The Multilayer Capacitated Survivable IP Network Design Problem: Valid inequalities and Branch-and-Cut

S. BORNE, E. GOURDIN, O. KLOPFENSTEIN, A. R. MAHJOUB
The Multilayer Capacitated Survivable IP Network Design Problem: valid inequalities and Branch-and-Cut

S. Borne1*, E. Gourdin2, Olivier Klopfenstein2 and A. R. Mahjoub1†

1 Laboratoire LIMOS, CNRS, Université Blaise Pascal - Clermont Ferrand II
Complexes scientifiques des Cézeaux, 63177 Aubière Cedex, France
sylvie.borne@lipn.univ-paris13.fr, mahjoub@lamsade.dauphine.fr
2 Orange Labs Research & Development
38-40 rue du Général-Leclerc, 92794 Issy-les-Moulineaux Cedex 9, France
(eric.gourdin,olivier.klopfenstein)@orange-ftgroup.com

Abstract: Telecommunication networks can be seen as the stacking of several layers like, for instance, IP-over-Optical networks. This infrastructure has to be sufficiently survivable to restore the traffic in the event of a failure. Moreover, it should have adequate capacities so that the demands can be routed between the origin-destinations. In this paper we consider the Multilayer Capacitated Survivable IP Network Design problem. We study two variants of this problem with simple and multiple capacities. We give two multicommodity flow formulations for each variant of this problem and describe some valid inequalities. In particular, we characterize valid inequalities obtained using Chvatal-Gomory procedure from the well known Cutset inequalities. We show that some of these inequalities are facet defining. We discuss separation routines for all the valid inequalities. Using these results, we develop a Branch-and-Cut algorithm and a Branch-and-Cut-and-Price algorithm for each variant and present extensive computational results.

Keywords: IP-over-optical network, survivability, capacities, Branch-and-Cut-and-Price algorithms.

1 Introduction

In the past years, telecommunication networks have seen a big development with the advances in optical technologies and the explosive growth of the Internet. Also the data traffic has increased dramatically and has now surpassed voice traffic in volume. Using the new optical technologies, different systems allow a very large increase of transport capacity and the transport of almost illimitated quantities of information. Hence, in the event of a catastrophic failure, a big amount of traffic may be lost. In consequence telecommunication networks must have a survivable topology, that is to say a topology that permits to the service to be restored and the network to remain functional in the event of a failure. For this, network survivability has become a major objective in the design of telecommunication networks.

Data networks have always been analysed, described and managed in a multilayer structure. Indeed, it is quite natural to assume that the more elaborate functionalities of a network rely on a set of simple ones provided by some lower layer. This is in particular the case of modern telecommunication networks where different technologies (SDH/SONET, WDM, Gigabit Ethernet, ATM, IP, . . . ) are combined in various ways on successive layers. From a practical point of view,
this means that, in order to carry its traffic on some layer, the network may need to use a lower-
level technology. Then several layers can be piled up in order to have an operational network
offering a variety of services. The advantage of this is that each technology can be used for its most
favorable features. Moreover, each technology is characterized by a certain range of traffic rates.
The drawback, however, is that each technology, and hence each layer, manages its own routing
control scheme independently from the others, and addresses its own survivability issues.

The capacities of a given layer correspond to the (worst-case) traffic demands that must be routed
on the layer just below. The process of determining the capacities to install on the different layers
of a network, usually called **dimensioning**, often reduces to a succession of multicommodity flow
problems. Usually there is an empirical relation between these problems, and the whole dimensioning
problem is never treated in an optimal way. As a consequence, in a network design problem, reliability
is considered layer by layer without tackling the redundancy and the non-optimality yielded by
the multilayer structure. Moreover, a failure in the network can be handled by several successive
layers. This results in a potential huge global over-provisioning of resources, each layer protecting
in turn the ones above. However the relation between technologies used in the different layers is
usually complex, and does not permit to efficiently correlate the control of the successive layers. In
consequence, the solution provided for this multilayer survivability problem usually consists of an
over protection of the whole network. But this may be very costly and sometimes not efficient.

The introduction of new protocols in telecommunication (like GMPLS) [44] gives a new trend
for multilayer data networks. This new system provides a common signaling and routing framework
between the different layers, and it does not restrict the way these layers work together. This
evolution is yielding new survivability issues in multilayer networks. In [43], Voge studies different
problems about the multilayer telecommunication networks based on MPLS and GMPLS.

In this paper we introduce a multilayer capacitated survivable network design problem that
may be of practical interest for the design and the dimensioning of IP-over-optical networks. These
networks, based on the GMPLS technology, consist of two layers, the IP (service, client) layer and
the optical (transport) layer. We give mixed integer programming formulations for this problem and
discuss Branch-and-Cut and Branch-and-Cut-and-Price algorithms.

Survivability and dimensioning have already been studied in the literature for multilayer net-
works. In particular, heuristic approaches have been proposed. In [18, 19], Gouveia and Patrício
study the design of MPLS-over-WDM networks. They address the dimensioning subject to some
path constraints in the WDM layer and hop constraints in the MPLS layer. They give an inte-
ger programming formulation and devise a heuristic technique based on that formulation. In [39],
Ricciato et al. consider the problem of off-line configuration of MPLS-over-WDM networks under
time-varying offered traffic. They present a mixed integer programming formulation for the problem
and discuss heuristic approaches. There are also some recent works carried out on two-layered net-
work design with or without dimensioning. In [34], Orlowsky and Wessäly describe a general integer
linear programming model for the design of multi-layer telecommunication network de sign problem
which integrates hardware, capacity, routing and grooming decisions. They give also a sketch of an
algorithmic approach. Orlowski et al. [33] develop three primal heuristics to be called in a Branch-
and-Cut algorithm to solve the problem with two layers. Knippel and Lardeux [27] study heuristic
and exact algorithms based on metric inequalities for a multilayer design problem (see also [28]).

The first major survivability requirement used in telecommunications networks is the so-called
**2-connectivity**. That is there must exist at least two edge-disjoint paths between every pair of nodes
in the network. This implies that the network remains connected in the event of any single edge
failure. The problem of finding a minimum cost 2-edge connected subgraph has been extensively
investigated in the past decade [3, 20, 25, 26, 32, 41].

Most of research on the design of networks concentrates on uncapacitated networks where each
link can support all the traffic at once. However, for many telecommunication networks, capacities
play a fundamental role. Nevertheless several problems with capacity loading have been studied. In
Dahl and Stoer present a cutting plane approach for solving the MULTIcommodity SUrvivable Network design problem (MULTISUN problem). This consists, given point-to-point traffic demands in a network, in finding minimum cost capacities that permit the routing of the given demands. The possible capacity choices on each edge give rise to a discrete cost function. Another problem called the Network Loading Problem (NLP) plays a central role in the design of telecommunication networks. It is a special case of the MULTISUN problem. For the NLP, a single type of capacitated facility is considered, and each link can be assigned one or several facilities in such a way that the network can carry given point-to-point demands at minimum cost. Barahona [4] studies this problem in the both nonbifurcated and bifurcated cases that is when the flow of each commodity is carried by a single path or when it could use several paths. He proposes a separation algorithm for the so-called cut inequalities, which seem to play a central role for solving the problem. In [30] Magnanti et al. study the polyhedral structure of two core subproblems of the NLP. And in [31] they discuss a further problem, called the Two-Facility capacitated network Loading Problem (TFLP), in which considers two types of capacities.

The paper is organised as follows. In the following section we discuss the IP-over-optical networks and the interaction between the different layers. We present a multilayer survivable network design problem with capacity constraint, called the multilayer capacitated survivable IP network design problem. We describe two versions of this problem : with and without multiple edges. We give mixed integer programming formulations for this problem. Section 3 presents the column generation algorithm to solve a linear relaxation of the problem. In Section 4, we study the associated polytopes. We identify a few classes of valid inequalities and describe conditions for these inequalities to be facet defining. In Section 5, we describe the Branch-and-Cut algorithms and Branch-and-Cut-and-Price algorithms for the problem. Our computational results are presented and discussed in Section 6. In Section 7, we give some concluding remarks.

2 The Multilayer Capacitated Survivable IP Network Design Problem

2.1 Multilayer telecommunication networks

Telecommunication networks are now moving toward a model of high-speed routers interconnected by intelligent optical core networks. Moreover, there is a general consensus that the control plan of the optical networks should utilize IP-based protocols for dynamic provisioning and restoration of lightpaths [9, 24, 35, 36, 37].

The optical network consists of multiple switches (also called Optical Cross-Connects (OXC)) interconnected by optical links. The IP and optical networks communicate through logical control interfaces called User-Network-Interfaces (UNI). The optical network essentially provides point-to-point connectivity between routers in the form of fixed bandwidth lightpaths. These lightpaths define the topology of the IP network.

Each router in the IP network is connected to at least one of the optical switches. Moreover to each link between two routers in the IP network corresponds a routing path in the optical one between two switches corresponding to these routers. Figure 1 shows an IP-over-optical network. The IP network has four routers \( R_1, \ldots, R_4 \) and the optical network has seven switches \( S_1, \ldots, S_7 \). Only the optical switches \( S_1, \ldots, S_4 \) communicate with one router throught the UNI.

The introduction of this new infrastructure of telecommunication networks gives rise to survivability issues. For example consider the IP-over-optical network given in Figure 1. Suppose that the link \( R_1 - R_2 \) of the IP network corresponds to the optical path \( S_1 - S_2 \), and the link \( R_1 - R_3 \) corresponds to the path \( S_1 - S_2 - S_6 - S_3 \). Here, the network is not survivable to single link failures. For instance, if the optical link \( S_1 - S_2 \) fails, then the links in the IP network \( R_1 - R_2 \) and
$R_1 - R_3$ are cut, and therefore the router $R_1$ is no more connected to the rest of the routers. As a consequence, survivability strategies have to be considered. If the transport network is fixed, one has to determine the suitable client network topology for the network to be survivable.

In addition to the survivability aspect, we may need to install capacities on the IP network in order to route commodities between some routers. In this paper we shall be concerned by this problem which considers simultaneously both the survivability and the dimensioning of the IP network when the transport network is fixed.

2.2 The problem

The first major survivability requirement used in telecommunication networks is the 2-connectivity. That is there must exist at least two edge-disjoint paths between every pair of nodes in the network. This assumption, that only one edge may fail at a time, is based on the naive idea that the links in the network are independent and no equipment can be commonly used by two distinct links. However, this is not the case, for instance, for the IP-over-optical networks, when the optical layer is taken into account in the management of the IP network.

In fact, any edge of the client network is supported by a path in the optical network (lightpaths). That is the traffic of an edge in the client network is routed in the optical network along the path corresponding to that edge. Therefore an edge of the optical network may appear in several paths supporting distinct edges. In consequence, the failure of an edge in the optical network may affect several optical paths, and hence the edges of the client network corresponding to these paths. As a result, several edges may fail at the same time in the IP layer (such a group of links is usually referred to as a Shared Risk Link Group, or SRLG).

The multilayer survivable IP network design problem (MSIPND problem) introduced by Borne et al. [7] consists in finding the set of links to be installed in the IP network so that if a failure occurs on an optical link, the IP subnetwork obtained by removing the corresponding edges is connected.

In our problem, we can install capacities of 2.5 Gbits or 10 Gbits on any link of the IP network. Usually, the installed capacities have to be symmetric. Hence, we consider that each time a certain capacity is installed from a router $R_1$ to a router $R_2$, one has to install the same capacity from $R_2$ to $R_1$. Figure 2 shows the eligible and the forbidden situations.

Consequently, a more realistic model which has to be investigated, would consist in setting up
capacities with a minimum cost to the client network that allows a multicommodity flow which respects the capacities for any simple edge failure in the transport network.

In this paper we consider this problem. More precisely, we consider the overlay model where the IP and the optical networks are separated. We suppose that the topology and the routing of the optical network are fixed and satisfy some survivability requirements. We also suppose that a set of IP routers (resp. optical switches) is given as well as the possible links between the routers (resp. switches). As the routing of the optical network is known, one can determine for each optical link $e$, the set of edges of the IP network that may be affected if $e$ is cut. If a certain cost is associated with each type of capacity on each edge of the IP network, the Multilayer Capacitated Survivable IP Network Design problem (MCSIPND problem) is to find the minimum cost set of links to be installed in the IP network and facilities to be loaded on these links so that if a failure occurs on an optical link, the IP subnetwork obtained by removing the corresponding edges allows a multicommodity flow which satisfies the capacities.

We consider two variants of the problem: the multiple MCSIPND (denoted by MCSIPND$_m$) which allows multiple links, and the simple MCSIPND (denoted by MCSIPND$_s$) where only one link of one type of capacity can be loaded between two routers. Figure 3 illustrates the two possibilities of installing capacities. The orientation of the links between the routers is omitted because the capacities are symmetric. A capacity installed in the two directions between two routers is represented by a cylinder (a small one for capacity 2.5 Gbits and a big one for 10 Gbits). Figure 3(a) corresponds to the multiple case and figure 3(b) to the simple one.

In what follows we give mixed integer programming formulations for the MCSIPND problem. To this end, we first introduce some definitions and notations.

### 2.3 Definitions and notation

We denote a graph by $G = (V, E)$ where $V$ is the node set and $E$ the edge set of $G$. If $e \in E$ is an edge between two nodes $u$ and $v$, then we also write $e = uv$ to denote $e$. We denote also by $D = (V, A)$ the bidirected graph associated with $G$ such that each edge $e = uv \in E$ is replaced by two arcs $(u, v)$ and $(v, u)$, respectively from $u$ to $v$ and from $v$ to $u$, in the arc set $A$ of $D$. For an
edge subset \( F \subseteq E \) we denote by \( \bar{F} \subseteq A \) the associated arc subset. For \( F \subseteq E \) we let \( G \setminus F \) denote the subgraph of \( G \) obtained by removing the edges of \( F \) and \( D \setminus \bar{F} \) the associated subgraph obtain from \( D \) by removing the arcs of \( \bar{F} \). Throughout the paper we will consider simple graphs.

Let \( G = (V, E) \) be an undirected graph. Given \( W \subseteq V \), we denote by \( \delta_G(W) \) the set of edges of \( G \) having exactly one node in \( W \). The edge set \( \delta_G(W) \) is called a cut. A subset \( F \subseteq E \) of \( G \) is called an edge cutset if \( F \) is a cut. For \( W \subseteq V \), we denote by \( G(W) \) the subgraph of \( G \) induced by \( W \). If \( W \subset V \), \( \overline{W} \) denotes \( V \setminus W \). If \( U \) and \( W \) are two node subsets such that \( U \cap W = \emptyset \), then we denote by \( [U, W] \) the set of edges having one node in \( U \) and the other in \( W \). If \( V_1, \ldots, V_p \) is a partition of \( V \), we let \( \delta_G(V_1, \ldots, V_p) \) denote the set of edges of \( G \) between the elements of the partition.

Given a graph \( G = (V, E) \), a path \( P \) in \( G = (V, E) \) is an alternate sequence of nodes and edges \((v_1, e_1, v_2, e_2, \ldots, v_p, e_p, v_{p+1})\) such that \( e_i = v_iv_{i+1} \) for \( i = 1, \ldots, p \) and \( v_i \neq v_j \) for \( i = 1, \ldots, p + 1, j = 1, \ldots, p + 1 \). Nodes \( v_1, v_{p+1} \) are the extremities of \( P \) and we will say that \( P \) goes from \( v_1 \) to \( v_{p+1} \) or \( P \) is between \( v_1 \) and \( v_{p+1} \).

Given a vector \( x \in \mathbb{R}^E \) and \( F \subseteq E \), we let \( x(F) = \sum_{e \in F} x(e) \).

![Graphs of an IP-over-optical network](image)

**Fig. 4** - Graphs of an IP-over-optical network

Throughout the paper, given an IP-over-optical network, we suppose that to each router of the IP layer corresponds exactly one optical switch. We will represent an IP-over-optical network by two graphs \( G^1 = (V^1, E^1) \) and \( G^2 = (V^2, E^2) \), that represent the IP and optical networks, respectively. The nodes of \( G^1 \) (resp. \( G^2 \)) correspond to the routers of the IP layer (resp. the optical switches), and the edges represent the possible links between the routers (resp. switches). A chaque sommet \( v_i \in V^1 \) est associé un sommet \( w_i \in V^2 \). For an edge \( f \in E^1 \), we denote by \( P_f \) the path in \( G^2 \) corresponding to \( f \). Figure 4 shows graphs \( G^1 \) and \( G^2 \) corresponding to the IP-over-optical network of Figure 1. In \( G^2 \), are indicated two paths \( P_e \) and \( P_f \) which correspond to the edges \( e \) and \( f \) of \( G^1 \).

### 2.4 Formulations

In terms of graphs, the MCSIPND problem, for both multiple and simple variants, can be presented as follows.

For an edge \( e \) of graph \( G^2 = (V^2, E^2) \) corresponding to the optical network, let \( F_e \) be the set of edges of the IP network that may be affected by a failure of \( e \), that is \( F_e = \{ f \in E^1 \mid e \in P_f \} \). We let \( \mathcal{F} = \{ F_e \mid e \in E^2 \} \). Also we denote by \( D^1 \) the directed graph associated with \( G^1 \) and \( \mathcal{D} = \{ F_e \mid F_e \in \mathcal{F} \} \). We denote by \( K \) the set of commodities. For each \( k \in K \), we know the origin \( o_k \), the destination \( d_k \) and the amount \( \omega_k \) of the demand \( k \).

