, it is not possible to obtain a disjoint path using active paths with minimal number of regenerators for a given demand. To address this issue, a relaxation of that value was considered, iteratively increasing it until a solution is found or up to a maximum allowed value $\delta_{\max}$ passed as input.

Algorithm 4, called *solveSrwarp*, implements the SRWARP heuristic where the inputs are a directed graph $\mathcal{G}(\mathcal{N}, \mathcal{A})$, an impairment threshold Δ, an array of demands $\mathcal{I}$ and $\delta_{\max}$. The set of topological candidate paths (for the active path), for demand i and a given δ ($\delta = 0, 1, \ldots, \delta_{\max}$) is returned by function *srwarpRoute* – see Algorithm 13 in appendix C. If this set is empty, the present demand has no solution (for the current δ), otherwise, the elements of the set (P_{sd}) obtained in step 6 are ordered according to the Fortz path cost function – see step 8. Note that each element of P_{sd} can correspond to an optical path (this is ensured by Algorithm 13).

The candidate optical path ($ap^{\lambda}_{i,sd}$) is calculated and the necessary resources are allocated, in step 10 of Algorithm 4, using function *crtOptPath* – see Algorithm 14 in appendix C. Then, an attempt is made to obtain the corresponding backup path using function *computeBPForAP* (Algorithm 5). If the attempt was successful, the pair of paths is stored – see steps 22 and 23. If none of the elements of P_{sd} allows to obtain a solution, δ is increased and a new attempt is made (P_{sd} is recalculated), until either a solution is found or $\delta_{\max}$ is exceeded.

Algorithm 5, as previously stated, computes a backup path ($bp^{\lambda}_{i,sd}$) for a given active path ($ap^{\lambda}_{i,sd}$) as input. It starts by creating an auxiliary graph $\mathcal{G}'(\mathcal{N}, \mathcal{A}')$, where $\mathcal{A}'$ is a subset of $\mathcal{A}$ after removing the fibres used by the bidirectional active path ($ap^{\lambda}_{i,sd}$ and $ap^{\lambda}_{i,ds}$) – see step 2. The cost of preferred arcs in $\mathcal{A}'$ is set equal to zero – see step 3 – to favour re-use of shareable regenerators by the set of paths obtained in step 4 of the algorithm. Note that the elements of this set have *a priori* a minimum number of regenerators (shareable or not). If the set obtained in step 4 has at least one path, the final backup path selection and wavelength assignment is done using function *generateFinalBP* – see Algorithm 6. Otherwise no solution is found.

Algorithm 4 SRWARP heuristic – $solveSrwarp(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,\mathcal{I},\delta_{\max})$

Input: A Graph $\mathcal{G}(\mathcal{N},\mathcal{A})$, an impairment threshold Δ, an array of demands $\mathcal{I}$ and $\delta_{\max}$ (the maximum integer increment to relax the minimum logical path cost)

Output: $DP^{\lambda}_{\mathcal{I}} = \{DP^{\lambda}_{i,sd} : i \in \mathcal{I}\}$, the set of optical path pairs $DP^{\lambda}_{i,sd}$ where i is an index of a demand from $\mathcal{I}$, and s and d the source and destination of i, respectively. A given $DP^{\lambda}_{i,sd}$ may be $(null, null)$, if either the primary path or the backup path fails to be obtained

1: $DP^{\lambda}_{\mathcal{I}} \leftarrow \emptyset$
2: Sort $\mathcal{I}$ ▷ By non-increasing distance of the corresponding shortest paths
3: **for** $i \in \mathcal{I}$ **do**
4: $\quad\delta \leftarrow 0$ ▷ Initially only the minimum number of regenerators paths is allowed
5: $\quad$**while** $\delta \leq \delta_{\max}$ **do**
6: $\quad\quad P_{sd} \leftarrow srwarpRoute(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,i,\delta)$ ▷ See Algorithm 13
7: $\quad\quad$**if** $P_{sd} \neq \emptyset$ **then**
8: $\quad\quad\quad$Sort P_{sd} by non-decreasing Fortz path cost function
9: $\quad\quad\quad$**for each** $p_{i,sd} \in P_{sd}$ **do**
10: $\quad\quad\quad\quad ap^{\lambda}_{i,sd} \leftarrow crtOptPath(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,p_{i,sd})$ ▷ See Algorithm 14
11: $\quad\quad\quad\quad bp^{\lambda}_{i,sd} \leftarrow null$ ▷ Initializes backup path to $null$
12: $\quad\quad\quad\quad bp^{\lambda}_{i,sd} \leftarrow computeBPForAP(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,ap^{\lambda}_{i,sd})$ ▷ $ap^{\lambda}_{i,sd}$ cannot be $null$
13: $\quad\quad\quad\quad$**if** $bp^{\lambda}_{i,sd} = null$ **then** ▷ May happen due to lack of capacity
14: $\quad\quad\quad\quad\quad ap^{\lambda}_{i,sd} \leftarrow null$, after releasing all resources assigned to $ap^{\lambda}_{i,sd}$
15: $\quad\quad\quad\quad$**else**
16: $\quad\quad\quad\quad\quad$**go to** step 22 ▷ Demand i was solved
17: $\quad\quad\quad\quad$**end if**
18: $\quad\quad\quad$**end for**
19: $\quad\quad\quad\delta \leftarrow \delta + 1$
20: $\quad\quad$**end if**
21: $\quad$**end while**
22: $\quad DP^{\lambda}_{i,sd} \leftarrow (ap^{\lambda}_{i,sd}, bp^{\lambda}_{i,sd})$ ▷ May be $(null, null)$ for capacity reasons
23: $\quad DP^{\lambda}_{\mathcal{I}} \leftarrow DP^{\lambda}_{\mathcal{I}} \cup \{DP^{\lambda}_{i,sd}\}$
24: **end for**
25: **return** $DP^{\lambda}_{\mathcal{I}}$

Algorithm 6 chooses from the candidate set of paths, passed as input, the final and minimal, in terms of the number of newly deployed regenerators, backup optical path. This algorithm is similar to Algorithm 3 where the only changes concern the addition of an auxiliary hash map $pathToWavelengthMap$ and the steps 4 and 11, which are modifications of steps 3 and 6 of Algorithm 3, respectively. The change in step 11 is simply to allow the re-use of regenerators at the head of lightpaths. If the lightpath has a re-usable regenerator for the lightpath's wavelength, then no new regenerator will be added to the node. The

Algorithm 5 *computeBPForAP*($\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ, $ap^{\lambda}_{i,sd}$)

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and a primary optical path $ap^{\lambda}_{i,sd}$

Output: $bp^{\lambda}_{i,sd}$, the computed backup path for the input active path $ap^{\lambda}_{i,sd}$

1: $bp^{\lambda}_{i,sd} \leftarrow null$
2: Create a graph $\mathcal{G}'(\mathcal{N},\mathcal{A}')$ where $\mathcal{A}' = \mathcal{A} \setminus \{\mathcal{A}_{ap^{\lambda}_{i,sd}} \cup \mathcal{A}_{ap^{\lambda}_{i,ds}}\}$
3: Set the cost of *preferred* arcs in $\mathcal{A}'$ to 0 ▷ Recall definition in page 33
4: $\hat{P}^{k}_{sd} \leftarrow$ the set of paths returned by using steps steps 4 to 9 of Algorithm 11 using $\mathcal{G}'(\mathcal{N},\mathcal{A}')$, considering usable wavelengths instead of free wavelength at step 4 of that algorithm
5: **if** $\hat{P}^{k}_{sd} \neq null$ **then**
6: $\quad bp^{\lambda}_{i,sd} \leftarrow generateFinalBP(\mathcal{G}'(\mathcal{N},\mathcal{A}'), \Delta, ap^{\lambda}_{i,sd}, \hat{P}^{k}_{sd})$
7: **end if**
8: **return** $bp^{\lambda}_{i,sd}$

addition of $pathToWavelengthMap$ allows the mapping between the logic arcs of p'^{*}_{sd} and the corresponding usable wavelength at step 10 (further explained at Algorithm 7). As for step 4, the initialization of the segments p_{uv} has now to take into account both re-usable regenerators and shareable wavelengths. As this change adds extra complexity, Algorithm 7 details this step in a step by step approach. Note that for the *dedicated-dedicated* variant of the SRWARP, one could use Algorithm 3 without any change instead of using Algorithm 6. Algorithm 6 can be used for the *dedicated-shared* variant if the definition of re-usable is restricted to the regenerators of the active path to be protected and no wavelength sharing between backup paths is allowed.

As stated, the creation of the reachability graph has now to take into account the possibility of re-using wavelengths and regenerators. Thus, as presented at Algorithm 7 the method will try to create an auxiliary graph $\mathcal{G}''(\mathcal{N}_{p_{sd}},\mathcal{A}'')$ where $\mathcal{A}''$ will contain logical arcs from every node u to every reachable (in terms of distance and common usable wavelengths) posterior node v in the path p_{uv}. Each of these segments u to v have an associated λ_{uv} which is mapped through the $pathToWavelengthMap$. That is, $pathToWavelengthMap$ is a hash map (a set of pairs key-value) where the key is a path p_{uv} and the value is the corresponding λ_{uv}. This hash map will be part of the output of Algorithm 7, so that in Algorithm 6 at step 10 the lightpaths may be formed with the selected wavelengths.

The creation of the reachability graph, as presented in Algorithm 7 has the following steps. In step 1 an auxiliary graph $\mathcal{G}''(\mathcal{N}_{p_{sd}},\mathcal{A}'')$ is created, where $\mathcal{A}''$ is initialized to an empty set. In step 2, the path-to-wavelength hash map is created, also in an empty state. The method will then procceed to create the feasible segments p_{uv} and associate an usable wavelength,

Algorithm 6 $generateFinalBP(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,ap^{\lambda}_{i,sd},\hat{P}^{k}_{sd}))$

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ, the active path $ap^{\lambda}_{i,sd}$ and a set of candidate paths $\hat{P}^{k}_{sd}$
Output: $bp^{\lambda}_{i,sd}$, the computed backup path for the input active path $ap^{\lambda}_{i,sd}$

1: Set $P'_{sd} \leftarrow \emptyset$ where P'_{sd} is an auxiliary set of logic paths from s to d
2: $pathToWavelengthMap \leftarrow \emptyset$
3: **for each** $(p_{sd} \in \hat{P}^{k}_{sd})$ **do**
4: $\quad\mathcal{G}''(\mathcal{N}_{p_{sd}},\mathcal{A}''), newPathToWavelengthMap \leftarrow createReachabilityGraph(p_{sd},\Delta)$
5: $\quad pathToWavelengthMap = pathToWavelengthMap \cup newPathToWavelengthMap$
6: $\quad$Compute the shortest path from s to d, p'_{sd} on $\mathcal{G}''$ and add it to the set of P'_{sd}
7: **end for**
8: **if** $P'_{sd} \neq \emptyset$ **then** ▷ A solution must exist if called from Algorithm 5
9: $\quad$Select the minimum cost path p'^{*}_{sd} from all paths in P'_{sd} ▷ Note that the cost is now different than the hop count
10: $\quad$Expand the logic arcs of p'^{*}_{sd} into the segments assigning the corresponding wavelength obtained in step 4 (using $pathToWavelengthMap$) to obtain the lightpaths ξ_w, $\forall w \in \{su_1, u_1u_2, \ldots, u_kd\}$. Let $p^{\lambda}_{sd} = \langle \xi_{su_1}, \xi_{u_1u_21}, \ldots, \xi_{u_{k-}d} \rangle$ be the final optical path
11: $\quad$If at the last node of each lightpath ξ_w, $\forall w \neq u_kd$ there is no regeneration for the corresponding wavelength, place a regenerator. Otherwise, use, if shareable, the existing regenerator
12: $\quad$**return** p^{λ}_{sd}
13: **end if**
14: **return** *null* ▷ Can not fail if called from Algorithm 5

through steps 3 to 26. With that intent, step 3 will iterate through every node $u \in \mathcal{N}_{p_{sd}}$ and step 4 will iterate through every downstream (w.r.t. node u) node $v \in \mathcal{N}_{p_{ud}}$, defining thus the segment p_{uv}; p_{uv} must have a cost (distance) lower or equal to the impairment threshold. If the path cost is higher than Δ, then step 6 is executed and the method returns to step 3 for the following node u. If the path cost is acceptable, then the method will try to find an usable wavelength and, in case of success, a logical arc is created for p_{uv} in $\mathcal{A}''$.

As the purpose of the SRWARP is to minimize the number of regenerators, finding a wavelength is not as simple as using the FF. Furthermore, the *shared-shared* variant also seeks to reduce the required bandwidth by sharing wavelengths between backup paths (when feasible). For this two reasons, steps 8 to 24 are required in Algorithm 7. Starting at step 8, the existence of re-usable regenerators at node v is checked. If there is at least one re-usable regenerator, a set of wavelengths Λ_{uv} is created at step 9 containing all these regenerators' wavelengths. Within this set and in step 10, a shareable common wavelength λ_{uv} to all arcs in $\mathcal{A}_{p_{uv}}$ is retrieved, if one exists. If no such wavelength exists, λ_{uv} is set to any usable

common wavelength (to all arcs in p_{uv}) or *null* if no usable common wavelength exists. If λ_{uv} is not *null* at step 11 then a feasible segment was successfully computed were λ_{uv} may be assigned. Thus, this segment is added to $\mathcal{A}''$ with a cost of 0 in step 12 and the pair key-value (p_{uv}, λ_{uv}) is added to the hash map $pathToWavelengthMap$ in step 13. Note that the assignment of a cost of 0 is associated to the fact that a regenerator will be re-used for this segment, which makes traversing this segment costless. As a segment p_{uv} was found, the method shall return to step 4 for the following v (step 14). If either λ_{uv} is *null* or no reusable regenerator was found in v at step 8, Algorithm 7 will proceed to step 17 where first a shareable common wavelength is checked to exist in the segment p_{uv} and set to be λ_{uv}. If no sharable common wavelength exists, then λ_{uv} is set to be an usable (either free or a combination of free-shareable through the arcs) wavelength or *null* if none is found. If a wavelength is found, then the segment is added to $\mathcal{A}''$ with a cost of 1 in step 19 and then the pair key-value (p_{uv}, λ_{uv}) is added to $pathToWavelegnthMap$ in step 20. As a segment was successfully formed, the algorithm then proceeds to return to step 5 for the next node u. If λ_{uv} is *null* after step 17, then it is not necessary to check for the following v has no usable wavelength exists in the segment. Therefore, step 23 makes the method return to step 3 for the following node u. At step 27, the output is finally returned where $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ contains the reachability graph and $pathToWavelengthMap$ the key-value pairs that map each physical path of the logical arcs in $\mathcal{A}''$ to an usable wavelength to be assigned at the lightpath formation step (step 10 of Algorithm 6).

