Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

SEBASTIAN ORLOWSKI
MICHAŁ PIÓRO
ARTUR TOMASZEWSKI
ROLAND WESSÄLY

SNDlib 1.0–Survivable Network Design Library

ZIB-Report 07-15 (June 2007)

SNDlib 1.0–Survivable Network Design Library*

S. Orlowski[†] M. Pióro[‡] A. Tomaszewski[‡] R. Wessäly[†§]

June 2007

Abstract

We provide information on the Survivable Network Design Library (SNDlib), a data library for fixed telecommunication network design that can be accessed at `http://sndlib.zib.de`. In version 1.0, the library contains data related to 22 networks which, combined with a set of selected planning parameters, leads to 830 network planning problem instances. In this paper, we provide a mathematical model for each planning problem considered in the library and describe the data concepts of the SNDlib. Furthermore, we provide statistical information and details about the origin of the data sets.

Keywords: telecommunication network design, data library, optimization
MSC classification (2000): 90B18, 90C11, 90C35

1 Introduction

In the field of telecommunication network design problems, proposed solution approaches range from (integer) linear programming models and corresponding algorithms such as branch-and-bound, row and column generation, or Lagrangian relaxation, to meta-heuristics such as evolutionary algorithms, simulated annealing and tabu search. For drawing credible conclusions on the competitiveness of a model or an algorithm, it is indispensable to have a set of representative and challenging test instances on which the new approach can be tested and compared with previous ones. Unfortunately, such comparisons are rare in the network design literature, which is partially due to unavailability of reference data sets. This is in contrast to other research areas where libraries of standardized benchmark instances are available. Examples are MIPLIB [1] for mixed-integer linear programs, TSPlib [2] for the Traveling Salesman Problem, SteinLib [3] for Steiner tree problems, FAP Web [4] for frequency assignment problems in GSM mobile phone networks.

Since December 2005, the Survivable Network Design Library (SNDlib), a library of standardized test instances for the design of survivable fixed telecommunication networks, has been available online at `http://sndlib.zib.de`. The instances, together with the best known solutions and dual bounds, can be viewed and downloaded from the SNDlib website in various formats.[1] In addition, the SNDlib contains a collection of references to relevant publications in network design in the form of BiBTeX entries, a list of conferences related to this area, and a mailing list.

SNDlib aims at becoming a future standard for testing and demonstrating effectiveness of new optimization models and algorithms.

1.1 Evolution of the library

Given the enormous variety of network design problems, we intend to extend the library step by step. In the first version, we have deliberately limited the scope of design problems to single-layer networks. The test instances are specified in a technology-independent way in terms of an abstract network model composed

*Supported by the German ministry of science (BMBF) project Eibone.
[†]Zuse Institute Berlin (ZIB), Takustr. 7, 14195 Berlin, Germany, `orlowski@zib.de`
[‡]Institute of Telecommunications, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland, {`mpp,artur`}`@tele.pw.edu.pl`
[§]atesio GmbH, Sophie-Taeuber-Arp-Weg 27, 12205 Berlin, Germany, `wessaely@atesio.de`
[1]XML, GML, and a native ASCII format.

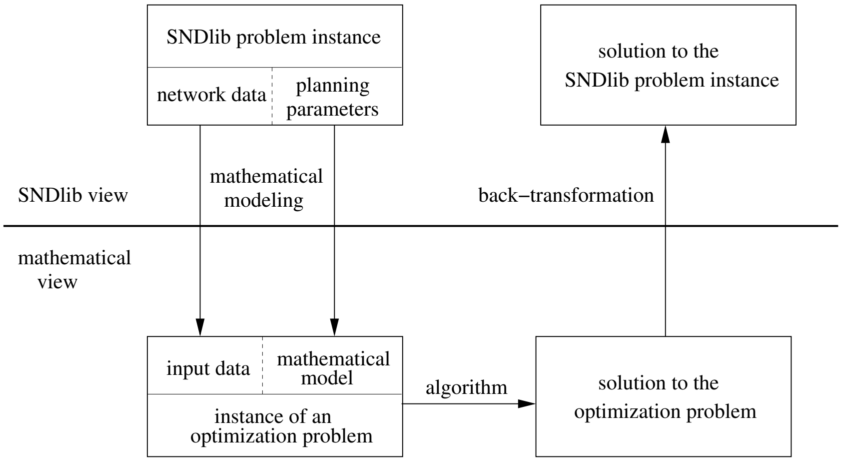

Figure 1: SNDlib-based planning process

of nodes, links, demands, and a set of installable capacity modules with corresponding cost parameters. In addition, certain problem parameters are given, e.g., whether links are directed or undirected, or which routing model and survivability concept should be used. Provided that this first version of the SNDlib is accepted by the network design community, we think of various extensions in subsequent phases, such as adding different types of multi-X (-hour, -period, -layer, -service, . . .) design problems, taking into account shortest-path routing constraints, including the cost of hardware, etc. This will mainly depend on the needs of the community and the availability of the appropriate data.

The current set of test instances is fixed until the next official version of the library. In future versions, new instances might be included and others might be removed from the official test set. Old instances will be always available on the website for reference.

The lists of solutions, dual bounds, conferences, and bibliography entries are being constantly updated. Users of the SNDlib are cordially invited to contribute to the library by submitting solutions, dual bounds, or references to new papers.