Let \( \mu^1 = 2.5 \) Gbit/s and \( \mu^2 = 10 \) Gbit/s be the possible facilities. For each \( ij \in E^1 \), let \( c_{ij} \) be
the cost of installing a capacity $\mu^l$ on $ij$ for $l = 1, 2$. Then, the MCSIPND$_m$ problem consists in finding a minimum cost subgraph $H$ of $G^1$ such that for every edge $e \in E^2$, the graph obtained from $H$ by removing the edges of $F_e$ has enough capacity to route the commodities of $K$ with respect to the capacity of the remaining edges.

In what follows we give two different formulations for the MCSIPND problem: the node-arc or conventional formulation and the path or column-generation formulation. A survey of linear multicommodity flow models and solution procedures is presented in [2].

### 2.4.1 Node-arc formulation

In order to give a node-arc formulation for the MCSIPND problem, let us denote by $f_{uv}^{k,e}$ for an arc $(u, v) \in A^1$, an edge $e \in E^2$ and a commodity $k \in K$, the flow of $k$ on $(u, v)$ from $u$ to $v$ in case of failure of $e$ (i.e. when the arcs of $F_e$ are removed in $D^1$). For an edge $uv \in E^1$ let $x_{uv}^l$ be the number of facilities $\mu^l$ installed on $uv$, for $l = 1, 2$. Set

$$b_k^v = \begin{cases} -\omega_k & \text{if } v = o_k, \\ \omega_k & \text{if } v \neq o_k, d_k, \text{ for all } v \in V^1, \text{ for all } k \in K. \\ 0 & \text{if } v = d_k, \end{cases}$$

Hence the multiple MCSIPND problem is equivalent to the following integer programming problem.

Minimize \[ \sum_{l=1,2} \sum_{uv \in E^1} c_{uv}^l x_{uv}^l \]

\[ \sum_{u:(v,u) \in A^1 \setminus F_e} f_{uv}^{k,e} - \sum_{u:(u,v) \in A^1 \setminus F_e} f_{vu}^{k,e} = b_k^v \quad \text{for all } v \in V^1, \text{ for all } k \in K, \text{ for all } e \in E^2, \quad (1) \]

\[ \sum_{k \in K} f_{uv}^{k,e} \leq \sum_{l=1,2} \mu^l x_{uv}^l \quad \text{for all } uv \in E^1, \text{ for all } e \in E^2, \quad (2) \]

\[ \sum_{k \in K} f_{vu}^{k,e} \leq \sum_{l=1,2} \mu^l x_{uv}^l \quad \text{for all } uv \in E^1, \text{ for all } e \in E^2, \quad (3) \]

\[ x_{uv}^l \geq 0 \text{ and integer} \quad \text{for all } uv \in E^1, l = 1, 2, \quad (4) \]

\[ f_{uv}^{k,e}, f_{vu}^{k,e} \geq 0 \quad \text{for all } uv \in E^1, \text{ for all } k \in K, \text{ for all } e \in E^2. \quad (5) \]

Inequalities (1) are called flow conservation constraints. Inequalities (2) and (3) express the fact that the sum of the flows of all commodities $k \in K$ on an edge has to be less than or equal to the capacity of this edge. They will be called capacity constraints. Inequalities (4) and (5) are called trivial inequalities.

By adding the following inequalities

\[ x_{uv}^1 + x_{uv}^2 \leq 1 \quad \text{for all } uv \in E^1, \quad (6) \]

and by replacing inequalities (4) by

\[ x_{uv}^l \in \{0,1\} \quad \text{for all } uv \in E^1, l = 1, 2, \quad (7) \]

we obtain a valid formulation for the simple MCSIPND problem.

Inequalities (6) express the fact that only one link can be used between two given nodes. Then we have only one type of capacity on an edge. Constraints (7) are the integrality constraints that express the fact that $x_{uv}^l = 1$ if capacity $\mu^l$, $l = 1, 2$, is installed on $uv$ and 0 otherwise.
Note that the following inequalities
\[ f_{uv}^{k,e} + f_{vu}^{k,e} \leq \omega_k \quad \text{for all } u, v \in V^1, \text{ for all } k \in K, \text{ for all } e \in E^2, \quad (8) \]
are valid for the MCSIPND problem. They are called **bound inequalities**.

The Arc Residual Capacity inequalities have been introduced in [30] for the Network Loading Problem and used for the Two-Facility capacitated network loading Problem [31]. In the following we extend these inequalities for our problem.

**Theorem 2.1** Let \( L \subseteq K \). Set
\[
Q_L = \sum_{k \in L} \omega_k \quad \text{and} \quad s_L = \begin{cases} 
2 & \text{if } Q_L \mod 4 = 0, \\
|Q_L| & \text{otherwise}.
\end{cases}
\]

Then we have \( Q_L = 4(\sigma_L - 1) + s_L \).

Let \( u, v \in V^1 \) and \( e \in E^2 \), then inequality
\[
\frac{1}{2.5} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) - 2x_{uv}^1 - s_L \times 2x_{uv}^2 \leq (\sigma_L - 1)(4 - s_L) \quad (9)
\]
is valid for the polytope associated with the MCSIPND problem.

**Proof.** Inequality (9) can be written as
\[
\frac{1}{2.5} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq Q_L - s_L(\sigma_L - 2x_{uv}^2) + 2x_{uv}^1
\]
for \( L \subseteq K \), \( u, v \in V^1 \) and \( e \in E^2 \) because \( (\sigma_L - 1)(4 - s_L) = Q_L - \sigma_L s_L \).

- If \( 2x_{uv}^2 \geq \sigma_L \) then \( Q_L - s_L(\sigma_L - 2x_{uv}^2) + 2x_{uv}^1 \geq Q_L \).

We know that inequalities (8) are valid for the problem. By summing these inequalities, we obtain
\[
\sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq \sum_{k \in L} \omega_k.
\]

Then
\[
\frac{1}{2.5} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq Q_L,
\]
which implies that inequality (9) is valid.

- If \( 2x_{uv}^2 \leq \sigma_L - 1 \), then
\[
Q_L - s_L(\sigma_L - 2x_{uv}^2) + 2x_{uv}^1
\]
\[
= 4(\sigma_L - 1) + s_L - s_L(\sigma_L - 2x_{uv}^2) + 2x_{uv}^1
\]
\[
= 4(\sigma_L - 1) + s_L(2x_{uv}^2 - (\sigma_L - 1)) + 2x_{uv}^1
\]

Let \( s_L = 4 - t \) with \( 0 \leq t < 4 \), we obtain
\[
Q_L - s_L(\sigma_L - 2x_{uv}^2) + 2x_{uv}^1
\]
\[
= 4(\sigma_L - 1) + (4 - t)(2x_{uv}^2 - (\sigma_L - 1)) + 2x_{uv}^1
\]
\[
= 8x_{uv}^2 + 2x_{uv}^1 + t(-2x_{uv}^2 + (\sigma_L - 1)).
\]

Then
\[
Q_L - s_L(\sigma_L - 2x_{uv}^2) + 2x_{uv}^1 \geq 8x_{uv}^2 + 2x_{uv}^1, \quad (10)
\]
because \( t \geq 0 \) and \( 2x_{uv}^2 \leq \sigma_L - 1 \). As
\[
\sum_{k \in K} f_{uv}^{k,e} \leq 2.5x_{uv}^1 + 10x_{uv}^2 \quad \text{and} \quad \sum_{k \in K} f_{vu}^{k,e} \leq 2.5x_{uv}^1 + 10x_{uv}^2,
\]

8
then \( \frac{1}{25} \sum_{k \in K} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq 2x_{uv}^1 + 8x_{uv}^2 \), and hence

\[
\frac{1}{25} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq 2x_{uv}^1 + 8x_{uv}^2.
\]

By (10) and (11), it follows that inequality (9) is valid.

\[
\square
\]

### 2.4.2 Path formulation

As in the node-arc formulation, for an edge \( uv \in E^1 \) we denote by \( x_{uv}^l \) the number of facilities \( \mu^l \) installed on \( uv \) for \( l = 1, 2 \). For an edge \( e \in E^2 \) and a commodity \( k \) we denote by \( P^e_k \) the set of paths from \( o_k \) to \( d_k \) in the graph \( D^1 \setminus \hat{F}_e \) (i.e., when the edge \( e \in E^2 \) fails). For a path \( P \) of \( P^e_k \), let \( y_k^e(P) \) be the amount of flow of commodity \( k \) on \( P \) in case of failure of \( e \). Set

\[
\tau_{uv}(P) = \begin{cases} 
1 & \text{if the arc } (u, v) \text{ belongs to } P, \\
0 & \text{otherwise},
\end{cases}
\]

for all \( u, v \in V \). Hence we notice that for all \( u, v \in V, k \in K, \) and \( e \in E^2 \),

\[
f_{uv}^{k,e} = \sum_{P \in P^e_k} \tau_{uv}(P) y_k^e(P) = \sum_{P \in P^e_k} y_k^e(P).
\]

By substituting the path variables in the node-arc formulation, we obtain the following mixed integer programming formulation which is valid for the multiple MCSIPND problem.

\[
\text{Minimiser } \sum_{l=1,2} \sum_{uv \in E^1} c_{uv}^l x_{uv}^l
\]

\[
\sum_{P \in P^e_k} y_k^e(P) = \omega_k \quad \text{for all } k \in K, \text{ for all } e \in E^2,
\]

\[
\sum_{k \in K} \sum_{P \in P^e_k \setminus \{(u,v)\} \in P} y_k^e(P) \leq \sum_{l=1,2} \mu^l x_{uv}^l \quad \text{for all } uv \in E^1, \text{ for all } e \in E^2,
\]

\[
\sum_{k \in K} \sum_{P \in P^e_k \setminus \{(v,u)\} \in P} y_k^e(P) \leq \sum_{l=1,2} \mu^l x_{uv}^l \quad \text{for all } uv \in E^1, \text{ for all } e \in E^2,
\]

\[
y_k^e(P) \geq 0 \quad \text{for all } e \in E^2, \text{ for all } k \in K, \text{ for all } P \in P^e_k,
\]

\[
x_{uv}^l \geq 0 \text{ and integer} \quad \text{for all } uv \in E^1, l = 1, 2.
\]

This formulation has a collection of \( |K| \) demand constraints (12) that represent the flow of each path \( P \) in \( P^e_k \), \( k \in K \) for each failure \( e \in E^2 \) and \( \sum_{P \in P^e_k} y_k^e(P) \) represents the amount of flow of commodity \( k \) passing through the set of paths from \( o_k \) to \( d_k \). This flow has to be equal to the amount \( \omega_k \) between \( o_k \) and \( d_k \). Inequalities (13) and (14) are called capacity constraints. The flow through the edge \( uv \) has to be less than the capacity of this edge from \( u \) to \( v \) (constraints (13)) and from \( v \) to \( u \) (constraints (14)). Inequalities (15) and (16) are the trivial constraints.

By adding inequalities (6) and replacing inequalities (16) by inequalities (7), we obtain a valid formulation for the simple MCSIPND problem.

The linear relaxation of the node-arc MCSIPND formulation contains a large number of constraints and a large number of variables. The linear relaxation of the path formulation, however, contains a moderate number of constraints (for each failure, one for each commodity and one for each arc) and a huge number of variables (one for each path for each commodity for each failure). These linear relaxations may require excessive memory and times to solve. An appropriate method to solve this second type of formulation would be the column generation approach. In the next section, we discuss this approach.
3 Column generation

Column generation will be used to solve the linear relaxation of the path based formulation of the MCSIPND problem (called the master problem). This approach has been extensively used for modeling and solving large versions of the linear multicommodity flow problem [2, 5, 29]. The general idea of column generation is to solve a restricted linear program with a small number of columns (variables) in order to determine an optimal solution for the master problem. In fact a limited number of variables may induce an optimal basis solution for the master problem. So the column generation algorithm solves the linear relaxation of the master problem by solving the linear relaxations of several restricted master problems. After determining the solution of the linear relaxation of a restricted master problem, we use the pricing problem which consists in finding whether there are any columns not yet in the restricted master problem with negative reduced cost. If none can be found, the current solution is optimal for the linear relaxation of the master problem. However, if one or more such columns do exist, then they are added to the restricted master problem and the process is repeated. This approach could be combined with row generation to obtain a very strong method to solve the linear relaxations (see [6]).

3.1 Initial Solution

To start the column generation scheme, an initial restricted master problem has to be provided. This initial problem must have a feasible solution to ensure that correct information is passed to the pricing problem.

For the version of the MCSIPND problem with multiple edges (MCSIPND\textsubscript{m} problem), finding an initial feasible solution is very easy. Indeed we look for shortest paths between the origin-destinations of all commodities. These paths are then used to carry the flow for each commodity. As we can install as much capacity as we want, this multicommodity flow is feasible.

For the MCSIPND\textsubscript{s} problem, we consider the following linear program obtained from the path based formulation by setting $x_{uv}^1 = 0$ and $x_{uv}^2 = 1$ for all edge $uv \in E^1$, that is to say by fixing a capacity of 10 Gbits for each edge. We also consider a new variable $\varepsilon$.

Minimiser $\varepsilon$

\begin{align}
\sum_{P \in \mathcal{P}_k^e} y_k^e(P) &= \omega_k && \text{for all } k \in K, \text{ for all } e \in E^2, \tag{17} \\
\sum_{k \in K} \sum_{P \in \mathcal{P}_k^e | (u,v) \in P} y_k^e(P) &\leq \mu^2 + \varepsilon && \text{for all } (u,v) \in A^1, \text{ for all } e \in E^2, \tag{18} \\
y_k^e(P) &\geq 0 && \text{for all } e \in E^2, \text{ for all } k \in K, \text{ for all } P \in \mathcal{P}_k^e, \tag{19} \\
\varepsilon &\geq 0. \tag{20}
\end{align}

At the optimum, variable $\varepsilon$ corresponds to the minimum amount of capacity we must add to each edge in order to allow a multicommodity flow. We solve this linear program using a column generation algorithm similar to the algorithm used to solve the MCSIPND\textsubscript{m} problem. If the optimal solution for this linear program imposes that $\varepsilon > 0$, we conclude that the MCSIPND\textsubscript{s} problem has no solution. On the other hand, if $\varepsilon = 0$, the set of variables used in the column generation permits to have an initial feasible solution for the restricted master problem.

When defining the initial restricted master problem, it is necessary to ensure the existence of a feasible solution. Finding a "good" initial restricted master problem could be important. Indeed this would permit to determine the initial dual variables which will be passed to the pricing problem.
3.2 Pricing problem

For any restricted master problem, let $\gamma^e_k$, $\vartheta^e_{(u,v)}$ and $\vartheta^e_{(v,u)}$ be the dual variables associated with constraints (12), (13) and (14), respectively. The reduced cost associated with the variable of a path $P \in \mathcal{P}_k$ is $R^e_P = \sum_{(u,v) \in P} \vartheta^e_{(u,v)} - \gamma^e_k$. The pricing problem can then be reduced to the search of several shortest path problem with non-negative costs. Indeed the pricing problem consist in finding for each commodity $k \in K$ and each edge $e \in E^2$, a path $P$ in $\mathcal{P}_k$ such that $R^e_P = \min_{P' \in \mathcal{P}_k} R^{k,e}_{P'}$ and $R^e_P < 0$. Therefore, we can identify columns which have to be added to the restricted master problem by solving one shortest path problem for each commodity $k \in K$ and each edge $e \in E^2$ in the graph with arc costs equal to $\vartheta^e_{(u,v)}$ for each $(u,v) \in A^1 \setminus \vec{F}_e$. If one or more paths have non-positive reduced cost, then they are added in the restricted master problem. Otherwise, the master problem has been solved to optimality.

Combining column and row generation can yield a very strong linear relaxation. In the next section, we describe some valid inequalities. These will be used as cutting planes in our Branch-and-Cut-and-Price algorithm for the two variants of the problem. We introduce some inequalities which are valid for the problem with or without multiple edges.

4 Valid inequalities and facets

Throughout the following sections we consider a graph $G = (V,E)$ and the associated digraph $D = (\bar{V},\bar{A})$ obtained from $G$ by substituting each edge of $E$ by two arcs. We consider also a family $\mathcal{F} = \{F_1, \ldots, F_t\} \subseteq 2^E$, $t \geq 2$ of edge subsets of $E$ and the family $\mathcal{F} = \{\bar{F}_1, \ldots, \bar{F}_t\} \subseteq 2^A$ of arc subsets associated with $\mathcal{F}$. Let $K$ be a set of demands. For an arc $(u,v) \in A$, a commodity $k \in K$ and $i \in \{1, \ldots, t\}$, let us denote by $f^{k,i}_{uv}$ the flow of $k$ on $(u,v)$ from $u$ to $v$ when the arcs of $\bar{F}_i$ are removed in $D^i$. For $i \in \{1, \ldots, t\}$, we will denote by $G_i = (\bar{V}, E_i)$ (resp. $D_i = (\bar{V}, A_i)$) the subgraph of $G$ (resp. $D$) obtained by removing the edges of $\bar{F}_i$ (resp. $\vec{F}_i$). Hence $E_i = E \setminus F_i$ (resp. $A_i = A \setminus \vec{F}_i$). In the following we consider $\mu^1 = 2.5$ and $\mu^2 = 10$.

