DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Solving Survivable Two-Layer Network Design Problems by Metric Inequalities

Sara Mattia

Technical Report n. 2, 2010

Solving Survivable Two-Layer Network Design Problems by Metric Inequalities

Sara Mattia
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Sapienza, Università di Roma

Abstract

We address the problem of designing a multi-layer network with survivability requirements. We are given a two-layer network: the lower layer represents the potential physical connections that can be activated, the upper layer is made of logical connections that can be set up using physical links. We are given origin-destination demands (commodities) to be routed at the upper layer. We are also given a set of failure scenarios and, for every scenarios, an associated subset of commodities. The goal is to install minimum cost integer capacities on the links of both layers in order to ensure that the commodities can be routed simultaneously on the network. In addition, in every failure scenario the routing of the associated commodities must be guaranteed. We consider two variants of the problem and develop a branch-and-cut scheme based on the capacity formulation. Computational results on instances derived from the SNDLib for single node failure scenarios are discussed.

keywords: optical network design, branch-and-cut

1 Introduction and motivation

In the past years, telecommunication networks have grown in dimension, supported services, transportation capacity, speed and reliability. In fact, the large number of new users and the growing request for high capacity, fast and reliable connections to support new services and applications, led to the development of new technologies. Optical networks based on Wavelength Division Multiplexing (WDM) have assumed a dominant role, since they provide a huge capacity increasing by allowing the creation of virtual point-to-point connections (lightpaths) that can share the same fiber. A wavelength of operation is assigned to each lightpath. The wavelength must be the same for the entire path, unless wavelength converters are used. Every physical connection (fiber) can support several lightpaths, but two lightpaths can share the same fiber only if they are assigned different wavelengths on that link. Because optical transport networks are designed to carry high volumes of traffic, network failures may have severe consequences. Therefore survivability requirements have became an integral part of every network design problem [8], [16], [29]. As a result, the complexity of the design and configuration process increases significantly. In fact, the design of optical networks includes: the definition of the virtual topology (lightpaths) in order to route a set of point-to-point traffic demands, the dimensioning of the physical network in order to support the virtual topology, the satisfaction of a set of survivability requirements, and the assignment of wavelengths to the lightpaths. In order to reduce problem complexity, a common approach in the literature is to consider the two-layer network design problem and the wavelength assignment problem as two separate problems to be solved in sequence. First the network design problem is solved under the hypothesis that wavelength converters are available at each node of the network. As a second step, the wavelength assignment is considered. If the design problem is solved making sure that the capacity constraints on the physical links are satisfied, it is always possible to compute a feasible assignment of wavelengths to the lightpaths placing wavelength converters, if needed. In this paper, following [3], [4], [8], [10], [6], [25], [18] and others, we focus our attention on the network design problem. We refer the reader to [19], [20] and references therein for the problem of installing wavelength converters in the network at minimum cost.

An optical network is a two-layer network: the lower layer is called *physical* or *optical* layer, the upper layer is called *logical* layer. Depending on the technology installed on the nodes, a node of the network can either operate at both levels, or it is equipped to operate only in the physical layer. Every link of the physical layer represents a potential physical connection that can be established between the nodes of the network. Every link of the logical layer is a point-to-point logical connection (lightpath) that corresponds to a path between its endpoints in the physical network. See for example Figure 1 for a simple network on three nodes.
In this example the physical network G is the complete graph on three nodes and all the nodes can operate at both levels. The logical network H includes four logical edges, each of them representing the physical path indicated in the edge label after the underscore. Origin-destination traffic demands (commodities) are given in the logical network. A set of failure scenarios is considered. In every scenario a given set of physical components (links and/or nodes) of the network are supposed to be affected by a fault that makes those elements unavailable. The fault propagates from the physical layer to the logical one. In fact, when a physical component fails, all logical elements using that component become unavailable. For a more detailed description of scenarios see Section 5. For every scenario, a subset of (protected) commodities whose routing must be guaranteed in that scenario is given. The aim is to choose minimum costs physical and logical capacities such that all the demands can be routed simultaneously on the network in the case without failures and, for every failure scenario, the routing of the corresponding protected commodities is ensured.

Due to the large number of applications, network design problems have received a huge amount of

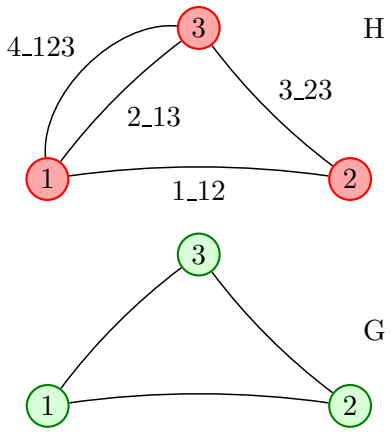

Figure 1: a two-layer network on three nodes

attention in the literature, both for single-layer networks and for multi-layer networks. Different versions of the problem have been investigated and different solution approaches are available in the literature. For single-layer network design problems without survivability, see for instance [2], [7], [14], [1], and references therein. As for the two-layer network design problem, see [27] for an overview of technological issues and related models. Several authors proposed MIP-based models and algorithms for different versions of the problem, see for example [3], [5]. In [10] they consider a problem where the physical capacities are fixed and logical edges (pipes) must be activated in order to support given demands: polyhedral results are given and a branch-and-cut approach is developed. In [18] a problem with survivability requirements and node hardware installation issues is addressed: the problem is solved by branch-and-cut using single-layer cuts. In [6] a model based on metric inequalities without survivability requirements, based on [17], is used to solve a variant of the problem belonging to the class that in this paper is called implicit lightpaths case (see Section 2). In [4] a Lagrangean approach is proposed. In [25] MIP-based heuristics to be used within a branch-and-cut scheme are presented. In [8], valid inequalities and a branch-and-cut approach are presented for the uncapacitated case with survivability requirements. In [13] multi-layer hop-constrained node-survivable networks are addressed and two different survivability mechanisms (path diversity and path protection) are considered.

The paper is structured as follows: in Section 2 we analyze different mathematical formulations for the problem, in Sections 3 and 4 we describe our solution approach for two variants of the problem, in Section 5 we discuss computational results and in Section 6 we present our conclusions.

2 Mathematical model

Let $G(V, E)$ be an undirected graph representing the physical network, where V is the set of nodes to be connected and E is the set of potential physical links. Let $H(V, L)$ be an undirected graph representing the logical network. In general, the nodes of the network can operate either at one single layer or in both layers depending on the technological components they have. In this paper, as done in [18], [25], [10], [6] and others, we suppose that both networks have the same set of nodes, that is that all the nodes of the network are equipped to operate at both levels. We use the words *edge*, *arc* and *link* as synonyms. We also use *lightpath* and *logical edge* as synonymous. Let us also suppose that in the physical network there are no loops and no parallel edges, while there can be (and in practice there are) parallel edges in the logical network. If we

do not consider technological limitations, any path in the physical network can be used to define a lightpath, therefore the lightpaths can be exponentially many with respect to $|V|$. In practice, the set of admissible lightpaths is restricted according to some criteria. A common used criterion is to consider only lightpaths corresponding to physical paths having at most a fixed number of intermediate nodes (*hops*). Let E_ℓ be the set of physical edges used by $\ell \in L$ and let L_e be the set of logical links using physical edge $e \in E$. Let L_{ij} be the set of (parallel) logical links $\ell \in L$ connecting i and j and let $L_{ii} = \emptyset$. Since the graph is undirected, $L_{ij} = L_{ji}$. Capacity on logical and physical links can only be installed in fixed amounts. Let U be the size of a capacity module for logical links and let B be the size of a capacity module for a physical link. That is, every time we buy one unit (or module) of capacity for a logical link we get a capacity of U on that link, every time we buy one module of capacity for a physical link we get a capacity of B on that link. Let c_e^E and c_ℓ^L be the cost of installing one module of capacity on edge $e \in E$ and $\ell \in L$, respectively. Let K be the set of commodities. Each commodity k is a triple (s^k,t^k,d^k) where s^k is the source node, t^k is the destination node and d^k is the demand to be routed from the source node to the destination node. Let S be the set of failure scenarios to be considered, plus an additional scenario s_0 representing the case without failures. Let $G^s(V^s, E^s)$ and $H^s(V^s, L^s)$ be the physical and logical graphs in scenario s, where V^s is the set of active nodes, E^s is the set of active physical edges, L^s is the set of active logical edges ($G^{s_0} = G$ and $H^{s_0} = H$). Let $K^s \subseteq K$ be the set of commodities whose routing must be ensured in scenario $s \in S$ ($K^{s_0} = K$). Given i and $j \in V^s$, let L_{ij}^s be the set of (parallel) logical links $\ell \in L^s$ connecting i and j in scenario s.