Structure of the paper The purpose of this paper is to serve as an official reference for a mathematical definition of the SNDlib problems, and to present the problem instances contained the library together with their background and statistical information. Section 2 describes how the data and different variants of network design problems are organized within the SNDlib. Section 3 provides a mathematical model that serves as a defining reference for each considered problem variant. Section 4 presents an overview of the problem instances included in the SNDlib, and the technological background of the networks. We conclude with some remarks in Section 5.

2 SNDlib data concepts

SNDlib provides benchmark instances of network planning problems together with their solutions and dual bounds. Similarly to the approach adopted by several modelling languages for mixed-integer programming like Zimpl [5] or AMPL [6], models and data are separated from each other in the SNDlib. An SNDlib network planning problem description consists of two parts:

- an *SNDlib network* part (also called *SNDlib network instance*) describing nodes, links, demands, capacities, cost, and other planning data, and
- an *SNDlib model* part (also called *SNDlib model instance*) specifying design parameters, e.g., whether links are directed or undirected, whether routing of a demand may be split on several paths or not, or which capacity model or survivability concept should be used.

A particular SNDlib network can usually be combined with various SNDlib models, leading to different instances of *SNDlib* network planning *problems*. For example, a given network might be a part of a planning problem without survivability and of another one with 1+1 protection, or of one with a bifurcated multi-commodity flow routing and of another one with a single-path routing. An *SNDlib solution* to a particular SNDlib problem instance consists of a routing of the demands specified in the SNDlib network and a suitable capacity installation on the links that satisfy the side constraints specified in the SNDlib model. Examples of an SNDlib network instance, an SNDlib model instance, and a corresponding SNDlib solution are given at the end of Sections 2.1, 2.2, and 2.3, respectively.

The SNDlib problems and their solutions are independent of any mathematical model or algorithm. To solve such a problem, a suitable *mathematical model* has to be chosen that reflects the design parameters specified in the SNDlib model. Often the same SNDlib model can be expressed with several mathematical models, like an edge-flow formulation or an equivalent path-flow formulation. Such a mathematical model combined with input data derived from an SNDlib network forms a particular problem instance in the mathematical sense, as illustrated in the left part of Figure 1. After solving the resulting mathematical optimization problem, its solution has to be transformed back into an SNDlib solution, as shown in the right part of Figure 1. The data provided by the library belongs to the upper part of the figure, whereas users of the library are invited to fill in the lower part by investigating different mathematical models and solution methods.

The following section informally characterizes network data and model parameters of SNDlib. A precise mathematical formulation for each of the SNDlib models is given in Section 3, serving both as an example formulation and as a definition of the sets of feasible and optimal solutions of the SNDlib problem. Examples and a detailed definition of the I/O formats can be found on the SNDlib website `http://sndlib.zib.de`.

2.1 SNDlib network

SNDlib network describes the structure of a network, the traffic to be routed, and the set of admissible routing paths. The network structure is defined as the sets of nodes and links. The set of links defines potential connections between the nodes that may be used to carry traffic; parallel links are allowed. For every link, capacity and cost information contains the amount of pre-installed capacity and its cost, and a list of capacity modules with associated cost, which can be installed on that link. Additionally, fixed-charge cost of using the link is defined, as well as flow cost incurred by routing traffic through that link.

The set of communication demands describes the traffic to be routed. Each demand is characterized by its end-nodes and the volume of traffic that has to be routed through the network expressed in multiples of some base routing unit. The routing unit of a demand defines the amount of link capacity consumed by one unit of the demand's traffic (corresponding, for instance, to 2 Mbit/s, 155 Mbit/s, or 2.5 Gbit/s).

Finally, the set of admissible routing paths is either defined as an empty list, meaning that every possible path is admissible, or a non-empty set of paths is specified for each demand. A hop limit may be imposed for each demand, i.e., a maximum number of links for each admissible routing path. If the SNDlib model specifies to use a particular hop limit, it further restricts the set of admissible paths.

Example of the SNDlib network Consider a three-node network EXAMPLE representing an SDH transport network. For the purpose of this example, assume that point-to-point demand requests are given as an integer multiple of a base *routing unit* of 2 Mbit/s or 155 Mbit/s. Link bandwidth can be installed in integer multiples of 155 Mbit/s or 622 Mbit/s, respectively. Due to the need for overhead bandwidth required for monitoring purposes, if a bitrate of 2 Mbit/s corresponds to 1 unit of capacity, then 155 Mbit/s correspond to 63 units and 622 Mbit/s correspond to 252 units.

An example network might be specified as follows (in the SNDlib native ASCII format):

```
NODES (
  # Format: name (longitude latitude)
  A ( 11.49 49.26 )
  B ( 11.80 48.65 )
  C ( 12.25 49.10 )
```

```
)

LINKS (
  # Format: name (src tgt) precap precost rcost fixcost ({cap cost}*)
  L_AB  ( A B )  63  100  10  1000 ( 63 1200 252 3700 )
  L_AC  ( A C )   0    0  10  1000 ( 63  500 252 1700 )
  L_BC  ( B C )   0    0  10  1000 ( 63  600 252 2100 )
)

DEMANDS (
  # Format: name (src tgt) routing_unit value hoplimit
  D_AB1  ( A B )   1 65  UNLIMITED
  D_AB2  ( A B )  63  4  1
)

ADMISSIBLE_PATHS (
  # Format: name ({path_id ( link_name+ )}+)
  D_AB1 ( P1 ( L_AB ) P2 ( L_AC L_BC ) )
  D_AB2 ( P1 ( L_AB ) )
)
```

The node section describes the set of node locations in the network. The coordinates are included for drawing purposes only.