Now, consider the following inequalities:

\[ \sum_{u:(u,v) \in A \setminus \vec{F}_i} f^{k,i}_{uv} - \sum_{u:(v,u) \in A \setminus \vec{F}_i} f^{k,i}_{uv} = b^v_k \quad \text{for all } v \in V, \text{ for all } k \in K, \ i = 1, \ldots, t, \quad (21) \]

\[ \sum_{k \in K} f^{k,i}_{uv} \leq 2.5x^1_{uv} + 10x^2_{uv} \quad \text{for all } uv \in E, \ i = 1, \ldots, t, \quad (22) \]

\[ \sum_{k \in K} f^{k,i}_{uv} \leq 2.5x^1_{uv} + 10x^2_{uv} \quad \text{for all } uv \in E, \ i = 1, \ldots, t, \quad (23) \]

\[ x^l_{uv} \geq 0 \quad \text{for all } uv \in E, \ l = 1, 2, \quad (24) \]

\[ f^{k,i}_{uv}, f^{k,i}_{vu} \geq 0 \quad \text{for all } uv \in E, \text{ for all } k \in K, i = 1, \ldots, t, \quad (25) \]

\[ x^1_{uv} + x^2_{uv} \leq 1 \quad \text{for all } uv \in E. \quad (26) \]

Let $\text{MCSIPND}^m(G, \mathcal{F}, K)$ and $\text{MCSIPND}^s(G, \mathcal{F}, K)$ be the polytopes associated with the MCSIPND_m problem and the MCSIPND_s problem (when $G = G^1$ and $\mathcal{F} = \{F_e, e \in E^2\}$), i.e.

\[
\text{MCSIPND}^m(G, \mathcal{F}, K) = \{(x, f) \in D^m \mid x \text{ and } f \text{ satisfy } (21)-(25)\},
\]

\[
\text{MCSIPND}^s(G, \mathcal{F}, K) = \{(x, f) \in D^s \mid x \text{ and } f \text{ satisfy } (21)-(26)\}
\]

with

\[ D^m = \{x \in \mathbb{R}^{2|E|}, f \in \mathbb{R}^{2|E|} \times |K| \times |\mathcal{F}|\} \]

and

\[ D^s = \{x \in \{0,1\}^{2|E|}, f \in \mathbb{R}^{2|E|} \times |K| \times |\mathcal{F}|\}. \]
Let $P_k$ be the set of paths between $o_k$ and $d_k$ in the graph $D \setminus F_i$ for $i \in \{1, \ldots, t\}$. Let $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$ denote the convex hull of the integer solutions of the system

$$\sum_{P \in P_k} y_k(P) = \omega_k$$
for all $k \in K$, $i = 1, \ldots, t$, \hspace{1cm} (27)

$$\sum_{k \in K} \sum_{P \in P_k} y_k(P) \leq 2.5x_{uv}^i + 10x_{uv}^j$$
for all $uv \in E$, $i = 1, \ldots, t$, \hspace{1cm} (28)

$$\sum_{k \in K} \sum_{P \in P_k} y_k(P) \leq 2.5x_{uv}^i + 10x_{uv}^j$$
for all $uv \in E$, $i = 1, \ldots, t$, \hspace{1cm} (29)

$$x_{uv}^i \geq 0$$
for all $uv \in E$, $l = 1, 2$, \hspace{1cm} (30)

$$y_k(P) \geq 0$$
for all $k \in K$, $i = 1, \ldots, t$, for all $P \in P_k$. \hspace{1cm} (31)

By adding constraints (26) to $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$, we obtained $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$.

We can remark that if $G = G^1$ and $\mathcal{F} = \{F_e, e \in E^2\}$, $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$ (resp. $\text{MCSIPND}_s^\rho(G, \mathcal{F}, K)$) is nothing but the polytope associated with the $\text{MCSIPND}_m$ (resp. $\text{MCSIPND}_s$) problem.

If no confusion may arise, we will sometimes write $\text{MCSIPND}_m(G, \mathcal{F}, K)$ for the two polytopes $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s^\rho(G, \mathcal{F}, K)$. Similarly the polytope $\text{MCSIPND}_s(G, \mathcal{F}, K)$ will correspond indifferently to the polytopes $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s^\rho(G, \mathcal{F}, K)$.

The following theorem gives the dimension of the polytope $\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)$.

**Theorem 4.1** $\text{dim}(\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)) = 2|E| + 2|\mathcal{F}| \times |K| - (|N| - 1)|\mathcal{F}| \times |K|.$

**Proof.** The node-arc formulation of the $\text{MCSIPND}_m$ problem contains $2|E| + 2|\mathcal{F}| \times |E| \times |K|$ variables and $(|N| - 1)|\mathcal{F}| \times |K|$ nonredondant equality constraints. Hence $\text{dim}(\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)) \leq 2|E| + 2|\mathcal{F}| \times |E| \times |K| - (|N| - 1)|\mathcal{F}| \times |K|.$

The proof of $\text{dim}(\text{MCSIPND}_m^\rho(G, \mathcal{F}, K)) \geq 2|E| + 2|\mathcal{F}| \times |E| \times |K| - (|N| - 1)|\mathcal{F}| \times |K|$ uses arguments similar to those used in theorem 4.9 and we, therefore, omit it. \hspace{1cm} $\square$

In the following, we introduce several classes of valid inequalities. We also give necessary and sufficient conditions for one of these inequalities to be facet defining. We assume that the reader is familiar with polyhedral combinatorics, for more details see [40].

As in [7], a subgraph $H = (W, F)$ of $G = (V, E)$ is said to be $\mathcal{F}$-connected with respect to $\mathcal{F} = \{F_1, \ldots, F_t\}$ if for all $i \in \{1, \ldots, t\}$, the graph $H \setminus F_i$ is connected.

### 4.1 Cut inequalities

For $W \subseteq V$, we denote by $\gamma^+(W) \subseteq K$ (resp. $\gamma^-(W) \subseteq K$) the set of demands which have their origin (resp. destination) in $W$ and their destination (resp. origin) in $V \setminus W$. We denote also by $\gamma(W)$ the set $\gamma^+(W) \cup \gamma^-(W)$.

**Theorem 4.2** Let $F_i \in \mathcal{F}$ be an edge subset of $E$ and $W \subseteq V$, $\emptyset \neq W \neq V$ such that $\gamma(W) \neq \emptyset$. Then the inequality

$$x^1(\delta_{C_i}(W)) + x^2(\delta_{C_i}(W)) \geq 1$$

is valid for $\text{MCSIPND}_m(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s(G, \mathcal{F}, K)$.

**Proof.** Trivial. \hspace{1cm} $\square$
Inequalities of type (32) will be called design cut inequalities. Theses inequalities express the fact that the graph $G$ keeps connectivity between the origin and the destination of each demand after removing the edges of $F_i$, for all $F_i \in \mathcal{F}$.

Given a set of nodes $W \subseteq V$, $\emptyset \neq W \neq V$, let

$$D_W = \left\lceil \max \left\{ \frac{\sum_{k \in \gamma^+(W)} \omega_k}{2.5}, \frac{\sum_{k \in \gamma^-(W)} \omega_k}{2.5} \right\} \right\rceil.$$ 

Let $F_i \in \mathcal{F}$ be an edge subset of $E$ and $W \subseteq V$, $\emptyset \neq W \neq V$. Consider the inequality

$$x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq D_W. \quad (33)$$

We have the following.

**Theorem 4.3** Inequality (33) is valid for both $\text{MCSIPND}_m(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s(G, \mathcal{F}, K)$.

**Proof.** The aggregate capacity across $\delta_{G_i}(W)$ must be no less than the demand across the cut from $W$ to $V \setminus W$ and from $V \setminus W$ to $W$. Thus one should have

$$2.5x^1(\delta_{G_i}(W)) + 10x^2(\delta_{G_i}(W)) \geq \max \left\{ \sum_{k \in \gamma^+(W)} \omega_k, \sum_{k \in \gamma^-(W)} \omega_k \right\}.$$ 

Dividing by 2.5 and rounding up the right hand side yields (33). \hfill \Box

Before introducing the next class of inequalities, we give a lemma.

**Lemma 4.4** Let $F_i \in \mathcal{F}$ be an edge subset of $E$ and $W \subseteq V$, $\emptyset \neq W \neq V$. Then inequality

$$x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil \quad (34)$$

is valid for both $\text{MCSIPND}_m(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s(G, \mathcal{F}, K)$.

**Proof.** The following inequalities are valid for both $\text{MCSIPND}_m(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s(G, \mathcal{F}, K)$,

$$x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq D_W,$$

$$x^1_{uv} \geq 0 \quad \text{for all } uv \in \delta_{G_i}(W).$$

By summing these inequalities, we obtain

$$2x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq D_W.$$ 

Dividing by 2 and rounding up the right hand side yields inequality (34). \hfill \Box

**Theorem 4.5** Let $F_i \in \mathcal{F}$ be an edge subset of $E$ and $W \subseteq V$, $\emptyset \neq W \neq V$. Then inequality

$$x^1(\delta_{G_i}(W)) + x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{4} \right\rceil \quad (35)$$

is valid for both $\text{MCSIPND}_m(G, \mathcal{F}, K)$ and $\text{MCSIPND}_s(G, \mathcal{F}, K)$.  

13
Proof. By adding the inequalities
\[ x_{uv}^1 \geq 0 \quad \text{for all } uv \in \delta_G(W). \]
to inequality (34), we get
\[ 2x^1(\delta_G(W)) + 2x^2(\delta_G(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil. \]
Dividing by 2 and rounding up the right hand side yields
\[ x^1(\delta_G(W)) + x^2(\delta_G(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil. \]
As \( \left\lceil \frac{D_W}{2} \right\rceil = \left\lceil \frac{D_W}{m} \right\rceil \), we get inequality (35). □

Theorem 4.6 Let \( F_i \in \mathcal{F} \) be an edge subset of \( E \) and \( W \subseteq V \), \( \emptyset \neq W \neq V \). Then inequality
\[ x^1(\delta_G(W)) + 3x^2(\delta_G(W)) \geq \left\lceil \frac{3D_W}{4} \right\rceil \] (36)
is valid for both \( \text{MCSIPND}_m(G, \mathcal{F}, K) \) and \( \text{MCSIPND}_s(G, \mathcal{F}, K) \).

Proof. The following inequalities are valid for both \( \text{MCSIPND}_m(G, \mathcal{F}, K) \) and \( \text{MCSIPND}_s(G, \mathcal{F}, K) \),
\[ x^1(\delta_G(W)) + 4x^2(\delta_G(W)) \geq D_W \]
\[ x^1(\delta_G(W)) + 2x^2(\delta_G(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil \]
By summing these inequalities we obtain
\[ 2x^1(\delta_G(W)) + 6x^2(\delta_G(W)) \geq D_W + \left\lceil \frac{D_W}{2} \right\rceil. \]
Dividing by 2 and rounding up the right hand side yields
\[ x^1(\delta_G(W)) + 3x^2(\delta_G(W)) \geq \left\lceil \frac{3D_W}{4} \right\rceil. \]
As \( \left\lceil \frac{3D_W}{2} \right\rceil = \left\lceil \frac{3D_W}{4} \right\rceil \), we get inequality (36). □

Theorem 4.7 Let \( F_i \in \mathcal{F} \) be an edge subset of \( E \) and \( W \subseteq V \), \( \emptyset \neq W \neq V \). Then inequality
\[ x^1(\delta_G(W)) + 2x^2(\delta_G(W)) \geq \begin{cases} \left\lfloor \frac{D_W}{2} \right\rfloor + 1 & \text{if } D_W \mod 4 = 2, \\ \left\lceil \frac{D_W}{2} \right\rceil & \text{otherwise}, \end{cases} \] (37)
is valid for both \( \text{MCSIPND}_m(G, \mathcal{F}, K) \) and \( \text{MCSIPND}_s(G, \mathcal{F}, K) \).
**Proof.** The following inequalities are valid for both \( \text{MCSIPND}_m(G, \mathcal{F}, K) \) and \( \text{MCSIPND}_s(G, \mathcal{F}, K) \),

\[
x^1(\delta_{G_i}(W)) + x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{4} \right\rceil,
\]
\[
x^1(\delta_{G_i}(W)) + 3x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{3D_W}{4} \right\rceil.
\]

By summing these inequalities we obtain

\[
2x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{4} \right\rceil + \left\lceil \frac{3D_W}{4} \right\rceil.
\]

By dividing by 2 and rounding up the right hand side, we get the inequality

\[
x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{4} \right\rceil + \left\lceil \frac{3D_W}{4} \right\rceil.
\]

As \( \left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{2} \right\rceil = \begin{cases} 
\left\lceil \frac{D_W}{2} \right\rceil + 1 & \text{if } D_W \text{ mod } 4 = 2, \\
\left\lceil \frac{D_W}{2} \right\rceil & \text{otherwise ,}
\end{cases} \)

the theorem follows. \( \square \)

Inequalities (33), (35), (36), (37) will be called *capacity demand cut inequalities.*

One may generate further cut based valid inequalities by combining inequalities of type (33), (35), (36), (37) and trivial inequalities. However all inequalities obtained this way are redundant with respect to the capacity demand cut inequalities (see [8]).

In the following, we give necessary conditions and sufficient conditions for inequality (33) to be facet defining for \( \text{MCSIPND}_m(G, \mathcal{F}, K) \).

**Theorem 4.8** Inequality (33) defines a facet of \( \text{MCSIPND}_m(G, \mathcal{F}, K) \) only if

1. \( G_i(W) \) and \( G_i(\overline{W}) \) are connected,
2. there is no \( j \in \{1, \ldots, t\} \setminus \{i\} \) such that \( F_i \cap \delta_G(W) \subset F_j \cap \delta_G(W) \),
3. \( G(W) \) and \( G(\overline{W}) \) are \( \mathcal{F} \)-connected, if \( \delta_G(W) \cap F_i = \emptyset \),
4. \( D_W > \max \left\{ \frac{\sum_{k \in \gamma^+(W) \omega_k}{2.5}, \sum_{k \in \gamma^-(W) \omega_k}{2.5}}{2.5} \right\} \),
5. \( D_W \geq 4 \).

**Proof.** 1. Suppose w.l.o.g., that \( G_i(W) \) is not connected. Hence there is a partition \( W_1, W_2 \) of \( W \) such that \( \delta_{G_i}(W_1, W_2) = \emptyset \) (see Fig. 5). Thus \( \delta_{G_i}(W_1) = \delta_{G_i}(W_1, \overline{W}) \) and \( \delta_{G_i}(W_2) = \delta_{G_i}(W_2, \overline{W}) \).

![Diagram](image.png)
This implies that
\[
x^1(\delta G_i(W)) + 4x^2(\delta G_i(W))
= x^1(\delta G_i(W_1, \overline{W})) + 4x^2(\delta G_i(W_1, \overline{W})) + x^1(\delta G_i(W_2, \overline{W})) + 4x^2(\delta G_i(W_2, \overline{W}))
\]
\[
= x^1(\delta G_i(W_1)) + 4x^2(\delta G_i(W_1)) + x^1(\delta G_i(W_2)) + 4x^2(\delta G_i(W_2))
\geq D_{W_1} + D_{W_2}.
\]

The last inequality is obtain from the inequalities of type (33) corresponding to \(W_1\) and \(W_2\). As \(D_{W_1} + D_{W_2} \geq D_W\), inequality (33) is then redundant with respect to these inequalities and hence (33) can not be facet defining.

2. Assume the contrary. Let \(j \in \{1, \ldots, t\} \setminus \{i\}\) such that \(F_i \cap \delta G(W) \subset F_j \cap \delta G(W)\). Then (33) can be obtained as the sum of the following valid constraints.

\[
x^1(\delta G_j(W)) + 4x^2(\delta G_j(W)) \geq D_W,
\]
\[
x^1_{uv} \geq 0 \quad \text{for all } uv \in (F_j \setminus F_i) \cap \delta G(W),
\]
\[
4x^2_{uv} \geq 0 \quad \text{for all } uv \in (F_j \setminus F_i) \cap \delta G(W).
\]

Hence it is not facet defining.

3. Suppose \(\delta G(W) \cap F_i = \emptyset\) and w.l.o.g., that \(G(W)\) is not \(\mathcal{F}\)-connected. Then there is \(j \in \{1, \ldots, t\}\) such that \(G_j(W)\) is not connected. If \(j = i\), then by condition 1), (33) can not define a facet. So suppose \(j \neq i\). If \(\delta G(W) \cap F_j \neq \emptyset\), then by condition 2), (33) can not also define a facet. Then, suppose that \(\delta G(W) \cap F_j = \emptyset\). Hence, \(\delta G(W) = \delta G_i(W) = \delta G_j(W)\). As \(G_j(W)\) is not connected, there is a partition \(W_1, W_2\) of \(W\) such that \(\delta G_i(W_1, W_2) = \emptyset\). Hence \(\delta G_j(W_1) = \delta G_i(W_1, \overline{W})\) and \(\delta G_j(W_2) = \delta G_j(W_2, \overline{W})\). This implies that

\[
x^1(\delta G_i(W)) + 4x^2(\delta G_i(W))
= x^1(\delta G_j(W)) + 4x^2(\delta G_j(W))
= x^1(\delta G_j(W_1, \overline{W})) + 4x^2(\delta G_j(W_1, \overline{W})) + x^1(\delta G_j(W_2, \overline{W})) + 4x^2(\delta G_j(W_2, \overline{W}))
= x^1(\delta G_j(W_1)) + 4x^2(\delta G_j(W_1)) + x^1(\delta G_j(W_2)) + 4x^2(\delta G_j(W_2))
\geq D_{W_1} + D_{W_2}.
\]

Since \(D_{W_1} + D_{W_2} \geq D_W\), as before, inequality (33) cannot then define a facet.