Algorithm 7 *createReachabilityGraph*(p_{sd}, Δ))

Input: A topological path p_{sd} and the impairment threshold Δ

Output: A reachability graph $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ where $\mathcal{A}''$ contains logical arcs from every node $u \in \mathcal{N}_{p_{sd}}$ to every reachable node $v \in \mathcal{N}_{p_{ud}} \setminus \{u\}$ and a hash map $pathToWavelengthMap = (p_{xy}, \lambda_{xy})$ which stores a given λ for a given p_{xy}, where $x, y \in \mathcal{N}_{p_{sd}}$

1: Create $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ with $\mathcal{A}'' = \emptyset$
2: $pathToWavelengthMap \leftarrow \emptyset$
3: **for** $u \in \mathcal{N}_{p_{sd}}$ **do**
4: **for** $v \in \mathcal{N}_{p_{ud}}$ **do**
5: **if** $(c(p_{uv}) > \Delta)$ **then**
6: **go to** step 3 for the next u ▷ All segments must have a maximum length of Δ
7: **end if**
8: **if** v has re-usable regenerators **then**
9: Let Λ_{uv} be the set of all re-usable regenerators' wavelengths
10: Let $\lambda_{uv} \in \Lambda_{uv}$ be a shareable common (to all arcs) wavelength in p_{uv} if there is one. Otherwise, let $\lambda_{uv} \in \Lambda_{uv}$ be an usable (mixture of free and shareable) common wavelength in p_{uv} or *null* if no usable wavelength is found
11: **if** $\lambda_{uv} \neq null$ **then**
12: $\mathcal{A}'' = \mathcal{A}'' \cup \{(u, v)\}$, with $c_{uv} = 0$
13: $pathToWavelengthMap = pathToWavelengthMap \cup \{(p_{uv}, \lambda_{uv})\}$
14: **go to** step 4 for the next v
15: **end if**
16: **end if**
 ▷ If it reaches here, either there are no re-usable regenerators in v or no usable wavelength λ_{uv} was found in the regenerated wavelengths set.
17: Let λ_{uv} be a shareable common wavelength in the p_{uv} or a free common wavelength if no shareable is found or *null* if nor a free nor a shareable is found
18: **if** $\lambda_{uv} \neq null$ **then**
19: $\mathcal{A}'' = \mathcal{A}'' \cup \{(u, v)\}$, with $c_{uv} = 1$
20: $pathToWavelengthMap = pathToWavelengthMap \cup \{(p_{uv}, \lambda_{uv})\}$
21: **go to** step 4 for the next v
22: **else**
23: **go to** step 3 for the next u ▷ No usable λ found
24: **end if**
25: **end for**
26: **end for**
27: **return** $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ and $pathToWavelengthMap$

3.4 Multi-Layer Heuristic

In this section, the introduction of the aforementioned RWARP approach in a multi-layer problem, referred to as Grooming and Routing and Wavelength Assignment with Regeneration (GRWAR), is described. This problem was addressed in the context of another MSc. Thesis and can be found thoroughly explained in [20]. A brief contextualization, followed by the definition of the problem and the proposed heuristics will be described in the following subsections, while highlighting this thesis' contribution. An extension to the original approach, where segment-based survivability scheme is added to the heuristics, will be also discussed.

3.4.1 Problem Context

Prior to the definition of the problem, some contextualization is required. In GRWAR, an abstraction in two layers may be considered: the *optical* layer and the *client* layer. The former layer is the network depicted in this work where fibres support wavelength division multiplexing and regenerators perform OEO conversions to regenerate the signal quality. A signal is added (at an OADM of a node) to the optical layer from the client layer, to be routed in the optical domain (except for regeneration purposes) and then dropped at the destination, to the client layer. In this upper layer, signals – client signals – typically require sub-wavelength data rates and thus there is the possibility of grooming multiple lower speed streams into high speed wavelength channels. Client signals may thus return to the client layer in intermediate nodes to be groomed.

Groomed or not, a client signal as to be converted from the electrical to the optical domain to be routed in the optical layer and converted back to the electrical domain. The devices that allow these conversions are known as transponders and, in fact, regenerators have two of these converters[4] that allow the OEO conversion. Thus, at both ends of a lightpath, a transponder is always required.

In summary, a wavelength channel may emerge at a node from the optical layer to the client layer, undergoing the optical to electrical conversion at a transponder, to reach the destination, for traffic grooming, regeneration and/or wavelength conversion. Regenerators may still perform regeneration and wavelength conversion at the optical layer.

[4]Recall that since regenerators are considered bidirectional, four transponders are in fact required. In order to simplify the explanation, transponders will also be assumed as bidirectional.

3.4.2 Problem Definition

The definition of the GRWAR problem is similar to the one presented in subsection 3.2.1 for the RWARP problem, where the difference lies on the possibility of grooming multiple demands in a wavelength. Given a single fibre network[5], a static set of demands and an impairment threshold Δ, the objective is to optically route all requests while minimizing the number of required transponders. Traffic may be groomed into a wavelength, at the client layer, as long as there is enough residual bandwidth, allowing a lightpath to carry data from multiple demands. Regeneration and wavelength conversion may be performed at the client layer and at the regenerators, where each of these devices will count as two transponders towards the objective value.

Since that at both ends of a lightpath (recall the definition of transparent lightpaths from section 3.1) a transponder is required, the dual problem of the GRWAR problem is the minimization of the number of lightpaths required to route all demands.

3.4.3 Heuristic Approach

The multi-layer implemented heuristic, illustrated in Algorithm 8, is essentially the execution of Algorithm 11, which computes a set of optical paths for the demands set, followed by the grooming heuristic (*runGR*) [20] which by using a given candidate set of lightpaths as a virtual topology, optically re-routes all demands applying grooming whenever possible (see steps 5 to 8). The output of *runGR* is a list of the selected lightpaths used to route the demands where each lightpath may be a segment of multiple demands.

Since Nogueira [20] found that *runGR* may depend on the maximum capacity in each arc (W), the aforementioned steps are now the body of a *while* cycle that decrements by 1 the arcs capacity from W until *runGR* cannot route all demands due to capacity shortage – steps 3 to 14. At the end of each iteration, the network is restored to the original state and thus the best solution (a set of lightpaths to route all demands with minimal regeneration) has to be stored in a auxiliary set (steps 1 and 10). Note that step 10 is only executed if current iteration results are a better than the previous better solution (or at first cycle). A better solution exists when all demand are served and less transponders are required. At the end of the cycle, the network state reflects the best solution found – step 15 and 16 – which is returned at step 17.

[5]See footnote 2 in page 17

Algorithm 8 $rwarp\_gr(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,\mathcal{I})$

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and $\mathcal{I}$

Output: The solution set of lightpaths $bestLP_{gr}^{\lambda}$

1: Let $bestLP_{gr}^{\lambda} \leftarrow null$ be the current best solution set of lightpaths
2: $wCounter \leftarrow W$ ▷ $wCounter$ will iterate from W (arc's original capacity) to 0
3: **repeat**
4: Set the number of wavelengths in all arcs in $\mathcal{A}$ to $wCounter$
5: $P_{\mathcal{I}}^{\lambda} \leftarrow solveRwarpImproved(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,\mathcal{I})$ (see Algorithm 11) ▷ Set of optical paths for the demands with the current arc's capacity; some paths in $P_{\mathcal{I}}^{\lambda}$ may be $null$
6: Clear network routing information, capacity usage and placed regenerators
7: Transform the set of paths in $P_{\mathcal{I}}^{\lambda}$ into a set of lightpaths LP^{λ}
8: $LP_{gr}^{\lambda} \leftarrow runGR(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,LP^{\lambda})$
9: **if** LP_{gr}^{λ} is better solution than $bestLP_{gr}^{\lambda}$ or $bestLP_{gr}^{\lambda} = null$ **then**
10: $bestLP_{gr}^{\lambda} \leftarrow LP_{gr}^{\lambda}$ ▷ Best solution is stored in $bestLP_{gr}^{\lambda}$
11: **end if**
12: Reset network routing information, capacity usage and transponder information
13: $wCounter \leftarrow (wCounter - 1)$
14: **until** fails to route all demands in step 8
15: Restore number of wavelengths in all arcs to W
16: Populate network with $bestLP_{gr}^{\lambda}$ information, including transponder placement.
17: **return** $bestLP_{gr}^{\lambda}$

3.4.4 Survivable Multi-Layer Heuristic

Segment-based survivability (from the point of view of the client layer) may be added to Algorithm 8, by calling Algorithm 9 with the solution returned from the former algorithm as input. The *addRecovery* method, presented in Algorithm 9, computes a backup optical path (segment) for every active lightpath in the input set of lightpaths (see steps 3 to 8). To do so, Algorithm 5 is called where the input active path is composed by a single lightpath (steps 4 and 5 of Algorithm 9), returning a backup optical path or *null* if for capacity reasons, no feasible segment was found. Since a shared variant of segment recovery may be used, the computation of the backup segments is ordered by the a non-decreasing hop count of the respective active lightpaths (see step 2) to increase the probabilities of sharing bandwidth and regenerators. Note that since the active lightpaths have no regenerators (albeit having transponders on both ends), sharing this resource occurs only between backup segments.

Algorithm 9 *addRecovery*$(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,LP^{\lambda})$

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and LP^{λ}, where LP^{λ} is the set of active lightpaths to be protected

Output: A list of active lightpath – backup segment pairs $(A\xi_w, BP_{w'})$.

1: $A\xi_w \leftarrow \emptyset$ and $BP_{w'} \leftarrow \emptyset$
2: Sort ligthpaths in LP^{λ} in a non-decreasing hop count way
3: **for each** $\xi_{w_0} \in LP^{\lambda}$ **do**
4: Let $ap^{\lambda}_{sd} \leftarrow \langle \xi_{w_0} \rangle$ where it is not relevant the association between this primary path/segment and the transported demands
5: Run Algorithm 5 with ap^{λ}_{sd} as the input primary optical path. Let $bp^{\lambda}_{sd} = \langle \xi'_{w_0}, \xi'_{w_1}, \cdots, \xi'_{w_k} \rangle$ be the output of this step, which may also be *null*
6: $A\xi_w \leftarrow A\xi_w \cup \{\xi_{w_0}\}$
7: $BP_{w'} \leftarrow BP_{w'} \cup \{bp^{\lambda}_{sd}\}$
8: **end for**
9: **return** $(A\xi_w, BP_{w'})$

4 Results and Discussion

In this chapter, the setup used to conduct the experimental tests will be first described, followed by the results for the defined problems presented in chapter 3, with the respective analysis for each of the approaches.

4.1 Experimental Setup And Terminology

Two computers were used in the experimental tests, a PC1, with an Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz and 16GB RAM, and a PC2, with a Inter(R) Xeon(R) CPU X5660 @ 2.80GHz processor and 48GB RAM. In both computers, the optimizer IBM ILOG CPLEX Optimization Studio V12.6.1 was used through the Java API. The optimizer was limited to 4 CPU cores and to an execution time of 24 hours for each test. The heuristics were implemented in Java, where no parallelization technique was used.

The simulations were performed in four different networks from SNDlib [21]: polska, abilene, nobel_germany and janos_us_ca. As the nobel_germany network is too small to use a realistic value for Δ, the links distances were doubled, and therefore this modified network will be denoted as nobel_germany*. SNDLib problem instances associate a demand set to each of the network topologies where a request for each pair of nodes exists. Experiments in this work made use of these base set of demands or multiples of the same set. Table 4.1 illustrates each of these problem instances with the base set of demands. Recall that n is the number of nodes and m the number of arcs.

Network	n	m	$\lvert\mathcal{I}\rvert$
polska	12	18	66
abilene	12	15	66
nobel germany*	17	26	121
janos_us_ca	39	122	1482

Table 4.1: SNDLib input networks with the base set of demands.

Concerning the survivable multi-layer heuristic, no results will be presented as no comparison may be done.

4.2 RWARP Results

Results for the RWARP ILP formulation from subsection 3.2.2 (CPLEX), *solveRwarp* (Algorithm 1) and *solveRwarpImproved* (Algorithm 11) are illustrated in table 4.2 where no limitation to the number of K shortest paths algorithm was given to Algorithm 11. Columns 1, 2, 3 and 4 are the input data corresponding to, respectively: the network, the number of demands $|\mathcal{I}|$, the threshold impairment value Δ and the capacity (number of wavelengths) in each arc $|\lambda| = W$. The output results are composed of three values, delimited by "/": the first value represents the CPLEX solution, the middle value is the *solveRwarp* and the last is the *solveRwarpImproved*. The outputs are, respectively, from column 5 to 10: the number of fulfilled demands, the number of lightpaths formed, the network capacity usage (in percentage), the maximum number of wavelengths used in any link of the network, the number of regenerators placed and the execution time (in seconds). The network capacity usage is simply the ratio between the total wavelengths in use and the total number of wavelengths in the network. Since the CPLEX execution time was limited to 24 hours, results with "time out" provided either a sub-optimal solution or no solution at all (marked with "–"). The "killed" time results means that the tests were terminated with an *out of memory* error.

Starting by comparing the execution time, it is clear from table 4.2, that both heuristics largely outperform the CPLEX. For most of the tests, *solveRwarp* took fractions of a second and *solveRwarpImproved* required from a little more than half a second till almost four seconds whereas CPLEX took hundreds or even thousands of seconds to execute, when the 24 hours limit was enough. Even for janos us_ca, the largest tested network, *solveRwarp* requires less than 10 seconds, whereas *solveRwarpImproved* took more than 1000 seconds. This difference is due to the fact that the at step 4 of Algorithm 11, K_{uv} segments are considered between every pair of nodes u, v where K_{uv} was not limited (aside from the segments length being required to respect Δ). Thus, bounding K_{uv} in the shortest K paths algorithm could improve the execution time but could also lead to a worse solution (a larger number of regenerators).

Considering the objective value, the number of regenerators placed, *solveRwarpImproved* is considerable better than *solveRwarp*. In fact, in almost all the tests,

Input data				Results CPLEX / Without Fortz / With Fortz					
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage(%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska	66	1000	48	66 / 66 / 66	134 / 134 / 134	17.94 / 16.67 / 16.67	16 / 15 / 14	1 / 1 / 1	206.47 / 0.14 / 1.08
	132	1000	24	132 / 132 / 132	268 / 284 / 268	66.67 / 71.76 / 70.60	24 / 24 / 24	2 / 10 / 2	4363.83 / 0.26 / 1.90
	132	1000	48	132 / 132 / 132	268 / 268 / 268	35.76 / 33.33 / 35.19	28 / 30 / 25	2 / 2 / 2	667.97 / 0.21 / 2.02
	264	1000	48	- / 264 / 264	- / 568 / 536	- / 71.76 / 71.64	- / 48 / 48	- / 20 / 4	killed / 0.31 / 3.72
abilene	66	3000	20	66 / 66 / 66	174 / 180 / 174	58.00 / 57.00 / 58.33	19 / 20 / 20	21 / 24 / 21	606.16 / 0.11 / 0.63
	66	3000	48	66 / 66 / 66	174 / 174 / 174	24.44 / 23.47 / 23.61	21 / 22 / 20	21 / 21 / 21	636.27 / 0.11 / 0.73
	132	3000	40	132 / 132 / 132	348 / 360 / 348	56.83 / 57.00 / 59.67	39 / 40 / 40	42 / 48 / 42	time out / 0.17 / 1.13
	132	3000	48	132 / 132 / 132	348 / 348 / 348	47.36 / 46.94 / 49.58	38 / 44 / 38	42 / 42 / 42	2399.51 / 0.16 / 1.17
nobel germany*	121	1000	32	121 / 121 / 121	346 / 330 / 332	40.87 / 40.38 / 41.47	31 / 32 / 32	52 / 44 / 45	time out / 0.25 / 2.41
janos us_ca	1482	2000	220	– / 1482 / 1482	- / 5388 / 5186	- / 51.74 / 54.78	- / 220 / 220	- / 1212 / 1111	killed / 8.24 / 1086.87

Table 4.2: Comparison between ILP and the heuristics for the RWARP problem, run on PC1

solveRwarpImproved provided the optimal solution to the problem. It can be seen that the constant use of the shortest path (either for the segments u to v or for the final paths) by Algorithm 1, may lead to a worse solution value due to the capacity shortage of some arcs – results where "Max in Use λ" value equals $|\lambda|$. The traffic distribution that the Fortz function introduces in the *solveRwarpImproved* allowed to greatly reduce the probability of arc capacity shortage, by using equivalent segments/paths (w.r.t. the number of regenerators), thus leading to better results. There is an exception observed in the nobel germany case, where *solveRwarpImproved* performed worse than *solveRwarp* by one regenerator as both lead to at least one arc to be fully occupied. Nevertheless both were better than the sub-optimal solution obtained by the CPLEX in 24 hours.

In terms of overall network capacity usage, *solveRwarp* has the best values when it computes the optimal number of regenerators placed, as it was to be expected since the shortest paths are used. In these cases, *solveRwarpImproved* has a higher capacity usage as longer paths (but still optimal in terms of regeneration) are considered. The ILP does not optimize path lengths (or path hop counts) as long as the paths are optimal in terms of the number of regenerators.