The link section describes the set of potential links with their installable capacity and cost. Suppose that unit 1 corresponds to 2 Mbit/s. Then link L_AB has 63 capacity units of preinstalled capacity with the cost of 100, whereas other links have no preinstalled capacity. For all links the routing cost of a flow unit is 10, and the fixed-charge cost of a link is 1000. Capacity modules of size 63 and 252 can be installed on every link; for link L_AB the cost of one module is 1200 or 3700, respectively (similarly for the other links).

The demand section describes the demands with their parameters. The above example contains two parallel demands from A to B, given in different routing units. Demand D_AB1 has the volume of 65 units of 2 Mbit/s (routing unit 1) and no hop limit. Demand D_AB2 has the volume of 4 units of 63x2Mbit/s (routing unit 63) and hop limit equal to 1; if the SNDlib model specifies to take the hop limit value into account (see the next section) then only direct paths with at most one link are admissible for this demand.

The admissible paths section describes the sets of admissible paths for demands D_AB1 and D_AB2. If in the SNDlib model the admissible path model is EXPLICIT_LIST then at most the paths listed here are admissible, otherwise all paths are admissible. The set of paths for demand D_AB2 may be further restricted by the hop limit value. Further details can be found in Section 3 and on the SNDlib website.

2.2 SNDlib model

SNDlib model specifies a selected combination of design parameter values. The design parameters are given as a set of attributes and their admissible values (exactly one value has to be selected for each attribute), and are described in the sequel. Table 1 gives an overview of the attributes and their admissible values. The abbreviations given in parentheses will be used in the example at the end of this section.

Demand model This attribute specifies the type of traffic. The admissible values are DIRECTED and UNDIRECTED. Depending on the network technology, traffic may correspond to (directed) flow of data (for example, a stream of IP packets) from one end-node of a demand to the other (DIRECTED). In other situations traffic may correspond to undirected transport connections (for example, VC-N in SDH networks) between a pair of nodes (UNDIRECTED).

Link model This attribute specifies how the capacity of a link is used. The admissible values are DIRECTED, BIDIRECTED, and UNDIRECTED. If the nature of traffic is such that the traffic is directed, the

Attribute	Admissible values
Demand model	DIRECTED (D), UNDIRECTED (U)
Link model	DIRECTED (D), UNDIRECTED (U), BIDIRECTED (B)
Link capacity model	LINEAR_LINK_CAPACITIES (L), MODULAR_LINK_CAPACITIES (M), EXPLICIT_LINK_CAPACITIES (E)
Fixed-charge model	YES (Y), NO (N)
Routing model	SINGLE_PATH (S), CONTINUOUS (C), INTEGER (I)
Admissible path model	EXPLICIT_LIST (E), ALL_PATHS (A)
Hop limit model	INDIVIDUAL_HOP_LIMITS (Y), IGNORE_HOP_LIMITS (N)
Survivability model	NO_SURVIVABILITY (N), ONE_PLUS_ONE_PROTECTION (P), SHARED_PATH_PROTECTION (S), UNRESTRICTED_FLOW_RECONFIGURATION (U)

Table 1: SNDlib model design parameters

capacity of a link may be either available only for traffic flowing in one direction (DIRECTED), or may be shared by traffic flows in opposite directions (UNDIRECTED). In the BIDIRECTED case, the same amount of capacity is provided in each direction of the link for routing directed traffic.

Link capacity model This attribute specifies how the capacity of a link is provided. The admissible values are LINEAR_LINK_CAPACITIES, MODULAR_LINK_CAPACITIES, and EXPLICIT_LINK_CAPACITIES. In the LINEAR_LINK_CAPACITIES case, any amount of capacity can be installed, even fractional values. Otherwise, only discrete values of capacity are allowed. With MODULAR_LINK_CAPACITIES, any combination of integer multiples of some capacity modules can be installed, representing for example a mixture of different lightpath bitrates in an optical network. With EXPLICIT_LINK_CAPACITIES, at most one single capacity out of a given list can be installed, e.g., to select a particular STM-N port capacity in an SDH network.

Fixed-charge model This attribute specifies whether using a link incurs a fixed-charge cost or not. The admissible values are YES and NO. If the value is YES then the fixed-charge parameter of each link from the network data is taken into account, otherwise it is ignored. The fixed-charge cost reflects the fact that providing capacity modules on a link may require initial investment which is independent of the actual amount of capacity. For example, to install a number of transport systems between a pair of nodes, a cable segment might be a capacity independent prerequisite.

Routing model This attribute specifies how traffic of a demand can be split among the available paths. The admissible values are SINGLE_PATH, CONTINUOUS, and INTEGER. For technological or operational reasons, e.g., to avoid reordering of data packets, it may be required to route all traffic of a demand on a single path (SINGLE_PATH). Otherwise, the traffic of a demand can be split (bifurcated) among several paths. In this case, either the amount of traffic flowing on any path can be arbitrary (CONTINUOUS), which is true, for example, in MPLS networks, or is required to be a multiple of the demand's routing unit (INTEGER), like in transport networks.

Admissible path model This attribute specifies the paths which can be used to route the traffic. The admissible values are EXPLICIT_LIST and ALL_PATHS. In general, the traffic of a demand can be routed on any path between the end-nodes of the demand (ALL_PATHS). Sometimes, however, it is desired to provide a set of carefully chosen paths and require the traffic to be routed only along these paths (EXPLICIT_LIST).