4. If \(D_W = \max \left\{ \frac{\sum_{k \in -\{w\}^\omega} \omega k}{2.5}, \frac{\sum_{k \in -\{w\}^\omega} \omega k}{2.5} \right\}\), then by Proposition ?? constraint (33) is redundant with respect constraints (21)-(23), and do not then facet defining.

5. If \(D_W < 4\), then for every solution of \(\text{MCSIPND}^n_{w,a}\) that satisfies (33) with equality we have \(x^2(e) = 0\) for all \(e \in \delta G_i(W)\). If \(|\delta G_i(W)| \geq 2\), then there are two edges \(e_1, e_2 \in \delta G_i(W)\) such that \(x^2(e_1) = x^2(e_2) = 0\) in every solution satisfying (33) with equality. But in this case, we can not construct \(\text{dim}(\text{MCSIPND}^n_{w,a}(G, \mathcal{F}, K))\) solutions satisfying (33) with equality, and affinely independant. This implies that (33) can not be facet defining.

So suppose that \(\delta G_i(W) = \{e = uv\}\) with \(u \in W\) and \(v \in \overline{W}\). Then in every solution satisfying (33) with equality, one should have

\[
x^1(e) = D_W,
\]
\[
x^2(e) = 0.
\]

Also, in every solution satisfying (33) with equality, one should have

\[
f^{k,i}_{uv} = \begin{cases} \omega_k + \varepsilon_k \text{ if } k \in \gamma^+(W) \\ \varepsilon_k \text{ if } k \in \gamma^-(W) \end{cases}
\]

16
Proof. \( m \) is positive and sufficiently small. In fact, we can see that the variables \( f \) are not involved in constraint (33). Furthermore, as by condition 4),

\[ D_W > \max \left\{ \frac{\sum_{k \in \gamma^+ (W)} \omega_k}{2}, \frac{\sum_{k \in \gamma^- (W)} \omega_k}{2} \right\} \]

the available capacity is yet more than the demand across the cut. This gives the possibility to route a lightly upper flow. This explains the addition of \( \varepsilon^k \) from \( u \) to \( v \) and from \( v \) to \( u \).

Set the matrix whose columns correspond to solutions satisfying (33) with equality and the lines are associated to \( x^1(e), x^2(e) \), the sum of variables \( f_{uv} \) for \( k \in \gamma(W) \) and the sum of variables \( f_{vu} \) for \( k \in \gamma(W) \). We denote by \( p \) the number of solutions satisfying (33) with equality. This matrix has the following form:

\[
\begin{pmatrix}
    x^1(e) \\
    x^2(e) \\
    \sum_{k \in \gamma(W)} f_{uv}^{k,i} \\
    \sum_{k \in \gamma(W)} f_{vu}^{k,i} \\
\end{pmatrix}
\begin{pmatrix}
    D_W \\
    0 \\
    \sum_{k \in \gamma^+(W)} \omega_k + \sum_{k \in \gamma^-(W)} \varepsilon^k \\
    \sum_{k \in \gamma^+(W)} \omega_k + \sum_{k \in \gamma^-(W)} \varepsilon^k \\
\end{pmatrix}
\begin{pmatrix}
    D_W \\
    0 \\
    \sum_{k \in \gamma^+(W)} \omega_k + \sum_{k \in \gamma^-(W)} \varepsilon^k \\
    \sum_{k \in \gamma^+(W)} \omega_k + \sum_{k \in \gamma^-(W)} \varepsilon^k \\
\end{pmatrix}
\]

By substracting the fourth line from the third one, we get a line which is multiple of the first one. As the second line is formed only by zeros, we can not get as many as necessary of solution affinelly independant for (33) to be facet defining.

\[ \square \]

**Theorem 4.9** Inequality (33) defines a facet of MCSIPND\(_m\)(\( G, \mathcal{F}, K \)) if

1. condition 1), 2), 4), 5) of Theorem 4.8 are satisfied,
2. \( G(W) \) and \( G(W) \) are \( \mathcal{F} \)-connected.

**Proof.** See Appendix. \( \square \)

In [31], Magnanti et al. introduce cutset inequalities valid for the Two-Facility Capacitated Network Loading Problem (TLFP). These can be easily extended to the MCSIPND\(_m\) and the MCSIPND\(_s\) problems. The extended ones are special cases of the capacity demand cut inequalities.

**Corollary 4.10** Let \( F_i \in \mathcal{F} \) be an edge subset of \( E \), \( W \subseteq V \), \( \emptyset \neq W \neq V \), and let

\[ r_W = \left\{ \begin{array}{ll} 4 & \text{if } D_W \mod 4 = 0, \\
                       D_W \mod 4 & \text{otherwise}. \end{array} \right. \]

Then the inequality

\[ \sum_{e \in \delta_{c_i}(W)} (x^1_e + r_W x^2_e) \geq r_W \left[ \frac{D_W}{4} \right] \quad (38) \]

is valid for both MCSIPND\(_m\)(\( G, \mathcal{F}, K \)) and MCSIPND\(_s\)(\( G, \mathcal{F}, K \)).

**Proof.**

- If \( r_W = 1 \), inequality (38) is nothing but inequality (35).
- If \( r_W = 2 \), as \( 2 \left[ \frac{D_W}{4} \right] = \left[ \frac{D_W}{2} \right] + 1 \), inequality (38) is nothing but inequality (37).
- If \( r_W = 3 \), as \( 3 \left[ \frac{D_W}{4} \right] = \left[ \frac{3D_W}{4} \right] \), inequality (38) is nothing but inequality (36).
- If \( r_W = 4 \), as \( 4 \left[ \frac{D_W}{4} \right] = D_W \), inequality (38) is nothing but inequality (33). \( \square \)
4.2 Saturation inequalities

In this section we introduce a further class of valid inequalities for only the polytope $\text{MCISPND}_s(G, \mathcal{F}, K)$. These inequalities are also induced by cuts. In these inequalities only the $x^2$ variables, associated with the links with big capacity (10Gbits), are involved. In fact, the idea behind these inequalities is that if the demand can not be routed on the links of a cut with small capacity (2.5 Gbits), then at least one big capacity must be installed on one of the links of the cut.

**Theorem 4.11** Let $F_i \in \mathcal{F}$ and $W \subseteq V, \emptyset \neq W \neq V$. Then inequality

$$x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{\max \left\{ \sum_{k \in \gamma^+(W)} \omega_k, \sum_{k \in \gamma^-(W)} \omega_k \right\} - |\delta_{G_i}(W)| \times 2.5}{7.5} \right\rceil$$  \hspace{1cm} (39)$$

is valid for $\text{MCISPND}_s(G, \mathcal{F}, K)$.

**Proof.** We suppose that $x^1(e) = 1$ for all $e \in \delta_{G_i}(W)$. The available capacity across the cut $\delta_{G_i}(W)$ is then equal to $|\delta_{G_i}(W)| \times 2.5$. Let $M = \max \left\{ \sum_{k \in \gamma^+(W)} \omega_k, \sum_{k \in \gamma^-(W)} \omega_k \right\} - |\delta_{G_i}(W)| \times 2.5$. Note that $M$ is nothing but the missing capacity across the cut $\delta_{G_i}(W)$.

- If $M \leq 0$, that is to say, the demand over the cut is less than or equal the available capacity on the small edges. Then the edges can carry the flow across $\delta_{G_i}(W)$. In this case, inequality (39) can be written as $x^2(\delta_{G_i}(W)) \geq 0$ and is thus valid for $\text{MCISPND}_s(G, \mathcal{F}, K)$.
- If $M > 0$, this means that the edges of $\delta_{G_i}(W)$ do not suffice to carry the whole flow between $W$ and $\overline{W}$, if they get all a small capacity of 2.5 Gbits. We then have to replace some small capacities by big ones. Let $f \in E^1$ be an edge on which we install a big capacity instead of a small one. As $x^1(e) + x^2(e) \leq 1$ for all $e \in E^1$, if $x^2(f) = 1$ this implies that $x^1(f) = 0$. Therefore, installing a big capacity on $f$ allows to add a capacity of $10-2.5=7.5$ Gbits over the cut. Hence the minimum number of edges with big capacity we need in order to route the demand between $W$ and $\overline{W}$ is

$$\left\lceil \frac{\max \left\{ \sum_{k \in \gamma^+(W)} \omega_k, \sum_{k \in \gamma^-(W)} \omega_k \right\} - |\delta_{G_i}(W)| \times 2.5}{7.5} \right\rceil.$$  \hspace{1cm} (39)$$

Then inequality (39) is valid for $\text{MCISPND}_s(G, \mathcal{F}, K)$.

\hfill \Box

In the two following sections we present further classes of valid inequalities which are extensions of valid inequalities introduced in [7] for the problem without capacities.

4.3 Cut-cycle inequalities

Let $W \subseteq V$ and $T_1 = \{e_1, \ldots, e_s\}, s \geq 3$, be an edge subset of $\delta_{G}(W)$. Let $1 \leq q < s$ be an integer. Suppose that for every $i = 1, \ldots, s$, there is $j_i \in \{1, \ldots, t\}$ such that $F_{j_i} \cap T_1 = \{e_i, \ldots, e_{i+q-1}\}$ (the indices are modulo $s$). Let $T_2 = \delta_{G}(W) \setminus (T_1 \cup (\bigcap_{i=1}^{s} F_{j_i}))$. Such a configuration $(W, T_1, T_2)$ will be called a cut-cycle configuration (see Figure 6).

**Theorem 4.12** Let $(W, T_1, T_2)$ be a cut-cycle configuration such that $\gamma(W) \neq \emptyset$. For $e \in \delta_{G}(W)$, let $r_e = |\{i \in \{1, \ldots, s\} | e \in \delta_{G}(W) \setminus F_{j_i}\}|$ and $r$ be the smallest integer such that $r(s - q) \geq \max_{e \in T_2} r_e$. Then inequality

$$\sum_{l=1,2} (x^l(T_1) + rx^l(T_2)) \geq \left\lfloor \frac{s}{s - q} \right\rfloor$$  \hspace{1cm} (40)$$

is valid for both $\text{MCISPND}_m(G, \mathcal{F}, K)$ and $\text{MCISPND}_s(G, \mathcal{F}, K)$. 

18
Proof. The following inequalities are valid for both \( \text{MCSIPND}_m(G, F, K) \) and \( \text{MCSIPND}_s(G, F, K) \),

\[
\sum_{l=1,2} x^l(\delta_{G,j_i}(W)) \geq 1 \quad \text{for } i = 1, \ldots, s,
\]

\[
(r(s - q) - r_e)x^l(e) \geq 0 \quad \text{for all } e \in T_2, l = 1, 2.
\]

By summing these inequalities, we obtain

\[
\sum_{l=1,2} ((s - q)x^l(T_1) + r(s - q)x^l(T_2)) \geq s.
\]

By dividing by \( s - q \) and rounding up the right hand side we get inequality (40).

Inequalities (40) will be called design cut-cycle inequalities.

**Theorem 4.13** Let \((W, T_1, T_2)\) be a cut-cycle configuration such that \( \gamma(W) \neq \emptyset \). For \( e \in \delta_G(W) \), let \( r_e = |\{i \in \{1, \ldots, s\} | e \in \delta_G(W) \setminus F_{j_i}\}| \), and \( r \) be the smallest integer such that \( r(s - q) \geq \max_{e \in T_2}\{r_e\} \).

Let \( r_W = \begin{cases} 4 & \text{if } D_W \mod 4 = 0 \\ D_W \mod 4 & \text{otherwise.} \end{cases} \)

Then the inequality

\[
x^1(T_1) + r_W x^2(T_1) + r x^1(T_2) + r x_W x^2(T_2) \geq \left[ \frac{s}{s - q} \times r_W \left[ \frac{D_W}{4} \right] \right]
\]

is valid for both \( \text{MCSIPND}_m(G, F, K) \) and \( \text{MCSIPND}_s(G, F, K) \).

**Proof.** By Theorem 4.10, the following inequalities are valid for both \( \text{MCSIPND}_m(G, F, K) \) and \( \text{MCSIPND}_s(G, F, K) \),

\[
\sum_{e \in \delta_{G,j_i}(W)} (x^1_e + r_W x^2_e) \geq r_W \left[ \frac{D_W}{4} \right] \quad \text{for } i = 1, \ldots, s.
\]

Also consider the following inequalities which are also valid for both \( \text{MCSIPND}_m(G, F, K) \) and \( \text{MCSIPND}_s(G, F, K) \).

\[
(r(s - q) - r_e)x^1(e) \geq 0 \quad \text{for all } e \in T_2,
\]

\[
(r(s - q) - r_e)r_W x^2(e) \geq 0 \quad \text{for all } e \in T_2.
\]
By summing these inequalities we obtain
\[
\sum_{e \in T_1} [(s-q)x_e^1 + (s-q)rWx_e^2] + \sum_{e \in T_2} [r(s-q)x_e^1 + r(s-q)rWx_e^2] \geq s \times rW \left\lceil \frac{D_W}{4} \right\rceil .
\]

By dividing this inequality by \( s-q \) and rounding up the right hand side we get inequality (41). \( \square \)

Inequalities (41) will be called capacity cut-cycle inequalities.

We can remark that if \( rW = 1 \) and \( \left\lceil \frac{D_W}{4} \right\rceil = 1 \) we then obtain the design cut-cycle inequality (40).

4.4 Star-partition inequalities

Let \( G = (V,E) \) be a graph and \( \mathcal{F} = \{F_1, \ldots, F_t\} \), with \( t \geq 2 \), a family of edge subsets of \( E \). Let \( V_0, V_1, \ldots, V_p \) be a partition of \( V \) with \( p \) odd. Suppose that for every \( i = 1, \ldots, p \), there is \( j_i \in \{1, \ldots, t\} \) such that \( F_{j_i} \cap [V_i, V_0] \neq \emptyset \). Let \( \Lambda = \{ e \in E \mid e \in [V_k, V_l] \cap F_{j_k} \cap F_{j_l}, \text{ for some } k, l \in \{1, \ldots, p\} \} \). Let \( F = \bigcup_{i=1}^p (F_{j_i} \cap [V_i, V_0]) \cup \Lambda \). Such a configuration will be called a star-partition configuration (see Figure 7).

\[
\text{Fig. 7 – A star-partition configuration}
\]

**Theorem 4.14** Let \( (V_0, V_1, \ldots, V_p, F) \) a star-partition configuration with \( p \) odd such that \( \gamma(W) \neq \emptyset \). Then inequality
\[
\sum_{l=1,2} x^l(\delta_G(V_0, \ldots, V_p) \setminus F) \geq \left\lceil \frac{p}{2} \right\rceil
\]
(42)
is valid for both \( \text{MCSIPN}_m(G, \mathcal{F}, K) \) and \( \text{MCSIPN}_s(G, \mathcal{F}, K) \).

**Proof.** It is clear that the following inequalities are valid for both \( \text{MCSIPN}_m(G, \mathcal{F}, K) \) and \( \text{MCSIPN}_s(G, \mathcal{F}, K) \).

\[
\sum_{l=1,2} x^l(\delta_{G \setminus F_{j_i}}(V_i)) \geq 1 \quad \text{for } i = 1, \ldots, p,
\]
\[
x^l(e) \geq 0 \quad \text{for all } e \in \delta(V_0) \setminus F, \ l = 1, 2,
\]
\[
x^l(e) \geq 0 \quad \text{for all } e \in (\delta_G(V_k, V_m) \cap F_{j_k}) \setminus F_{j_m},
\]
\[
k = 1, \ldots, p, \ m = 1, \ldots, p, \ k \neq m, \ l = 1, 2.
\]
By summing these inequalities, we obtain inequality
\[ 2 \sum_{i=1}^{2} x^i \delta(V_0, \ldots, V_p) \leq F \geq p. \]

By dividing by 2 and rounding up the right hand side, we obtain inequality (42).

Inequalities (42) will be called design star-partition inequalities.

**Theorem 4.15** Let \((V_0, V_1, \ldots, V_p, F)\) be a star-partition configuration.

Let \(r_{V_i} \in \{ \begin{cases} 4 & \text{if } D_{V_i} \mod 4 = 0, \\ D_{V_i} \mod 4 & \text{otherwise.} \end{cases} \)

for \(i = 1, \ldots, p.\)

Set
\[ r_e = \sum_{i = 1, \ldots, p : e \in \delta_G \setminus F_i(V_i)} r_{V_i} \]

and
\[ \lambda_e = \begin{cases} r_e & \text{if } r_e \text{ is even,} \\ r_e + 1 & \text{otherwise,} \end{cases} e \in \delta_G(V_0, \ldots, V_p) \setminus F. \]

Then inequality
\[ \sum_{e \in \delta_G(V_0, \ldots, V_p) \setminus F} \left( x^1_e + \frac{\lambda_e}{2} x^2_e \right) \geq \left[ \sum_{i=1}^{p} r_{V_i} \left[ \frac{D_{V_i}}{4} \right] \right] \]

is valid for both MCSIPND_m(G, F, K) and MCSIPND_s(G, F, K).