Table 4.3 provides a time benchmark between the RWARP ILP variants defined in subsection 3.2.3. Results show that the wavelength conversion capability improves the execution time (ILP-1 to ILP-2) which was to be expected as wavelength conversion relaxes the wavelength continuity constraint. The removal of the capacity constraints also improves the execution time (ILP-1 to ILP-3), as it relaxes the path selection. Note that without capacity constraints, wavelength assignment is irrelevant. Lastly, Beshir *et al.* [3] implementation (ILP-4) was the fastest which was to be expected since both capacity and bidirectional constraints were absent.

Input data					CPLEX
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	ILP Variation	**Time (s)**
polska	66	1000	32	ILP-1	180.16
	66	1000	32	ILP-2	219.55
	66	1000	32	ILP-3	172.15
	66	1000	32	ILP-4	108.63
	132	1000	48	ILP-1	765.89
	132	1000	48	ILP-2	963.53
	132	1000	48	ILP-3	781.14
	132	1000	48	ILP-4	503.43

Table 4.3: RWARP ILP variants execution time comparison, run on PC2

4.3 SRWARP Results

Similarly to the previous section, two sets of tests were performed for the SRWARP problem. The first set seeks to compare the results of the ILP formulation presented in subsection 3.3.2 with the results of Beshir *et al.* formulation in [3], whereas the second set of results is the comparison between the optimal and the heuristic results. Note that since Beshir *et al.* formulation does not take into account arcs capacity, using these formulation results for comparisons is only valid for tests where the CPLEX solution does not "overuse" any arc, i.e. tests where "Max in Use λ" is less or equal to $|\lambda|$ (this overuse can occur only in Beshir *et al.* formulation) and thus, only tests of this type are presented.

In this subsection, subsets of the polska and abilene networks were used, where " * " will be used to identify these subsets. The polska* network corresponds to a subset of 7 nodes and 9 links (18 arcs), where each node is connected by at least two links. The definition of this network may be found in appendix D. In abilene*, node *ATLAM5* was removed, since only one link connected this node which makes it impossible to provide recovery for link failures (no disjoint path pair exists).

Table 4.4 presents the results for Beshir *et al.* formulation (recall each demand is solved sequentially) and for the formulation presented in subsection 3.3.2. The "Variant" column was added to the "Input data" columns, where d-d refers to the *dedicated-dedicated* variant and d-s is the *dedicated-shared* version. The network capacity usage was divided in two columns, the "Capacity Usage for APs(%)" and "Capacity Usage for BPs(%)", where the first has the network capacity in use for the active paths, and the second has the network capacity usage for the backup paths (both in percentage).

As one can see in the execution time column of table 4.4, for the ILP formulation from

subsection 3.3.2, all tests were terminated with a "time out". In fact, other tests were performed where no results were obtained after the 24 hours time limit (or even "killed" before that time frame). On the other hand, Beshir *et al.* formulation, which was run for each demand sequentially, executed in less than 1 second for all these cases. Thus, table 4.4 confirms that the ILP formulation from subsection 3.3.2 is impractical for real size networks. This result is a consequence of the addition of capacity constraints that remove the possibility of running the formulation independently for each demand.

Input data					**Results** Beshir *et al.* CPLEX / Implemented CPLEX						
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	Variant	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	d-d	21 / 21	98 / 98	6.02 / 5.56	5.79 / 5.90	12 / 12	7 / 7	0.41 / time out
				d-s	21 / 21	98 / 98	6.02 / 6.02	5.67 / 6.02	13 / 14	6 / 6	0.46 / time out
	42	1000	96	d-d	42 / -	196 / -	12.04 / -	11.57 / -	24 / -	14 / -	0.71 / time out
				d-s	42 / 42	196 / 196	12.04 / 11.57	11.34 / 12.04	26 / 27	12 / 12	0.82 / time out

Table 4.4: Results for the Beshir *et al.* [3] formulation and the ILP formulation from subsection 3.3.2, run on PC1

Since the ILP formulation from subsection 3.3.2 may only run in a limited number of networks, Beshir *et al.* formulation was used instead, to properly compare the heuristic results. Recall that the comparison is valid for tests where the CPLEX solution does not overuse any arc.

Table 4.5 presents the comparison results between the CPLEX solution (with Beshir *et al.* formulation) and the SRWARP heuristic presented in subsection 3.3.3, with an input $\delta_{\max} = 4$. This input value was chosen such that, in all tested networks, an active path and a backup path could be assigned to all demands. Note that, for these particular test cases, if $\delta_{\max}$ was larger, the results would still be the same. Nevertheless, a lower $\delta_{\max}$ could cause blocking of some demands (see appendix E). Although the network capacity should be sufficient for routing all demands, the heuristic may be unable to route all requests when the utilization rate of the network's bandwidth is high.

It is observable from table 4.5 that the SRWARP heuristic provides acceptable results, where the average relative error is approximately 8.5% and the maximum 21% (for the abilene* network with 165 demands). Note that in this particular test, the total network capacity usage (sum of the capacity usage for the active paths with the capacity usage for the backup paths) is around 80%, which, for the heuristic, leads to the capacity shortage of some arcs, thus worsening the results.

Other interesting result from table 4.5 is that in the heuristic, active paths capacity usage is, at all times, lower than the CPLEX result, whereas in the backups capacity, it

Input data				Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 4$ Beshir *et al.* CPLEX / Heuristic						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21 / 21	98 / 98	6.02 / 4.28	5.79 / 7.41	12 / 13	7 / 7	0.40 / 0.14
	42			42 / 42	196 / 196	12.04 / 8.57	11.57 / 14.93	24 / 27	14 / 14	0.81 / 0.22
polska	66	1000	96	66 / 66	294 / 300	10.42 / 8.51	11.52 / 12.56	33 / 32	15 / 18	2.93 / 1.22
	132			132 / 132	588 / 600	20.83 / 18.056	23.03 / 23.90	66 / 60	30 / 36	6.13 / 2.21
abilene*	55	3000	96	55 / 55	364 / 366	13.84 / 10.79	12.95 / 16.00	31 / 32	72 / 73	2.63 / 0.53
	110			110 / 110	728 / 732	27.68 / 21.58	25.89 / 32.00	62 / 64	144 / 146	3.95 / 1.10
	165			165 / 165	1092 / 1182	41.52 / 32.37	38.84 / 48.00	93 / 96	216 / 261	6.15 / 1.43
nobel germany*	121	1000	96	121 / 121	882 / 900	18.23 / 17.11	18.35 / 18.79	64 / 67	199 / 208	18.21 / 2.96

Table 4.5: Results for the *dedicated-dedicated* variant of the Beshir *et al.* [3] formulation and the SRWARP Heuristic, run on PC1

is the opposite. This comes as a consequence of using the Min-Min approach depicted in subsection 3.3.3.

Table 4.6 presents the results for the *dedicated-shared* variant, where the abilene* test with 165 demands was removed as at least one arc would be overused if the ILP would consider capacity constraints. It is clear that the heuristic results are worse than the results with CPLEX. In fact, comparing table 4.6 with table 4.5 shows that the heuristic was only able to share regenerators for the nobel_germany* network. The ILP solution on the other hand, was able to share regenerators in all networks, with the exception in the abilene* network. This result suggests that routes that pass through re-usable regenerators are not being selected for the backup path. It is possible that setting the cost of preferred arcs to 0 (see step 3 of Algorithm 5) is not enough. Re-using a regenerator may decrease the impairment accumulated effects up to Δ kilometres (the impairment threshold) but decreasing an arc cost to 0 only reduces the arc length to the path cost, which is at most the impairment threshold.

The SRWARP heuristic (*dedicated-shared*) is not very effective in sharing regenerators. However note that even the optimal solutions do not present a significant reduction in the total number of regenerators, for the tested cases. It should be pointed out that the heuristic is able to solve, in a very short time, relatively large problems. If capacity constraints were taken into account, no solution would be obtained by the corresponding ILP.

Since no ILP formulation was given (or developed) for the *shared-shared* variant, the results comparison will be restricted to the comparison between variants. Table 4.7 presents the results for this last variant, where it is clear that the sharing of bandwidth greatly reduced the capacity needed for the backup paths and, consequently, lowered the "Max in Use λ" column value. The number of regenerators placed is less than in the *dedicated-shared*

Input data				Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 4$ Beshir *et al.* CPLEX / Heuristic						
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21 / 21	98 / 98	6.02 / 4.28	5.67 / 7.41	13 / 13	6 / 7	0.46 / 0.15
	42			42 / 42	196 / 196	12.04 / 8.56	11.34 / 14.93	26 / 27	12 / 14	0.82 / 0.29
polska	66	1000	96	66 / 66	294 / 300	10.94 / 8.51	11.05 / 12.56	30 / 32	14 / 18	3.56 / 1.20
	132			132 / 132	588 / 600	21.86 / 18.06	22.11 / 23.90	60 / 60	28 / 36	7.21 / 2.19
abilene*	55	3000	96	55 / 55	364 / 366	13.69 / 10.79	13.17 / 16.00	33 / 32	72 / 73	2.57 / 0.60
	110			110 / 110	728 / 732	27.38 / 21.58	26.34 / 32.00	66 / 64	144 / 146	4.92 / 1.24
nobel germany*	121	1000	96	121 / 121	894 / 900	18.91 / 17.10	18.39 / 18.83	69 / 67	181 / 206	22.84 / 3.00

Table 4.6: Results for the *dedicated-shared* variant of the Beshir *et al.* [3] formulation and the SRWARP Heuristic, run on PC1

Input data				Results for the *shared-shared* SRWARP variant with $\delta_{\max} = 4$ Heuristic						
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	100	4.28	5.79	12	7	0.20
	42			42	200	8.56	12.38	23	14	0.30
polska	66	1000	96	66	344	8.56	4.86	18	17	1.27
	132			132	662	17.30	10.24	34	30	2.83
abilene*	55	3000	96	55	372	10.71	11.83	30	70	0.64
	110			110	740	21.58	23.66	60	140	1.46
	165			165	1110	32.37	35.49	90	210	2.90
nobel germany*	121	1000	96	121	958	17.07	11.86	60	178	3.47

Table 4.7: Results for the *shared-shared* SRWARP Heuristic, run on PC1

variant optimal solution in both abilene* and nobel_germany*. In both polska* and polska networks, the objective value is greater than the *dedicated-shared* variant optimal solution. Nevertheless, in polska, there is a slight improvement in relation to the *dedicated-dedicated* and *dedicated-shared* heuristic. In a final note, the execution times of the three heuristic variants are similar.

To summarize, figure 4.1 shows the number of regenerators obtained for all variants of the SRWARP heuristic and ILP formulation where the conclusions were already mentioned. Figures 4.2a and 4.2b illustrate the network capacity usage for the active paths and for the backup paths, respectively, obtained with the CPLEX and with the heuristic variants solutions. As previously mentioned, the active paths capacity is considerably lower for all heuristic variants when compared to the CPLEX solution, which comes as a consequence of using the Min-Min approach (see figure 4.2a). The inverse can be seen in figure 4.2b for the backup paths case, where higher bandwidth is required in the heuristic except for the *shared-shared* variant. In this variant and as expected, wavelength sharing possibility greatly decreases the required capacity.

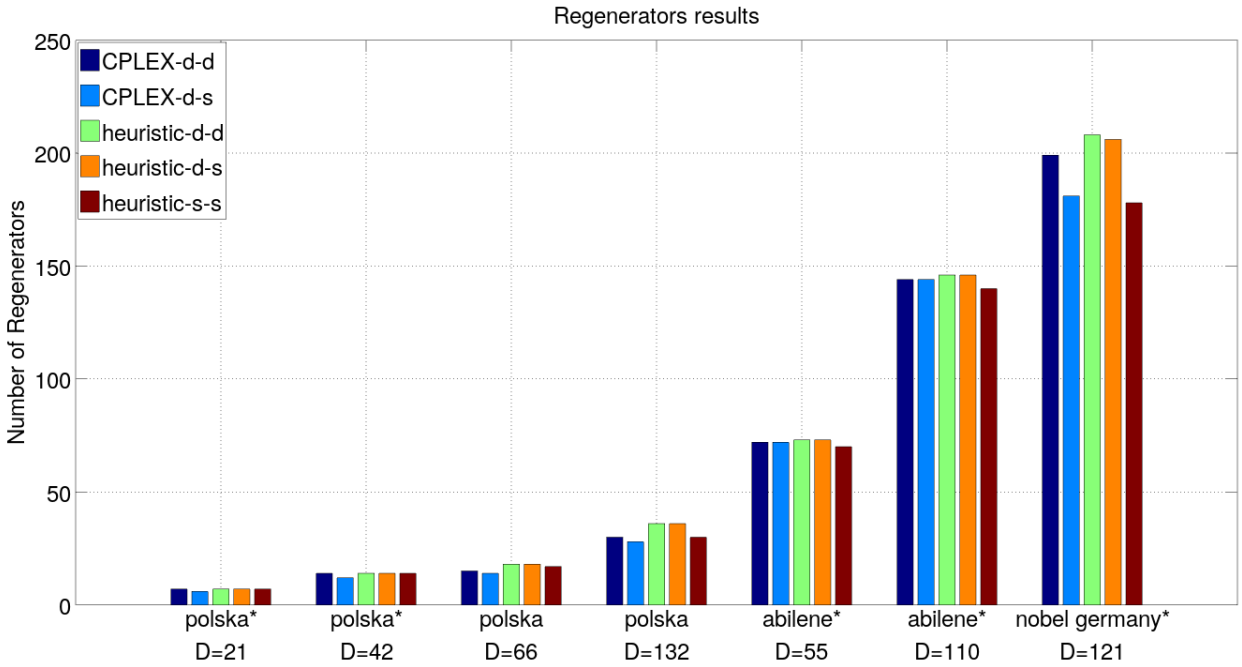
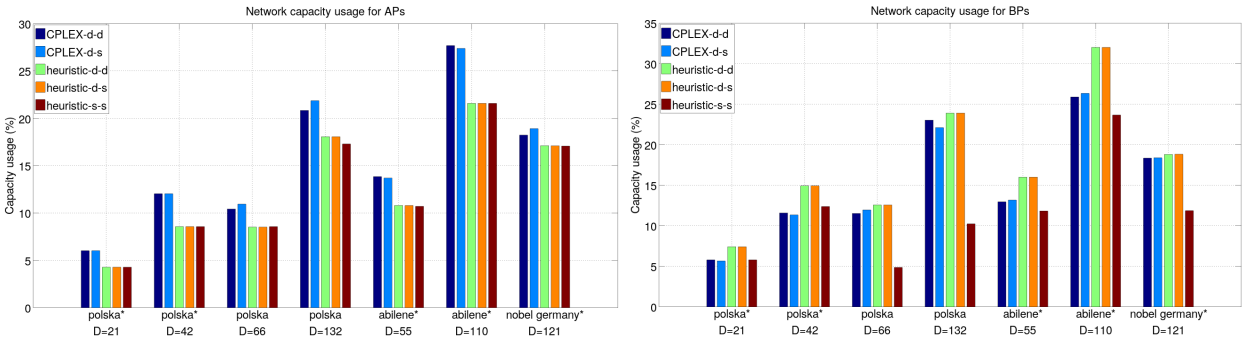


Figure 4.1: Number of regenerators obtained with the different heuristic variants and with the CPLEX solutions.



(a) Capacity usage for active paths

(b) Capacity usage for backup paths

Figure 4.2: Comparison between the network capacity usage for the active paths and backup paths obtained with the different heuristic variants and with the CPLEX solutions.

5 Conclusions and Future Work

Today's society is strongly dependent on uninterrupted network services which require increasing bandwidth. WDM networks have made possible to provide the necessary capacity to support these services. An optical transport network can interconnect an entire nation or nations. Due to the long distances an optical path may have to cover, regeneration of the optical signal is unavoidable.

The cost of a regenerator is a significant part of a network cost. In this work, the regenerator placement problem was addressed with the objective of minimizing the number of per-wavelength regenerators. This required solving a RWA problem, where it was assumed that wavelength conversion could only be performed on a network nodes with a regenerator for the input wavelength.

Two efficient heuristics for the impairment-aware placement and minimization of the number of regenerators in an optical transport networks were proposed. The impairment threshold considered was the optical reach (identical for all fibres). The first heuristic [18] does not take into account traffic distribution in the network; The second heuristic, which is an improvement from the first, uses load costs [10] to achieve a better in-use capacity distribution, which results in a smaller number of regenerators for the same set of demands.