2.1 The flow formulation

A first possible formulation for the problem is a two-level version of the well known arc-flow formulation. Let x_e be an integer variable representing the number of capacity modules installed on physical edge $e \in E$, and let y_ℓ be an integer variable representing the number of capacity modules installed on logical edge $\ell \in L$. Let $f_{\ell,ij}^{k,s}$ and $f_{\ell,ji}^{k,s}$ be continuous variables representing the flow for commodity $k \in K^s$ directed from i to j and viceversa on edge $\ell = (i,j) \in L^s$ in scenario $s \in S$. The arc-flow formulation (EFF) is:

$$
\begin{aligned}
(EFF) \qquad & \min \sum_{e \in E} c_e^E x_e + \sum_{\ell \in L} c_\ell^L y_\ell \\
& \sum_{j \in V^s} \sum_{\ell \in L_{ij}^s} (f_{\ell,ij}^{k,s} - f_{\ell,ji}^{k,s}) = d_i^k \qquad i \in V^s, k \in K^s, s \in S && (1) \\
& \sum_{k \in K^s} (f_{\ell,ij}^{ks} + f_{\ell,ji}^{k,s}) \leq U y_\ell \qquad \ell = (i,j) \in L^s, s \in S && (2) \\
& \sum_{\ell \in L_e} y_\ell \leq B x_e \qquad e \in E && (3) \\
& f \geq 0 \\
& x_e, y_\ell \in \mathbb{Z}_+
\end{aligned}
$$

Constraints (1) are flow conservation constraints, where $d_i^k = d^k$ if $i = s_k$, $d_i^k = -d^k$ if $i = t_k$ and $d_i^k = 0$ otherwise. Constraints (2) are capacity constraints for logical links for every scenario. They ensure that the total flow traversing a lightpath does not exceed the installed capacity. Constraints (3) are capacity constraints for physical edges and they ensure that enough physical capacity is installed to support all the lightpaths.

Formulation (EFF) can handle the presence of integrality requirements for the flows and simple routing constraints, but it can not handle general routing restrictions for the commodities. If there

are path restrictions for the commodities, we have to use a path-flow formulation. In this paper we suppose that there are no path restrictions for the commodities. Under this assumption, a different formulation based on the so-called metric inequalities can be written. Metric inequalities allow us to eliminate flow variables at the cost of obtaining a non compact formulation, which requires the use of a cutting-plane approach. Metric inequalities have already been used in the literature, both for single-layer problems (see for example [7], [2]) and for two-layer problems (see for example [6], [17]) without survivability requirements. They have also been used when survivability issues must be taken into account (see for example [30], [29] and references therein). In the following section we formally define metric inequalities, clarify their importance for the single-layer problem, and use them to formulated the two-layer problem.

2.2 Metric inequalities

DEFINITION 2.1 *Given a set of nodes N, a function $\pi : N \times N \to \mathbb{R}_+$ is a semi-metric on N if and only if:*

1. $\pi_{ii} = 0 \qquad i \in N$
2. $\pi_{ij} \geq 0 \qquad i, j \in N$
3. $\pi_{ij} = \pi_{ji} \qquad i, j \in N$
4. $\pi_{ij} \leq \pi_{ik} + \pi_{kj} \qquad i, j, k \in N$

More precisely, if condition 2. holds with strict inequality we have a *metric*, otherwise we have a *semi-metric*. If symmetry condition 3. does not hold we have an oriented distance function or *quasi-(semi)-metric* [11],[12]. Since we are working on undirected graphs and we allow π values to be zero, when we say *metric*, we are technically speaking of a semi-metric. Let Met_N the cone generated by all non zero metrics.

DEFINITION 2.2 *Let G(N,A) be a graph, a function $\mu : A \to \mathbb{R}_+$ defines a metric on G if and only if:*

1. $\mu_a \geq 0 \qquad a \in A$
2. $\mu_a \leq \mu(P_a) \qquad a \in A$

where $\mu(P_a)$ is the length of shortest path between the endpoints of edge a using μ as weights.

Let Met_A be the cone of all non zero metrics defined on $G(N, A)$. Given $u, v \in N$ we denote by π^μ_{uv} the length of the shortest path in G between u and v using μ as weights, $\pi^\mu_{ii} = 0$ for all i. If $(u, v) \in A$ then, by definition, $\pi^\mu_{uv} = \mu_{uv}$. In this way a metric $\mu \in Met_A$ can be extended to a metric $\pi^\mu \in Met_N$. If G is directed we get a quasi-(semi)-metric, otherwise we get a (semi-)metric.

Let us consider for the moment a single-layer network design problem, also known as Network Loading Problem. The problem can be stated as follows: let $G(N, A)$ be an undirected graph and let D be a set of traffic demands, we want to choose minimum cost integer capacities for the edges so that the demands can be routed simultaneously on the network.

THEOREM 2.3 *[23] [15] An edge capacity vector z (not necessarily integer) is feasible for the problem, that is, it can support a feasible flow on the network, if and only if it satisfies inequalities:*

$$\sum_{a \in A} \mu_a z_a \geq \sum_{q \in D} \pi^\mu_q d^q \qquad \mu \geq 0 \tag{4}$$

where π_q^μ is the length of the shortest between the source node and the destination node of demand $q \in D$ using μ as edge weights. Theorem 2.3 is known as *Japanese theorem* and inequalities (4) are called *metric inequalities*. A stronger feasibility condition (see [21]) can be obtained replacing $\mu \geq 0$ by $\mu \in Met_A$ in (4). Therefore, the so-called capacity formulation of the problem is:

$$
\begin{aligned}
(NLF) \qquad & \min \sum_{a \in A} c_a z_a \\
& \sum_{a \in A} \mu_a z_a \geq \sum_{q \in D} \pi_q^\mu d^q \qquad \mu \in Met_A \\
& z_a \in \mathbb{Z}_+ \qquad a \in A
\end{aligned}
$$

where z_a is an integer variable representing the capacity installed on edge $a \in A$. Let $NL(G, D)$ be the convex hull of integer feasible solutions of the problem. The following results hold.

THEOREM 2.4 *[2] If $a^T z \geq b$ is valid for $NL(G, D)$ then there exists a metric $\mu \in Met_A$ such that $\mu^T z \geq b$ is still valid and $\mu_{ij} \leq a_{ij}$ for all $(i, j) \in A$.*

DEFINITION 2.5 *[2] Given a metric $\mu \in Met_A$, let $R_\mu = \min\{\mu^T z : z \in NL(G, D)\}$ be its rank.*

THEOREM 2.6 *[2] Every constraint of type $\mu^T z \geq b$ is dominated by $\mu^T z \geq R_\mu$.*

The above results imply that all facet-defining inequalities are of the form $\mu^T z \geq R_\mu$, where μ is a metric on G. This kind of inequalities are called *tight metric inequalities*.

LEMMA 2.7 *[22] Given $\mu, \eta, \nu \in Met_A$ such that $\mu = \eta + \nu$, then $R_\mu \geq R_\eta + R_\nu$*

DEFINITION 2.8 *Given η and $\nu \in Met_A$, then $\eta \neq \mu$ if there does not exist $\lambda \in \mathbb{R}_+$ such that $\eta = \lambda \nu$*

DEFINITION 2.9 *A metric $\mu \in Met_A$ is decomposable with respect to D if there exist $\eta, \nu \in Met_A$ such that $\eta \neq \mu$, $\nu \neq \mu$, $\eta \neq \nu$ and:*

1. *$\mu = \eta + \nu$*
2. *$R_\mu = R_\eta + R_\nu$*

LEMMA 2.10 *[22] $\mu^T z \geq R_\mu$ is facet-defining for $NL(G, D)$ only if μ is not decomposable with respect to D.*

In particular, metrics corresponding to extreme rays of the metric cone (extreme metrics) are non decomposable with respect to every possible D and the following result holds.

THEOREM 2.11 *[2] If μ is an integer valued extreme ray of the metric cone having greatest common divisor equal to one, then $R_\mu = \lceil \pi^T d \rceil$.*

Examples of extreme rays inducing facet-defining inequalities for $NL(G, D)$ are the well-known cut inequalities. However, in general, extreme rays are not sufficient to describe all facets.

The above results show that metric inequalities are a powerful tool, but there are limitations in using them that one must be aware of. In fact (NLF) is not a formulation for unsplittable flow problems or, in general, in any case where path restrictions (maximum number of crossed nodes, unavailable paths, predetermined set of available paths, ...) are given. In those situations you have to use a different framework, that can be either (i) a capacity formulation based on tight metric inequalities where $NL(G, D)$ is defined according to the routing requirements, or (ii) a general Benders decomposition approach, or (iii) a branch-and-cut(-and-price) method based on the path-flow formulation of the problem. Additional details about the relationship between metric inequalities and Benders cuts can be found in [9].

2.3 The capacity formulation

The limitations discussed in the previous section have a huge impact on a two-layer problem. Here you can potentially use metric inequalities for both layers: in the logical layer logical capacities play the role of capacities and original commodities play the role of demands, in the physical layer physical capacities play the role of capacities and logical capacities play the role of demands.