**Proof.** The following inequalities are valid for both MCSIPND_m(G, F, K) and MCSIPND_s(G, F, K).
\[ \sum_{e \in \delta_G \setminus F_i(V_i)} (x^1_e + r_{V_i} x^2_e) \geq r_{V_i} \left[ \frac{D_{V_i}}{4} \right] \text{ for } i = 1, \ldots, p, \]

\[ x^1_e \geq 0 \quad \text{for all } e \in \delta_G(V_0, \ldots, V_p) \setminus F, \]
\[ x^2_e \geq 0 \quad \text{for } i = 1, \ldots, p, \text{ for all } e \in \delta_G(V_i, V_0) \setminus F, \text{ such that } r_{V_i} \text{ is odd,} \]
\[ x^2_e \geq 0 \quad \text{for all } e \in \delta_G(V_k, V_m) \setminus (F_{j_k} \cap F_{j_m}), \text{ such that } r_e \text{ is odd,} \]
\[ k = 1, \ldots, p, m \neq k, \]
\[ x^1_e \geq 0 \quad \text{for all } e \in (\delta_G(V_k, V_m) \setminus F_{j_k}) \setminus F_{j_m}, k = 1, \ldots, p, m \neq k, \]
\[ x^2_e \geq 0 \quad \text{for all } e \in (\delta_G(V_k, V_m) \setminus F_{j_k}) \setminus F_{j_m}, \text{ such that } r_{V_m} \text{ is odd,} \]
\[ k = 1, \ldots, p, m \neq k, \]

By summing these inequalities we get
\[ \sum_{e \in \delta_G(V_0, \ldots, V_p) \setminus F} (2x^1_e + \lambda_e x^2_e) \geq \sum_{i=1}^{p} r_{V_i} \left[ \frac{D_{V_i}}{4} \right]. \]

As \(\lambda_e\)'s, \(e \in \delta_G(V_0, \ldots, V_p) \setminus F\) are all even, by dividing this inequality by 2 and rounding up the right hand side we get the inequality
\[ \sum_{e \in \delta_G(V_0, \ldots, V_p) \setminus F} \left( x^1_e + \frac{\lambda_e}{2} x^2_e \right) \geq \left[ \sum_{i=1}^{p} r_{V_i} \left[ \frac{D_{V_i}}{4} \right] \right]. \]

Inequalities (43) are called capacity star-partition inequalities.
5 Branch-and-cut and branch-and-cut-and-price algorithms

In this section, we describe four algorithms for the MCSIPND problem. We consider the two variants of the problem (simple and multiple) and for each variant we propose a Branch-and-Cut algorithm based on the node-arc formulation and a Branch-and-Cut-and-Price algorithm based on the path formulation. Our aim is to address the algorithmic applications of the previous results.

We now describe the framework of our algorithms. For the Branch-and-Cut algorithms based on the node-arc formulation, we start the optimization with the linear relaxations of the formulations. The optimal solutions \((x^1, x^2, f)\) of these relaxations are feasible for the MCSIPND problems if \(x^1\) and \(x^2\) are integral.

For the Branch-and-Cut-and-Price algorithms, we also start the optimization by solving the linear relaxation of the path formulations. For this we use the column generation algorithms described in Section 3. As for the node-arc formulations, the solutions then obtained are feasible for the MCSIPND problems only if \(x^1\) and \(x^2\) are integral.

Usually, a solution \((x^1, x^2, f)\) is not feasible, and thus, in each iteration of the Branch-and-Cut and the Branch-and-Cut-and-Price algorithms, it is necessary to generate further inequalities that are valid for the MCSIPND problem but violated by the current solution \((x^1, x^2, f)\). For this, one has to solve the so-called separation problem. This consists, given a class of inequalities, in deciding whether the current solution \((x^1, x^2, f)\) satisfies the inequalities, and if not, in finding an inequality that is violated by \((x^1, x^2, f)\). An algorithm which solves this problem is called a separation algorithm. The inequalities given above are all valid for the four polytopes MCSIPND\(_m^a\), MCSIPND\(_m^p\), MCSIPND\(_s^a\) et MCSIPND\(_s^p\) except the saturation inequalities which are valid only for the simple version of the problem. Hence these inequalities are used in our algorithms. The separation is performed in the following order:

1. arc residual capacity constraints (9) (for the Branch-and-Cut algorithms only),
2. design cut constraints (32) and capacity demand cut constraints (33)-(37),
3. saturation constraints (39) (for MCSIPND\(_a\) only),
4. design cut-cycle constraints (40),
5. capacity cut-cycle constraints (41),
6. design star-partition constraints (42),
7. capacity star-partition constraints (43).

We remark that all inequalities are global (i.e. valid in all the Branch-and-Cut tree and the Branch-and-Cut-and-Price tree) and several constraints may be added at each iteration. Moreover, we go to the next class of inequalities only if we have found no violated inequality. Our strategy is to try to detect violated constraints at each node of the tree in order to obtain the best possible lower bound and thus limit the number of generated nodes. Generated inequalities are added by sets of at most 200 inequalities at a time.

Now we describe the separation procedures used in our algorithms. All our separation algorithms are applied on \(G_{(\vec{x}^1, \vec{x}^2)} = (V, E_{(\vec{x}^1, \vec{x}^2)})\) where \((\vec{x}^1, \vec{x}^2)\) is the restriction on \(x^1\) and \(x^2\) of the current LP solution, and \(E_{(\vec{x}^1, \vec{x}^2)}\) contains all the edges \(uv\) of \(E\) such that \(\vec{x}_{uv}^1 + \vec{x}_{uv}^2 \neq 0\).

To separate the design cut inequalities and the capacity demand cut inequalities, we have developed a fast heuristic. We first check whether a degree cut \(\delta_{G_{(\vec{x}^1, \vec{x}^2)}}(v), v \in V\), is violated. Then we start contracting edges \(uv\) with high value \(2.5\vec{x}_{uv}^1 + 10\vec{x}_{uv}^2 - \sum_{k \in \gamma((u) \cup \gamma((v)))} \omega_k\) until we get a graph on two nodes. In each iteration we check if the cut associated with the node arising from the contraction, induces a violated constraint of type (33), (35), (36) or (37).

When this heuristic does not any more allow to find violated inequalities, we compute the so-called Gomory-Hu tree [17] on the graph \(G_{(\vec{x}^1, \vec{x}^2)}\) with the weight \(\vec{x}_{uv}^1 + \vec{x}_{uv}^2\) for each edge \(uv\). This
tree has the property that for all pairs of nodes \( s, t \in V \) the minimum \((s,t)\)-cut in the tree is also a minimum \((s,t)\)-cut in \( G \). Actually, we use the algorithm developed by Gusfield [21] which requires \(|V| - 1\) maximum flow computations. The maximum flow computations are handled by the efficient Goldberg and Tarjan algorithm [16] that runs in \( O(mn \log n) \) time where \( m \) and \( n \) are the number of edges and nodes of \( G \), respectively. Then we calculate the right hand side for all the cuts in the Gomory-Hu tree and check if the found constraints are violated.

During the separation of the saturation constraints (39), we consider the cuts \( \delta_{G_{(x_1,x_2)}}(v), v \in V \). We test if these inequalities are violated and if so we add them to the program. We don’t consider the other cuts of the graph \( G_{(x_1,x_2)} \) induced by more than two nodes which seem almost never be violated.

Now we turn our attention to the separation of the cut-cycle inequalities (40) and (41). For more efficiency, we have used these constraints only when \( q = 1 \). In fact we remarked that the design cut-cycle inequalities and the capacity cut-cycle inequalities which are violated are usually of this type.

To separate the design cut-cycle constraints with \( q = 1 \), we compute the Gomory-Hu tree of the graph \( G_{(x_1,x_2)} \) with the weight for each edge \( uv \) equal to the sum \( \bar{x}_u^1 + \bar{x}_u^2 \). Then for each cut given by the Gomory-Hu tree, with value less than 2, we test if the cut intersects at least one demand. If this is the case, then it yields a design cut-cycle inequality (40) violated by \((\bar{x}_1,\bar{x}_2)\). Then sets \( T_1 \) and \( T_2 \) are determined so that \( T_1 \) is maximal, using the following greedy procedure (Algorithm 1).

Since the Gomory-Hu algorithm runs with a large complexity, in order to accelerate our separation for the cut-cycle inequalities, we first consider the degree cuts \( \delta_{G_{(x_1,x_2)}}(v), v \in V \). The computation of the Gomory-Hu tree is considered only if no cuts of this type of value less than 2 are found.

Algorithm 1

```plaintext
T_1 \leftarrow \emptyset; T_2 \leftarrow \emptyset; \mathcal{F} \leftarrow \emptyset;

for i = 1 to m do
    if \( f_i \in F_{j_0} \) for some \( j_0 \in \{1, \ldots, t\} \) and \( f_i \notin F_j \) for all \( F_j \in \mathcal{F} \) then
        \( T_1 \leftarrow T_1 \cup \{f_i\} \);
        \( \mathcal{F} \leftarrow \mathcal{F} \cup \{F_{j_0}\} \);
    else
        \( T_2 \leftarrow T_2 \cup \{f_i\} \);
    end if
end for

for all \( f_i \in T_2 \) do
    if \( f_i \in F_j \) for all \( F_j \in \mathcal{F} \) then
        \( T_2 \leftarrow T_2 \setminus \{f_i\} \);
    end if
end for
```

For the separation of the capacity cut-cycle constraints, we first consider the degree cuts \( \delta_{G_{(x_1,x_2)}}(v), v \in V \). We calculate the right hand side. If the associated constraint is violated, we then determine the sets \( T_1 \) and \( T_2 \) using Algorithm 1. Then we start contracting edges until we get a graph on two nodes. In each iteration we contract an edge \( uv \) with the biggest value for \( 2.5\bar{x}_u^1 + 10\bar{x}_u^2 - \sum_{k \in \gamma(v)} w_k \) and check whether the new node obtained by contraction together with \( T_1 \) and \( T_2 \) induces a violated capacity cut-cycle inequality.

We now discuss our separation routine for the star-partition inequalities (42) and (43). Our algorithm consists in determining fractional cycles in the supporting graph, satisfying some conditions. These cycles have to be odd, in order to have a chance to find a violated design star-partition inequality. Thus, for each detected cycle \((v_1, \ldots, v_p)\) we try to find edge subsets \( F_j, j_i \in \{1, \ldots, t\}, i = 1, \ldots, p \) among the edges of \([v_i, V \setminus \{v_1, \ldots, v_p\}]\) in such a way that either the design star-partition inequality or the capacity star-partition induced by \( V \setminus \{v_1, \ldots, v_p\}, \{v_1\}, \ldots, \{v_p\} \), and
\( F_{ji}, i = 1, \ldots, p \) is violated by \((x^1, x^2)\).

To store the generated inequalities, we created a pool whose size increases dynamically. All the generated inequalities are put in the pool and are dynamic, i.e. they are removed from the current LP when they are not active. We first separate inequalities from the pool. If all the inequalities in the pool are satisfied by the current LP-solution, we separate the classes of inequalities in the order given above.

In the following section, we give some computational results obtained with the algorithms presented above for random instances and for real instances provided by France Télécom.

6 Computational results

The Branch-and-Cut and Branch-and-Cut-and-Price algorithms described in the previous section have been implemented in C++, using ABACUS (A Branch-And-Cut System) 2.4 alpha \[1, 14, 42\] to manage the Branch-and-Cut tree and Cplex 9.0 as LP-solver \[11\]. It was tested on a Pentium IV 2.4 GHz with 1 Gb RAM, running under Linux. We fixed the maximum CPU time to 5 hours.

Results are presented here for instances coming from real applications and instances obtained from problems of the TSP Library \([38]\) by randomly generating the node set, the edge sets \(F_e\) and the set of demands \(K\). For all the instances, the graph \(G^1\), representing the IP network, is considered complete.

These instances were generated with 6, 8 and 10 nodes, \(|F| = 10, 20\) and \(|K| = 5, 10, 20\). Five instances of each size, each \(|F|\) and each \(|K|\) were tested. We will consider the average results obtained for these instances.

The real instances are extracted from operational networks and have been provided by the french telecommunications operator France Télécom. These instances have 6 to 18 nodes and \(F\) with 11 to 32 edge sets. Actually France Télécom has provided the optical network and the routing between every pair of nodes in this network. With an edge \(e\) of the IP network, we associate the routing path of the optical network between the switches corresponding to the IP router extremities of \(e\).

Using these paths, we have computed \(F = \{F_e \subseteq E^1, e \in E^2\}\) where \(F_e\) is the set of edges \(f\) of \(E^1\) such that \(e\) belongs to the path associated with \(f\).

The number of commodities is between 5 and 20. We randomly generated the extremities of the commodities. The amount of each commodity is calculated with the gravity model which uses the distance and the population of both the origin and destination cities. The general expression of the gravity model for a commodity \((o_k, d_k, v_k)\) of \(K\) is \(v_k = \frac{P_o^k P_d^k}{d_{o_k, d_k}}\) where \(P_o^k\) and \(P_d^k\) are the populations of the origin and destination cities, respectively and \(d_{o_k, d_k}\) represents the euclidian distance between \(o_k\) and \(d_k\). We have fixed \(\alpha = 1.2\) and \(\beta = 0.8\) to have different volume between two towns in the two directions.

Usually the cost associated with a link in the client network is related to the corresponding routing path in the optical network, and then depends on the cost of this path. Actually, the cost \(c(f)\) of link \(f\) in the IP network is given by

\[
c(f) = c + \kappa(f),
\]

where \(c\) is a fixed cost representing the equipments of the extremity ports on the routers of \(f\) in the IP layer, and \(\kappa(f)\) is a cost depending on the length of the path \(P_f\) corresponding to \(f\) in the optical network.

The installation of an optical segment usually yields a fixed cost on each extremity of this segment. Hence a first estimation of the optical cost \(\kappa(f)\) is the sum of the fixed costs of the optical
segments on $P_f$. As these fixed costs can be considered the same in the optical network, a good approach would be to consider a cost $\kappa(f)$ proportional to the number of the optical segments on $P_f$. So, a first natural function $\kappa(f)$ consists of the number of links (hops) in the optical path between the switching nodes corresponding to the extremities of $f$. Here we assume that there is a fixed cost associated with each optical link. This cost is considered once the corresponding link is used. Then the cost $c(f)$ is given in this case by $c + |P_f|$. 

In the various tables, the entries are:

- $|V^1|$: the number of nodes of $G^1$,
- $|\mathcal{F}|$: the number of sets $F_e$,
- $|K|$: the number of demands,
- Algo: the type of algorithm ($na$ (resp. $p$ means that the used algorithm is based on the node-arcs (resp. path) formulation.
- FV: the number of flow variables,
- NC: the number of generated cut inequalities,
- NRC: the number of generated arc residual capacity inequalities,
- NS: the number of generated saturation inequalities (only for MCSIPND$_n$ problem),
- NCC: the number of generated cut-cycle inequalities,
- NSP: the number of generated star-partition inequalities,
- NT: the number of generated nodes in the Branch-and-Cut tree,
- o/p: the number of problems solved to optimality over the number of instances tested (only for random instances),
- Copt: the value of the optimal solution,
- Gap: the relative error between the best upper bound (the optimal value if the problem has been solved to optimality) and the lower bound achieved by the cutting plane phase (before branching),
- TT: the total CPU time in h:mm:ss.

Our first series of experiments concerns the problem MCSIPND$_m$ (with multiple links) for the random instances. In these experiments, we have considered three instances for each size. Table 1 reports the average results obtained for these instances with both algorithms based on the node-arc and the path formulations.

As we can observe, all the instances with 6 nodes have been solved to optimality. Moreover, they have been solved in less than 5 minutes (and in less than 2 minutes using the path formulation based algorithm), except those with $|\mathcal{F}| = 20$ and $|K| = 20$ which needed around 33 minutes to be solved to optimality using the node-arc formulation based algorithm. We also remark that for the instances with a reduced number of commodities, the problem seems to be much easier to solve. In fact, for all the instances with 5 commodities, less than one hour was needed to get the optimal solution except for the instances with 10 nodes and $|\mathcal{F}| = 20$.

The instances with 10 and 20 commodities and $|\mathcal{F}| = 20$ seem to be harder to solve. In fact, none of these instances could be solved to optimality with the node-arc formulation within the time limit, when the number of nodes exceeds 8. The results are more promising with the path formulation where 5 among the 12 instances have nevertheless been solved in an average time sometimes less than two hours.

Among the 54 tested instances, 17 could not be solved within the time limit of five hours using the node-arc formulation based algorithm, and only 12 instances with that based on the path formulation. So, this would imply that the latter algorithm is more efficient. However for 10 nodes, $|\mathcal{F}| = 10$ and $|K| = 20$, two instances have been solved with the first algorithm but they did not with the second one.

The number of flow variables of the node-arc formulation is fixed and depends only on $|V^1|$, $|\mathcal{F}|$ and $|K|$. This is not however the case for the flow variables of the path formulation. We remark
that the number of the flow variables generated, using this formulation, is generally smaller than that of the node-arc formulation. This may explain the fact that some instances could be solved to optimality using the path formulation whereas they could not using the node-arc one. Moreover the CPU times obtained with the path formulation based algorithm are less than those obtained with the node-arc formulation based algorithm except for the instances with 10 nodes and $|F| = 10$. The difference of time is sometimes big. For example, for the instances with 6 nodes, $|F| = 20$ and $|K| = 20$, the time goes from less than 2 minutes to more than 30 minutes and for 8 nodes, $|F| = 10$ and $|K| = 20$ it goes from 20 minutes to more than 4 hours. Therefore the path formulation seems to give better results than the node-arc formulation.