ILP formulations were adapted/extended to take into account capacity constraints, the regenerators wavelength conversion capability and bidirectional traffic. Then, they were used to evaluate the performance of the proposed heuristics.

Four variants of the ILP formulation for the RWARP problem were also compared and it was concluded that the addition of wavelength conversion capabilities to regenerators in network nodes, when capacity constraints are considered, may speed up the ILP. However, the ILPs were only able to solve small problems in a reasonable amount of time.

The improved version of the RWARP heuristic was shown to obtain solutions which were optimal or very close to optimal in the tested problems, in a very short time.

Significant traffic losses may occur due to single fibre cuts. Optical transport networks

have evolved to have a mesh topology that have embedded recovery procedures, mainly protection mechanisms. Dedicated protection is expensive, and considering that single fibre failures are the most frequent event, sharing resources among backup paths that protect fibre disjoint active paths seems to be an attractive trade-off between resilience and cost. Hence, this work, also addressed the survivable version of the RWARP problem, designated SRWARP, where path-based recovery was utilized. Three recovery approaches were considered: *dedicated-dedicated*, *dedicated-shared* and *shared-shared*.

The SRWARP heuristic (*dedicated-shared*) did not seem to be very effective sharing regenerators. However, for the tested cases, the optimal solutions did not present a significant reduction in the number of regenerators with respect to the *dedicated-dedicated* variant either. Nevertheless, all three heuristic variants were able to solve, in a very short time, relatively large problems, that the ILP formulation would have been unable to solve if capacity constraints had been taken into account – recall that, in these variants, the ILP was used iteratively to solve each demand and the results were only considered if the final solution did not exceed any link capacity.

When backup bandwidth sharing was considered (*shared-shared*) variant, the heuristic presented some reduction in the number of regenerators (for two of the four tested networks) and, as expected, a decrease in the capacity used by the backup paths.

The results of the integration of RWARP in a multi-layer grooming proposed in [20], can be found in that work. The output of that work is a set of transparent lightpaths that were then protected (segment-base protection) with backup bandwidth sharing.

Regarding future work, more networks could be considered for confirming the trend of the presently obtained results. Also, one could use Bhandari's algorithm [5] before the first iteration of the SRWARP heuristic, to verify the existence of a topological solution, to reduce the CPU time. Finally, the proceedings to encourage the re-use of shareable regenerators needs to be improved.

Appendices

Appendix A

Dijkstra's Shortest Path Algorithm

One of the most common problems in network operation and management is the Shortest Path Problem. This problem concerns the optimal way to traverse a network from one point, the source, to another one, the sink. This optimal way may refer, for instance, to the cheapest, the fastest or the most reliable way to traverse the network between two points.

Dijkstra's algorithm is a well known algorithm for the computation of the Shortest Path and one that is widely used. This algorithm finds the shortest paths from a source node s to all other nodes in a network with non-negative arc lengths (or costs). In [1], it is stated that the worst case scenario of the Dijkstra's algorithm implementation has a complexity of $O(n^2)$. Nevertheless there a few more efficient implementations of this algorithm, using dynamic structures, that allow to reduce this complexity.

In the Dijkstra's algorithm it is defined that each node maintains a distance label $d(i)$ to the source node s. Initially, no path is known from s to every node and the distance label is infinity, except for the source node which as a distance 0 to itself. Also, two sets of nodes are defined: permanently labeled and temporarily labeled nodes. For the nodes in the first set the distance label contains the minimum distance from s (the source) to that node. For the second set of nodes the value of the label is an upper bound for the distance of the shortest path from s to that node, meaning that it can be the shortest distance or it can still be reduced. Besides the distance label each node also maintains the predecessor node index. The predecessor is the node that precedes the current node on the shortest path. Note that while the node or all its adjacent nodes are not permanently labeled this predecessor node may change. Thus, one can state that the Dijkstra's algorithm maintains a directed out-tree rooted at the source that spans the nodes with finite distance labels.

The main step of the algorithm will now be described. The algorithm selects, from the set of temporarily labeled nodes, the one with minimum distance label, and this node (let it be designated by v) becomes permanently labeled. Then the algorithm evaluates if any of the temporarily labeled nodes, adjacent to v, can have this distance to s reduced. If that is the case, label distance is updated and v becomes the node predecessor. This procedure is repeated until all nodes are permanently labeled.

From this very short description one verifies that the main step of the algorithm must be able to find among the temporarily labeled nodes the one with minimum distance to s. If no additional structure is used, this will require $\mathcal{O}(n)$ operations, for each labeled node. The following subsection presents the d-heap - and the improvements achieved, in terms of complexity, with its use on the algorithm.

d-Heap And The Dijkstra's Algorithm

This section will introduce the d-heap as a data structure with the corresponding manipulation functions in order to evaluate the possibilities of improving the Dijkstra's Algorithm efficiency using this type of structure.

An heap or priority queue is a structure that adds to each element a key value. This key is a dynamic real number and will define how the object associated with this key will be stored.

A d-heap may be seen as a rooted tree of nodes whose arcs represent a predecessor-successor relationship. Thus we can define pred(i) as the predecessor of node i in the d-heap and succ(i) the set of successors of the node i. Note that these successors and predecessors of node i in the d-heap may not be the same as the successors and predecessors of the node in the Dijkstra's algorithm. Also important to clarify is that the root node has no predecessor, that is, $pred(i) = -1$, for $i = root$.

The d in d-heap defines the number of successors a node can have. These successors are known as siblings. In the heap, the nodes are added in an increasing depth order and from left to right – *contiguity property* -, where the depth is defined as the number of arcs in the unique path from node i to the root. Even though the addition of nodes follows the contiguity property, a d-heap needs to sort it's nodes to satisfy the *order property*. The

order property states that the key of node i in the heap is less than or equal to the key of each of its successors. That is, for each node i, $key(i) \leq key(j)$ for every j in succ(i). As an important consequence of this last property, it is possible to state that at the root of a d-heap lies the node with the smallest key.

The order property may be temporarily violated during a heap operation. Nonetheless, at the end of the operation the heap will be restored to conform to this property. Thus, the heap structure has a set of procedures that are used for this purpose:

- *swap(i,j)*: swaps the object i with the object j in the heap. This operation will be used to sort the heap and it requires $\mathcal{O}(1)$ time. This operation may still violate the order property.

- *siftup(i)* – while i is not root and its key value is smaller than the key of its predecessor then $swap(i, pred(i))$. This procedure requires $\mathcal{O}(\log_d n)$ time.

- *siftdown(i)* – while node i is not leaf and $key(i) > key(minchild(i))$ then *swap(i, minchild(i))*. Minchild in this context is the successor/child with the smallest key. This procedure requires $\mathcal{O}(d \log_d n)$ time.

The previous procedures will be used in the context of the following d-heap operations in order to provide the functionalities that will allow this dynamic structure to be used on the Dijkstra's algorithm:

- *insert(i)* – This operation will insert the object i into the heap at the last position. Note that this operation can violate the heap order property. Thus, after the insertion, this operation also performs a $siftup(i)$ operation to re-order the heap and therefore requiring $\mathcal{O}(\log_d n)$ time.

- *decreaseKeyValue(i, value)* – Decreases the key value of i to the value *value*. This may require a $siftup(i)$ operation, consuming $\mathcal{O}(\log_d n)$ time for this operation.

- *deleteMin()* – Simply returns the min key value node – the root – while also removing it from the heap. After removing, the last element of the heap will became the root, using swap, followed by $siftdown$ on that node to restore the order property. Given that the $siftdown$ operation requires $\mathcal{O}(d \log_d n)$ time and the other two operations $\mathcal{O}(1)$, this operation requires $\mathcal{O}(d \log_d n)$ time.

- *delete(i)* – removes the object i from the Heap then follows by swapping the last node to the i position followed by the $siftdown$ to restore the order.

- *increaseKeyValue(i,value)* – Increases the key value of node i to *value* and performs the $siftdown$ operation.

A heap can be used in Dijkstra's algorithm by defining the heap objects as nodes and the value field as the distance label $d(i)$. With this setup and using graph theory notation, the algorithm is implemented as presented in procedure 10 where a binary heap (2-heap) was used for its simplicity of implementation. The implementation starts by initializing the binary heap ($bHeap$) and the permanently labeled node set/array. The heap structure will be used to keep the temporarily labeled nodes ordered by the distance label: the root will contain the min distance value node. Lines 4 to 6 set up the distance labels of all nodes as previously described in the main step. Line 7 inserts the source into the heap before heading to the main iteration cycle, in order to access the adjacent nodes further on. Each iteration of the cycle will permanently label a node until the heap is empty (all nodes have been permanently labeled). Lines 9 and 10 remove the min distance node, referred as $minNode$ from the heap and add it to the permanently labeled node array *permLabeled*. Note that this array is of type DijkstraNodeArray, where each DijkstraNode contains the distance to s and a reference to the predecessor node that is set throughout the algorithm.

Each time a new $minNode$ is selected from the $bHeap$, all its adjacent nodes (with temporary labels), $nodeI$ in line 11, will be accessed and checked if $d(nodeI)$ is higher than the sum of $d(minNode)$ with the arc cost - value computed at line 13. If $nodeI$ current distance label is higher, and if it is the first time (condition in line 15 resolves as true) the distance label of the node is set, the $pred(nodeI)$ is set to $minNode$ and $nodeI$ is then added to the heap - lines 16 to 18. If $nodeI$ was previously labeled, then the node is already in the heap which will only require to update the predecessor to be the $minNode$ and the reduction of the distance value label - lines 20 and 21. If $nodeI$ distance value is lower than the distance computed at line 13 then nothing is changed.

Note that the previous implementation may be used to compute the shortest path distance from a source node s to any destination node d if, after adding $minNode$ at line 10, it is checked whether $minNode$ is node d and if that proves to be the case then d is stored at the last slot of *permLabeled* with the distance label and predecessor set and thus the procedure

may then return *permLabeled*.

In [1] it is shown that the Dijkstra's algorithm implemented with a binary heap runs in $\mathcal{O}(m \log n)$ which greatly improves efficiency for sparse networks but may be slower than the original implementation ($\mathcal{O}(n^2)$) otherwise.

Algorithm 10 Dijkstra's algorithm using a binary heap

```
1:  procedure DIJKSTRA(G(N, A), NODE s)
2:      bHeap ← new BHeap();
3:      permLabeled ← new DijkstraNodeArray();          ▷ The permanently labeled nodes
4:      d(j) ← ∞,   ∀j ∈ N;
5:      d(s) ← 0;
6:      pred(s) ← −1;                                   ▷ Source has no predecessor
7:      bHeap.insert(s);
8:      while bHeap.size > 0 do
9:          minNode ← bHeap.deleteMin();
10:         permLabeled.add(minNode);                   ▷ Stores minNode
11:         for each nodeI ∈ N+(minNode) do             ▷ N+ is the adjacency node list
12:             if (nodeI ∉ permLabeled) then           ▷ nodeI is not permanently labeled
13:                 possibleNewMinCost ← minNode.getCost() + arcCost(minNode, nodeI);
14:                 if (nodeI.getCost() > possibleNewMinCost) then
15:                     if (nodeI.getCost() = ∞) then   ▷ Set temporarily label
16:                         nodeI.setCost(possibleNewMinCost);   ▷ Update nodeI distance
17:                         nodeI.setPred(minNode);     ▷ Update nodeI predecessor
18:                         bHeap.insert(nodeI);        ▷ Node is inserted into the heap
19:                     else                            ▷ nodeI was already temporarily labeled
20:                         nodeI.setPred(minNode);     ▷ Update predecessor
21:                         bHeap.decreaseKeyValue(nodeI, possibleNewMinCost);
                                ▷ KeyValue is also the cost/distance of the node
22:                     end if
23:                 end if
24:             end if
25:         end for
26:     end while
27:     return permLabeled;
28: end procedure
```

Appendix B

Improved RWARP heuristic

This appendix presents the improvements stated in subsection 3.2.5 for the RWARP heuristic. Algorithm 11 and Algorithm 12 illustrate these enhancements.

Algorithm 11 Improved RWARP heuristic – *solveRwarpImproved*($\mathcal{G}(\mathcal{N}, \mathcal{A})$, Δ, $\mathcal{I}$)

Input: A Graph $\mathcal{G}(\mathcal{N}, \mathcal{A})$, an impairment threshold Δ and an array of demands $\mathcal{I}$

Output: $P_{\mathcal{I}}^{\lambda} = \{p_{i,sd}^{\lambda} : i \in \mathcal{I}\}$, the set of optical paths $p_{i,sd}^{\lambda}$ where i is an index of a demand from $\mathcal{I}$, and s and d the source and destination of i, respectively. A given $p_{i,sd}^{\lambda}$ may be *null*, due to the lack of residual capacity in the network

1: $P_{\mathcal{I}}^{\lambda} \leftarrow \emptyset$
2: Sort $\mathcal{I}$
3: **for** $i \in \mathcal{I}$ **do**
4: For each pair of nodes $u, v \in \mathcal{N}$, compute all paths p_{uv}^{k}, where $k = 1 \ldots K_{uv}$ is an index to this paths, such that $c(p_{uv}^{k}) \leq \Delta$, $\forall k$ and each $arc \in \mathcal{A}_{p_{uv}^{k}}$ has at least a free wavelength. Select, from the K_{uv} paths, the final u to v path as the path whose cost, according to the Fortz capacity cost function, is minimal. That is, let p_{uv}^{*} be the selected path, $C_F(p_{uv}^{*}) \leq C_F(p_{uv}^{k})$, $\forall k$
5: Create a graph $\mathcal{G}'(\mathcal{N}, \mathcal{A}')$ where $\mathcal{A}'$ contains the arcs (u, v) where each arc of cost $1 + (|\mathcal{A}_{p_{uv}^{*}}|/m)$ logically represents a path p_{uv}^{*}
6: Find all paths of minimum cost $p_{sd}^{'k}$ in $\mathcal{G}'$, where k is used as an index for these paths. If no path was found, **go to** step 4 for the following demand
7: Expand the sequence of arcs of $p_{sd}^{'k}$ to the corresponding sequence of paths p_{uv}^{*} in $\mathcal{G}$ to obtain p_{sd}^{k}
8: Remove possible existing loops in all p_{sd}^{k} paths
9: Remove identical paths, if any, from the paths obtained in step 8 to obtain $\hat{P}_{sd}^{k} = \{p_{sd}^{k}, k = 1, \ldots, K_{sd}\}$ where K_{sd} is the number of the remaining paths
10: $p_{i,sd}^{\lambda} \leftarrow tryWavelengthAssignmentWithFortz(\hat{P}_{sd}^{k})$ ▷ See Algorithm 12
11: **if** $p_{i,sd}^{\lambda} = null$ **then** ▷ Wavelength assignment attempt failed
12: $p_{i,sd}^{\lambda} \leftarrow assignWavelength(\hat{P}_{sd}^{k})$
13: **end if**
14: $P_{\mathcal{I}}^{\lambda} \leftarrow P_{\mathcal{I}}^{\lambda} \cup \{p_{i,sd}^{\lambda}\}$ ▷ Add new optical path to the solution set
15: **end for**
16: **return** $P_{\mathcal{I}}^{\lambda}$

Algorithm 12 $tryWavelengthAssignmentWithFortz(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,\hat{P}^k_{sd})$

Input: $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and a set of candidate paths $\hat{P}^k_{sd}$

Output: An optical path p^λ_{sd} such that its corresponding topological path is the first element of $\hat{P}^k_{sd}$ for which a successful wavelength assignment was possible; if the wavelength assignment failed for every path in $\hat{P}^k_{sd}$ then the output is a *null* path

1: Sort $\hat{P}^k_{sd}$ by non-decreasing Fortz path cost function
2: **for each** $p_{sd} \in \hat{P}^k_{sd}$ **do**
3: Create consecutive maximum distance segments from s to d $(p_{su_1},\ p_{u_1u_2},\dots p_{u_kd})$ respecting Δ and try to assign a wavelength to each of these segments. Let $\xi_w = (p_w,\lambda_w),\ \forall w \in \{su_1,u_1u_2,\dots,u_kd\}$ if a wavelength was successfully assigned to a p_w or *null* otherwise
4: **if** $\xi_w \neq null,\ \forall w$ **then**
5: $p^\lambda_{sd} \leftarrow \langle \xi_{su_1},\xi_{u_1u_2},\dots,\xi_{u_kd}\rangle$
6: Place regenerators at the last node of each lightpath $\xi_w\ \forall w \neq u_kd$, for the assigned wavelength λ_w
7: **return** p^λ_{sd}
8: **end if**
9: **end for**
10: **return** *null* ▷ No wavelength assignment was possible

Appendix C

Auxiliary SRWARP methods

An iterative two step approach was used in Algorithm 4, which required a set of candidate topological paths and an algorithm for assigning wavelengths to a single candidate path. Therefore, two algorithms were extracted from Algorithm 11.