As in the single-layer problem, you can use metric inequalities for the logical layer only if there are no routing constraints for the commodities. The formulation is given below.

$$
\begin{aligned}
(ECF) \qquad & \min \sum_{e \in E} c^E_e x_e + \sum_{\ell \in L} c^L_\ell y_\ell \\
& \sum_{\ell=(i,j) \in L^s} \mu_{ij} U y_\ell \geq \sum_{k \in K^s} \pi^\mu_k d^k \qquad \mu \in Met_{L^s}, s \in S && (5) \\
& \sum_{\ell \in L_e} y_\ell \leq B x_e \qquad e \in E && (6) \\
& x_e, y_\ell \in \mathbb{Z}_+
\end{aligned}
$$

Constraints (5) are metric inequalities for logical layer, which ensure that the capacity installed on logical links can support the demand in every scenario. Capacity constraints for the physical layer (6) guarantee that the capacity installed on physical edges can support the logical links.

The situation is much more complex for the physical layer and some remarks are needed before replacing (6) by metric inequalities. A capacity formulation using metric inequalities for both layers should have the following structure.

$$
\begin{aligned}
(ICF) \qquad & \min \sum_{e \in E} c^E_e x_e + \sum_{\ell \in L} c^L_\ell y_\ell \\
& \sum_{\ell=(i,j) \in L^s} \mu_{ij} U y_\ell \geq \sum_{k \in K^s} \pi^\mu_k d^k \qquad \mu \in Met_{L^s}, s \in S && (7) \\
& \sum_{e=(i,j) \in E^s} \mu_{ij} B x_e \geq \sum_{\ell \in L^s} \pi^\mu_\ell y_\ell \qquad \mu \in Met_{E^s}, s \in S && (8) \\
& x_e, y_\ell \in \mathbb{Z}_+
\end{aligned}
$$

where (7) and (8) are metric inequalities for the logical and the physical layer. Unfortunately, formulation (ECF) and (ICF) are not equivalent.
Consider for example the problem of Figure 2 a). Let the demands be $d_{12} = d_{23} = 0$ and $d_{13} = 1$. Let $S = \{s_0\}$ (there are no failure scenarios), let $U = B = 1$. Let (x, y) be the solution having $y_{4\_123} = 1$, $y_{1\_12} = y_{2\_13} = y_{3\_23} = 0$ and $x_{13} = 1$, $x_{12} = x_{23} = 0$. It is easy to see that (x, y) is feasible for (ICF), but it is not feasible for (ECF), and therefore for (EFF), because no physical capacity is installed to support logical edge 4_123. Things are no better if we consider logical networks without parallel edges. Let us consider for example the two-layer network of Figure 2 b). Let $B = U = 1$, let the demands be $d_{13} = 1$, $d_{12} = d_{23} = 0$. Solution (x, y) defined as $y_{2\_13} = 1$, $y_{1\_12} = y_{3_23} = 0$, $x_{12} = x_{23} = 1$, $x_{13} = 0$, is feasible for (ICF), but not for (ECF).

In fact, even the simple assumption of choosing in advance physical paths for the lightpaths, as we do in (EFF) and (ECF), is a routing constraint that can not be handled by metric inequalities (8). Formulation (ICF) can be used only under the assumption that the physical paths for the

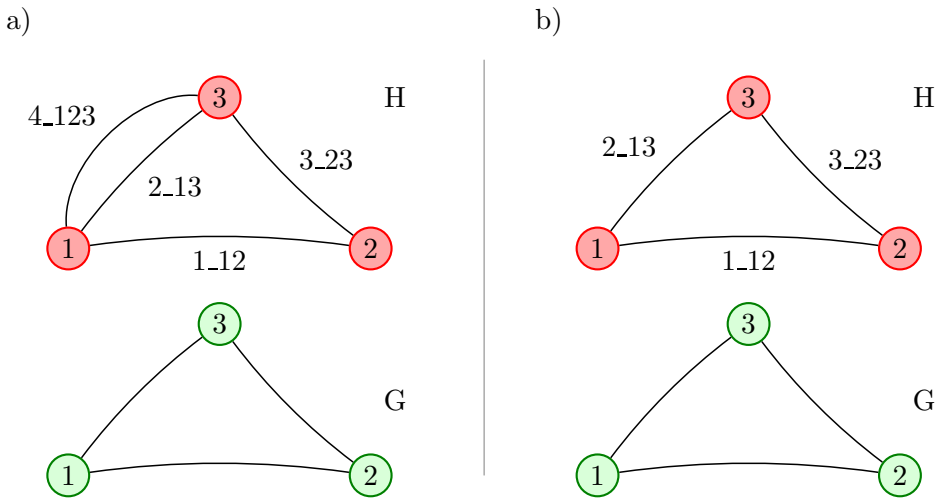

Figure 2: explicit VS implicit lightpaths

lightpaths are not known in advance. For this reason we distinguish between two possible lightpaths definitions: explicit lightpaths and implicit lightpaths (see also [24]). By *explicit* lightpaths we mean that a set of available lightpaths is given: for each lightpath ℓ a physical routing E_ℓ is known a priori, and there may be parallel lightpaths. This is the usual definition of lightpaths, the one we used so far. In this case it is not possible to use metric inequalities for the physical layer and we must use formulation (EFF) or (ECF). By *implicit* lightpaths, we mean that only a set of node pairs that can be potentially connected by a lightpath is given: the physical routing of the lightpaths is not known in advance. To each logical capacity module installed on a lightpath can correspond a different physical path. Moreover a lightpath can be routed on different paths depending on the scenario. In this case the problem can be modeled as (ICF). The corresponding arc-flow formulation (IFF) is given below.

Let $f^{k,s}_{\ell,ij}$, and $f^{k,s}_{\ell,ji}$ be the usual flow variables for the commodities. Let $p^{\ell,s}_{ij}$ and $p^{\ell,s}_{ji}$ be the flow on the physical layer corresponding to the routing of the logical capacity installed on lightpath $\ell \in L$, going from i to j and viceversa on $e = (i,j) \in E$ for every scenario $s \in S$. For a lightpath $\ell \in L$ let s_ℓ and t_ℓ be the endpoints of ℓ.

$$
\begin{aligned}
(IFF) \qquad & \min \sum_{e \in E} c^E_e x_e + \sum_{\ell \in L} c^L_\ell y_\ell \\
& \sum_{\ell \in L^s} (p^{\ell,s}_{ij} + p^{\ell,s}_{ij}) \leq B x_e \qquad e = (i,j) \in E^s, s \in S && (9) \\
& \sum_{k \in K^s} (f^{k,s}_{\ell,ij} + f^{k,s}_{\ell,ji}) \leq U y_\ell \qquad \ell = (i,j) \in L^s, s \in S && (10) \\
& \sum_{j \in V^s} \sum_{\ell \in L^s_{ij}} (f^{k,s}_{ij} - f^{k,s}_{ji}) = d^k_i \qquad i \in V^s, k \in K^s, s \in S && (11) \\
& \sum_{j \in V^s : (i,j) \in E^s} (p^{\ell,s}_{ij} - p^{\ell,s}_{ji}) = y_\ell(i) \qquad i \in V^s, \ell \in L^s, s \in S && (12) \\
& f, p \geq 0 \\
& x_e, y_\ell \in \mathbb{Z}_+
\end{aligned}
$$

Where $y_\ell(i) = y_\ell$ if $i = s_\ell$, $y_\ell(i) = -y_\ell$ if $i = t_\ell$, $y_\ell(i) = 0$ otherwise. Constraints (9) and (10) are capacity constraints for the physical layer and for the logical layer respectively. Constraints (11), (12) are flow conservation constraints for the logical layer (commodities) and for the physical layer (lightpaths) respectively.
Formulations (IFF) and (ICF) also include the implicit assumption (not always reasonable in practice) that the cost of a lightpath depends only on its endpoints and not on the physical path where it is routed. To take into account lightpath routing costs, in formulation (IFF) it is possible to associate costs to variables p. Nothing can be done for (ICF) as it is.

An additional remark about the relationship between the flow (or path) formulation of the problem and the corresponding capacity version is the following. The flow formulation explicitly computes at the same time optimal capacities and the corresponding feasible routing. A capacity formulation only computes optimal capacities ensuring that, with the given capacities, such a routing exists. It can be computed in a second time by fixing the capacity variables to their optimal values and solving the flow formulation. Therefore solving (EFF) or (IFF) we get both the capacity values and the routing, solving their capacity versions (ECF) and (ICF) we only get the capacity values while the routing must be computed in a second time solving an LP problem.

The two-layer metric formulation and the flow formulation for implicit lightpaths for the case without failures are given in [17] and [6]. We extend them to take into account failure scenarios. We were not able to find previous references for formulation (ECF).

Usually, only one of the two lightpath models is considered. The aim of this work is to propose a solution approach both for the explicit lightpaths model and for the implicit case. For each problem we present a branch-and-cut algorithm based on the capacity formulation. We use formulation (ECF) for explicit lightpaths and (ICF) for implicit lightpaths.