We can also remark that for most of the instances, a significant number of cut and arc residual capacity inequalities (for the node-arc formulation) have been generated. This implies that these inequalities are useful for the random problems. However the cut-cycle and star-partition inequalities do not seem to play an important role for this type of instances. This can be explained by the fact that, for these instances, the edges do not necessarily belong to some $F_e$'s, and hence, it could be hard to find cut-cycle and star-partition configurations.

Table 2 presents the results for the MCSIPND$_m$ problem for the real instances with both path and node-arc formulation based algorithms. Here the cost function represents, for each link of the IP network, the number of links of the optical network used in the associated path. This cost function is then integral.

It appears from the table that the difficulty to solve the instances increases with the number of

| $|V|$ | $|F|$ | $|K|$ | Algo | P | NC | NRC | NCC | NEP | NT | o/p | Gap | TT |
|------|------|------|------|----|----|------|------|------|----|-----|-----|----|
| 6    | 10   | 5    | $n_a$ | 1500 | 85.67 | 78.00 | 0.67 | 0.67 | 152.33 | 3/3 | 9.47 | 0:00:13.52 |
| 6    | 10   | 10   | $p$   | 3000 | 80.00 | 89.00 | 0.00 | 0.00 | 109.67 | 3/3 | 5.86 | 0:00:40.11 |
| 6    | 10   | 20   | $n_a$ | 6000 | 71.33 | 201.00 | 0.33 | 0.33 | 123.00 | 3/3 | 6.39 | 0:00:22.54 |
| 6    | 10   | 5    | $p$   | 590.67 | 93.33 | - | 0.33 | 0.67 | 112.33 | 3/3 | 9.40 | 0:00:07.25 |
| 6    | 10   | 10   | $p$   | 899.33 | 92.67 | - | 0.00 | 0.00 | 95.00 | 3/3 | 5.24 | 0:00:08.38 |
| 6    | 10   | 20   | $n_a$ | 1363.33 | 124.67 | - | 0.00 | 0.00 | 157.00 | 3/3 | 5.96 | 0:00:15.40 |
| 6    | 20   | 5    | $p$   | 3000 | 105.00 | 226.33 | 2.00 | 0.67 | 220.33 | 3/3 | 10.74 | 0:01:38.04 |
| 6    | 20   | 10   | $p$   | 6000 | 125.67 | 427.67 | 2.67 | 0.00 | 109.67 | 3/3 | 8.41 | 0:00:08.38 |
| 6    | 20   | 20   | $n_a$ | 12000 | 153.33 | 153.33 | 0.33 | 0.33 | 153.33 | 3/3 | 7.48 | 0:00:29.84 |

Tab. 1 – Result for random instances for the MCSIPND$_m$ problem
nodes and the number of commodities in the network. All the instances with 5 commodities and up to 14 nodes have been solved in less than one hour. Also for the instances with 10 commodities, we have obtained an optimal solution for all the instances with no more than 10 nodes in less than 3 hours with the node-arc formulation and in less than 45 minutes with the path formulation. The instance with 16 nodes and 10 commodities has also been solved to optimality.

In addition, we remark that for the instances with 15 and 20 commodities only those with no more than 8 nodes could be solved to optimality. For all the instances which have not been solved in the time limit, we have nevertheless obtained feasible solutions (given in italic). Also, we can remark that for some instances, we have obtained a relatively small gap. This is for example the case for the instance with 12 nodes and 10 commodities where the gap is 6%. For the larger instances, the gap does not exceed 18%.

Moreover we notice that all the instances solved with the algorithm based on the node-arc formulation have also been solved with the one based on the path formulation, except that with 14 nodes and 10 commodities. Furthermore, the time needed was much lower. In fact, as it can be observed, for the instance with 8 nodes and 15 commodities, the time passes from 2 hours 12 minutes to 18 minutes and for that with 10 nodes and 10 commodities, it passes from 2 hours 45 minutes to 2 minutes 17 seconds.

Tab. 2 – Results for real instances for the MCSIPND$_m$ problem

<table>
<thead>
<tr>
<th>V</th>
<th>F</th>
<th>Algo</th>
<th>Ev</th>
<th>Nc</th>
<th>NRC</th>
<th>NCC</th>
<th>NEP</th>
<th>NT</th>
<th>Copt</th>
<th>Gap</th>
<th>TT</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>11</td>
<td>5</td>
<td>na</td>
<td>1650</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>57</td>
<td>0.00</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>10</td>
<td>p</td>
<td>3300</td>
<td>56</td>
<td>111</td>
<td>2</td>
<td>3</td>
<td>65</td>
<td>99</td>
<td>8.08</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>15</td>
<td></td>
<td>4950</td>
<td>99</td>
<td>779</td>
<td>1</td>
<td>0</td>
<td>157</td>
<td>105</td>
<td>7.62</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>20</td>
<td></td>
<td>6600</td>
<td>199</td>
<td>1194</td>
<td>1</td>
<td>0</td>
<td>375</td>
<td>115</td>
<td>9.57</td>
</tr>
<tr>
<td>8</td>
<td>17</td>
<td>5</td>
<td>na</td>
<td>4760</td>
<td>173</td>
<td>114</td>
<td>2</td>
<td>0</td>
<td>115</td>
<td>81</td>
<td>11.11</td>
</tr>
<tr>
<td>8</td>
<td>17</td>
<td>10</td>
<td>p</td>
<td>9520</td>
<td>191</td>
<td>717</td>
<td>2</td>
<td>7</td>
<td>567</td>
<td>109</td>
<td>15.60</td>
</tr>
<tr>
<td>8</td>
<td>17</td>
<td>15</td>
<td></td>
<td>14280</td>
<td>512</td>
<td>815</td>
<td>4</td>
<td>2</td>
<td>621</td>
<td>123</td>
<td>13.01</td>
</tr>
<tr>
<td>8</td>
<td>17</td>
<td>20</td>
<td></td>
<td>23400</td>
<td>262</td>
<td>279</td>
<td>4</td>
<td>2</td>
<td>369</td>
<td>156</td>
<td>3.21</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>5</td>
<td></td>
<td>11250</td>
<td>71</td>
<td>157</td>
<td>4</td>
<td>0</td>
<td>113</td>
<td>129</td>
<td>7.75</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>10</td>
<td>na</td>
<td>23400</td>
<td>262</td>
<td>279</td>
<td>4</td>
<td>2</td>
<td>369</td>
<td>156</td>
<td>3.21</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>15</td>
<td></td>
<td>33750</td>
<td>772</td>
<td>1616</td>
<td>6</td>
<td>0</td>
<td>473</td>
<td>123</td>
<td>13.01</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>20</td>
<td></td>
<td>45000</td>
<td>447</td>
<td>2754</td>
<td>7</td>
<td>5</td>
<td>709</td>
<td>160</td>
<td>33.75</td>
</tr>
<tr>
<td>12</td>
<td>32</td>
<td>5</td>
<td>na</td>
<td>21120</td>
<td>85</td>
<td>180</td>
<td>5</td>
<td>0</td>
<td>99</td>
<td>129</td>
<td>8.53</td>
</tr>
<tr>
<td>12</td>
<td>32</td>
<td>10</td>
<td>na</td>
<td>42240</td>
<td>333</td>
<td>777</td>
<td>5</td>
<td>1</td>
<td>303</td>
<td>194</td>
<td>12.37</td>
</tr>
<tr>
<td>12</td>
<td>32</td>
<td>15</td>
<td></td>
<td>63360</td>
<td>251</td>
<td>1129</td>
<td>8</td>
<td>0</td>
<td>127</td>
<td>231</td>
<td>18.18</td>
</tr>
<tr>
<td>12</td>
<td>32</td>
<td>20</td>
<td></td>
<td>84380</td>
<td>357</td>
<td>2001</td>
<td>7</td>
<td>0</td>
<td>89</td>
<td>392</td>
<td>41.58</td>
</tr>
<tr>
<td>14</td>
<td>27</td>
<td>5</td>
<td>na</td>
<td>24570</td>
<td>55</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>150</td>
<td>8.00</td>
</tr>
<tr>
<td>14</td>
<td>27</td>
<td>10</td>
<td></td>
<td>49140</td>
<td>342</td>
<td>298</td>
<td>2</td>
<td>0</td>
<td>175</td>
<td>198</td>
<td>6.60</td>
</tr>
<tr>
<td>14</td>
<td>27</td>
<td>5</td>
<td>na</td>
<td>24570</td>
<td>55</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>150</td>
<td>8.00</td>
</tr>
<tr>
<td>14</td>
<td>27</td>
<td>10</td>
<td>na</td>
<td>49140</td>
<td>342</td>
<td>298</td>
<td>2</td>
<td>0</td>
<td>175</td>
<td>198</td>
<td>6.60</td>
</tr>
<tr>
<td>16</td>
<td>29</td>
<td>5</td>
<td>na</td>
<td>34800</td>
<td>55</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>73</td>
<td>150</td>
<td>8.00</td>
</tr>
<tr>
<td>16</td>
<td>29</td>
<td>10</td>
<td>na</td>
<td>69600</td>
<td>88</td>
<td>206</td>
<td>2</td>
<td>0</td>
<td>67</td>
<td>185</td>
<td>4.32</td>
</tr>
<tr>
<td>16</td>
<td>29</td>
<td>15</td>
<td></td>
<td>13988</td>
<td>1109</td>
<td>- 7</td>
<td>0</td>
<td>681</td>
<td>378</td>
<td>39.15</td>
<td>5 :00 :00:00</td>
</tr>
<tr>
<td>18</td>
<td>30</td>
<td>5</td>
<td>na</td>
<td>45900</td>
<td>87</td>
<td>234</td>
<td>2</td>
<td>0</td>
<td>229</td>
<td>218</td>
<td>17.43</td>
</tr>
<tr>
<td>18</td>
<td>30</td>
<td>5</td>
<td>p</td>
<td>21494</td>
<td>47</td>
<td>- 0</td>
<td>0</td>
<td>63</td>
<td>220</td>
<td>18.64</td>
<td>5 :00 :00:00</td>
</tr>
</tbody>
</table>
minutes to 43 minutes. As regards the instances which have not been solved with the two algorithms in the time limit, we can remark that the value of the best feasible solution we have found with the path formulation is cheaper than that obtained with the node-arc one and this is for most of the instances.

As for the random problems, for most of the real instances, a significant number of cut inequalities have been generated. It also appears that the cut-cycle inequalities and star-partition inequalities are more effective for this type of instances. This can be explained by the fact that the way the sets $F_e$'s are built, it is easier to find cut-cycle and star-partition configurations. It seems that the node-arc formulation permits to generate more cut-cycle inequalities (72 constraints generated for all the instances against 58 with the path formulation) and the path formulation privileges rather the star-partition inequalities (37 constraints generated against only 22 with the node-arc formulation).

For these instances, we can also observe that the number of flow variables generated for the path formulation is much lower than the one used in the node-arc one. For example, the instance with 16 nodes and 10 commodities has used 69600 flow variables for the first formulation and only 19934 flow variables in the second one.

**Table 3 – Result for random instances for the MCSIPNd problem**

| $|V|^1$ | $|J|^2$ | $|K|^3$ | Algo | FV | NC | NRC | NS | NCC | NEP | NT | o/p | Gap | TT |
|-------|-------|-------|-------|-----|-----|-----|-----|-----|-----|-----|----|-----|-----|-----|
| 6     | 10    | 5     | 1500  | 56.67 | 49.67 | 3.67 | 0.33 | 0.00 | 46.33 | 3/3 | 14.04 | 0.00 :07.70 |
| 6     | 10    | 10    | 3000  | 43.67 | 47.33 | 4.67 | 0.00 | 0.00 | 23.67 | 3/3 | 4.48  | 0.00 :15.00 |
| 6     | 10    | 20    | 6000  | 48.67 | 96.67 | 3.33 | 0.00 | 0.00 | 36.33 | 3/3 | 6.63  | 0.01 :17.19 |
| 6     | 10    | 5     | 550.67 | 85.00 | -     | 3.33 | 0.33 | 0.33 | 52.33 | 3/3 | 13.93 | 0.00 :04.58 |
| 6     | 10    | 10    | 776.67 | 39.57 | -     | 4.67 | 0.33 | 0.00 | 29.67 | 3/3 | 4.54  | 0.00 :02.95 |
| 6     | 10    | 20    | 886.67 | 21.00 | -     | 1.33 | 0.00 | 0.00 | 17.00 | 3/3 | 4.93  | 0.00 :01.59 |
| 6     | 20    | 5     | 3000  | 75.67 | 151.00 | 3.00 | 1.67 | 0.33 | 106.33 | 3/3 | 16.16 | 0.00 :14.04 |
| 6     | 20    | 10    | 3000  | 70.67 | 187.33 | 9.67 | 1.00 | 0.00 | 41.00 | 3/3 | 9.93  | 0.01 :29.15 |
| 6     | 20    | 20    | 12000 | 44.67 | 94.33 | 2.67 | 0.00 | 0.00 | 17.00 | 3/3 | 16.63 | 0.00 :00.00 |
| 6     | 20    | 5     | 812.00 | 55.67 | -     | 3.00 | 2.00 | 0.00 | 93.67 | 3/3 | 15.82 | 0.00 :16.27 |
| 6     | 20    | 10    | 1194.33 | 49.33 | -     | 8.67 | 2.00 | 0.00 | 50.33 | 3/3 | 9.16  | 0.00 :10.58 |
| 6     | 20    | 20    | 1532.33 | 44.33 | -     | 3.67 | 0.00 | 0.00 | 24.33 | 3/3 | 7.58  | 0.00 :08.16 |

Tables 3 and 4 report the results for the MCSIPNd problem (single version). As in the previous experiments related to the MCSIPNd problem, we have considered random and real instances. We have tested our two algorithms on the same instances. Table 3 reports the results for the random instances. As for the multiple case, we have tested three instances for each size, Table 3 gives the average result. Also here we have used the saturation constraints (39) which are valid for this variant of the problem.
We can note that all the instances with 6 and 8 nodes have been solved to optimality with at least one of the algorithms. This is also the case for all the instances with $|F| = 10$ except that with 10 nodes and 20 commodities, and for the instances with $|K| = 5$, except two of the instances with 10 nodes and $|F| = 20$.

Among the 9 instances with 10 nodes and $|F| = 20$, only one has been solved in the time limit of 5 hours. The gap obtained is generally not very high (less than 18%). Also here both algorithms seem to have similar performance. In fact most of the instances solved by one have also been solved by the other.

| $|V|$ | $|F|$ | $|K|$ | Algo | FV | NC | NRC | NS | NCC | NEP | NT | Copt | Gap | TT |
|------|------|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 6    | 11   | 5    | na   | 1650 | 12  | 21  | 0   | 0   | 1   | 3   | 57  | 0.00 | 0:00:06.34 |
| 6    | 11   | 10   | p    | 3300 | 46  | 137 | 1   | 2   | 0   | 79  | 104 | 10.58 | 0:00:58.33 |
| 6    | 11   | 15   | 4950 | 71   | 389 | 1   | 1   | 0   | 75  | 109 | 11.01 | 0:01:39.03 |
| 6    | 11   | 20   | 6600 | 72   | 705 | 1   | 1   | 0   | 191 | 122 | 9.84  | 0:09:39.32 |
| 8    | 17   | 5    | 4150 | 101 | 1   | 0   | 97  | 82  | 12.20 | 0:00:27.73 |
| 8    | 17   | 10   | na   | 9520 | 165 | 955 | 4   | 5   | 77  | 115 | 20.00 | 1:23:37.11 |
| 8    | 17   | 15   | 14280| 320 | 3983| 3   | 8   | 981 | 129 | 22.48 | 4:00:51.52 |
| 8    | 17   | 20   | 19040| 550 | 8849| 3   | 10  | 173 | 122 | 9.84  | 0:00:00.00 |
| 10   | 25   | 5    | 11250| 44   | 135 | 1   | 0   | 31  | 129 | 8.53  | 0:01:45.53 |
| 10   | 25   | 10   | na   | 23400| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 10   | 25   | 15   | 33750| 553 | 1743| 2   | 4   | 291 | 226 | 13.90 | 5:00:00.00 |
| 10   | 25   | 20   | 4500 | 336 | 1087| 2   | 6   | 227 | 261 | 24.52 | 5:00:00.00 |
| 12   | 32   | 5    | na   | 42240| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 12   | 32   | 10   | na   | 42240| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 12   | 32   | 15   | 33750| 553 | 1743| 2   | 4   | 291 | 226 | 13.90 | 5:00:00.00 |
| 12   | 32   | 20   | 4500 | 336 | 1087| 2   | 6   | 227 | 261 | 24.52 | 5:00:00.00 |
| 14   | 27   | 5    | na   | 42240| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 14   | 27   | 10   | na   | 42240| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 16   | 29   | 5    | na   | 42240| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 16   | 29   | 10   | 42240| 82  | 47  | 1   | 0   | 17  | 157 | 3.82  | 0:05:29.97 |
| 18   | 30   | 5    | na   | 45900| 68  | 262 | 1   | 3   | 229 | 271 | 14.69 | 5:00:00.00 |
| 18   | 30   | 5    | p    | 24332| 43  | 1   | 1   | 0   | 77  | 231 | 22.08 | 5:00:00.00 |

Tab. 4 – Results for real instances for the MCSIPNDs problem

The saturation constraints appear in a small proportion with respect to the cut and arc residual capacity inequalities. This does not imply that these inequalities are not necessary for this variant of the problem. On the contrary, they have been quite useful for solving the random instances. Also we notice that for the multiple variant of the problem, the number of generated cut-cycle and star-partition inequalities is also not very significant in this case.