Hop limit model This attribute specifies whether the number of links in a path is explicitly bounded or not. The admissible values are INDIVIDUAL_HOP_LIMITS and IGNORE_HOP_LIMITS. Very often, switching and transmission delays are limited indirectly by bounding the number of intermediate nodes (for example, IP routers) or links (for instance, optical transport systems) on allowed routing paths. This further restricts the set of available paths that can be used for a demand. If the value of this model parameter is set to INDIVIDUAL_HOP_LIMITS, the number of links in any admissible routing path is limited by the hop limit of the corresponding demand from the network data. If, on the other hand, the value is IGNORE_HOP_LIMITS, all hop limit values are ignored.

Survivability model This attribute specifies the survivability concept to be used. The admissible values for this attribute are NO_SURVIVABILITY, ONE_PLUS_ONE_PROTECTION, SHARED_PATH_PROTECTION, and UNRESTRICTED_FLOW_RECONFIGURATION. In the simplest case the network does not provide any protection against node and link failures (NO_SURVIVABILITY). Otherwise, several mechanisms are considered to protect traffic against failures. With 1+1 protection (ONE_PLUS_ONE_PROTECTION), the traffic of a demand is routed simultaneously on exactly two link- or node-disjoint paths, one of which is the working path and the other is the hot-standby backup path. Traffic restoration is more complicated as it requires to reroute traffic flows in the case of a failure. For example, one may choose to restrict rerouting to the flows affected by the failure (SHARED_PATH_PROTECTION), or to allow rerouting of unaffected flows as well (UNRESTRICTED_FLOW_RECONFIGURATION).

Other models Certain design parameters, such as a hardware model, are not considered in the current version of the SNDlib. Furthermore, the only considered design objective so far is to minimize the total link cost. The cost of each link includes the fixed-charge cost, the cost of the pre-installed capacity and of the selected capacity modules, and routing cost.

Example of the SNDlib model Selecting one of the admissible values for each attribute defines an SNDlib model instance. Notice that not all combinations of admissible values make sense; for example, UNDIRECTED demands can only be combined with UNDIRECTED link capacities.

A complete model instance in the native ASCII format of the SNDlib might look as follows:

```
DEMAND_MODEL          = UNDIRECTED                # U
LINK_MODEL            = UNDIRECTED                # U
LINK_CAPACITY_MODEL   = MODULAR_LINK_CAPACITIES   # M
FIXED_CHARGE_MODEL    = NO                        # N
ROUTING_MODEL         = CONTINUOUS                # C
ADMISSIBLE_PATH_MODEL = ALL_PATHS                 # A
HOP_LIMIT_MODEL       = IGNORE_HOP_LIMITS         # N
SURVIVABILITY_MODEL   = NO_SURVIVABILITY          # N
```

Naming conventions We have introduced a naming convention for the SNDlib problems that reflects both the name of the network and the characteristics of the model. Such a name consists of an abbreviation of the SNDlib network name and of a sequence of letters specifying the selected design parameter values. These letters are given in Table 1, and they are ordered as in the example above, i.e., first the demand model, then the link model, and so on. That is, the problem instance built from the network EXAMPLE and the above model would be named EXAMPLE–U-U-M-N-C-A-N-N.

2.3 SNDlib solution

Basically, a solution to an SNDlib problem consists of a link configuration and a routing configuration of demands such that all side constraints are satisfied. Without going into details, this is illustrated by the following solution to the problem instance EXAMPLE–U-U-M-N-C-A-N-N introduced previously.

```
LINK-CONFIGURATIONS (
   # Format: link_name ( {module_capacity  install_count}+ )
   L_AB ( 63 1  252 1 )
   L_AC ( 63 1 )
```

```
    L_BC ( 63 1 )
)

ROUTINGS (
    # Format: demand-name ( {flow_path_value ( link-name+ )}+ )
    D_AB1 ( 63 ( L_AB ) 2 ( L_AC L_BC ) )
    D_AB2 ( 4  ( L_AB ) )
)
```

In this solution, link L_AB is equipped with one module of size 252 (i.e., 622 Mbit/s) in addition to the preinstalled capacity of size 63 (corresponding to 155 Mbit/s). On links L_AC and L_BC, one module of size 63 is installed. The 65 units of 2 Mbit/s for demand D_AB1 are routed on two paths. In contrast, all 4 units of size 63 for demand D_AB2 are routed on the direct link L_AB. A detailed description of the available solution description formats, as well as a slightly larger example with a detailed cost analysis can be found on the SNDlib website.

3 Mathematical reference model

This section serves as a reference for a precise definition of the sets of feasible and optimal solutions for network design problems. The design problems currently available in SNDlib are defined mathematically in terms of mixed-integer programming (MIP) formulations. Starting with basic graph notation, we introduce parameters and variables, discuss different types of constraints, and define objective functions.

We would like to emphasize that the primary goal of this section is to present a compact and comprehensible MIP formulation that covers *all* SNDlib planning problems. From a computational point of view, some parts of this formulation can certainly be modeled in a more efficient way. It is a part of SNDlib's purpose that its users identify the best mathematical models.

3.1 Network

A network instance $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{D})$ is given by a set of nodes $\mathcal{V}$, a set of links $\mathcal{E}$, and a set of point-to-point demands $\mathcal{D}$. Links and demands can be directed or undirected depending on the context; in addition, links can be bidirected. This will be discussed in more detail later.