3 The algorithm for explicit lightpaths

In this section we present a branch-and-cut approach for solving the problem when explicit lightpaths are given. Separation and heuristic techniques to be used within a branch-and-cut framework are presented.

Branch-and-cut combines branch-and-bound and cutting plane methods, see also [28]. Starting from an initial formulation including only a reduced number of inequalities, the current problem without the integrality requirements is solved. The optimal solution of the current problem is tested for feasibility. If a violated inequality is found, the inequality is added to the current formulation and the problem is solved again. This process is repeated until a feasible solution is found. If the solution is fractional, then a branching step is performed.

3.1 Preprocessing and initial formulation

Let us consider the single-layer problem described in Section 2. Let a be an edge of the network and let c_a be its cost. If a is not a bridge, which means that the removal of a does not disconnect the network, let P_a be the shortest path between the endpoints of a after removing a from the network, and let $c(P_a)$ be the corresponding cost. If $c_a \geq c(P_a)$ then a can be removed from the network, since the capacity installed on a can be installed on P_a at a lower cost. Even this simple consideration is not valid, in general, for a two-layer problem. Consider for example the simple network on three nodes given in Figure 3. Let the physical costs be $c^E_{12} = 3$, $c^E_{13} = 1$ and $c^E_{23} = 5$. Let the logical costs be $c^L_{1\_12} = 2$, $c^L_{2\_13} = 9$ and $c^L_{3\_23} = 1$. Let us consider a problem without

scenarios and let the demands be $d_{23} = 1$, $d_{12} = d_{23} = 0$. Let $B = 2$ and let $U = 1$. Even if for physical edge $(2,3)$ we have that $c_{23} > c(P_{23})$, where $P_{23} = \{(1,2),(1,3)\}$, edge $(2,3)$ can not be eliminated. In fact the optimal solution of the problem is $x_{23} = y_{3\_23} = 1$, $x_{12} = y_{1\_12} = 0$, $x_{13} = y_{1\_13} = 0$.

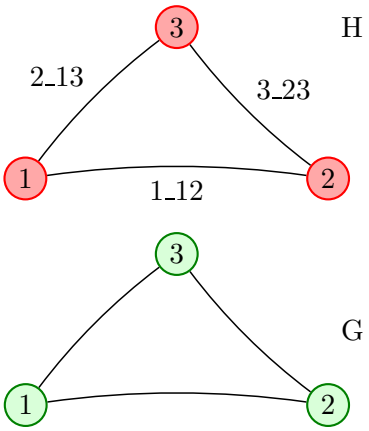

Figure 3: preprocessing example

The following (very weak) condition can be used. Let $\ell, t \in L$ such that $c^L_\ell > c^L_t + c^E(E_t)$, where $c^E(E_t)$ is the cost of installing capacities on the physical path corresponding to lightpath t, then ℓ can be eliminated. In fact we get a lower cost using t instead of ℓ, even if it implies to buy additional physical capacities for all the physical edges involved. In addition, since we have scenarios, before removing any edge, we have to check each scenario. If the above condition is satisfied in every scenario then the edge can be removed. If a physical edge e does not support any lightpath, it can be eliminated from the problem.

Let $d_{K^s}(i)$ be the sum of all the demands in K^s for which i is the source node or the destination node, that is: $d_{K^s}(i) = \sum_{k \in K^s : s^k = i \text{ or } t^k = i} d^k$. Let $\delta_{E^s}(i)$ and $\delta_{L^s}(i) = \cup_{j \in V^s} L^s_{ij}$ be the set of physical links and logical links incident to i in scenario s. We start with the following initial formulation:

$$
\begin{aligned}
&\min \sum_{e \in E} c^E_e x_e + \sum_{\ell \in L} c^L_\ell y_\ell \\
&\sum_{\ell \in \delta_{L^s}(i)} y_\ell \geq \left\lceil \frac{d_{K^s}(i)}{U} \right\rceil \qquad i \in V^s, s \in S \qquad (13) \\
&\sum_{\ell \in L_e} y_\ell \leq B x_e \qquad e \in E \\
&x_e, y_\ell \in \mathbb{Z}_+
\end{aligned}
$$

where constraints (13) are single-node cut inequalities (and therefore metric inequalities) for the logical layer. They simply say that the capacity on edges incident to node i must support the total demand of node i, see [2] and [7] for similar constraints. To help reducing the size of the branch-and-bound tree, upper bounds for physical and logical capacities can be computed as follows.

$$
UB_\ell = \left\lceil \frac{\sum_{k \in K} d^k}{U} \right\rceil \qquad \ell \in L
$$

$$UB_e = \left\lceil \frac{\sum_{\ell \in L_e} UB_\ell}{B} \right\rceil \qquad e \in E$$

To obtain better x values in the first iterations of the algorithm, we also add to the initial formulation the following constraints.

$$\sum_{e \in \delta_{E^s}(i)} x_e \geq \left\lceil \frac{\left\lceil \frac{d_{Ks}(i)}{U} \right\rceil}{B} \right\rceil \qquad i \in V^s, s \in S \tag{14}$$

They are single node cut inequalities for the physical layer.

3.2 Separation procedure

Let $(\bar{x}, \bar{y})$ be the optimal solution of the current problem. Let D^s be the set of the source nodes of the commodities in K^s and let $C^s = \{(i,j) \in N \times N : i < j, L^s_{ij} \neq \emptyset\}$ in scenario $s \in S$. To verify whether $\bar{y}$ is a (non necessarily integer) feasible logical capacity vector for scenario s, we solve:

$$\begin{aligned}
(sepY^s) \qquad & \min \sum_{(i,j) \in C^s} \sum_{\ell \in L^s_{ij}} U \bar{y}_\ell \mu_{ij} - \sum_{k \in K^s} \pi^k_\mu d^k \\
& \pi^{oj}_\mu \leq \pi^{oi}_\mu + \mu_{ij} \qquad o \in D^s, (i,j) \in C^s \\
& \pi^{oi}_\mu \leq \pi^{oj}_\mu + \mu_{ji} \qquad o \in D^s, (i,j) \in C^s \\
& \sum_{(i,j) \in C^s} \mu_{ij} = 1 \qquad (15) \\
& \mu \geq 0
\end{aligned}$$

Where (15) is a normalization constraint to avoid unboundedness. The above formulation is the two-layer version of the separation oracle used in [2] for finding violated strong metric inequalities. For other separation oracles for metric inequalities see for example [7], [17] and references therein.

If the optimal value of $(sepY^s)$ is greater than or equal to zero, then $\bar{y}$ is a feasible capacity vector for the logical layer in scenario s, otherwise, let $(\bar{\mu}, \bar{\pi})$ be the optimal solution of the separation problem, we get the violated inequality:

$$\sum_{(i,j) \in C^s} \sum_{\ell \in L^s_{ij}} \bar{\mu}_{ij} U y_\ell \geq \sum_{k \in K^s} \bar{\pi}^\mu_k d^k \tag{16}$$

We strengthen the above inequality by dividing all coefficients and the right-hand-side by $\bar{m} = \min\{\bar{\mu}_{ij} : (i,j) \in C^s\}$ and rounding to the upper nearest integer. In addition, we apply a second rounding after dividing by U, obtaining constraint:

$$\sum_{(i,j) \in C^s} \sum_{\ell \in L^s_{ij}} \left\lceil \frac{\bar{\mu}_{ij}}{\bar{m}} \right\rceil y_\ell \geq \left\lceil \frac{\left\lceil \frac{\sum_{k \in K^s} \bar{\pi}^\mu_k d^k}{\bar{m}} \right\rceil}{U} \right\rceil \tag{17}$$

If (17) is violated, we use it. Otherwise we use inequality (16). Let us note that the dimension of the separation problem does not depend on the number of parallel lightpaths, therefore the separation scheme is, in a sense, robust with respect to the parallel lightpaths.

At each iteration, we look for a violated inequality by checking all scenarios and we stop as soon as we found one. Therefore we add only one violated inequality per iteration. In addition we set a maximum number of cuts to be added at each node of the branch-and-bound tree. When this number is reached, we look for a violated inequality only if the solution is integer, otherwise we branch. We do that for several reasons. The most important ones are: (i) to keep the problem small, (ii) because there is a trade-off between cutting and branching and, sometimes, it is easier to cut a fractional solution by branching than by cutting-plane. Preliminary comparative tests confirmed the effectiveness of this strategy. The maximum number of cuts allowed must be carefully chosen, in order to avoid the generation of a huge number of branch-and-bound nodes.

3.3 Heuristic procedure

In this section, we describe the heuristic procedure we use during the exploration of the branch-and-bound tree. Let $(\bar{x}, \bar{y})$ be the capacity vector corresponding to the current LP solution, non necessarily integer. We first get rid of $\bar{x}$ variables and then apply Algorithm 1.