Table 4 presents the results obtained for the MCSIPNDs problem for the real instances. We can remark that in general, the results based on the path formulation, obtained with the Branch-and-Cut-and-Price algorithm, are better than those based on the node-arc formulation. In fact, several
instances which have not been solved in the time limit with the latter formulation, have been solved with the former one, and even in less than 30 minutes for some of them like that with 8 nodes and 20 commodities. Furthermore, for the path formulation based algorithm, the CPU time is much smaller. Also for most of the instances which could not be solved in the time limit, the gap has significantly decreased using the path formulation, as for example for the instance with 12 nodes and 15 commodities, the gap passes from 18.22 to 9.82.

It seems however that the MCSIPND problem in its simple version is easier to solve. In fact, for the random instances and the node-arc formulation, only 11 among the 54 instances could not be solved for the MCSIPND_s problem against 17 for the MCSIPND_m problem. The difference is less important for the path formulation. Here the number of solved instances is almost the same, but the CPU time needed for the MCSIPND_s problem is generally lower.

We can also remark that the optimal solution for the MCSIPND_m problem has a cost lower than that of the MCSIPND_s. This is because each feasible solution for the latter problem is also feasible for the former one.

For several instances presented in the previous tables, we have not obtained an optimal solution. In order to evaluate the performance of our algorithms in providing provably near-optimal solutions in reasonable time, we have noted the value of the best feasible solution after each hour of computation. Table 5 shows the evolution of the current feasible solution for these instances. The different columns of this table represent:

- $|V^1|$: the number of nodes of $G^1$,
- $|\mathcal{F}|$: the number of sets $F_e$,
- $|K|$: the number of demands,
- Algo: the type of algorithm ($na$ (resp. $p$) means that the used algorithm is based on the node-arc (resp. path) formulation.
- Pb: the problem ($m$ represents the multiple version and $s$ the simple one)
- BS: the value of the best founded solution,
- Gap2: the relative error between the best feasible solution and the best lower bound.

| $|V^1|$ | $|\mathcal{F}|$ | $|K|$ | Algo | Pb  | BS   | Gap2 | BS   | Gap2 | BS   | Gap2 | BS   | Gap2 |
|-------|----------|-------|------|-----|------|------|------|------|------|------|------|------|
| 10    | 25       | 15    | na   | m   | 243  | 25.91| 231  | 19.07| 231  | 18.46| 231  | 18.46|
| 10    | 25       | 15    | na   | m   | 218  | 10.66| 218  | 10.10| 216  | 8.54 | 213  | 7.04 |
| 12    | 32       | 10    | na   | m   | 272  | 57.23| 207  | 18.29| 194  | 10.23| 194  | 9.60 |
| 12    | 32       | 10    | na   | m   | 189  | 8.00 | 182  | 2.82 | 182  | 2.25 | 182  | 2.25 |
| 14    | 27       | 10    | na   | m   | 246  | 26.15| 198  | 1.02 | 198  | 1.02 | 198  | 0.51 |
| 16    | 29       | 5     | na   | m   | 202  | 36.49| 159  | 7.33 | 159  | 6.00 | 153  | 3.12 |
| 16    | 29       | 5     | na   | m   | 202  | 36.49| 167  | 12.84| 167  | 12.84| 167  | 12.08|
| 18    | 30       | 5     | na   | m   | 323  | 73.66| 264  | 39.68| 239  | 25.13| 218  | 13.54|
| 18    | 30       | 5     | na   | m   | 222  | 21.98| 222  | 20.65| 222  | 19.35| 222  | 17.46|
| 10    | 25       | 15    | na   | s   | 240  | 21.21| 227  | 13.50| 227  | 12.94| 227  | 12.94|
| 12    | 32       | 10    | na   | s   | 217  | 24.71| 201  | 14.20| 193  | 8.43 | 186  | 3.91 |
| 12    | 32       | 10    | na   | s   | 188  | 5.62 | 182  | 1.68 | 182  | 1.11 | 182  | 1.11 |
| 14    | 27       | 10    | na   | s   | 226  | 17.10| 226  | 15.90| 226  | 13.57| 226  | 13.00|
| 14    | 27       | 10    | na   | s   | 225  | 12.50| 225  | 11.94| 214  | 5.94 | 208  | 2.46 |
| 16    | 29       | 5     | na   | s   | 163  | 7.95 | 163  | 5.84 | 163  | 5.16 | 158  | 1.28 |
| 16    | 29       | 5     | na   | s   | 169  | 14.19| 169  | 11.92| 163  | 7.24 | 163  | 6.54 |
| 18    | 30       | 5     | na   | s   | 241  | 25.52| 211  | 8.21 | 211  | 7.11 | 211  | 6.37 |
| 18    | 30       | 5     | na   | s   | 231  | 20.94| 231  | 20.31| 231  | 19.69| 231  | 19.69|

**Tab. 5 – Unsolved instances**

The first part of the table concerns the MCSIPND_m problem whereas the second one is related to the MCSIPND_s one. The lines are presented in pairs, one for the node-arc formulation and one for the path formulation. When there is only one line, this means that one of the formulation has permitted to find the optimal solution in the time limit.
We remark that for most of the instances, we have a very near-optimal solution. For example, for the MCSIPND$_m$ problem and the instance with 16 nodes, the gap is 1.32%. However for harder instances, the best feasible solution is still distant of the optimal one. For example, the gap for the instance with 18 nodes is near 15% for the MCSIPND$_m$ problem and 6% (resp. 19%) for the MCSIPND$_s$ problem for the node-arc formulation (resp. the path formulation).

We can note that, as time elapsed, the gap decreases. Actually we have observed that the best solution decreases and the global lower bound increases for almost all the instances. A last remark we can give is that a very good feasible solution could be found in 1 or 2 hours. For example, for the instance with 12 nodes the gap is 5.62% (resp. 1.68%) after 1 hour (resp. 2 hours) with the path formulation for the MCSIPND$_s$ problem.

Finally, we present a little real french instance with 10 nodes, $|F| = 25$ and $|K| = 10$. Figure 8 presents the optical network and the set of commodities.

![Optical network](image1.png)

**Fig. 8** – A real french instance with 10 nodes, $|F| = 25$ and 10 commodities

![Commodities](image2.png)

![MCSIPND$_m$ problem](image3.png)

**Fig. 9** – IP network solutions
Figure 9 gives the solutions for the two variants of the problem. The dashed lines represent the edges with capacity 2.5 Gbits and the solid ones the edges with capacity 10 Gbits. For the MCSIPND\(_m\) problem, the number 2 on some edges indicates that we have installed two links between the towns, extremities of the edge. The optimal solutions shown in Figure 9 have been obtained with the Branch-and-Cut-and-Price algorithm based on the path formulation in 43 minutes for the multiple version and in only 10 seconds for the simple one.

7 Concluding remarks

In this paper we have considered the multilayer survivable network design problem which has applications to the design of reliable IP-over-optical network. We have considered the capacity dimensioning of the network. We have proposed two integer programming formulations for each of the two variants of the problem, simple and multiple. We have identified some valid inequalities, and described necessary conditions and sufficient conditions for a class of inequalities to define facets. Using this, we have developed Branch-and-Cut and Branch-and-Cut-and-Price algorithms for the problems and presented extensive computational results. These ones show that the path formulation based algorithm performs better than the one based on the node-arc formulation. The experimental results also show the effectiveness of the capacity demand cut, the cut-cycle and the star-partition inequalities for the problem.

Other variants of the multilayer network design problem are of interest for telecommunication operators and merit to be investigated. In particular those in which IP and optical layers should be treated simultaneously. This is our direction of future research.

Also a more general variant of the problem, which merits to be studied, is when bounds are considered on the paths of the IP network. This issue has been treated in the literature in the monolayer case [12, 15, 22, 23]. An other interesting question would be to consider integer flows. This concept has also been investigated for the monolayer networks but without considering survivability [6, 10].

Références


Appendix : Proof of Theorem 4.9

Proof. We use ideas similar to those developed in [31]. Let us denote inequality (33) by

\[ a^1x^1 + a^2x^2 \geq \alpha, \]  

(44)

and let

\[ b^1x^1 + b^2x^2 + \lambda f \geq \beta, \]  

(45)

be a facet defining inequality of MCSIPND\(^{\alpha}\)\(_m\)\((G, F, K)\) such that the face defined by (44) is contained in that defined by (45). Let \( L = \{(x^1, x^2, f) \in \text{MCSIPND}^{\alpha}\(_m\)\((G, F, K)\) | a^1x^1 + a^2x^2 = \alpha\} \).

We first construct a feasible solution which satisfies (44) with equality. For each commodity \( k \) such that \( o_k \) and \( d_k \) are in \( W \), as \( G(W) \) is \( F \)-connected, for all \( j \in \{1, \ldots, t\} \), there exists a path \( P^k_j \) fully contained in \( G_j(W) \), which connects \( o_k \) and \( d_k \). If we install \( \left\lceil \frac{x^o_k}{10} \right\rceil \) high capacities on each edge which belongs to at least one path \( P^k_j \), then we can send a flow of \( \omega_k \) along the path \( P^k_j \) from \( o_k \) to \( d_k \) for each failure \( j \). By adding successively, in a similar way, the necessary capacity for all commodities, we obtain a feasible dimensioning for the edges of \( E(W) \) according to the commodities of \( W \).

For the edges in \( E(\overline{W}) \), we similarly associate, for each commodity \( k \) such that \( o_k \) and \( d_k \in \overline{W} \) and for each \( j \in \{1, \ldots, t\} \), a path \( \overline{P}^k_j \) between \( o_k \) and \( d_k \). We install capacities on these paths in a similar way as in \( G(W) \).

Now, consider an edge \( u_iv_i \) of \( \delta_{G_i}(W) \) such that \( u_i \in W \) and \( v_i \in \overline{W} \). Let \( I_i = \{j \in \{1, \ldots, t\} \setminus \{i\} | u_iv_i \in F_j\} \). Then for all \( j \in I_i \), there exists an edge, say \( u_jv_j \), in \( \delta(W) \cap F_i \) such that \( u_jv_j \notin F_j \). In fact, if this is not the case, then there would exist \( j \in I_i \) such that \( \delta(W) \cap F_i \subseteq \delta(W) \cap F_j \). As by definition of \( I_i \), \( u_iv_i \in F_j \setminus F_i \), we would have \( \delta(W) \cap F_i \subseteq F_j \). But this is a contradiction with Condition 3) of Theorem 4.8 (see Figure 10).

\[ \begin{array}{c}
\text{W} \\
\text{Q}_i \\
\text{Q}_j \\
\text{W} \\
\hline
\end{array} \quad \begin{array}{c}
\text{W} \\
\text{Q}_i \\
\text{Q}_j \\
\text{W} \\
\hline
\end{array} \]

\[ \text{Fig. 10} \]

Let \( k \in \gamma(W) \) be a commodity across the cut. As the graph \( G(W) \) (resp. \( G(\overline{W}) \)) is \( F \)-connected, for all \( j \in \{1, \ldots, t\} \setminus I_i \), there exists a path \( Q^k_j \) (resp. \( \overline{Q}^k_j \)) between \( o_k \) and \( u_i \) (resp. \( v_i \) and \( d_k \)) in \( G_j(W) \) (resp. \( \overline{G}_j(W) \)). For \( j \in I_i \), by the previous remark, there is an edge \( u_jv_j \) of \( (\delta(W) \cap F_i) \setminus F_j \). We may suppose that \( u_j \in W \) and \( v_j \in \overline{W} \). For the commodity \( k \), and for all \( j \in I_i \), similarly, there is a path \( Q^k_j \) (resp. \( \overline{Q}^k_j \)) between \( o_k \) and \( u_j \) (resp. \( v_j \) and \( d_k \)) in \( G_j(W) \) (resp. \( \overline{G}_j(W) \)).

Now, we can complete the partial dimensioning already performed in \( W \) and \( \overline{W} \). We install for commodity \( k \), \( \left\lceil \frac{\omega_k}{10} \right\rceil \) high capacities on each edge that belongs to at least one path \( Q^k_j \), \( \overline{Q}^k_j \), \( j = 1, \ldots, t \). We then add successively these capacities for each commodity of \( \gamma(W) \). These new capacities, added to those already installed in \( W \) and \( \overline{W} \), permit to have a feasible dimensioning for all the commodities and all the edges of \( E(W) \cup E(\overline{W}) \). This dimensioning of the edges of
The solution \( P \) can be given as follows,

\[
\begin{align*}
    x_{uv}^1 &= 0 \quad \text{for all } uv \in E(W) \cup E(\overline{W}), \\
    x_{uv}^2 &= \sum_{k \mid \alpha_k, \delta_k \in W} \left[ \frac{\omega_k}{10} \right] + \sum_{k \in \gamma(W)} \left[ \frac{\omega_k}{10} \right] \quad \text{for all } uv \in E(W), \\
    x_{uv}^2 &= \sum_{k \mid \alpha_k, \delta_k \in W} \left( \frac{\omega_k}{10} \right) + \sum_{k \in \gamma(W)} \left( \frac{\omega_k}{10} \right) \quad \text{for all } uv \in E(\overline{W}).
\end{align*}
\] (46)

As \( D_W \geq 4 \), for the edges of \( \delta(W) \), we can consider the following dimensioning,

\[
\begin{align*}
    x_{uv}^1_{u,v_j} &= D_W \mod 4 \quad \text{for all } j \in I_i \cup \{ i \}, \\
    x_{uv}^1_{u,v} &= 0 \quad \text{for all } uv \in \delta(W) \setminus \bigcup_{j \in I_i \cup \{ i \}} \{ u_j v_j \}, \\
    x_{uv}^2_{u,v_j} &= \left\lfloor \frac{D_W}{4} \right\rfloor \quad \text{for all } j \in I_i \cup \{ i \}, \\
    x_{uv}^2_{u,v} &= 0 \quad \text{for all } uv \in \delta(W) \setminus \bigcup_{j \in I_i \cup \{ i \}} \{ u_j v_j \}. \quad \text{(47)}
\end{align*}
\]

Also consider the flows on the paths \( P_j^k, \overline{P}_j^k, Q_j^k, \overline{Q}_j^k, j = 1, \ldots, t \), given by

\[
\begin{align*}
    f_{uv}^{k,i} &= \omega_k \quad \text{for all } k \text{ such that } \alpha_k, \delta_k \in W \text{ (resp. } \alpha_k, \delta_k \in \overline{W} \text{)}, \\
    f_{uv}^{k,i} &= \omega_k \quad \text{for all } k \in \gamma(W) \text{ and } uv \text{ traversed from } u \text{ to } v \text{ in } P_j^k \text{ (resp. } \overline{P}_j^k \), \\
    f_{uv}^{k,i} &= \omega_k \quad \text{for all } k \in \gamma^+(W) \text{ (resp. } \gamma^-(W) \), \\
    f_{uv}^{k,i} &= 0 \quad \text{otherwise.}
\end{align*}
\] (48)

The solution \((x^1, x^2, f)\) given by (46), (47) and (48) is feasible for \( \text{MCISPND}^{na}_n(G, \mathcal{F}, K) \).

Let \( pq \in E(W) \cup E(\overline{W}) \cup (F_i \cap \delta(W)) \). And let \((x'^1, x'^2, f')\) be the solution such that

\[
\begin{align*}
    x_{uv}^{1'} &= x_{uv}^1 \quad \text{for all } uv \neq pq, \\
    x_{uv}^{1pq} &= x_{uv}^1 + 1, \\
    x_{uv}^{2} &= x_{uv}^2, \\
    f' &= f.
\end{align*}
\]

Clearly, \((x'^1, x'^2, f')\) is feasible for \( \text{MCISPND}^{na}_n(G, \mathcal{F}, K) \). Furthermore, \((x^1, x^2, f)\) and \((x'^1, x'^2, f')\) satisfy inequality (44) with equality. In consequences, both solutions satisfy inequality (45) with equality. This yields \( b_{pq}^1 = 0 \). Similarly, one can show that \( b_{pq}^2 = 0 \). As \( pq \) is an arbitrary edge of \( E(W) \cup E(\overline{W}) \cup (F_i \cap \delta(W)) \), we then have \( b_{uv}^1 = b_{uv}^2 = 0 \) for all \( uv \in E(W) \cup E(\overline{W}) \cup (F_i \cap \delta(W)) \).