3.2 Routing

To each demand $d \in \mathcal{D}$ there corresponds a set $\mathcal{P}_d$ of admissible paths with the same end-nodes. The sets $\mathcal{P}_d$, $d \in \mathcal{D}$, are assumed to be pairwise disjoint, i.e., if there are parallel demands, we use different symbols for the same path depending on the currently considered demand. In the DIRECTED demand model, each path $p \in \mathcal{P}_d$ has to be directed from the source node to the target node of demand d, i.e., the direction of links must be consistent with the direction of the path. In the UNDIRECTED demand model, p connects the end-nodes of d in both directions since links are also undirected.

By default, all simple routing paths (i.e., without node repetitions) are allowed. Optionally, a fixed set of admissible routing paths can be specified for each demand, which must be the case if the admissible path model is set to EXPLICIT_LIST. In both cases, the set of admissible paths can be additionally restricted by a hop limit ℓ_d for each demand d. If the hop limit model is INDIVIDUAL_HOP_LIMITS then every routing path used for demand d must traverse at most ℓ_d links; otherwise, the hop limit is ignored.

The total set $\mathcal{P}$ of all admissible paths is denoted by $\mathcal{P} = \bigcup_{d \in \mathcal{D}} \mathcal{P}_d$. For each demand $d \in \mathcal{D}$, a routing unit $r_d \in \mathbb{Z}_+$ and a demand value $h_d \in \mathbb{Z}_+$ is specified, meaning that a flow equal to $r_d h_d$ has to be routed between the end-nodes of the demand. Using path-flow variables x_p to specify the number of units of size r_d routed on path $p \in \mathcal{P}_d$, the demand constraints are

$$\sum_{p \in \mathcal{P}_d} x_p = h_d, \quad d \in \mathcal{D}. \tag{1}$$

The domain of the path-flow variables depends on the routing model. The three possibilities are $x_p \in \mathbb{R}_+$ (CONTINUOUS), $x_p \in \mathbb{Z}_+$ (INTEGER), or $x_p \in \{0, h_d\}$ (SINGLE_PATH). Consequently, in the latter two cases, the flow on each routing path is always an integer multiple of the routing unit r_d.

3.3 Link capacity model

For each link $e \in \mathcal{E}$, a set $\mathcal{T}_e$ of installable capacity modules is given. Each module $t \in \mathcal{T}_e$ provides capacity $c_e^t \in \mathbb{Z}_+$. Furthermore, each link e is equipped with pre-installed capacity $C_e \in \mathbb{Z}_+$ (which can be equal to zero). In the case of MODULAR_LINK_CAPACITIES or EXPLICIT_LINK_CAPACITIES, the integer variable $y_e^t \in \mathbb{Z}_+$ denotes the number of modules of type $t \in \mathcal{T}_e$ installed on link e, and the auxiliary variable

$$Y_e := C_e + \sum_{t \in \mathcal{T}_e} c_e^t y_e^t \tag{2}$$

defines the total capacity on link e. With LINEAR_LINK_CAPACITIES, we assume $y_e^t \in \mathbb{R}_+$ for all $e \in \mathcal{E}$. In this case, only one capacity module with the lowest unit cost needs to be kept on each link, so $|\mathcal{T}_e| = 1$ can be assumed after preprocessing. With EXPLICIT_LINK_CAPACITIES, an additional constraint specifies that at most one capacity module can be chosen for each link:

$$\sum_{t \in \mathcal{T}_e} y_e^t \leq 1, \quad e \in \mathcal{E}. \tag{3}$$

The integer capacity variables y_e^t are effectively binary in this setting.

3.4 Link capacity constraints

Now we introduce the capacity constraints relating path-flow and capacity variables. With the DIRECTED or BIDIRECTED link capacity models, demands must be DIRECTED, whereas with the UNDIRECTED capacity model we assume UNDIRECTED demands. Let $\mathcal{Q}_e := \{p \in \mathcal{P} \mid e \in p\}$ denote the set of all admissible paths traversing link $e \in \mathcal{E}$. For BIDIRECTED link capacities, let $\mathcal{Q}_e^+ \subseteq \mathcal{Q}_e$ denote the set of routing paths traversing link e in the forward direction, i.e., from source to target of the link. Accordingly, $\mathcal{Q}_e^- := \mathcal{Q}_e \setminus \mathcal{Q}_e^+$ consists of the routing paths using link e in the backward direction.

To simplify the notation, define the routing unit r_p of routing path p to be equal to the routing unit r_d of the unique demand with $p \in \mathcal{P}_d$, and define an auxiliary variable

$$f_e := \begin{cases} \sum_{p \in \mathcal{Q}_e} r_p x_p & \text{(DIRECTED/UNDIRECTED)} \\ \max\left\{ \sum_{p \in \mathcal{Q}_e^+} r_p x_p,\ \sum_{p \in \mathcal{Q}_e^-} r_p x_p \right\} & \text{(BIDIRECTED)} \end{cases} \tag{4}$$

for each link $e \in \mathcal{E}$, denoting the amount of capacity induced by the flow on link e. Using these variables, the capacity constraint simply reads:

$$f_e \leq Y_e, \qquad e \in \mathcal{E}. \tag{5}$$

3.5 Survivability

To cope with the survivability models of SNDlib, we introduce a set $\mathcal{S}$ of network (failure) states. In the current version of SNDlib, the set $\mathcal{S}$ consists of the *normal state* with no link failed (this particular state is denoted by $\mathcal{O}$) and the *failure states* corresponding to all single link failures. Links are assumed to be either fully available or totally failed so that partial link failures are not considered. The set of all links that are available (not failed) in state $s \in \mathcal{S}$ will be denoted by $\mathcal{E}^s$. Further, for each state $s \in \mathcal{S}$ and each demand $d \in \mathcal{D}$ the set of all admissible paths of demand d that survive in state s is denoted by $\mathcal{P}_d^s$, i.e., $\mathcal{P}_d^s := \{p \in \mathcal{P}_d \mid p \subseteq \mathcal{E}^s\} \subseteq \mathcal{P}_d$. Similarly, for each link $e \in \mathcal{E}$ and each state $s \in \mathcal{S}$, the set of all surviving paths that traverse link e is denoted by $\mathcal{Q}_e^s$, $\mathcal{Q}_e^s \subseteq \mathcal{Q}_e$.

3.5.1 Dedicated protection

To model ONE_PLUS_ONE_PROTECTION, we replace the demand constraint (1) by

$$\sum_{p \in \mathcal{P}_d} x_p = 2h_d, \quad d \in \mathcal{D} \tag{6}$$

to route twice the demand value in the normal state, and add the constraints

$$\sum_{p \in \mathcal{P}_d^s} x_p \geq h_d, \quad s \in \mathcal{S} \backslash \{\mathcal{O}\}, \ d \in \mathcal{D} \tag{7}$$

to ensure that in any failure state s, the surviving flow can satisfy the demand volume. Furthermore, we assume the routing model SINGLE_PATH, i.e., $x_p \in \{0, h_d\}$ for all $d \in \mathcal{D}$ and $p \in \mathcal{P}_d$. This can also be modeled using binary flow variables.

3.5.2 Shared protection

To model SHARED_PATH_PROTECTION and UNRESTRICTED_FLOW_RECONFIGURATION, we use state-dependent path-flow variables x_p^s ($s \in \mathcal{S}$, $d \in \mathcal{D}$, $p \in \mathcal{P}_d^s$) denoting the amount of flow of demand d allocated to path p in state s. To model the flow through a link in a given network state, we use auxiliary state-dependent variables

$$f_e^s := \begin{cases} \sum_{p \in \mathcal{Q}_e} r_p x_p^s & \text{(DIRECTED/UNDIRECTED)} \\ \max\left\{ \sum_{p \in \mathcal{Q}_e^+} r_p x_p^s, \sum_{p \in \mathcal{Q}_e^-} r_p x_p^s \right\} & \text{(BIDIRECTED)} \end{cases} \tag{8}$$

for each failure state $s \in \mathcal{S} \backslash \{\mathcal{O}\}$ and each surviving link $e \in \mathcal{E}^s$. Using these variables, we can define the dimensioning problem with UNRESTRICTED_FLOW_RECONFIGURATION by adding the constraints

$$\sum_{p \in \mathcal{P}_d^s} x_p^s \geq h_d, \quad s \in \mathcal{S} \backslash \{\mathcal{O}\}, \ d \in \mathcal{D} \tag{9}$$

and

$$f_e^s \leq Y_e, \quad s \in \mathcal{S} \backslash \{\mathcal{O}\}, \ e \in \mathcal{E}^s \tag{10}$$

to the model of the normal state.

Since the above formulation allows for arbitrary flow reconfiguration in case of a failure, it is primarily of theoretical importance. As the routing problem decomposes into independent subproblems for each network state, it can easily be used to derive a lower bound on the network cost with more realistic restoration mechanisms such as SHARED_PATH_PROTECTION. The latter can be modeled by adding the constraints

$$x_p^s \geq x_p, \quad s \in \mathcal{S} \setminus \{\mathcal{O}\}, \ d \in \mathcal{D}, \ p \in \mathcal{P}_d^s \tag{11}$$

to the dimensioning problem with unrestricted flow reconfiguration to ensure that only failed flows are rerouted. Recall that x_p denotes the flow on path $p \in \mathcal{P}_d$ in the normal state.

3.5.3 Summary

In summary, the relevant constraints for the different survivability concepts are

NO_SURVIVABILITY	(1), (2), (4), (5)
ONE_PLUS_ONE_PROTECTION:	(2), (4), (5), (6), (7), SINGLE_PATH
UNRESTRICTED_FLOW_RECONFIGURATION:	(1), (2), (4), (5), (8), (9), (10),
SHARED_PATH_PROTECTION:	(1), (2), (4), (5), (8), (9), (10), (11)

In addition, constraints (3) have to be added to any of these if the link capacity model is set to EXPLICIT_LINK_CAPACITIES.

3.6 Objective

Currently, the only optimization objective considered in SNDlib is to minimize the total network cost that, in its most general form, is composed of fixed-charge cost for installing a link, capacity cost, and routing cost. We will now introduce the necessary cost parameters and define the objective function.

Fixed-charge cost: For each link $e \in \mathcal{E}$, a fixed-charge cost $\kappa_e \in \mathbb{R}_+$ is incurred if link e is used, i.e., if a non-zero capacity Y_e is installed on e. We thus introduce a binary variable $z_e \in \{0, 1\}$ indicating whether link $e \in \mathcal{E}$ is used or not and require that

$$z_e = 0 \implies Y_e = 0, \quad e \in \mathcal{E}. \tag{12}$$

In terms of a mixed-integer programming model, this translates to

$$Y_e \leq M z_e, \quad e \in \mathcal{E}, \tag{13}$$

with some sufficiently large fixed number M. In particular, z_e is fixed to 1 as soon as link e has a non-zero pre-installed capacity.