Algorithm 13 Routing phase of SRWARP – $srwarpRoute(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,i,\delta)$

Input: A Graph $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ, a single demand i from node s to d, and δ (an integer increment to relax the minimum logical path cost)

Output: P_i, the set of topological paths for demand i ($\emptyset$, if no solution was found)

1: For each pair of nodes $u, v \in \mathcal{N}$, compute all paths p_{uv}^k, where $k = 1 \ldots K_{uv}$ is an index to this paths, such that $c(p_{uv}^k) \leq \Delta$, $\forall k$ and each $arc \in \mathcal{A}_{p_{uv}^k}$ has at least a free wavelength. Select, from the K_{uv} paths, the final u to v path as the path whose cost, according to the Fortz capacity cost function, is minimal. That is, let p_{uv}^* be the selected path, $C_F(p_{uv}^*) \leq C_F(p_{uv}^k)$, $\forall k$

2: Create a graph $\mathcal{G}'(\mathcal{N},\mathcal{A}')$ where $\mathcal{A}'$ contains the arcs (u,v) where each arc of cost $1 + (|\mathcal{A}_{p_{uv}^*}|/m)$ logically represents a path p_{uv}^*

3: Find all paths $p_{sd}^{'k}$ in $\mathcal{G}'$, where k is used as an index for these paths, with a cost less or equal to the minimum cost path plus δ.

4: **if** no path was found in step 3 **then**

5: **return** $\emptyset$

6: **end if**

7: Expand the sequence of arcs of $p_{sd}^{'k}$ to the corresponding sequence of paths p_{uv}^* in $\mathcal{G}$ to obtain p_{sd}^k

8: Remove possible existing loops in all p_{sd}^k paths

9: Remove identical paths, if any, from the paths obtained in step 8 to obtain $\hat{P}_{sd}^k = \{p_{sd}^k, k = 1, \ldots, K_{sd}\}$ where K_{sd} is the number of the remaining paths

10: **return** $\hat{P}_{sd}^k$

Algorithm 13, corresponds (approximately) to steps 4 to 9 and calculates the set of candidate topological paths. Note that the routing phase of Algorithm 13 extends the

routing phase of Algorithm 11 by allowing active paths to contain more than the minimum number of regenerators. This is achieved using the parameter δ – see step 3. This reduces the likelihood of falling into a trap (being unable to find a pair of disjoint paths when it exists). Algorithm 14, receives a topological path as input and tries to create the corresponding optical path.

Algorithm 14 Wavelength assignment phase of SRWARP – $crtOptPath(\mathcal{G}(\mathcal{N},\mathcal{A}),\Delta,p_{sd})$

Input: A Graph $\mathcal{G}(\mathcal{N},\mathcal{A})$, Δ and p_{sd}, a topological path

Output: p^{λ}_{sd} an optical path, such that its topological is the input path p_{sd}

1: $p^{\lambda}_{i,sd} \leftarrow tryWavelengthAssignmentWithFortz(\{p_{sd}\})$ ▷ See Algorithm 12
2: **if** $p^{\lambda}_{i,sd} = null$ **then** ▷ Wavelength assignment attempt failed
3: $\quad p^{\lambda}_{i,sd} \leftarrow assignWavelength(\{p_{sd}\})$
4: **end if**
5: **return**

Appendix D

SNDlib file for polska*

In section 4.3 a subset of the polska network was used, denominated polska*. Below, the network definition file, in SNDlib format, is presented, where the "Admissible Paths" section was removed as it was not used through this thesis.

```
?SNDlib native format; type: network; version: 1.0
# network polska

# NODE SECTION
#
# <node_id> [(<longitude>, <latitude>)]

NODES (
  Gdansk ( 18.60 54.20 )
  Bydgoszcz ( 17.90 53.10 )
  Kolobrzeg ( 16.10 54.20 )
  Krakow ( 19.80 50.00 )
  Bialystok ( 23.10 53.10 )
  Rzeszow ( 21.90 50.00 )
  Warsaw ( 21.00 52.20 )
)

# LINK SECTION
#
# <link_id> ( <source> <target> ) <pre_installed_capacity> <pre_installed_capacity_cost>
# <routing_cost> <setup_cost> ( {<module_capacity> <module_cost>}* )

LINKS (
  Link_0_10 ( Gdansk Warsaw ) 0.00 0.00 0.00 156.00 ( 155.00 156.00 622.00 468.00 )
  Link_0_2 ( Gdansk Kolobrzeg ) 0.00 0.00 0.00 272.00 ( 155.00 272.00 622.00 816.00 )
  Link_1_2 ( Bydgoszcz Kolobrzeg ) 0.00 0.00 0.00 156.00 ( 155.00 156.00 622.00 468.00 )
  Link_1_10 ( Bydgoszcz Warsaw ) 0.00 0.00 0.00 272.00 ( 155.00 272.00 622.00 816.00 )
  Link_4_8 ( Krakow Rzeszow ) 0.00 0.00 0.00 250.00 ( 155.00 250.00 622.00 750.00 )
  Link_4_10 ( Krakow Warsaw ) 0.00 0.00 0.00 324.00 ( 155.00 324.00 622.00 972.00 )
  Link_5_8 ( Bialystok Rzeszow ) 0.00 0.00 0.00 324.00 ( 155.00 324.00 622.00 972.00 )
```

```
  Link_5_10 ( Bialystok Warsaw ) 0.00 0.00 0.00 250.00 ( 155.00 250.00 622.00 750.00 )
  Link_0_5 ( Gdansk Bialystok ) 0.00 0.00 0.00 294.00 ( 155.00 294.00 622.00 882.00 )
)

# DEMAND SECTION
#
# <demand_id> ( <source> <target> ) <routing_unit> <demand_value> <max_path_length>

DEMANDS (
  Demand_0_1 ( Gdansk Bydgoszcz ) 1 195.00 UNLIMITED
  Demand_0_2 ( Gdansk Kolobrzeg ) 1 158.00 UNLIMITED
  Demand_0_4 ( Gdansk Krakow ) 1 101.00 UNLIMITED
  Demand_0_5 ( Gdansk Bialystok ) 1 198.00 UNLIMITED
  Demand_0_8 ( Gdansk Rzeszow ) 1 154.00 UNLIMITED
  Demand_0_10 ( Gdansk Warsaw ) 1 122.00 UNLIMITED
  Demand_1_2 ( Bydgoszcz Kolobrzeg ) 1 179.00 UNLIMITED
  Demand_1_4 ( Bydgoszcz Krakow ) 1 105.00 UNLIMITED
  Demand_1_5 ( Bydgoszcz Bialystok ) 1 159.00 UNLIMITED
  Demand_1_8 ( Bydgoszcz Rzeszow ) 1 166.00 UNLIMITED
  Demand_1_10 ( Bydgoszcz Warsaw ) 1 137.00 UNLIMITED
  Demand_2_4 ( Kolobrzeg Krakow ) 1 159.00 UNLIMITED
  Demand_2_5 ( Kolobrzeg Bialystok ) 1 164.00 UNLIMITED
  Demand_2_8 ( Kolobrzeg Rzeszow ) 1 130.00 UNLIMITED
  Demand_2_10 ( Kolobrzeg Warsaw ) 1 173.00 UNLIMITED
  Demand_4_5 ( Krakow Bialystok ) 1 124.00 UNLIMITED
  Demand_4_8 ( Krakow Rzeszow ) 1 144.00 UNLIMITED
  Demand_4_10 ( Krakow Warsaw ) 1 119.00 UNLIMITED
  Demand_5_8 ( Bialystok Rzeszow ) 1 140.00 UNLIMITED
  Demand_5_10 ( Bialystok Warsaw ) 1 104.00 UNLIMITED
  Demand_8_10 ( Rzeszow Warsaw ) 1 181.00 UNLIMITED
)
```

Appendix E

Results of varying $\delta_{\max}$ in the SRWARP heuristic

The following results represent the dependence of the SRWARP heuristic and the input value δ_{max}, for the *dedicated-dedicated* (tables E.1 to E.3) and *dedicated-shared* (tables E.4 to E.6) variants.

Input data				**Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 0$ Heuristic**						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	98	4.28	7.41	13	7	0.19
	42			42	196	8.56	14.93	27	14	0.22
polska	66	1000	96	66	300	8.51	12.56	32	18	1.22
	132			132	600	18.06	23.90	60	36	2.18
abilene*	55	3000	96	50	322	8.93	14.66	28	61	0.47
	110			99	634	17.56	29.02	55	119	0.86
nobel germany*	121	1000	96	121	900	17.11	18.79	67	208	2.82

Table E.1: Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 0$, run on PC1 (see section 4.1)

Input data				**Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 1$ Heuristic**						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	98	4.28	7.41	13	7	0.15
	42			42	196	8.56	14.93	27	14	0.21
polska	66	1000	96	66	300	8.51	12.56	32	18	1.20
	132			132	600	18.06	23.90	60	36	2.21
abilene*	55	3000	96	53	346	9.90	15.33	30	67	0.52
	110			106	692	19.79	30.65	60	134	0.90
nobel germany*	121	1000	96	121	900	17.11	18.79	67	208	2.84

Table E.2: Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 1$, run on PC1 (see section 4.1)

Input data				**Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 2$ Heuristic**						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	98	4.28	7.41	13	7	0.14
	42			42	196	8.56	14.93	27	14	0.21
polska	66	1000	96	66	300	8.51	12.56	32	18	1.18
	132			132	600	18.06	23.90	60	36	2.16
abilene*	55	3000	96	55	366	10.79	16.00	32	73	0.56
	110			110	732	21.58	31.99	64	146	0.94
nobel germany*	121	1000	96	121	900	17.11	18.79	67	208	2.79

Table E.3: Results for the *dedicated-dedicated* SRWARP variant with $\delta_{\max} = 2$, run on PC1 (see section 4.1)

Input data				**Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 0$ Heuristic**						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	98	4.28	7.41	13	7	0.15
	42			42	196	8.56	14.93	27	14	0.22
polska	66	1000	96	66	300	8.51	12.56	32	18	1.35
	132			132	600	18.06	23.90	60	36	2.15
abilene*	55	3000	96	50	322	8.93	14.66	28	61	0.47
	110			99	634	17.56	29.02	55	119	0.82
nobel germany*	121	1000	96	121	900	17.11	18.83	67	206	2.80

Table E.4: Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 0$, run on PC1 (see section 4.1)

Input data				**Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 1$ Heuristic**						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	98	4.28	7.41	13	7	0.15
	42			42	196	8.56	14.93	27	14	0.22
polska	66	1000	96	66	300	8.51	12.56	32	18	1.22
	132			132	600	18.06	23.90	60	36	2.40
abilene*	55	3000	96	53	346	9.90	15.33	30	67	0.65
	110			106	692	19.79	30.65	60	134	0.96
nobel germany*	121	1000	96	121	900	17.11	18.83	67	206	2.78

Table E.5: Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 1$, run on PC1 (see section 4.1)

Input data				**Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 2$ Heuristic**						
Network	$\lvert\mathcal{I}\rvert$	Δ	$\lvert\lambda\rvert$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage For APs (%)**	**Capacity Usage For BPs (%)**	**Max in Use λ**	**Regen Placed**	**Time(s)**
polska*	21	1000	96	21	98	4.28	7.41	13	7	0.15
	42			42	196	8.56	14.93	27	14	0.22
polska	66	1000	96	66	300	8.51	12.56	32	18	1.19
	132			132	600	18.06	23.90	60	36	2.27
abilene*	55	3000	96	55	366	10.79	16.00	32	73	0.73
	110			110	732	21.58	31.99	64	146	1.02
nobel germany*	121	1000	96	121	900	17.11	18.83	67	206	2.78

Table E.6: Results for the *dedicated-shared* SRWARP variant with $\delta_{\max} = 2$, run on PC1 (see section 4.1)

Appendix F

Accepted paper

Impairment-aware optimization strategies to dimension optical transport networks with minimal regeneration requirements

Ricardo Mendes*, Bruna Nogueira†, Teresa Gomes‡, Lúcia Martins§ and João Santos¶
*†‡§Department of Electrical and Computer Engineering
University of Coimbra, Pinhal de Marrocos 3030-290 Coimbra
Email: *rmendes@student.uc.pt, †ban@student.uc.pt, ‡teresa@deec.uc.pt and §lucia@deec.uc.pt
‡§Institute for Systems Engineering and Computers at Coimbra
Pólo II, R. Silvio Lima, 3030-290 Coimbra
¶Coriant Portugal
Rua Irmãos Siemens, 1-1A 2720-093 Amadora
Email: joao.m.santos@coriant.com

***Abstract*—Physical impairments restrain the maximum length a signal can travel without regeneration. The quality of a signal in optical wavelength division multiplexing (WDM) networks must thus be restored with opto-electro-optical (OEO) regeneration in order to reach its destination. As OEO regenerators are costly devices, sparse deployment using routing optimization is the key to reduce the network cost. This paper tackles the problem of impairment aware routing and wavelength assignment with regeneration placement (RWARP), considering capacity constraints, while focusing on minimizing the number of regenerators. We will firstly, extend an Integer Linear Programming (ILP) formulation for the RWARP problem which provides, when feasible, an optimal solution, and secondly, propose an efficient heuristic. Results show that the heuristic provides satisfactory results, optimal for small problems, in a fraction of the ILP execution time.**

I. Introduction

Optical wavelength division multiplexing (WDM) networks have been established as the backbone networks to face the increase in bandwidth demand of today's applications. In this type of networks, the signals travel through lightpaths which may span multiple consecutive fibres on a given wavelength. Physical impairments in the fibres degrade the signal quality as the propagation distance increases. This degradation occurs at the wavelength level and may lead to unacceptable bit error rates (BER). In order to recover its quality, regeneration at the network nodes may be needed so that the signal reaches its destination with a BER below a given threshold. Lightpaths that have regeneration at the intermediate nodes (and consequently conversion to the electrical domain) are referred to as translucent lightpaths as opposed to transparent lightpaths that are unregenerated segments. Optical networks regenerators regenerate the signal per wavelength and thus a node may have multiple regenerators. Opto-electro-optical(OEO) regenerators are still the practical choice as all-optical regenerators are still under development [1], [2]. Since OEO regenerators are expensive devices (CAPEX) and have an high power consumption (OPEX), minimizing the number of these elements in the network greatly reduces the network costs. The objective is thus to route the network traffic in a manner that the minimum number of regenerators is needed. Many strategies for Impairment Aware Routing and Wavelength Assignment and Regenerator Placement (IA-RWARP) have been proposed, in order to minimize their deployment throughout the network, where single [3]–[5] and multiple [6]–[10] impairment metrics were taken into consideration.

The contribution of this paper is an extension and modification of the ILP formulation in [5], where we introduced capacity and bidirectional traffic considerations and wavelength conversion capabilities in the network, without considering path protection. To tackle the RWARP problem for large networks, we also extend the heuristic Exact Single Request Regenerator Placement (ESRRP) in [6] to take into account the added constraints of the problem. The networks used on the study were backbone networks and thus the demands will be considered bidirectional symmetric. We will also assume that the regenerators are bidirectional devices. Results are presented for several networks showing the effectiveness of the presented heuristic.