Algorithm 1 branch-and-bound heuristic

procedure B&B-HEURISTIC($\bar{y}$)
  compute $\hat{y}$ ▷ round components of $\bar{y}$ to the nearest integer
  compute $\hat{x}$ supporting $\hat{y}$ ▷ $\hat{x}_e = \left\lceil \frac{\sum_{\ell \in L_e} \hat{y}_\ell}{B} \right\rceil$
  response = false
  if $\hat{y}$ is feasible **then** ▷ step 1: use algorithms of Section 3.2
    response = true
  end if
  if response = false **then** ▷ step 2: use Algorithm 2
    ROUTE($\hat{x}, \hat{y}$)
    response = true;
  end if
end procedure ▷ return $(\hat{x}, \hat{y})$ and the response

Algorithm 1 is made of two components (step 1 and step 2). In step 1 we round each component of $\bar{y}$ to the nearest integer, obtaining the integer vector $\hat{y}$. Vector $\hat{y}$ is tested for feasibility using the oracle described in Section 3.2. If $\hat{y}$ is a feasible capacity vector, we use the physical capacity constraints to compute physical capacities $\hat{x}$ supporting $\hat{y}$, thus obtaining a feasible solution for the problem. Otherwise we perform step 2 by applying Algorithm 2.
Algorithm 2 can also be used as a stand alone heuristic with initial capacities $x = 0$ and $y = 0$. The algorithm works as follows. At each iteration one scenario is considered. Physical and logical available capacities are fixed according to the current capacity values. The algorithm tries to route the commodities corresponding to the given scenario using a shortest path strategy. A commodity having a positive demand is chosen and the shortest path between its endpoints is computed. Logical edge costs are set as follows. The cost of an edge is zero if there is an available capacity on that edge, otherwise it is equal to the cost of installing a capacity module, plus the cost of installing additional physical capacity to support it, if needed (see Algorithm 3). The demand is routed on the computed path such that either the demand or the available capacity of at least one of the edges of the path is saturated. Demands, available capacities and edge costs are updated accordingly. The algorithm terminates when all the scenarios have been considered.
Since the two steps of Algorithm 1 are independent, each step can be separately turned off. If step 2 is not turned off, the response of the algorithm is always true, that is, it always returns a feasible solution. If step 2 is turned off, the algorithm returns a feasible solution only if $\hat{y}$ is a feasible capacity vector, which is not always the case.

Algorithm 2 routing procedure

procedure ROUTE(x,y)
  for all $s \in S$ **do**
    $Y_\ell = y_\ell$, $X_e = x_e$ ▷ X,Y vector of available capacities
    while $\exists k \in K^s : d^k > 0$ **do**
      choose $k : d^k > 0$
      compute weights c^R and capacities u ▷ see Algorithm 3
      find P^k, shortest s^k-t^k path in H^s
      for all $\ell \in P^k$ **do**
        if $Y_\ell = 0$ **then** ▷ install logical capacity
          $Y_\ell = U$,
          $y_\ell = y_\ell + 1$
          for all $e \in E_\ell$ **do**
            if $X_e = 0$ **then** ▷ install physical capacity
              $X_e = B - 1$
              $x_e = x_e + 1$
            end if
          end for
        end if
      end for
      compute $\delta = \min\{d^k, \min\{u_\ell : \ell \in P^k\}\}$
      $d^k = d^k - \delta$ ▷ route δ on P^k
      $Y_\ell = Y_\ell - \delta$, for $\ell \in P^k$ ▷ update available logical capacities
    end while
  end for
end procedure ▷ return (x, y)

Algorithm 3 procedure that compute edge weights and capacities

procedure COMPUTE EDGE WEIGHTS(Y,X)
  for all $\ell \in L$ **do**
    if $Y_\ell > 0$ **then**
      $c^R_\ell = 0$
      $u_\ell = Y_\ell$
    else
      $u_\ell = U$
      $c^R_\ell = c^L_\ell$
      for all $e \in E_\ell$ **do**
        if $X_e = 0$ **then**
          $c^R_\ell = c^R_\ell + c^E_e$
        end if
      end for
    end if
  end for
end procedure ▷ return c^R, u

4 The algorithm for implicit lightpaths

In this section we present a branch-and-cut approach for solving the problem in the case of implicit lightpaths. We remark that, in this case, no explicit routing for the lightpaths (E_ℓ) is given, but the routing of the lightpaths on the physical network is computed as a side result of the optimization process.

4.1 Preprocessing and initial formulation

In the case of implicit lightpaths, we can eliminate physical edges using the preprocessing technique for the single-layer problem described in Section 3.1. We can also apply the condition given in Section 3.1 for eliminating logical edges. In this case E_t is the longest path between the endpoints of t.

The initial formulation to be used in the branch-and-cut algorithm is the following.

$$\min \sum_{e \in E} c_e^E x_e + \sum_{\ell \in L} c_\ell^L y_\ell$$

$$\sum_{\ell \in \delta_{L^s}(i)} y_\ell \geq \left\lceil \frac{d_{K^s}(i)}{U} \right\rceil \qquad i \in V^s, s \in S \tag{18}$$

$$\sum_{e \in \delta_{E^s}(i)} x_e \geq \left\lceil \frac{\left\lceil \frac{d_{K^s}(i)}{U} \right\rceil}{B} \right\rceil \qquad i \in V^s, s \in S \tag{19}$$

$$x_e, y_\ell \in \mathbb{Z}_+$$

Constraints (18) and (19) are single node cut inequalities for the logical and the physical layer. Also in this case, to help reducing the size of the branch-and-bound tree, upper bounds on physical and logical capacities can be computed as follows.

$$UB_\ell = \left\lceil \frac{\sum_{k \in K} d^k}{U} \right\rceil \qquad \ell \in L$$

$$UB_e = \left\lceil \frac{\sum_{\ell \in L} UB_\ell}{B} \right\rceil \qquad e \in E$$

Since the routing of the lightpaths is not known in advance and a lightpath can use any physical link, upper bounds on physical capacities are weaker with respect to the ones that can be computed when explicit lightpaths are given.

4.2 Separation procedure

Given a scenario s, to verify if the current (possibly fractional) solution $(\bar{x}, \bar{y})$ is feasible for s, we apply two times the separation oracle for a single-layer problem. For logical feasibility, we use original commodities as demands and logical capacities as capacities. For physical feasibility, we use logical capacities as demands and physical capacities as capacities. The separation oracle for the single-layer problem is the following. Let $G(N, A)$ be a single-layer graph and let Q be a set of commodities to be routed on G. Let O be the set of source nodes of commodities Q and let q_i^o the amount to be sent from o to i. To test if a given vector of capacities $\bar{z}$ is feasible for the problem we can solve the problem below (see [2]).

$$
\begin{aligned}
(sepZ) \qquad & \min \sum_{(i,j)\in A} \bar{z}_{ij}\mu_{ij} - \sum_{o\in O}\sum_{i\in N} q_i^o \pi_\mu^{oi} \\
& \pi_\mu^{oj} \le \pi_\mu^{oi} + \mu_{ij} \qquad o \in O, (i,j) \in A \\
& \pi_\mu^{oi} \le \pi_\mu^{oj} + \mu_{ji} \qquad o \in O, (i,j) \in A \\
& \sum_{(i,j)\in A} \mu_{ij} = 1 \\
& \mu \ge 0
\end{aligned}
$$

If the optimal value of $(sepZ)$ is less than zero then we get a violated inequality, otherwise $\bar{z}$ is feasible. When applied to the logical layer, the resulting violated metric inequality is:

$$
\sum_{\ell=(i,j)\in L^s} \bar{\mu}_{ij} U y_\ell \ge \sum_{k\in K^s} \pi_k^\mu d^k \tag{20}
$$

We strengthen the above inequality by dividing the right-hand-side and the coefficient by $\bar{m} = \min\{\mu_{ij} : \ell = (i,j) \in L^s\}$ and rounding to the nearest integer, obtaining inequality:

$$
\sum_{\ell=(i,j)\in L^s} \left\lceil \frac{\bar{\mu}_{ij}}{\bar{m}} \right\rceil y_\ell \ge \left\lceil \frac{\left\lceil \frac{\sum_{k\in K^s} \bar{\pi}_k^\mu d^k}{\bar{m}} \right\rceil}{U} \right\rceil \tag{21}
$$

We use inequality (21), if it is still violated, otherwise we use inequality (20).
If applied to the physical layer, the above oracle returns, if any, the violated inequality below.

$$
\sum_{e=(i,j)\in E^s} \bar{\mu}_{ij} B x_e \ge \sum_{\ell\in L^s} \pi_\ell^\mu y_\ell \tag{22}
$$

We first check the logical feasibility, if no violated constraint for the logical layer exists, we look for a violated constraint for the physical layer.