Link capacity cost: Each module of type $t \in \mathcal{T}_e$ installed on a link $e \in \mathcal{E}$ incurs a cost of $k_e^t \in \mathbb{R}_+$. For the sake of computing the total network cost, the instances included in SNDlib provide an additional parameter for each link specifying the cost of pre-installed capacity. Since this value is irrelevant for the optimization process, we simply define C to be the sum of all pre-installed cost values for the purpose of this paper.

Routing cost: In addition to fixed-charge and capacity cost, there may also be a cost component which is proportional to the flow sent through each link. To this end, we introduce a cost parameter K_e for each link $e \in \mathcal{E}$ incurred for every unit of (working or backup) flow routed through e. The total routing cost of a link $e \in \mathcal{E}$ is computed as K_e times the maximum used capacity (f_e or f_e^s, respectively) on the link in any operating state. In particular, notice that for bidirected links, routing cost is incurred for the maximum of the two flow values in each direction, not for the sum of these values.

Objective function: Using the components defined above, the optimization objective reads:

$$\min\ C + \sum_{e \in \mathcal{E}} \Big(\kappa_e z_e + \sum_{t \in \mathcal{T}_e} k_e^t y_e^t + K_e f_e \Big). \tag{14}$$

for NO_SURVIVABILITY and ONE_PLUS_ONE_PROTECTION, or

$$\min\ C + \sum_{e \in \mathcal{E}} \Big(\kappa_e z_e + \sum_{t \in \mathcal{T}_e} k_e^t y_e^t + K_e \max_{s \in \mathcal{S}} f_e^s \Big). \tag{15}$$

for SHARED_PATH_PROTECTION and UNRESTRICTED_FLOW_RECONFIGURATION. By setting the values of some of the cost coefficients κ_e, k_e^t, or K_e to 0, classical planning problems like the fixed-charge network design problem or the modular link dimensioning problem can be obtained.

4 Network design instances

Currently, SNDlib contains 22 networks with topologies depicted in Figure 2. Each of these networks has been combined with several SNDlib models (basically, all reasonable SNDlib models for each network) to construct 830 problem instances.

Table 2 provides an overview of the networks together with their statistical information and with an indication which SNDlib models have been used for each network. The name of the network is followed by the number of its nodes ($|\mathcal{V}|$), links ($|\mathcal{E}|$), and demands ($|\mathcal{D}|$). The next column ($|I|$) provides the total