The remainder of the paper is organized as follows. In Section II we present a brief review on the RWARP problem. In Section III we present our ILP formulation and the heuristic formulation. In Section IV we show our simulation data and results and we conclude in Section V.

II. Related Work

Earlier works on routing and wavelength assignment in optical networks considered ideal physical layer conditions [11]–[13] and thus, the regeneration placement was not taken into account. Nowadays, even with the advance in the fibre technologies, current line rates of 40Gb/s and 100Gb/s with over 160 wavelengths [14] demand high transmission energy

which pose a serious limitation on the distance a signal can travel without regeneration.

Both linear and nonlinear impairments have been defined and modeled (cf. [15]) and several multiple Impairment Aware RWARP (IA-RWARP) approaches have been studied [6]–[10]. A good survey on the subject may be found in [16]. As the impairment constraints highly depend on the network architecture [15] (fibre characteristics, transmission and switching equipment, channel speed, etc.), studies often simplify these constraints to a single impairment metric [3]–[5] where the terms Quality of Transmission (QoT) and Optical Reach ($\mathcal{R}$) are used as metric. The QoT is often a function of the linear impairments added to an overestimation of the nonlinear impairments whereas the Optical Reach is the maximum distance a signal can travel without regeneration. In [5], Beshir and Kuipers $et\ al.$ stated that for regenerator placement problems, single impairment metric representing the worst impairment among all the impairments on a link suffices. They also note that the distance may be also used as a single metric as it is a good indicator of the signal quality.

The IA-RWARP approaches have two main variants. The first variant, which was taken as a first approach [10], [17], [18], is to find feasible routes for the demands while minimizing the number of regeneration nodes (where a node can have several regenerators). On the other hand, the second variant focus on minimizing the number of regenerators [3], [5]–[9]. In [3], Rahman $et\ al.$ demonstrated intuitively that the impairment aware routing in optical networks should focus on minimizing the number of regenerators rather than minimizing the number of nodes with regeneration as by doing so, the total number of regenerators can be minimal.

III. RWARP Problem

Throughout this section we will focus on the WDM Impairment Aware Routing and Wavelength Assignment and Regenerator Placement (IA-RWARP) problem which we will simply call RWARP. Given a network with single fibre links (in fact, each link will have a fibre pair to support traffic in both directions) and a static set of demands, the objective is to route each request through the network by allocating feasible lightpaths while minimizing the number of required regenerators. Bidirectional traffic will be considered and we will assume that each demand fully occupies a single wavelength on each link. That is, a transparent lightpath (or simply, a lightpath) will be a segment of a single end-to-end path for a given request. Of course, if no regeneration is required, the end-to-end path will coincide with one single lightpath. We will also assume that every fibre has the same number of wavelengths. Furthermore, the metric used for the impairment of each link is the length that an unregenerated segment can have (with a maximum of Δ kilometres, where Δ is the optical reach). The regeneration will be per wavelength – selective regeneration – and each regenerator is capable to shift the input wavelength to any output wavelength – wavelength conversion capability.

Beshir and Kuipers $et\ al.$ in [5] provided an exact Integer Linear Programming (ILP) formulation for the RWARP problem with link-disjoint dedicated path protection without wavelength conversion. In this section we present a new formulation based on their work with the following modifications to conform the aforementioned considerations:

- The need for path protection was suppressed;
- Capacity and Bidirectional traffic restrictions were added;
- Replacement of a constraint to ensure that regenerators can introduce wavelength conversion capability in the network.

A. Notation

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ denote the physical directed network where $\mathcal{N}$ is the set of n nodes composing the graph and $\mathcal{A}$ the set of m arcs connecting the nodes. An arc is referred as an ordered pair (i, j) where i and j are the nodes connected by the arc, with $i, j \in \mathcal{N}$ and $(i, j) \in \mathcal{A}$. $\mathcal{A}^+(i) = (i, j) \in \mathcal{A} : j \in \mathcal{N}$ denotes the set of outgoing arcs from node i and $\mathcal{A}^-(i) = (j, i) \in \mathcal{A} : j \in \mathcal{N}$ denotes the set of incoming arcs. The cost of an arc (i, j) is represented by c_{ij}.

In $\mathcal{G}$, a topological path p (or simply a path, when there is no ambiguity) from a source node s to a destination node d is defined as a sequence of nodes $p_{sd} = \langle s, u_0, u_1, \ldots, u_{w-1}, d\rangle$ where $s, d, u_k \in \mathcal{N}$ for $k = 0, 1, \ldots, w-1$. The set of nodes in a path p_{sd} shall be referred to as $\mathcal{N}_{p_{sd}}$ and the set of arcs $\mathcal{A}_{p_{sd}}$. A path p_{sd} may also be defined as a concatenation of sub-paths such that $p_{sd} = p_{su} \diamondsuit p_{ud}$, that is, p_{sd} coincides with path p_{su} from s to u and with p_{ud} from u to d. The number of arcs in p_{sd} shall be called $|\mathcal{A}_{p_{sd}}|$ and the cost of the path $c(p_{sd}) = \sum_{(i,j)\in p_{sd}} c_{ij}$. A set of paths is referred to as P where, more specifically, P_{sd} is a set of paths from a node s to a node d.

Let $\mathcal{I}$ denote the set of demands where a single demand i is defined by its source node $s_i \in \mathcal{N}$ and the destination node $d_i \in \mathcal{N}$. The topological end-to-end path of a demand i is referred to as $p_{i,sd}$. The optical path of the same demand i, $p^{\lambda}_{i,sd}$, is defined as a sequence $p^{\lambda}_{i,sd} = \langle lp_{sv_0}, lp_{v_0v_1}, \ldots, lp_{v_{w'-1}d}\rangle$ where lp_{xy} is a lightpath from $x \in \mathcal{N}$ to $y \in \mathcal{N}$ on some wavelength λ_{xy}. That is, a lightpath is defined as $lp_{xy} = (p_{xy}, \lambda_{xy})$ where p_{xy} is the topological path of the lightpath and λ_{xy} is the assigned wavelength. The topological path of $p^{\lambda}_{i,sd}$ is simply the concatenation of the paths in the lightpaths belonging to $p^{\lambda}_{i,sd}$, that is $p_{i,sd} = p_{sv_0} \diamondsuit p_{v_0v_1} \diamondsuit \ldots \diamondsuit p_{v_{w'-1}d}$.

B. Problem Formulation and Respective Justification

We present here an ILP formulation which is a modification (removal of protection) and extension (capacity and bidirectional traffic constraints and wavelength conversion capability at the regenerators) of the dedicated-dedicated ILP formulation proposed in [5].

The underlying network graph will be represented by a directed graph. The links will be transformed into two symmetrical arcs. That is, each link (i, j) is replaced by the arcs $l = (i, j)$ and $l' = (j, i)$ of equal capacity. Note that as this study focus on the core of the network, the assumptions that l and l' have equal capacity and that all demands are bidirectional and symmetric are realistic.

Indices:

$i = 0, \ldots, D-1$	Request ID.
$u, v = 0, \ldots, N-1$	Node ID.
$l, l' = 0, \ldots, L-1$	Arc ID.
$\lambda = 0, \ldots, W-1$	Wavelength ID.
$\mathcal{A}^-(u)/\mathcal{A}^+(u)$	Incoming/outgoing arcs of node u

Binary Variables:

$x_{i,l,u,\lambda}$	Is 1 if wavelength λ on arc l is used by demand i and the last regenerator node on the path before getting to arc l is node u. Node u can also be the source node.
$\tau_{i,u,v,\lambda}$	Is 1 if the path of demand i has a regenerator at node u immediately followed by a regenerator at node v on wavelength λ. Node u can also be the source node.

Objective:

Minimize the total number of regenerators needed on the network:

$$\sum_{i} \sum_{\lambda} \sum_{u \in \mathcal{N}} \sum_{v \in \mathcal{N} \setminus \{u\}} (\tau_{i,u,v,\lambda}) \tag{1}$$

Constraints:

Flow Conservation constraints:

At the source node of each demand only a single flow (for that request) can leave the node:

$$\sum_{l \in \mathcal{A}^+(s_i)} \sum_{\lambda} x_{i,l,s_i,\lambda} = 1 \quad \forall i \tag{2}$$

where s_i is the source node of the demand i.

For the intermediate nodes of each demand i, i.e. nodes that are not the source nor the destination, the incoming flow has to match the outgoing flow regardless of that node having (or not) regenerator for the given demand:

$$\sum_{l \in \mathcal{A}^-(v)} x_{i,l,u,\lambda} - \sum_{l \in \mathcal{A}^+(v)} x_{i,l,u,\lambda} = \tau_{i,u,v,\lambda} \qquad \forall i; \forall \lambda; \\ \forall v \in \mathcal{N} \setminus \{s_i, d_i\}; \forall u \in \mathcal{N} \setminus \{v\} \tag{3}$$

where v is the intermediate node and d_i is the destination node of demand i. Note that $\tau_{i,u,v,\lambda}$ is equal to 1 when wavelength λ is used in the segment between u and v (nodes with regeneration or u as source and v with regeneration) which is a transparent lightpath associated with demand i.

If a node v has a regenerator used by demand i, the last node with a regenerator in the next segment should be node v:

$$\sum_{l \in \mathcal{A}^+(v)} \sum_{\lambda} x_{i,l,v,\lambda} - \sum_{u \in \mathcal{N} \setminus \{v\}} \sum_{\lambda} \tau_{i,u,v,\lambda} = 0 \\ \forall i; \forall v \in \mathcal{N} \setminus \{s_i, d_i\} \tag{4}$$

v is thus the *tail* of the next lightpath and the *head* of the previous lightpath (from u to v). Note that the sum for all λ's is needed as the outgoing λ may be different from the regenerated λ, as we are assuming that regenerators may be simultaneously wavelength converters. This new constraint (w.r.t. [5]) may be also stated as: if v has a regenerator for wavelength λ required by demand i (previously regenerated/originated at an upstream node u), then an outgoing flow for demand i must exist on node v, using one of the emergent arcs from v and any available wavelength on that arc.

Wavelength constraints:

At a given arc l one wavelength λ may be used, at most, by a single lightpath:

$$\sum_{i} \sum_{u \in \mathcal{N}} x_{i,l,u,\lambda} \leq 1 \quad \forall l \in \mathcal{A}; \forall \lambda \tag{5}$$

Simple Path constraints:

The end-to-end path should be a simple path, i.e. without cycles. Thus the source node of a given request should not have any incoming flow relative to that request:

$$\sum_{l \in \mathcal{A}^-(s_i)} \sum_{u \in \mathcal{N}} \sum_{\lambda} x_{i,l,u,\lambda} = 0 \quad \forall i \tag{6}$$

Also at the source node and for each demand, the outgoing flow that is not originated on this node should be 0:

$$\sum_{l \in \mathcal{A}^+(s_i)} \sum_{u \in \mathcal{N} \setminus \{s_i\}} \sum_{\lambda} x_{i,l,u,\lambda} = 0 \quad \forall i \tag{7}$$

For the intermediate nodes case, the incoming flow for a given request i should be either 1 or 0, that is only one lightpath of a give demand may be incident to the node:

$$\sum_{l \in \mathcal{A}^-(v)} \sum_{u \in \mathcal{N}} \sum_{\lambda} x_{i,l,u,\lambda} \leq 1 \quad \forall v \in \mathcal{N} \setminus \{s_i\}; \forall i \tag{8}$$

Impairment constraints:

Any lightpath must satisfy an impairment threshold of Δ:

$$\sum_{l \in \mathcal{A}} \sum_{\lambda} r(l) \cdot x_{i,l,u,\lambda} \leq \Delta \quad \forall u \in \mathcal{N}; \forall i \tag{9}$$

with $r(l)$ the distance cost of arc l. Note that $r(l) = c_{ij}$, with $l = (i, j)$.

Bidirectional Traffic constraints:

$$\sum_{i} \sum_{u \in \mathcal{N}} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda}) \leq 1 \quad \forall l; \forall \lambda \tag{10}$$

this constraint assures that if a given wavelength is in use on a given arc the symmetrical arc on the same wavelength can't be used, as it is needed (already in use) by the bidirectional traffic.

Capacity constraints:

Finally, the capacity constraint must take into account the bidirectional traffic:

$$\sum_{i} \sum_{u} \sum_{\lambda} (x_{i,l,u,\lambda} + x_{i,l',u,\lambda}) \leq C_l \quad \forall l \tag{11}$$

where C_l is the arc l capacity in terms of number of wavelengths. This constraint states that the traffic that goes through l and l' is at most C_l as the remaining capacity (recall that a pair of fibres is to be considered) will be needed for the bidirectional traffic.

Algorithm 1 RWARP heuristic

Input: A Graph $\mathcal{G}(\mathcal{N}, \mathcal{A})$, an impairment threshold Δ and an array of demands $\mathcal{I}$

Output: $P_{\mathcal{I}}^{\lambda} = \{p_{i,sd}^{\lambda} : i \in \mathcal{I}\}$, the set of optical paths $p_{i,sd}^{\lambda}$ where i is an index of a demand from $\mathcal{I}$, and s and d the source and destination of i, respectively. A given $p_{i,sd}^{\lambda}$ may be $null$, due to the lack of capacity in the network

1: $P_{\mathcal{I}}^{\lambda} \leftarrow \emptyset$
2: Sort $\mathcal{I}$ ▷ By decreasing distance of the corresponding shortest path
3: **for** $i \in \mathcal{I}$ **do**
4: For each pair of nodes $u, v \in \mathcal{N}$, compute the shortest path p_{uv}^{*} such that $c(p_{uv}^{*}) \leq \Delta$ and each $arc \in A_{p_{uv}^{*}}$ has at least a free wavelength
5: Create a graph $\mathcal{G}'(\mathcal{N}, \mathcal{A}')$ where $\mathcal{A}'$ contains the arcs (u, v) where each arc of cost $1+(|\mathcal{A}_{p_{uv}^{*}}|/m)$ logically represents a path p_{uv}^{*}
6: Find all paths of minimum cost $p_{sd}^{'k}$ in $\mathcal{G}'$, where k is used as an index for these paths. If no path was found, increment i and **go to** step 4
7: Expand the sequence of arcs of $p_{sd}^{'k}$ to the corresponding sequence of paths p_{uv}^{*} in $\mathcal{G}$ to obtain p_{sd}^{k}
8: Remove possible existing loops in all p_{sd}^{k} paths
9: Remove identical paths, if any, from the paths obtained in step 8 to obtain $\hat{P}_{sd}^{k} = \{p_{sd}^{k}, k_{sd} = 1, \ldots, K_{sd}\}$ where K_{sd} is the number of the remaining paths
10: $p_{i,sd}^{\lambda} \leftarrow tryFastWavelengthAssignment(\hat{P}_{sd}^{k})$
11: **if** $p_{i,sd}^{\lambda} = null$ **then** ▷ Wavelength assignment failed
12: $p_{i,sd}^{\lambda} \leftarrow assignWavelength(\hat{P}_{sd}^{k})$
13: **end if**
14: $P_{\mathcal{I}}^{\lambda} \leftarrow P_{\mathcal{I}}^{\lambda} \cup \{p_{i,sd}^{\lambda}\}$ ▷ New path added to solution set
15: **end for**
16: **return** $P_{\mathcal{I}}^{\lambda}$