4.3 Heuristic procedure

Let $(\bar{x}, \bar{y})$ be the current solution, we apply Algorithm 4. As in the previous case, Algorithm 4 consists of two steps and, since the steps are independent, each step can be separately turned off. In the first step each component of the current solution is rounded to the nearest integer, obtaining an integer vector of capacities. The vector is tested for feasibility using the separation oracle of Section 4.2. If it is feasible, then we stop, otherwise we perform step 2 that consists of applying Algorithm 5.
Algorithm 5 finds a feasible solution decomposing the problem by layer and sequentially solving the two single-layer sub-problems for every scenario. When the sub-problem corresponding to the logical layer is solved, the commodities play the role of demands and the logical capacities play the role of capacities. When the sub-problem corresponding to the physical layer is solved, the logical capacities play the role of demands and the physical capacities play the role of capacities. As in the previous case, Algorithm 5 can also be used as a stand alone heuristic. The algorithm solves a sequence of problems, one for each scenario.

Algorithm 6 is used to solve the single-layer problem for a given scenario. Available capacities are fixed according to the given capacity vector. Until a commodity with a positive demand exists, the algorithm selects such a commodity and tries to route it using a shortest path approach. Edge costs

Algorithm 4 branch-and-bound heuristic for implicit logical edges

```
1: procedure B&B-HEURISTIC_I((x, y))
2:     compute (x̂, ŷ)                          ▷ round x and y to the nearest integer
3:     response = false
4:     if x̂ and ŷ are feasible then              ▷ step 1: use algorithms of Section 4.2
5:         response = true
6:     end if
7:     if response = false then                  ▷ step 2: use Algorithm 5
8:         HEURISTIC_I(x,y)
9:         response = true;
10:    end if
11: end procedure                                ▷ return (x̂, ŷ) and the response
```

Algorithm 5 step 2 of the heuristic for implicit logical edges

```
1: procedure HEURISTIC_I(x,y)
2:     initialize (x̂, ŷ)                        ▷ round x and y to the nearest integer
3:     for all s ∈ S do
4:         ROUTE_I(ŷ,K^s)                        ▷ see Algorithm 6
5:         for all s ∈ S do
6:             ROUTE_I(x̂,ŷ)                      ▷ see Algorithm 6
7:         end for
8:     end for
9: end procedure                                 ▷ return (x̂, ŷ)
```

are defined as follows (see Algorithm 7): the cost of an edge is zero if there is an available capacity on that edge, otherwise it is equal to the cost of installing a capacity module. The demand is routed on the computed path such that either the demand or the available capacity of at least one edge of the path is saturated. Demands, capacities and edge costs are updated accordingly. Parameter M of Algorithm 6 is the capacity module and c are the costs. If the routing is for the logical network $F(N, A) = H^s(V^s, L^s)$, $c = c^L$ and $M = U$, if it is for the physical network $F(N, A) = G^s(V^s, E^s)$, $c = c^E$ and $M = B$. The same happens for Algorithm 7.

Algorithm 6 routing procedure for a single-layer problem

```
1: procedure ROUTE_I(z,D)
2:     Z_a = ẑ_a                                 ▷ Z vector of available capacities
3:     while ∃k ∈ D : d^k > 0 do
4:         choose k : d^k > 0
5:         compute edge weights c^R and capacities u   ▷ see Algorithm 7
6:         find P^k, shortest s^k-t^k path in F(N, A)
7:         for all a ∈ P^k do
8:             if Z_a = 0 then                   ▷ install additional capacity
9:                 Z_a = M,
10:                z_a = z_a + 1
11:            end if
12:        end for
13:        compute δ = min{d^k, min{u_a : a ∈ P^k}}
14:        d^k = d^k − δ                         ▷ route δ on P^k
15:        Z_a = Z_a − δ, for a ∈ P^k            ▷ update available capacities
16:    end while
17: end procedure                                ▷ return z
```

Algorithm 7 procedure that compute edge weights and capacities for a single-layer problem

procedure COMPUTE_EDGE_WEIGHTS_AND_COSTS_I(Z)
  for all $a \in A$ **do**
    if $Z_a > 0$ **then**
      $c_a^R = 0$
      $u_a = Z_a$
    else
      $u_a = M$
      $c_a^R = c_a$
    end if
  end for
end procedure ▷ return c^R, u

5 Computational results

The branch-and-cut algorithms described in the previous sections have been implemented in $C++$, using `CPLEX 11.2`. It was tested on a laptop having a 2.20 GHz Intel Core 2 Duo processor, with 4Gb RAM. We turned off most of CPLEX features: presolve, CPLEX cuts and heuristics, the option to purge cuts if they seem ineffective. This was done in order to prove the effectiveness of our cutting plane and heuristic scheme. We set a time limit of 1 hour for every problem. We present results on 84 instances. Experiments have been made using instances derived from SNDlib networks [26]. We used SNDlib instances as physical networks and derived logical networks. We used original demands but, during optimization, we replace commodities (i, j, d_{ij}) and (j, i, d_{ji}), if any, by a unique commodity $(i, j, d_{ij} + d_{ji})$, since the graphs are undirected. In Table 1 we summarize the physical networks data reporting for every instance, the number of nodes, edges and commodities.

name	nodes	edges	commodities
atlanta	15	22	210
nobel-germany	16	26	121
nobel-us	14	21	91
polska	12	18	66
di-yuan	11	42	22
pdh	11	36	24
cost266	37	57	1368
nobel-eu	28	41	378

Table 1: physical networks data

We chose those instances in order to test our approach in three different settings: (i) in a standard situation, (ii) when the number of edges increases, and (iii) when the number of nodes increases. Instances of group 1 (atlanta, nobel-germany, nobel-eu, polska) are regular-size instances on sparse physical networks. Instances of group 2 (di-yuan, pdh) are regular-size instances on (almost) complete physical networks. Instances of group 3 (cost266, nobel-eu) are big-size instances on sparse physical networks. Instance of group 1 are the usual target of optical network planning. Instances of group 2 have complete physical networks, which is not true in real-life instances, corresponding to very dense logical networks. Instances of group 3 have been included since we think that optical network planning algorithms will be forced to move towards those kind of networks in order to address real-life problems.

For the case of explicit lightpaths (EL), we derived logical networks with and without hop limits. For the case of implicit lightpaths (IL), we considered logical networks having a lightpath between every pair of nodes. In Table 2 we report, for every instance, the number of logical edges and the maximum number of hops allowed, if any. For all instances we set the size of the physical capacity module to $B = 8$. The size of the logical capacity module is set to a value depending on the mean demand of each instance. Physical costs are derived from the costs of the first available capacity module of the original problem. Logical costs are randomly generated. As for the scenarios, we generated a single node failure scenario for every node of the network, therefore the number of scenarios for every instance is equal to the number of nodes. We choose to generate node failure scenarios instead of the usual edge failure scenarios (see for example [8]), because the failure of a node is, in our opinion, a much more critical event than the failure of an edge. In fact, the failure of a node also implies the failure of all edges (physical and logical) incident to that node. However, the proposed approach is general and it can be used for any kind of scenarios. When explicit lightpaths are given, a lightpath $\ell \in L$ non incident to the disconnected node, is active under failure scenario s ($\ell \in L^s$) if all physical edges in E_ℓ are active. When implicit lightpaths are given, a lightpath ℓ

name	logical edges	max hops allowed
atlanta-3hop-EL	295	3
atlanta-5hop-EL	899	5
atlanta-Thop-EL	5436	-
nobel-germany-3hop-EL	560	3
nobel-germany-5hop-EL	1989	5
nobel-germany-Thop-EL	13641	-
nobel-us-3hop-EL	314	3
nobel-us-5hop-EL	1107	5
nobel-us-Thop-EL	7113	-
polska-3hop-EL	273	3
polska-5hop-EL	810	5
polska-Thop-EL	2457	-
di-yuan-3hop-EL	10809	3
di-yuan-5hop-EL	201930	5
pdh-3hop-EL	4706	3
pdh-5hop-EL	54582	5
cost266-3hop-EL	1133	3
cost266-5hop-EL	5096	5
nobel-eu-3hop-EL	718	3
nobel-eu-5hop-EL	2900	5
atlanta-IL	105	-
nobel-germany-IL	120	-
nobel-us-IL	91	-
polska-IL	66	-
di-yuan-IL	55	-
pdh-IL	55	-
cost266-IL	666	-
nobel-eu-IL	378	-