Now, consider the edge \( u_iv_i \) (introduced above) and the solution \((\tilde{x}^1, \tilde{x}^2, \tilde{f})\) such that

\[
\begin{align*}
    \tilde{x}_{u_iv_i}^1 &= x_{u_iv_i}^1 + 4, \\
    \tilde{x}_{uv}^1 &= x_{uv}^1, \\
    \tilde{x}_{u_iv_i}^2 &= x_{u_iv_i}^2 - 1, \\
    \tilde{x}_{uv}^2 &= x_{uv}^2, \\
    \tilde{f} &= f.
\end{align*}
\]

The solution \((\tilde{x}^1, \tilde{x}^2, \tilde{f})\) is defined from \((x^1, x^2, f)\) by replacing a high capacity on \( u_iv_i \) by 4 small ones. Hence, this solution is feasible. Furthermore, as \((\tilde{x}^1, \tilde{x}^2, \tilde{f})\) satisfies (44) with equality, it also
satisfies (45) with equality. This implies that \( 4b^1_{uv,i} = b^2_{uv,i} \). As \( u_i v_i \) is an arbitrary edge of \( \delta_G(W) \), we then get
\[
b^2_{uv} = 4b^1_{uv} \text{ for all } uv \in \delta(W) \backslash F_i.
\]

Consider again an edge \( pq \in E(W) \cup E(\overline{W}) \cup F_i \) and the solution \( (x^1, x^2, f) \) introduced above. Let \( k' \in K \) be a commodity and \( j' \in \{1, \ldots, t\} \) a failure. Consider \( (\hat{x}^1, \hat{x}^2, \hat{f}) \) given by
\[
\hat{x}^1_{uv} = x^1_{uv}, \quad \hat{x}^2_{uv} = x^2_{uv}, \quad \hat{f}_{pq}^{k,j'} = f_{pq}^{k,j'} + \varepsilon, \\
\hat{f}_{qp}^{k,j'} = f_{qp}^{k,j'} + \varepsilon, \quad \hat{f}_{uv}^{k,j} = f_{uv}^{k,j} \text{ for all } uv \in E \backslash \{pq\}, \text{ for all } k \in K, \text{ for all } j \in \{1, \ldots, t\},
\]
where \( 0 < \varepsilon < \frac{1}{4} \). Note that as the capacity of edge \( pq \) is sufficiently big, we have a residual capacity to carry more flow. In consequences, \((\hat{x}^1, \hat{x}^2, \hat{f})\) is feasible. Moreover, this solution satisfies inequality (44), and hence (45), with equality. As \((x^1, x^2, f)\) also satisfies (45) with equality, we obtain \( \lambda^{k,j'}_{pq} \varepsilon + \lambda^{k,j'}_{qp} = 0 \) implying that \( \lambda^{k,j'}_{pq} = -\lambda^{k,j'}_{qp} \). As \( pq \) is an arbitrary edge of \( E(W) \cup E(\overline{W}) \cup F_i \), and \( k' \) and \( j' \) are arbitrary in \( K \) and \( \{1, \ldots, t\} \), respectively, we obtain
\[
\lambda^{k,j}_{uv} = -\lambda^{k,j}_{vu} \text{ for all } uv \in E(W) \cup E(\overline{W}) \cup F_i, \text{ for all } k \in K \text{ and all } j \in \{1, \ldots, t\}.
\]

Now consider again the edge \( u_i v_i \) of \( \delta(W) \backslash F_i \). By Condition 5) of Theorem 4.8, if we consider the solution \((x^1, x^2, f)\), as \( u_i v_i \) has a sufficiently big capacity, we can add more flow on this edge. In a similar way we can show that \( \lambda^{k,j}_{u_i v_i} = -\lambda^{k,j}_{v_i u_i} \). As \( u_i v_i \) is chosen arbitrarily in \( \delta_G(W) \), like \( k' \) and \( j' \) in \( K \) and \( \{1, \ldots, t\} \), we obtain
\[
\lambda^{k,i}_{uv} = -\lambda^{k,i}_{vu} \text{ for all } uv \in \delta_G(W), \text{ for all } k \in K \text{ and all } j \in \{1, \ldots, t\}.
\]

We have then shown that
\[
\lambda^{k,i}_{uv} = -\lambda^{k,i}_{vu} \text{ for all } uv \in E, \text{ for all } k \in K \text{ and all } j \in \{1, \ldots, t\}. \tag{49}
\]

We now show that \( \sum_{k \in K} \sum_{uv \in E} (\lambda^{k,i}_{uv} f^{k,j}_{uv} + \lambda^{k,i}_{vu} f^{k,j}_{vu}) \) is a constant for all failure \( j = 1, \ldots, t \), by showing that the sum of the coefficients, corresponding to any cycle in the network, equals zero. Let \( \Delta \) denote the set of cycles in \( D = (V, A) \). Consider a failure \( j \in \{1, \ldots, t\} \) and a particular cycle \( \xi \in \Delta \). Let \( \lambda^s_{\xi} = \sum_{(u,v) \in \xi} \lambda^{k,i}_{uv} \). We will show that \( \lambda^s_{\xi} = 0 \) for all cycle \( \xi \in \Delta \). Call \( \xi \) an \( s \)-intersection cycle with respect to the cut \( \delta_G(W) \) if \( \xi \) contains exactly \( s \) arcs of \( \delta_G(W) \). Note that \( s \) must be even as \( \xi \) is a cycle.

If \( \xi \) is a 0-intersection cycle, then \( \xi \) is completely contained in \( W \) or \( \overline{W} \). We suppose w.l.o.g., that \( \xi \) is in \( W \). Let \( k \in K \) and \((\hat{x}^1, \hat{x}^2, \hat{f})\) be the solution given by
\[
\hat{x}^1_{uv} = x^1_{uv} + 1 \quad \text{for all } uv \in E(W) \text{ such that } (u, v) \text{ or } (v, u) \in \xi, \\
\hat{x}^2_{uv} = x^2_{uv} \quad \text{for all } uv \in E(W) \text{ such that } (u, v) \text{ or } (v, u) \notin \xi, \\
\hat{f}_{uv}^{k,j} = f^{k,j}_{uv} + 1 \quad \text{for all } (u, v) \in \xi, \\
\hat{f}_{uv}^{k,j} = f^{k,j}_{uv} \quad \text{for all } (u, v) \notin \xi, \\
\hat{f}_{uv} = f_{uv} \quad \text{for all } (u, v) \in A, \text{ for all } h \in \{1, \ldots, t\} \backslash \{j\}.
\]

Here, we construct solution \((\hat{x}^1, \hat{x}^2, \hat{f})\) from solution \((x^1, x^2, f)\) by installing a small capacity on all the edges in \( \xi \) and by sending an additional unit of flow on \( \xi \). Solution \((\hat{x}^1, \hat{x}^2, \hat{f})\) is still feasible and satisfies (44) with equality. Hence \((\hat{x}^1, \hat{x}^2, \hat{f})\) satisfies (45) with equality, and in consequence we get
\[
\sum_{uv \in E(W)} b_{uv}^1 x_{uv}^1 - \sum_{uv \in E(W)} b_{uv}^2 x_{uv}^1 + \sum_{(u,v) \in \xi} (\lambda^{k,j}_{uv} f^{k,j}_{uv} - \lambda^{k,j}_{vu} f^{k,j}_{vu}) = 0.
\]
As \( b_{uv}^1 = 0 \) for all \( uv \in E(W) \), we obtain \( \sum_{(u,v) \in \xi} -\lambda_{uv}^{k,j} = 0 \), and then \( \lambda_{\xi}^{k,j} = 0 \). As cycle \( \xi \), the commodity \( k \) and the failure \( j \) are arbitrary, we have then shown that

\[
\lambda_{\xi}^{k,j} = 0 \quad \text{for all 0-intersection cycle } \xi \in \Delta, \text{ for all } k \in K \\
\text{and for all } j \in \{1, \ldots, t\}. \tag{50}
\]

Now suppose that \( \xi \) is a 2-intersection cycle. Assume that \((p,\bar{p})\) and \((q,\bar{q})\) are the cutset arcs belonging to \( \delta_G(W) \cap \xi \) with \( p,q \in W \) and \( \bar{p},\bar{q} \in \overline{W} \).

If \( p = q \) and \( \bar{p} = \bar{q} \), then \( \xi \) can be decomposed into three cycles: a 0-intersection cycle denoted by \( \xi_1 \) contained in \( W \), a 0-intersection cycle denoted by \( \xi_2 \) contained in \( \overline{W} \) and a cycle formed by the arcs \((p,\bar{p})\) and \((\bar{p},p)\) = \((\bar{q},q)\).

We then have \( \lambda_{\xi}^{k,j} = \lambda_{\xi_1}^{k,j} + \lambda_{\xi_2}^{k,j} + \lambda_{pp}^{k,j} + \lambda_{\bar{p}\bar{q}}^{k,j} \).

As \( \xi_1 \) and \( \xi_2 \) are 0-intersection cycles, \( \lambda_{\xi_1}^{k,j} = \lambda_{\xi_2}^{k,j} = 0 \) and by (49), we have \( \lambda_{pp}^{k,j} = -\lambda_{\bar{p}\bar{q}}^{k,j} \).

Thus \( \lambda_{\xi}^{k,j} = 0 \).

Now suppose that \( pp \neq qq \) and one of the two edges \( pp \) or \( qq \) belongs to \( F_i \). We suppose, w.l.o.g., that \( qq \in F_i \) and \( pp \notin F_i \). Let \( pp \) be the edge which belongs to \( \delta_G(W) \). One can construct a feasible solution \((\tilde{x}^1,\tilde{x}^2,\tilde{f})\) in a similar way as \((x^1,x^2,f)\) (given above for the edge \( uv \)).

Remark that \( \tilde{x}_{pp}^1 = D_W \mod 4 \) and \( \tilde{x}_{pp}^2 = \left[ \frac{D_W}{4} \right] - 1 \), Consider the solution obtained from \((\tilde{x}^1,\tilde{x}^2,\tilde{f})\) by installing a small capacity on all the edges of \( \xi \cap (E(W) \cup E(\overline{W}) \cup F_i) \), and by sending \( \varepsilon \) units of flow on \( \xi \) (for a certain \( \varepsilon > 0 \)). The additional flow is possible since by Condition 5) of Theorem 4.8, there is a positive residual capacity on the edge \( pp \). This new solution is feasible. As both solutions satisfy (44) with equality and hence (45), we get \( \lambda_{\xi}^{k,j} = 0 \).

If \( pp = qq \) and \( pp,qq \notin F_i \), we define a solution \((\tilde{x}^1,\tilde{x}^2,\tilde{f})\) by considering for the edges of the cut \( \delta_G(W) \) the following values

\[
\tilde{x}_{pp}^1 = D_W \mod 4, \\
\tilde{x}_{uv}^1 = 0 \quad \text{for all } uv \in \delta_G(W), \\
\tilde{x}_{pp}^2 = \left[ \frac{D_W}{4} \right] - 1, \\
\tilde{x}_{qq}^2 = 1, \\
\tilde{x}_{uv}^2 = 0 \quad \text{for all } uv \in \delta_G(W) \setminus \{pp,qq\}.
\]

For the dimensioning of \( E(W) \) and \( E(\overline{W}) \), we suppose that we have installed a sufficient capacity which permits to carry the flows of the commodities in \( W \) and \( \overline{W} \) on the paths fully contained in \( G(W) \) and \( \overline{G(W)} \). These capacities can be taken as big as we want. So, by Condition 5) of Theorem 4.8, the commodities of \( \gamma(W) \) can be routed in such a way that \( pp \) and \( qq \) contain at least \( \varepsilon \) (for a certain \( \varepsilon > 0 \)) units of residual capacity. This solution is feasible and satisfies (44) and then (45) with equality. Using the residual capacity \( \varepsilon \) on \( pp \) and \( qq \), one can obtain a new feasible solution from \((\tilde{x}^1,\tilde{x}^2,\tilde{f})\) by adding an additional flow \( \varepsilon \) along \( \xi \) for commodity \( k \), and installing a small capacity on the edges of \( \xi \) not in \( \delta_G(W) \). This new solution is also feasible and satisfies (44) and (45) as equalities. As commodity \( k \in K \), the failure \( j \in \{1, \ldots, t\} \) and the 2-intersection cycle \( \xi \in \Delta \) are arbitrary, we have that

\[
\lambda_{\xi}^{k,j} = 0 \quad \text{for all 2-intersection cycle } \xi \in \Delta, \text{ for all } k \in K \\
\text{and for all } j \in \{1, \ldots, t\}. \tag{51}
\]

Now consider an arbitrary \( s \)-intersection cycle \( \xi \). Let \( k \) be a commodity and \( j \) be a failure. Let \( \xi \) be the cycle given by \( \{(s_1,s_2), (s_2,s_3), \ldots, (s_T,s_1)\} \) with \( s_1 \in W \). Let \( (s_{t_1},s_{t_2}) \) be the first arc of the cycle \( \xi \) that crosses \( \delta(W) \) and \( (s_{t_3},s_{t_4}) \) the first subsequent arc that re-enters set \( W \). Note that cycle \( \xi' = \{(s_{t_1},s_{t_3}), \ldots, (s_{t_3},s_{t_4}), (s_{t_4},s_{t_1})\} \) is a 2-intersection cycle. We then have \( \lambda_{\xi'}^{k,j} = \lambda_{\xi}^{k,j} \).

So, we can replace the path \( \xi' \setminus (s_{t_1},s_{t_4}) \) by the arc \( (s_{t_1},s_{t_4}) \). Repeating this argument, one can construct a 0-intersection cycle \( \psi \) that satisfies \( \lambda_{\psi}^{k,j} = \lambda_{\psi}^{k,j} \).

As \( k \) and \( j \) are arbitrary, we obtain that

\[
\lambda_{\xi}^{k,j} = 0 \quad \text{for all cycle } \xi \in \Delta, \text{ for all } k \in K \\
\text{and for all } j \in \{1, \ldots, t\}. \tag{52}
\]

38
We have shown that $\sum_{k \in K} \sum_{uv \in E} (\lambda_{uv}^k f_{uv}^k + \lambda_{uv}^k f_{uv}^k) + \lambda_{uv}^k f_{uv}^k$ is a constant, say $\lambda_j$, for any failure $j \in \{1, \ldots, t\}$.

We can now show that $b^1_{uv} = \rho^1$ and $b^2_{uv} = \rho^2$ for all $uv \in \delta_G(i, W)$. Consider the solution $(x^1, x^2, f)$ given in the beginning of the proof, an edge $pq \in \delta_G(W)$ such that $pq \neq u_i v_i$, a commodity $k' \in K$ and a failure $j \in \{1, \ldots, t\}$. Let $P(u_i, p)$ be a path from node $u_i$ to $p$ fully contained in $W$ and $P(q, v_i)$ a path from $q$ to $v_i$ in $\overline{W}$. Consider the solution $(\tilde{x}^1, \tilde{x}^2, \tilde{f})$ such that

$$\begin{align*}
\tilde{x}^1_{uv} &= x^1_{uv} - 1 = D_W \mod 4 - 1, \\
\tilde{x}^2_{uv} &= x^2_{uv} + 1 & \text{for all } uv \in \delta_G(W) \setminus \{u_i v_i, pq\}, \\
\tilde{x}^1_{pq} &= 1, & \text{for all } (u, v) \in P(u_i, p) \cup P(q, v_i), \\
\tilde{x}^2_{uv} &= x^2_{uv} & \text{for all } (u, v) \in E(W) \cup E(\overline{W}) \setminus (P(u_i, p) \cup P(q, v_i)),
\end{align*}$$

Define the flows as follows.

$$\begin{align*}
\tilde{f}^k_{uv} &= f^k_{uv} + 1 & \text{for all } (u, v) \in P(u_i, p) \cup P(q, v_i), \\
\tilde{f}^k_{pq} &= f^k_{pq}, & \text{for all } uv \in E \setminus (P(u_i, p) \cup P(q, v_i) \cup \{u_i v_i, pq\}), \\
\tilde{f}^k_{uv} &= f^k_{uv} & \text{for all } k \in K \setminus \{k'\}, \text{ for all } h \in \{1, \ldots, t\} \setminus \{j\}.
\end{align*}$$

Solutions $(x^1, x^2, f)$ and $(\tilde{x}^1, \tilde{x}^2, \tilde{f})$ are feasible and satisfy constraint (44) with equality. In consequence, they also satisfy (45) with equality. This implies that

$$\begin{align*}
b^1_{uv} - b^1_{pq} - \sum_{uv \in P(u_i, p) \cup P(q, v_i)} \rho^1_{uv} - \sum_{(u, v) \in P(u_i, p)} \lambda^k_{uv} - \sum_{(u, v) \in P(q, v_i)} \lambda^k_{uv} + \lambda^k_{uv} - \lambda^k_{pq} = 0.
\end{align*}$$

As $b^1_{uv} = 0$ for all $uv \in E(W) \cup E(\overline{W})$ and by (49) $\lambda^k_{uv} = \lambda^k_{uv}$, we have

$$\begin{align*}
b^1_{uv} - b^1_{pq} - \sum_{uv \in P(u_i, p)} \lambda^k_{uv} - \lambda^k_{uv} - \sum_{uv \in P(q, v_i)} \lambda^k_{uv} - \lambda^k_{uv} = 0.
\end{align*}$$

As $P(u_i, p), pq, P(q, v_i)$ and $v_i u_i$ form a cycle, we have $b^1_{uv} - b^1_{pq} = 0$. As $pq$ was chosen arbitrarily, we obtain $b^1_{uv} = \rho^1$ for all $uv \in \delta_G(i, W)$ for $\rho^1 \in IR$. Since $b^2_{uv} = 4b^1_{uv}$ for all $uv \in \delta_G(i, W)$, we have also $b^2_{uv} = 4\rho^1$ for all $uv \in \delta_G(i, W)$.

Thus, inequality (45) is equivalent to

$$\begin{align*}
\rho^1 x^1(\delta_G(i, W)) + 4\rho^1 x^2(\delta_G(i, W)) + \frac{t}{\rho^1} \sum_{j=1}^l \lambda_j = \lambda,
\end{align*}$$

which implies that

$$\begin{align*}
\rho^1 x^1(\delta_G(i, W)) + 4\rho^1 x^2(\delta_G(i, W)) = \lambda',
\end{align*}$$

where $\lambda' = \lambda - \sum_{j=1}^l \lambda_j$. As the face defined by inequality (45) is not empty, $\rho^1 \neq 0$ and we obtain that

$$\begin{align*}
x^1(\delta_G(i, W)) + 4x^2(\delta_G(i, W)) = \frac{\lambda'}{\rho^1} = D_W.
\end{align*}$$

\qed