C. Heuristic Formulation

Algorithm 1 illustrates the RWARP heuristic in pseudo code where conceptually three main parts may be observed: demand selection (steps 2 and 3), routing for the selected demand (steps 4 to 9) and finally the wavelength assignment and regenerator placement for each path (steps 10 to 13). Starting at the demand selection, step 2 sorts the array of demands according to a predefined criteria. In [3], *Rahman et al.* observed that for the routing and wavelength assignment the deployment of lightpaths in a longest-route-first provides, in general, better performance when comparing to the shortest-route-first and random selection. As each demand will be satisfied with a set of consecutive lightpaths (and their symmetric), sorting by the longest-route-first will be advantageous to ensure that this higher demanding requests have a free wavelength on each arc. As at step 2 the routes are still unknown, the shortest path from source to destination of each demand is used as an approximation to the distance of the final demand path. In this work, the shortest path from a source to a destination in a given network was computed with Dijkstra shortest path algorithm implemented with a binary heap. After sorting the demands in $\mathcal{I}$, the heuristic proceeds to find a feasible path from source to destination for each demand sequentially (steps 4 to 15). In step 4 a set of shortest paths from each node u to every node v that have at least a free wavelength (enough capacity for i) and with a cost (impairment value) below or equal to the threshold is obtained. This set of paths is used in step 5 to create an auxiliary graph $\mathcal{G}'(\mathcal{N}, \mathcal{A}')$ where $\mathcal{A}' = \{(u, v) : c(p_{uv}^{*}) < \Delta, A_{p_{uv}^{*}} \subset \mathcal{A}\}$, that is, $\mathcal{A}'$ contains logical arcs that connect the source u to the destination v of each path obtained in step 4. These logical arcs are set to have a cost of 1 (a hop) plus the division of the number of arcs in the path p_{uv}^{*}, $|\mathcal{A}_{p_{uv}^{*}}|$, by m (the number of arcs in $\mathcal{G}$). This cost value will always be greater than 1 by an increment that depends on the number of physical arcs in the path. This increment discriminates longer paths with the same number of logical hops which consequently may conduce to a better overall network capacity usage. Note that the floor of the sum of the cost of any path from s to d in $\mathcal{G}'$ will always equal the number of segments in the path and subtracting 1 to this number will equal the number of regenerators needed if no wavelength assignment was to be considered. Using $\mathcal{G}'$, all minimum cost paths from s to d are computed in step 6 of algorithm 1 to form the paths $p_{sd}^{'k}$ where k is an index to these paths. In this work, the K shortest paths were computed with Ernesto Martins's and Marta Pascoal's implementation of the Yen's K shortest path algorithm [19], as presented in [20]. If no path was found at this step, then there's a capacity shortage and no path will be assigned to the demand, the heuristic will thus return to the beginning of the cycle at step 4 for the next demand. Step 7 transforms/maps the logical arcs in $p_{sd}^{'k}$ to the corresponding p_{uv}^{*} in $\mathcal{G}$ producing the real end-to-end paths p_{sd}^{k}. At this step loops may arise. Step 8 removes these loops but may consequently create identical paths which are then removed at step 9 to form $\hat{P}_{sd}^{k}$ – the set of candidate paths. The final path – optical path – is selected from the set of candidate paths either in algorithm 2 ($tryFastWavelengthAssignment()$) or in algorithm 3 ($assignWavelength()$) where the wavelength assignment and regenerator placement take place. The former algorithm is faster but may fail often when the network starts to have low spare capacity whereas the latter algorithm will always provide a solution when called from the RWARP heuristic and will thus be called in case of failure of the former. Both algorithms are responsible for creating the lightpaths, assigning the wavelength and placing the respective regenerators and thus, at the end of step 13 an optical path $p_{i,sd}^{\lambda}$ was successfully computed. Having $p_{i,sd}^{\lambda}$, computing the inverse optical path ($p_{i,ds}^{\lambda}$) is as simple as using the same wavelengths in the symmetric arcs of $p_{i,sd}^{\lambda}$. The regenerators are considered bidirectional and thus are already computed.

Note that this heuristic is an extension and modification of the Exact Single Request Regenerator Placement (ESRRP) –

Algorithm 2 tryFastWavelengthAssignment

Input: A set of paths $\hat{P}^k_{sd}$ and an impairment threshold Δ
Output: An optical path p^{λ}_{sd}. The output path is the first element of $\hat{P}^k_{sd}$ for which a successful wavelength assignment was possible; if the wavelength assignment failed for every path in $\hat{P}^k_{sd}$ then the output is a $null$ path

1: Sort $\hat{P}^k_{sd}$ ▷ Sorted according to non-decreasing distance
2: **for** $p_{sd} \in \hat{P}^k_{sd}$ **do**
3: Create consecutive max possible distance segments from s to d (p_{su_0}, $p_{u_0u_1}, \ldots p_{u_{k-1}d}$) respecting Δ and try to assign a wavelength to each of these segments. Let $lp_w = (p_w, \lambda_w),\ \forall w \in \{su_0, u_0u_1, \ldots, u_{k-1}d\}$ if a wavelength was successfully assigned to a p_w or $null$ otherwise
4: **if** $lp_w \neq null,\ \forall w$ **then**
5: $p^{\lambda}_{sd} \leftarrow \langle lp_{su_0}, lp_{u_0u_1}, \ldots, lp_{u_{k-1}d} \rangle$
6: Place regenerators at the last node of each lightpath $lp_w\ \forall w \neq u_{k-1}d$, for the assigned wavelength λ_w
7: **return** p^{λ}_{sd}
8: **end if**
9: **end for**
10: **return** $null$ ▷ No wavelength assignment was possible

algorithm 4 in [6]. The initial base steps were based in the ESRRP but the following changes were introduced: First, we ensure that in step 4, each arc in the shortest path p^*_{uv} has a free wavelength. As we are considering capacity constraints, an arc may have a capacity shortage turning it unusable for the next demands. Second, the cost of the arcs of graph $\mathcal{G}'$ (see step 5 of algorithm 1) are not equal to one, as in ESRRP. Finding the shortest path in $\mathcal{G}'$ will not only provide the minimum number of regenerators but also the path that requires less physical hops (in $\mathcal{G}$) among the paths that require minimal regeneration. We then used Yen's algorithm [20] to obtain a set of alternative shortest paths instead of a single one which allows to explore the wavelength assignment on multiple optimal paths (w.r.t. the number of regenerators). Finally, the wavelength assignment and regeneration placement is made so that the wavelength continuity constraint is respected in unregenerated segments. Without capacity constraints, there's always a free wavelength that can be use in any segment and thus this step is not required. This is not the case in this work. Thus, we attempt to assign wavelengths taking only into account the distance of the segments in step 10 (see algorithm 2). If that fails due to wavelength continuity constraints, the regenerators placement is re-examined, in step 12 (see algorithm 3) in order to obtain a solution.

The $tryFastWavelengthAssignment()$ routine starts by sorting the input set of paths $\hat{P}^k_{sd}$ accordingly to the path distances. The shortest path will be the path requiring less regeneration (recall that distance is the impairment taken into account) so that every lightpath has a distance as close to Δ as possible. Starting with the most favourable path, in step 3, ligthpaths are attempted to be formed by creating maximum distance (unregenerated) segments with respect to the impairment threshold and then assigned a wavelength to each of these segments. Wavelength assignment was implemented using a First-Fit (FF) approach [11]. For efficiency purposes, each arc contains an array of bits where each index represent a wavelength and the value of 1 represents a free wavelength. To obtain the FF free common wavelength on a segment, consecutive and operations through the arcs of the segment are executed and at the end, the index of the first bit with a value of 1 is returned. Note that as the regenerators have wavelength conversion capabilities and at the end of the segments a regenerator is needed, each segment may have a different wavelength but a lightpath in the segment has to comply to the wavelength continuity constraint in all of its composing arcs. The arcs belonging to a given segment may have different free wavelengths and thus finding a common free wavelength to assign to the segment may fail. If wavelength assignment failed in step 3 of algorithm 2, the for loop proceeds to the next favourable path and if all paths fail, the method shall return a $null$ path – step 10. If all segments of a path are successfully assigned a wavelength – forming the respective lightpaths – the optical path p^{λ}_{sd} is thus the sequence of these lightpaths from s to d in step 5. The routine continues at step 6 by placing regenerators for the assigned wavelength at the head of each lightpath except at the last head as it will be d, the destination node. Step 7 returns the formed path.

As one can see, the $tryFastWavelengthAssignment()$ routine prioritizes the distance when creating the segments without taking into account if there's a free common wavelength at each arc of a segment (albeit having at least a free wavelength on each arc). This may fail often when there's a shortage of free capacity on the arcs as the demands start to occupy some arcs in the network. The routine $assignWavelength()$ (see algorithm 3) on the other hand tries to create the longest segments on a free common wavelength, respecting the impairment distance threshold. The $tryFastWavelengthAssignment()$ routine provides the same number of regenerators as the $assignWavelength()$ when there's enough capacity for the maximum distance segments to be formed. When this is not the case, the former will fail to retrieve a path but the latter will successfully compute one as algorithm 1 ensures that, after step 6, a path with enough capacity does exist. In short, algorithm 2 is faster and will work for the vast majority of cases but will fail for a large number of demands or for poorly capacitated networks whereas algorithm 3 is slower but will always provide a solution. Algorithm 3 ilustrates the $assignWavelength()$ routine described next. For each path p_{sd} in the set of input paths $\hat{P}^k_{sd}$, a corresponding path composed of logical arcs will be created and added to the set of logical paths P'_{sd} in steps 1 to 5. A logical path p'_{sd} is created using a reachability graph $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ where $\mathcal{N}_{p_{sd}}$ is the set of nodes in the corresponding p_{sd} and $\mathcal{A}''$ is a set of logical arcs of cost 1 from every $u \in \mathcal{N}_{p_{sd}}$ to every downstream node $v \in \mathcal{N}_{p_{ud}} \setminus \{u\}$, with $p_{sd} = p_{su} \diamondsuit p_{ud}$, at a distance lower or equal to the impairment threshold and have at least a free

Algorithm 3 assignWavelength

Input: A set of paths $\hat{P}^k_{sd}$

Output: An optical path p^λ_{sd}. The output path is the first element of $\hat{P}^k_{sd}$ for which a successful wavelength assignment was possible; if the wavelength assignment failed for every path in $\hat{P}^k_{sd}$ then the output is a $null$ path.

1: Set $P'_{sd} \leftarrow \emptyset$ where P'_{sd} is an auxiliar set of logic paths from s to d
2: **for each** ($p_{sd} \in \hat{P}^k_{sd}$) **do**
3: Create a reachability graph $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ where $\mathcal{A}''$ is a set of logical arcs with a cost of 1 (hop) from every $u \in \mathcal{N}_{p_{sd}}$ to every $v \in \mathcal{N}_{p_{ud}} \setminus \{u\}$, with p_{ud} the remaining segment from u to d (that is $p_{sd} = p_{su} \diamondsuit p_{ud}$), as long as the segment from u to v has a distance lower than Δ and there is at least a free common wavelength along the segment
4: Compute the shortest path from s to d, p'_{sd} on $\mathcal{G}''$ and add it to the set of P'_{sd}
5: **end for**
6: **if** $P'_{sd} \neq \emptyset$ **then** ▷ A solution must exist if called from algorithm 1
7: Select the minimum hop count path p'^*_{sd} from all paths in P'_{sd}
8: Expand the logic arcs of p'^*_{sd} into the segments assigning the corresponding free wavelength obtained in step 3 to obtain the lightpaths $lp_w,\ \forall w \in \{su_0, u_0u_1, \ldots, u_{k-1}d\}$. Let $p^\lambda_{sd} = \langle lp_{su_0}, lp_{u_0u_1}, \ldots, lp_{u_{k-1}d} \rangle$ be the final optical path
9: Place regenerators at the last node of each lightpath $lp_w,\ \forall w \neq u_{k-1}d$, for the assigned wavelength λ_w
10: **return** p^λ_{sd}
11: **end if**
12: **return** $null$ ▷ If called from the RWARP heuristic in algorithm 1 it won't reach here.

common wavelength in the segment from u to v. That is, the $\mathcal{G}''(\mathcal{N}_{p_{sd}}, \mathcal{A}'')$ graph connects every node u in path p_{sd} to all reachable (w.r.t. impairment threshold and a free common wavelength of the connecting arcs) downstream nodes. Recall that by assigning a cost of 1 (a hop) to a segment from u to v, the cost of a logic path from s to d in $\mathcal{G}''$ minus 1 will be the number of regenerators needed to traverse the path. After generating this auxiliary graph in step 3, the shortest path p'_{sd} is computed at step 4 as a set of consecutive logic arcs and added to the set of logical paths P'_{sd}. This logical path creation is repeated for each input path which will then be used to create a final path in steps 7 to 9. The final path is selected as the minimum hop count path in P'_{sd} (see step 7 in algorithm 3). As the objective of the RWARP heuristic is to minimize the number of regenerators, the minimum hop count path will be most suitable path. In step 8, the logical arcs in p'^*_{sd} are mapped into the real corresponding segments obtained in step 3 and assigned the respective wavelength to form the lightpaths of the optical path p^λ_{sd}. Step 9 places regeneration as in step 6 of algorithm 2 and the optical path is finally returned at step 10.

IV. Simulations and Results

The simulations were performed in four different networks from SNDlib [21]: polska, abilene, nobel_germany and janos_us_ca. As the links in nobel_germany are short, the links distance were doubled, and therefore this network will be denote as nobel_germany*.

Table I presents the comparison results between the ILP and the Heuristic. Columns 1, 2, 3 and 4 are the input data corresponding to: the network, the number of demands $|\mathcal{I}|$ [1], the threshold value Δ and the capacity (number of wavelengths) in each arc $|\lambda|$, respectively. Columns 5 to 10 are the output results where the left value of "/" presents the CPLEX result and the right value is the heuristic result. The outputs are, respectively, from column 5 to 10: the number of fulfilled demands, the number of lightpaths formed, the network capacity usage (in percentage), the maximum number of wavelengths used in any link of the network, the number of regenerators placed and the execution time (in seconds). The network capacity usage is simply the ratio between the total wavelengths in use and the total number of wavelengths in the network (recall that each demand requires one full wavelength on each arc of its path). The CPLEX execution time was limited to 24 hours, where the results with "time out" provided either a sub-optimal solution or no solution at all (marked with "–"). The "killed" time results means that the tests were terminated with an $out\ of\ memory$ error. Also note that table I and II were run on different computers. Table I results suggest that our heuristic provides the minimum number of regenerators for the case where no link is used at maximum capacity (values in column "Max in use λ" are inferior to $|\lambda|$), that is, the shortest path is always the final path. This result is in fact proven by Kuipers $et\ al.$ in [6], for their algorithm 4 – Exact Single Request Regenerator Placement (ESRRP) – where similar base steps are taken, without taking capacity into account. For the cases where at least one link is at maximum capacity usage (maximum number of wavelengths in use in a link equals $|\lambda|$, the link capacity), our heuristic places slightly more regenerators than the optimal solution ILP except in one case, but in another case, less than the sub-optimal solutions.

Most of the tested networks that timed out at the 24 hours limit providing either a sub-optimal, or no solution at all, took fractions of a second in our heuristic (except for the largest network). Also interesting to note is that by using the shortest path in the heuristic, the network capacity usage is, for most cases, inferior to the ILP solution. This was to be expected as the ILP does not distinguish paths that need the same number of regenerators, but have different number hops. Lastly, note that even for small networks like Polska, when the number of demands is high, the ILP timed out but the heuristic takes less

[1]The SNDlib networks contain demands between each pair of nodes. We either used this base set of demands or multiples of them.

TABLE I
COMPARISON BETWEEN ILP AND HEURISTIC

Input data				CPLEX / Heuristic results					
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	**Fulfilled Demands**	**Lighpaths Formed**	**Capacity Usage(%)**	**Max in Use λ**	**No. Reg. Placed**	**Time(s)**
polska	66	1000	48	66 / 66	134 / 134	17.94 / 16.67	16 / 15	1 / 1	206.47 / 0.14
	132	1000	24	132 / 132	268 / 284	66.67 / 71.76	24 / 24	2 / 10	4363.83 / 0.26
	132	1000	48	132 / 132	268 / 268	35.76 / 33.33	28 / 30	2 / 2	667.97 / 0.21
	264	1000	48	- / 264	- / 568	- / 71.76	- / 48	- / 20	killed / 0.31
abilene	66	3000	20	66 / 66	174 / 180	58.00 / 57.00	19 / 20	21 / 24	606.16 / 0.11
	66	3000	48	66 / 66	174 / 174	24.44 / 23.47	21 / 22	21 / 21	636.27 / 0.11
	132	3000	40	132 / 132	348 / 360	56.83 / 57.00	39 / 40	42 / 48	time out / 0.17
	132	3000	48	132 / 132	348 / 348	47.36 / 46.94	38 / 44	42 / 42	2399.51 / 0.16
nobel germany*	121	1000	32	121 / 121	346 / 330	40.87 / 40.38	31 / 32	52 / 44	time out / 0.25
janos us_ca	1482	2000	220	– / 1482	- / 5388	- / 51.74	- / 220	- / 1212	killed / 8.24

than one second to find a solution. For larger networks, like janos_us_ca, the heuristic only took about eight seconds and a quarter for solving all demands whereas the CPLEX returned "Out of Memory".