Table 2: logical networks size for explicit and implicit lightpaths

	branch-and-cut				flow formulation			
problem	bestUB	bestLB	gap	sec	bestUB	bestLB	gap	sec
atlanta-50-3hop-EL	2475	2475	0%	775.71	4469	686.56	84.63%	1h
atlanta-50-5hop-EL	2470	2470	0%	1378.95	-	-	-	1h
atlanta-50-Thop-EL	2470	2470	0%	1102	-	-	-	1h
nobel-germany-50-3hop-EL	5238	5238	0%	427.54	-	-	-	1h
nobel-germany-50-5hop-EL	5214	5214	0%	383.74	-	-	-	1h
nobel-germany-50-Thop-EL	5214	5214	0%	2119.34	-	-	-	1h
nobel-us-50-3hop-EL	1366	1366	0%	215.60	2946	299.86	89.82%	1h
nobel-us-50-5hop-EL	1366	1366	0%	184.18	-	-	-	1h
nobel-us-50-Thop-EL	1366	1366	0%	512.80	-	-	-	1h
polska-50-3hop-EL	3063	3063	0%	32.57	4811	985.87	79.50%	1h
polska-50-5hop-EL	3063	3063	0%	24.05	5347	824.09	84.58%	1h
polska-50-Thop-EL	3063	3063	0%	42.88	-	-	-	1h
di-yuan-50-3hop-EL	423	423	0%	2704.54	-	-	-	1h
di-yuan-50-5hop-EL	423	360	14.89%	1h	-	-	-	1h
pdh-50-3hop-EL	1186	1186	0%	1101.38	-	-	-	1h
pdh-50-5hop-EL	1186	1186	0%	3306.28	-	-	-	1h
cost266-50-3hop-EL	24888	20976	15.71%	1h	-	-	-	1h
cost266-50-5hop-EL	24888	20951	15.81%	1h	-	-	-	1h
nobel-eu-50-3hop-EL	1710	1655	3.21%	1h	-	-	-	1h
nobel-eu-50-5hop-EL	1707	1657	2.92%	1h	-	-	-	1h

Table 3: results for explicit lightpath instances with 50% of protected commodities

	branch-and-cut				flow formulation			
problem	bestUB	bestLB	gap	sec	bestUB	bestLB	gap	sec
atlanta-70-3hop-EL	2503	2503	0%	490.51	-	-	-	1h
atlanta-70-5hop-EL	2500	2500	0%	865.24	-	-	-	1h
atlanta-70-Thop-EL	2500	2500	0%	631.55	-	-	-	1h
nobel-germany-70-3hop-EL	5310	5310	0%	354.55	-	-	-	1h
nobel-germany-70-5hop-EL	5280	5280	0%	2341.75	-	-	-	1h
nobel-germany-70-Thop-EL	5280	5264	0.30%	1h	-	-	-	1h
nobel-us-70-3hop-EL	1375	1375	0%	176.74	-	-	-	1h
nobel-us-70-5hop-EL	1375	1375	0%	82.16	-	-	-	1h
nobel-us-70-Thop-EL	1375	1375	0%	224	-	-	-	1h
polska-70-3hop-EL	3151	3151	0%	17.12	4908	982.49	79.98%	1h
polska-70-5hop-EL	3151	3151	0%	21.38	-	-	-	1h
polska-70-Thop-EL	3151	3151	0%	29.74	-	-	-	1h
di-yuan-70-3hop-EL	423	423	0%	537.29	-	-	-	1h
di-yuan-70-5hop-EL	423	423	0%	1973.51	-	-	-	1h
pdh-70-3hop-EL	1223	1223	0%	315.94	-	-	-	1h
pdh-70-5hop-EL	1223	1223	0%	1446.51	-	-	-	1h
cost266-70-3hop-EL	24858	20983	15.58%	1h	-	-	-	1h
cost266-70-5hop-EL	24858	20908	15.89%	1h	-	-	-	1h
nobel-eu-70-3hop-EL	1714	1684	1.75%	1h	-	-	-	1h
nobel-eu-70-5hop-EL	1714	1667	2.74%	1h	-	-	-	1h

Table 4: results for explicit lightpath instances with 70% of protected commodities

	branch-and-cut				flow formulation			
problem	bestUB	bestLB	gap	sec	bestUB	bestLB	gap	sec
atlanta-100-3hop	2575	2575	0%	164.50	-	-	-	1h
atlanta-100-5hop	2568	2568	0%	285.71	-	-	-	1h
atlanta-100-Thop	2568	2568	0%	432.40	-	-	-	1h
nobel-germany-100-3hop	5437	5437	0%	210.77	-	-	-	1h
nobel-germany-100-5hop	5406	5406	0%	268.72	-	-	-	1h
nobel-germany-100-Thop	5406	5377	0.53%	1h	-	-	-	1h
nobel-us-100-3hop	1420	1420	0%	139.91	-	-	-	1h
nobel-us-100-5hop	1420	1420	0%	82.74	-	-	-	1h
nobel-us-100-Thop	1420	1420	0%	304.80	-	-	-	1h
polska-100-3hop	3254	3254	0%	24.07	4978	1195.1	75.99%	1h
polska-100-5hop	3245	3245	0%	22.29	-	-	-	1h
polska-100-Thop	3245	3245	0%	28.89	-	-	-	1h
di-yuan-100-3hop	423	423	0%	655.18	-	-	-	1h
di-yuan-100-5hop	423	423	0%	1189.84	-	-	-	1h
pdh-100-3hop	1274	1274	0%	881.71	-	-	-	1h
pdh-100-5hop	1274	1250	1.88%	1h	-	-	-	1h
cost266-100-3hop	26862	21113	21.40%	1h	-	-	-	1h
cost266-100-5hop	26862	21055	21.61%	1h	-	-	-	1h
nobel-eu-100-3hop	1739	1710	1.66%	1h	-	-	-	1h
nobel-eu-100-5hop	1739	1701	2.18%	1h	-	-	-	1h

Table 5: results for explicit lightpath instances with 100% of protected commodities

	branch-and-cut				flow formulation			
problem	bestUB	bestLB	gap	sec	bestUB	bestLB	gap	sec
atlanta-50-IL	2068	2068	0%	3227.11	-	927.88	-	1h
nobel-germany-50-IL	6564	4948	24.61%	1h	-	-	-	1h
nobel-us-50-IL	1455	1193	18%	1h	-	540.63	-	1h
polska-50-IL	2872	2872	0%	1281.77	3140	1501.01	52.19%	1h
di-yuan-50-IL	336	336	0%	5.14	656	38.41	94.14%	1h
pdh-50-IL	1075	1075	0%	12.19	2016	208.43	89.66%	1h
cost266-50-IL	33306	19263	42.16%	1h	-	-	-	1h
nobel-eu-50-IL	2241	1622	27.62%	1h	-	-	-	1h
atlanta-70-IL	2070	2070	0%	1401.79	-	-	-	1h
nobel-germany-70-IL	6757	4951	26.72%	1h	-	-	-	1h
nobel-us-70-IL	1462	1193	18.39%	1h	-	408.74	-	1h
polska-70-IL	2881	2881	0%	1405.67	7.16e+012	1489.04	100%	1h
di-yuan-70-IL	383	383	0%	75.81	-	21.50	-	1h
pdh-70-IL	1135	1135	0%	45.64	1629	424.3	73.95%	1h
cost266-70-IL	38370	19266	49.78%	1h	-	-	-	1h
nobel-eu-70-IL	2233	1622	27.36%	1h	-	-	-	1h
atlanta-100-IL	3258	2140	34.31%	1h	-	-	-	1h
nobel-germany-100-IL	7534	4983	33.85%	1h	-	-	-	1h
nobel-us-100-IL	1478	1205	18.47%	1h	-	410.28	-	1h
polska-100-IL	2915	2910	0.17%	1h	-	1669.39	-	1h
di-yuan-100-IL	383	383	0%	60.04	-	22.74	-	1h
pdh-100-IL	1146	1146	0%	136.87	1.07e+013	456.22	100%	1h
cost266-100-IL	42460	19281	54.59%	1h	-	-	-	1h
nobel-eu-100-IL	2384	1623	31.92%	1h	-	-	-	1h

Table 6: results for implicit lightpath instances

not incident to the disconnected node is active in s if there is a path in G^s between its endpoints. We partitioned the original demands K into two subsets: *protected* commodities (PK) and *simple* commodities (SK). We generated instances where 50%, 70% and 100% of the original demands are protected. For every scenarios $s \in S, s \neq s_0$ we set $K^s = PK - IPK^s$, where IPK^s is the set of protected commodities that are not active in scenario s. A commodity k is active in scenario s, if s_k and t_k are active and there is a path in the logical network H^s between s_k and t_k. We set $K^{s_0} = K$. We chose to generate K^s in this way according to [18], but the proposed algorithm can be used for every possible definition of K^s.

In Tables and 3, 4, 5 we summarize computational results for explicit lightpath instances when 50%, 70%, and 100% respectively of the demands are protected. We compare the results obtained by our branch-and-cut algorithm with one ones obtained by CPLEX. CPLEX solves formulation (EFF) and it is used with default settings and a time limit of 1 hour. The tables can be read as follows. Column *problem* is the instance name. Column *bestUB* reports the best lower bound for the problem produced by the given algorithm. Column *bestLB* is the best lower bound. Column *gap* indicates the gap between the best lower bound and the best upper bound computed as $100 * (bestUB - bestLB)/bestUB$. Column *sec* is the computational time (in seconds) needed to solve the problem, a value $1h$ in this column means that the time limit has been reached. CPLEX is not able to solve any of the problems to optimality. A missing result (-) for an instance means that (i) either CPLEX is not able to solve the LP relaxation within the given time limit, or (ii) it goes out of memory, or (iii) it is not possible to build/load the problem within the given time limit. On the other hand, results for the branch-and-cut algorithm are very good. We are able to solve to optimality most of the instances and to obtain small gap values for the others, including the ones of group 3. Before starting branch-and-bound for 3-hop instances, we use algorithm 2 to find a first feasible solution for the problem. We use the 3-hop solution in the 5-hop problem, and

the solution of the 5-hop problem in the T-hop problem. In our experience, the improvement in the objective function due to long lightpaths is minimal, at least for the tested instances.