Table II provides a second set of results were the objective was to compare the execution time of four variations of our ILP: ILP-1 has wavelength conversion capabilities at regenerators and capacity constraints (ILP presented in section III); ILP-2 has capacity constraints but no wavelength conversion capabilities (this corresponds to removing the summation in λ in equation (4) from ILP-1); ILP-3 does not have capacity constraints (this corresponds to suppressing equation (11) from ILP-1) but has wavelength conversion; ILP-4 is the ILP presented in [5] without path protection, that is, no wavelength conversion neither capacity or bidirectional traffic constraints. The results show that the wavelength conversion capability improves the execution time (ILP-1 to ILP-2). That is to be expected as wavelength conversion relaxes the wavelength continuity constraint. The removal of the capacity constraints also improves the execution time (ILP-1 to ILP-3) as it relaxes the path selection. Note that without capacity constraints, wavelength assignment is irrelevant. Lastly, Beshir's and Kuipers's implementation was the fastest which was to be expected since both capacity and bidirectional constraints were absent.

TABLE II
ILP VARIATIONS

Input data					CPLEX
Network	$\|\mathcal{I}\|$	Δ	$\|\lambda\|$	ILP Variation	**Time (s)**
polska	66	1000	32	ILP-1	180.16
	66	1000	32	ILP-2	219.55
	66	1000	32	ILP-3	172.15
	66	1000	32	ILP-4	108.63
	132	1000	48	ILP-1	765.89
	132	1000	48	ILP-2	963.53
	132	1000	48	ILP-3	781.14
	132	1000	48	ILP-4	503.43

V. CONCLUSION

This work addressed the problem of impairment aware routing and wavelength assignment with regeneration placement (RWARP), considering capacity constraints, with the objective on minimizing the number of regenerators. A previously existing ILP formulation [5] for the RWARP problem was modified and extended to consider capacity constraints, bidirectional traffic and regenerator nodes with wavelength conversion capability. We have also compared some variants of the ILP and concluded that the addition of wavelength conversion capabilities to the network nodes, when capacity constraints are considered, may speed up the ILP. Nevertheless regarding the complexity of the problem, the ILP can only solve small problems in a reasonable amount of time. Therefore we proposed an heuristic which has shown to give accurate results for small networks and is capable of solving large problems in a small amount of time.

In the near future, we intend to extend this work to a multilayer optimization framework considering a MPLS layer over a protected optical network.

ACKNOWLEDGMENT

Lúcia Martins and Teresa Gomes acknowledge financial support by Fundação para a Ciência e a Tecnologia (FCT) under project grant UID/MULTI/00308/2013. Bruna Nogueira and Ricardo Mendes acknowledge financial support by Coriant Portugal and the University of Coimbra.

REFERENCES

[1] R. Slavík, F. Parmigiani, J. Kakande, C. Lundström, M. Sjödin, P. a. Andrekson, R. Weerasuriya, S. Sygletos, A. D. Ellis, L. Grüner-Nielsen, D. Jakobsen, S. Herstrøm, R. Phelan, J. O'Gorman,

A. Bogris, D. Syvridis, S. Dasgupta, P. Petropoulos, and D. J. Richardson, "All-optical phase and amplitude regenerator for next-generation telecommunications systems," *Nature Photonics*, vol. 4, no. 10, pp. 690–695, 2010. [Online]. Available: http://www.nature.com/doifinder/10.1038/nphoton.2010.203

[2] M. Matsumoto, "Fiber-based all-optical signal regeneration," *IEEE Journal on Selected Topics in Quantum Electronics*, vol. 18, no. 2, pp. 738–752, 2012.

[3] Q. Rahman, S. Bandyopadhyay, and Y. Aneja, "On static rwa in translucent optical networks," in *Computers and Communications (ISCC), 2012 IEEE Symposium on*, July 2012, pp. 000 171–000 176.

[4] ——, "A branch and price approach for optimal regenerator placement in translucent networks," in *Optical Network Design and Modeling (ONDM), 2011 15th International Conference on*, Feb 2011, pp. 1–6.

[5] A. Beshir, F. Kuipers, A. Orda, and P. V. Mieghem, "Survivable routing and regenerator placement in optical networks," in *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, oct 2012, pp. 684–690.

[6] F. Kuipers, A. Beshir, A. Orda, and P. Van Mieghem, "Impairment-aware path selection and regenerator placement in translucent optical networks," *Proceedings - International Conference on Network Protocols, ICNP*, pp. 11–20, 2010.

[7] B. Garcia-Manrubia, P. Pavon-Marino, R. Aparicio-Pardo, M. Klinkowski, and D. Careglio, "Offline impairment-aware RWA and regenerator placement in translucent optical networks," *Journal of Lightwave Technology*, vol. 29, no. 3, pp. 265–277, 2011.

[8] S. Rai, C. F. Su, and B. Mukherjee, "On provisioning in all-optical networks: An impairment-aware approach," *IEEE/ACM Transactions on Networking*, vol. 17, no. 6, pp. 1989–2001, 2009.

[9] M. A. Ezzahdi, S. Al Zahr, M. Koubàa, N. Puech, and M. Gagnaire, "LERP: A quality of transmission dependent heuristic for routing and wavelength assignment in hybrid WDM networks," *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, pp. 125–136, 2006.

[10] W. Zhang, J. Tang, K. Nygard, and C. Wang, "REPARE: Regenerator placement and routing establishment in translucent networks," *GLOBECOM - IEEE Global Telecommunications Conference*, 2009.

[11] H. Zang, J. P. Jue, and B. Mukherjee, "A review of routing and wavelength assignment approaches for wavelength- routed optical WDM networks," *Optical Networks Magazine*, vol. 1, no. January, pp. 47–60, 2000.

[12] Z. Zhang and A. Acampora, "A heuristic wavelength assignment algorithm for multihop wdm networks with wavelength routing and wavelength reuse," in *INFOCOM '94. Networking for Global Communications., 13th Proceedings IEEE*, Jun 1994, pp. 534–543 vol.2.

[13] J. M. Yates, M. P. Rumsewicz, and J. P. R. Lacey, "Wavelength converters in dynamically-reconfigurable wdm networks," *IEEE Communications Surveys*, vol. 2, no. 2, pp. 2–15, Second 1999.

[14] G. Charlet, J. Renaudier, H. Mardoyan, P. Tran, O. B. Pardo, F. Verluise, M. Achouche, A. Boutin, F. Blache, J. Y. Dupuy, and S. Bigo, "Transmission of 16.4-bit/s capacity over 2550 km using PDM QPSK modulation format and coherent receiver," *Journal of Lightwave Technology*, vol. 27, no. 3, pp. 153–157, 2009.

[15] J. Strand and A. Chiu, "RFC 4054, impairments and other constraints on optical layer routing," IETF Network Working Group, May 2005. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4054.txt

[16] S. Azodolmolky, M. Klinkowski, E. Marin, D. Careglio, J. S. Pareta, and I. Tomkos, "A survey on physical layer impairments aware routing and wavelength assignment algorithms in optical networks," *Computer Networks*, vol. 53, no. 7, pp. 926–944, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2008.11.014

[17] D. Lucerna, N. Gatti, G. Maier, and A. Pattavina, "On the efficiency of a game theoretic approach to sparse regenerator placement in WDM networks," *GLOBECOM - IEEE Global Telecommunications Conference*, 2009.

[18] W. Xie, J. P. Jue, X. Wang, Q. Zhang, Q. She, P. Palacharla, and M. Sekiya, "Regenerator site selection for mixed line rate optical networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 6, no. 3, pp. 291–302, March 2014.

[19] J. Y. Yen, "Finding the K Shortest Loopless Paths in a Network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[20] E. Q. V. Martins and M. M. B. Pascoal, "A new implementation of Yen's ranking loopless paths algorithm," *4or*, vol. 1, no. 2, pp. 121–133, 2003.

[21] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "SNDlib 1.0–Survivable Network Design library," *Networks*, vol. 55, no. 3, pp. 276–286, 2010, http://sndlib.zib.de.

Bibliography

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms and applications*, volume 1. 1993.

[2] S. Azodolmolky, M. Klinkowski, E. Marin, D. Careglio, J. S. Pareta, and I. Tomkos. A survey on physical layer impairments aware routing and wavelength assignment algorithms in optical networks. *Computer Networks*, 53(7):926–944, 2009.

[3] A. Beshir, F. Kuipers, A. Orda, and P. Van Mieghem. Survivable routing and regenerator placement in optical networks. In *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, pages 684–690, Oct 2012.

[4] A. Beshir, F. Kuipers, A. Orda, and P. Van Mieghem. Survivable impairment-aware traffic grooming in WDM rings. In *Proceedings of the 23rd International Teletraffic Congress*, ITC '11, pages 158–165. International Teletraffic Congress, 2011.

[5] R. Bhandari. *Survivable Networks: Algorithms for Diverse Routing*. The Springer International Series in Engineering and Computer Science. Springer, 1999.

[6] G. Charlet, J. Renaudier, H. Mardoyan, P. Tran, O. B. Pardo, F. Verluise, M. Achouche, A. Boutin, F. Blache, J. Y. Dupuy, and S. Bigo. Transmission of 16.4-bit/s capacity over 2550 km using PDM QPSK modulation format and coherent receiver. *Journal of Lightwave Technology*, 27(3):153–157, 2009.

[7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1(1):269–271, December 1959.

[8] M. A. Ezzahdi, S. Al Zahr, M. Koubàa, N. Puech, and M. Gagnaire. LERP: A quality of transmission dependent heuristic for routing and wavelength assignment in hybrid WDM networks. *Proceedings - International Conference on Computer Communications and Networks, ICCCN*, pages 125–136, 2006.

[9] M. Flammini, A. Marchetti-Spaccamela, G. Monaco, L. Moscardelli, and S. Zaks. On the complexity of the regenerator placement problem in optical networks. *IEEE/ACM Transactions on Networking*, 19(2):498–511, April 2011.

[10] B. Fortz and M. Thorup. Internet traffic engineering by optimizing OSPF weights. *Ieee Infocom*, 2:519–528, 2000.

[11] B. Garcia-Manrubia, P. Pavon-Marino, R. Aparicio-Pardo, M. Klinkowski, and D. Careglio. Offline impairment-aware RWA and regenerator placement in translucent optical networks. *Journal of Lightwave Technology*, 29(3):265–277, 2011.

[12] S. Gringeri, B. Basch, V. Shukla, R. Egorov, and T. J. Xia. Flexible architectures for optical transport nodes and networks. *IEEE Communications Magazine*, 48(7):40–50, July 2010.

[13] P.-H. Ho and H. T. Mouftah. Shared protection in mesh WDM networks. *IEEE Communications Magazine*, 42(1):70–76, Jan 2004.

[14] F. Kuipers. An overview of algorithms for network survivability. *ISRN Communications and Networking*, 2012, 2012.

[15] F. Kuipers, A. Beshir, A. Orda, and P. Van Mieghem. Impairment-aware path selection and regenerator placement in translucent optical networks. *Proceedings - International Conference on Network Protocols, ICNP*, pages 11–20, 2010.

[16] E. Q. V. Martins and M. M. B. Pascoal. A new implementation of Yen's ranking loopless paths algorithm. *4or*, 1(2):121–133, 2003.

[17] M. Matsumoto. Fiber-based all-optical signal regeneration. *IEEE Journal on Selected Topics in Quantum Electronics*, 18(2):738–752, 2012.

[18] R. Mendes, B. Nogueira, T. Gomes, L. Martins, and J. Santos. Impairment-aware optimization strategies to dimension optical transport networks with minimal regeneration requirements. In *8th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, Oct 2016. Accepted paper to be presented.

[19] H. T. Mouftah and P.-H. Ho. *Optical Networks: Architecture and Survivability*. Springer US, 2003.

[20] B. Nogueira. Regeneration and traffic grooming in MPLS over WDM networks. Master's thesis, University of Coimbra, Portugal, 2016. Submitted.

[21] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski. SNDlib 1.0–Survivable Network Design library. *Networks*, 55(3):276–286, 2010. http://sndlib.zib.de.

[22] Q. Rahman, Y. Aneja, S. Bandyopadhyay, and A. Jaekel. Optimal regenerator placement in survivable translucent networks. In *Design of Reliable Communication Networks (DRCN), 2014 10th International Conference on the*, pages 1–7, April 2014.

[23] Q. Rahman, S. Bandyopadhyay, and Y. Aneja. A branch and price approach for optimal regenerator placement in translucent networks. In *Optical Network Design and Modeling (ONDM), 2011 15th International Conference on*, pages 1–6, Feb 2011.

[24] Q. Rahman, S. Bandyopadhyay, and Y. Aneja. On static RWA in translucent optical networks. pages 000171–000176, July 2012.

[25] S. Rai, C. F. Su, and B. Mukherjee. On provisioning in all-optical networks: An impairment-aware approach. *IEEE/ACM Transactions on Networking*, 17(6):1989–2001, 2009.

[26] L. Sahasrabuddhe, S. Ramamurthy, and B. Mukherjee. Fault management in IP-over-WDM networks: WDM protection versus IP restoration. *IEEE Journal on Selected Areas in Communications*, 20(1):21–33, Jan 2002.

[27] J. M. Simmons. Network design in realistic "all-optical" backbone networks. *IEEE Communications Magazine*, 44(11):88–94, November 2006.

[28] R. Slavík, F. Parmigiani, J. Kakande, C. Lundström, M. Sjödin, P. Andrekson, R. Weerasuriya, S. Sygletos, A. D. Ellis, L. Grüner-Nielsen, D. Jakobsen, S. Herstrøm, R. Phelan, J. O'Gorman, A. Bogris, D. Syvridis, S. Dasgupta, P. Petropoulos, and D. J. Richardson. All-optical phase and amplitude regenerator for next-generation telecommunications systems. *Nature Photonics*, 4(10):690–695, 2010.

[29] P. Soproni, T. Cinkler, and J. Rak. Methods for physical impairment constrained routing with selected protection in all-optical networks. *Telecommunication Systems*, 56(1):177–188, 2014.

[30] J. Strand and A. Chiu. RFC 4054, impairments and other constraints on optical layer routing. IETF Network Working Group, May 2005.

[31] J. W. Suurballe. Disjoint paths in a network. *Networks*, 4(2):125–145, 1974.

[32] J. W. Suurballe and R. E. Tarjan. A quick method for finding shortest pairs of disjoint paths. *Networks*, 14(2):325–336, 1984.

[33] J.-P. Vasseur, M. Pickavet, and P. Demeester. *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[34] J. M. Yates, M. P. Rumsewicz, and J. P. R. Lacey. Wavelength converters in dynamically-reconfigurable WDM networks. *IEEE Communications Surveys*, 2(2):2–15, Second 1999.

[35] J. Y. Yen. Finding the K Shortest Loopless Paths in a Network. *Management Science*, 17(11):712–716, 1971.

[36] H. Zang, J. P. Jue, and B. Mukherjee. A review of routing and wavelength assignment approaches for wavelength- routed optical WDM networks. *Optical Networks Magazine*, 1(January):47–60, 2000.

[37] W. Zhang, J. Tang, K. Nygard, and C. Wang. REPARE: Regenerator placement and routing establishment in translucent networks. *GLOBECOM - IEEE Global Telecommunications Conference*, 2009.

[38] Z. Zhang and A. Acampora. A heuristic wavelength assignment algorithm for multihop WDM networks with wavelength routing and wavelength reuse. In *INFOCOM '94. Networking for Global Communications., 13th Proceedings IEEE*, pages 534–543 vol.2, Jun 1994.