In Table 6, we summarize computational results for implicit lightpaths instances. CPLEX solves formulation (IFF) and it is used with default settings and a time limit of 1 hour. The columns have the same meaning of the ones in Tables 3, 4 and 5. Also in this case, CPLEX is not able to solve any of the instances to optimality. Results for the branch-and-cut algorithm for implicit lightpaths problems are interesting, even if not as good as the ones for explicit lightpaths. We think that this is due to the nature of the problem. In fact, in the case of explicit lightpath, for each lightpath a path in the physical layer is given in advance. This lead to a more rigid structure of the problem and to a very strict connection between the two layers: given optimal logical capacities, optimal physical capacities can be easily determined using physical capacity constraints. The implicit lightpath problem instead, has more degrees of freedom, since the physical paths associated to the lightpaths are not set in advance and must be computed during the optimization process. Unlike the previous case, given optimal logical capacities, it is necessary to solve an optimization problem to compute the corresponding optimal physical capacities.

There is no common testbed for optical network design problems and many variants of the problem has been proposed, making difficult to compare computational results. In [3] instances based on NSFNET, a network having 14 nodes/22 edges, are considered. They do not solve the problem to optimality but use truncated branch-and-bound to produce feasible solutions. In [6] randomly generated instances and instances derived from SNDlib networks are used. The randomly generated networks have 8 nodes/14 edges, 9 nodes/16 edges, 10 nodes/20 edges, respectively. The largest network picked from the SNDlib has 15 nodes/22 edges. In [25] instances based on six SNDlib networks are used. Here the largest network has 40 nodes/89 edges, but they propose MIP-based primal heuristics, while we propose an exact approach. The largest network used in [18] has 17 nodes/26 edges. In [13] instances based on NSFNET and on EON, a network having 19 nodes/39 edges are presented. It is not possible to directly compare the results of the above papers with the ones presented in this paper, since different versions of the problem are solved. However, we notice that optimal solutions are produced only in very few cases and for small size instances. On the other hand, we are able to solve most of the problems to optimality. We can not solve to optimality the problems on big instances, but in some cases gap values are really very small.

6 Conclusions and future research

In this paper we address the problem of designing a two-layer network with survivability requirements. We presented mathematical models and solution algorithms for two versions of the problem: when explicit lightpaths are given, and when the implicit lightpath approach is used.

For both problems we proposed a branch-and-cut algorithm based on a capacity formulation of the problem that uses metric inequalities. We also presented a heuristic algorithm to compute feasible solutions that can be used both within a branch-and-cut framework, and as a stand alone heuristic.

We presented computational results on 84 instances derived from SNDLib networks. The results show the proposed approach can be attractive to solve both problems, even if the results for the explicit model are better than the ones for the implicit case.

A direction for further research could be a polyhedral study of the proposed models, and a comparison of this approach with a branch-and-price algorithm based on the path formulation.

References

[1] A. Atamtürk. On capacitated network design cut-set polyhedra. *Mathematical Programming*, 92:425–437, 2002.

[2] P. Avella, S. Mattia, and A. Sassano. Metric inequalities and the network loading problem. *Disc. Opt.*, 4:103–114, 2007.

[3] D. Banerjee and B. Mukherjee. Wavelength-routed optical networks: Linear formulatation, resource budgeting tradeoffs, and a reconfiguration study. *IEEE Transactions on Networking*, 8(5):598–607, October 2000.

[4] P. Belotti, A. Capone, G. Carello, F. Malucelli, F. Senaldi, and A. Totaro. Design of multi-layer networks with traffic grooming and statistical multiplexing. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, April 2007.

[5] P. Belotti and F. Malucelli. Multilayer network design: a row-column generation algorithm. In *Proceedings of the 2nd International Network Optimization Conference (INOC 2005), Lisbon, Portugal*, volume 3, pages 422–427, March 2005.

[6] B. Bernard and M. Poss. An improved benders decomposition applied to a multi-layer network design problem. *Oper. Res. Lett.*, 37(5):777–795, 2009.

[7] D. Bienstock, S. Chopra, O. Günlük, and C.Y. Tsai. Mininum cost capacity installation for multicommodity flows. *Mathematical Programming*, 81:177–199, 1998.

[8] S. Borne, E. Gourdin, B. Liau, and A. Mahjoub. Design of survivable IP-over-optical networks. *Ann. Oper. Res.*, 146:41–73, 2006.

[9] A. M. Costa, JF Cordeau, and B. Gendron. Metric inequalities, cutset inequalities and benders feasibility cuts for multicommodity capacitated network design. *Computational Optimization and Applications*, 42(3):371–392.

[10] G. Dahl, A. Martin, and M. Stoer. Routing through virtual paths in layered telecommunication networks. *Operations Research*, 47(5):693–702, 1999.

[11] M. Deza. and M. Laurent. *Geometry of Cuts and Metrics*. Springer-Verlag, Berlin, 1997.

[12] M. Deza and E. Pantaleeva. Quasi-semi-metrics, oriented multi-cuts and related plyhedra. *Europ. J. Combinatorics*, 21:777–795, 2000.

[13] L. Gouveia, P. Patricio, and A. de Sousa. Hop-constrained node survivable network design: An application to MPLS over WDM. *Networks and Spatial Economics*, 8(1):3–21, March 2008. http://ideas.repec.org/a/kap/netspa/v8y2008i1p3-21.html.

[14] O. Günlük. A branch-and-cut algorithm for capacitated network design problems. *Mathematical Programming*, 86:17–39, 1999.

[15] M. Iri. On an extension of the max-flow min-cut theorem to multicommodity flows. *Journal of the Operations Research Society of Japan*, 13:129–135, 1971.

[16] H. Kerivin and A. Mahjoub. Design of survivable networks: A survey. *Networks*, 46(1):1–21, 2005. http://www.isima.fr/limos/publi/RR-05-04.pdf.

[17] A. Knippel and B. Lardeaux. The multi-layered network design problem. *Europ. J. Op. Res.*, 183:87–99, 2007.

[18] A. Koster, S. Orlowski, C. Raack, G. Baier, and T. Engel. *Single-layer Cuts for Multi-Layer Network Design Problems*, chapter 1, pages 1–23. Springer, 2008. Selected proceedings 9th INFORMS Telecommunications Conference.

[19] A. M. C. A. Koster and A. Zymolka. Provably good solutions for wavelength assignment in optical networks. In *Proceedings of ONDM 2005*, pages 335–345, Milan, Italy, 2005. The 9th IFIP Working Conference on Optical Network Design & Modelling.

[20] A.M.C.A. Koster and A. Zymolka. Tight LP-based lower bounds for wavelength conversion in optical networks. *Statistica Neerlandica*, 61(1):115–136, 2007.

[21] M. Lomonosov. Combinatorial approaches to multiflow problems. *Disc. Appl. Math.*, 11:1–93, 1985.

[22] S. Mattia. *The Network Loading Problem*. PhD thesis, Università degli Studi di Roma "La Sapienza", 2004.

[23] K. Onaga and O. Kakusho. On feasibility conditions of multicommodity flows in network. *IEEE Trans. Circuit Theory*, 18(4):425–429, 1971.

[24] S. Orlowski. *Optimal Design of Survivable Multi-layer Telecommunication Networks*. PhD thesis, TU Berlin. http://opus.kobv.de/tuberlin/volltexte/2009/2275/.

[25] S. Orlowski, A. Koster, C. Raack, and R. Wessäly. Two-layer network design by branch-and-cut featuring MIP-based heuristics. In *Proceedings of the INOC 2007*, Spa, Belgium, 2007. http://www.poms.ucl.ac.be/inoc2007/Papers/author.89/paper/paper.89.pdf.

[26] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly. SNDlib 1.0–Survivable Network Design Library. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, April 2007. http://sndlib.zib.de.

[27] S. Orlowski and R. Wessäly. An integer programming model for multi-layer network design. ZIB Preprint ZR-04-49, Konrad-Zuse-Zentrum für Informationstechnik Berlin, December 2004. http://www.zib.de/PaperWeb/abstracts/ZR-04-49.

[28] M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large scale symmetric traveling salesman problems. *SIAM review*, (33):60–100, 1991.

[29] D. Rajan. *Designing capacitated survivable networks: polyhedral analysis and algorithms*. PhD thesis, UC Berkeley, 2004.

[30] M. Stoer and G. Dahl. A polyhedral approach to multicommodity survivable network design. *NUMERISCHE MATHEMATIK*, 68(1), 1994.