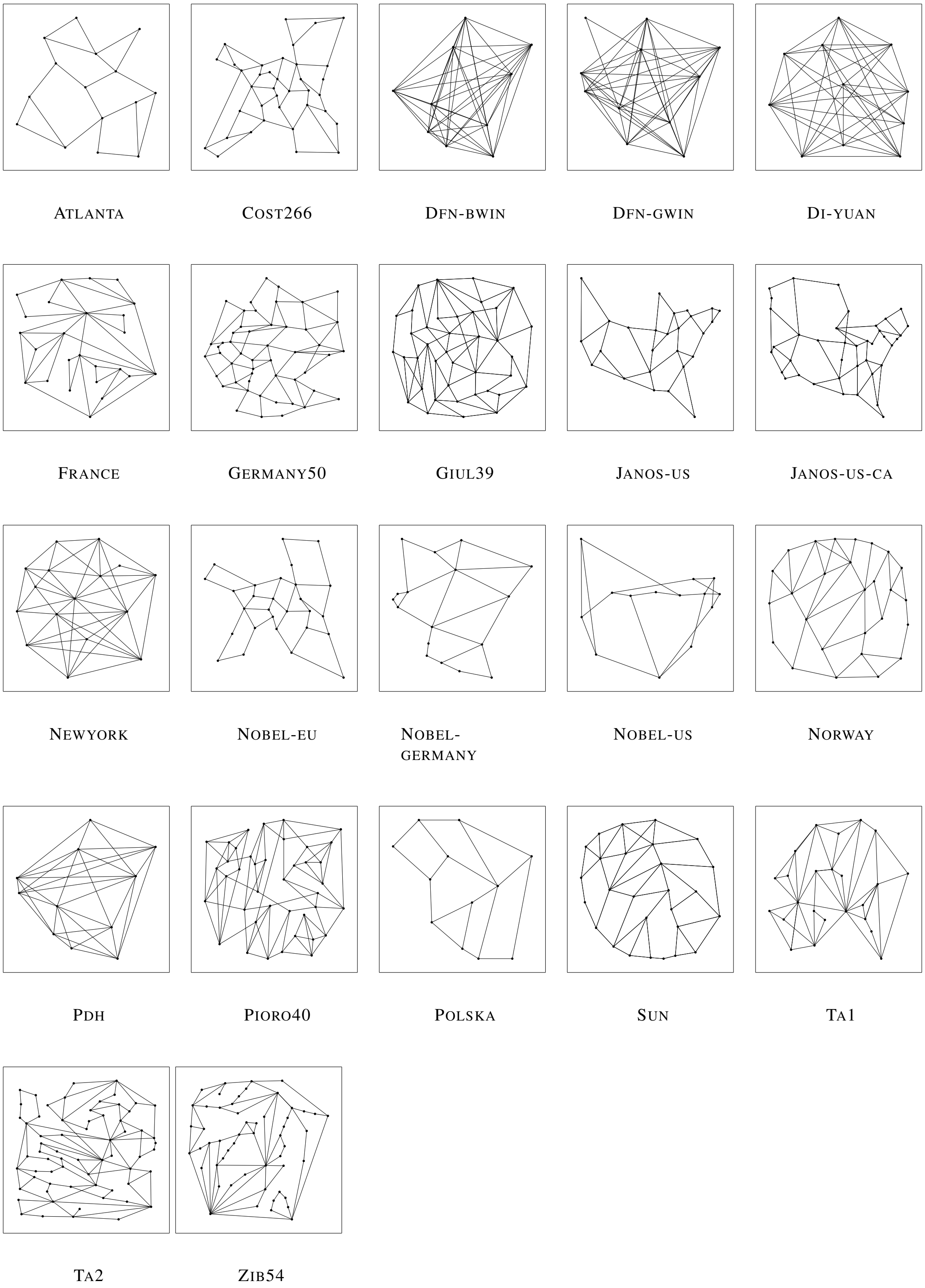

Figure 2: Topologies of the 22 SNDlib networks.

network	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	$\|\mathcal{D}\|$	$\|I\|$	DL	CAP	surv	pre	fix	rcost	exp	hop
ATLANTA	15	22	210	20	UU/DB	M	×	×	-	×	-	-
COST266	37	57	1332	18	UU/DB	ELM	-	-	-	×	-	-
DFN-BWIN	10	45	90	20	UU/DB	E	×	-	-	-	-	-
DFN-GWIN	11	47	110	24	UU/DB	EM	-	×	-	-	-	×
DI-YUAN	11	42	22	20	UU/DB	E	×	-	-	-	-	-
FRANCE	25	45	300	80	UU/DB	LM	×	-	×	-	-	-
GERMANY50	50	88	662	40	UU/DB	LM	×	-	-	-	-	-
GIUL39	39	172	1471	10	DD	E	×	-	-	-	-	-
JANOS-US	26	84	650	40	DD	LM	×	-	-	-	-	-
JANOS-US-CA	39	122	1482	20	DD	LM	×	-	-	-	-	-
NEWYORK	16	49	240	60	UU/DB	ELM	×	-	-	×	-	-
NOBEL-EU	28	41	378	20	UU/DB	E	×	-	-	-	-	-
NOBEL-GERMANY	17	26	121	20	UU/DB	E	×	-	-	-	-	-
NOBEL-US	14	21	91	6	UU/DB	E	-	-	-	-	-	-
NORWAY	27	51	702	60	UU/DB	ELM	×	-	-	-	-	-
PDH	11	34	24	60	UU/DB	ELM	×	-	-	-	-	-
PIORO40	40	89	780	80	UU/DB	M	×	-	×	-	×	-
POLSKA	12	18	66	80	UU/DB	M	×	-	×	-	×	-
SUN	27	102	67	10	DD	M	×	-	-	-	-	-
TA1	24	55	396	120	UU/DB	ELM	×	-	-	-	-	×
TA2	65	108	1869	36	UU/DB	ELM	-	×	-	-	-	×
ZIB54	54	81	1501	6	UU/DB	E	-	-	-	-	-	-

Table 2: SNDlib problem instances

number of problem instances constructed from each network. More information about the used SNDlib models is given in the next two columns. Column DL specifies the combinations of demand model and link model applied to this network: UU (both UNDIRECTED), DD (both DIRECTED), DB (DIRECTED demands and BIDIRECTED links). This is followed by the information on used link capacity models (E=EXPLICIT_LINK_CAPACITIES, L=LINEAR_LINK_CAPACITIES, M=MODULAR_LINK_CAPACITIES) provided in column CAP.

The columns in the last block indicate whether the network supports survivability (*surv*), whether links with positive preinstalled capacity (*pre*), fixed-charge cost (*fix*), or routing cost (*rcost*) exist, whether explicit lists of paths are provided (*exp*), and whether demands with hop limits exist (*hop*). The table shows that the problem instances cover the whole range from small to relatively large problems, with different densities, cost structures, and routing constraints.

To avoid infeasible problem instances as much as possible, we have run some heuristic tests on the networks to see which of them would allow for a solution with 1+1 protection. For those networks where we could show infeasibility in such a setting, we have generated problem instances without survivability only. The other networks have been combined with all survivability models. Due to this, we hope that there are no infeasible problem instances in SNDlib, but we cannot guarantee this. So, if you happen to find such an instance in SNDlib, we kindly request you to notify us, preferably providing information on how the infeasibility was detected (finding algorithms for quickly detecting infeasibility is a practically relevant problem).

The network instances included in the SNDlib have different backgrounds. Some of them stem from industrial projects, others have been defined as reference networks in international research projects involving a variety of industrial and academic partners. For the third group of network instances, the background is not publicly available, for instance, due to non-disclosure agreements. Below we will summarize the background and specific features of the network instances as close as possible.

Instances with industrial background The following network instances have been provided by network operators or by manufacturers of network equipment, either directly or via academic partners. We know that in some cases the data has been modified by the industrial or academic partners in order to avoid revealing internal information. From our experience with industry projects, we suspect this to be the case for other network instances as well. Nevertheless, we assume that the cost and demand structures provided in these data sets are not too far away from reality.

DFN-BWIN, DFN-GWIN	Derived from IP/OSPF planning problems on the German research network, which is operated by DFN-Verein and connects most universities and research institutes. The demands are based on traffic measurements in the network.
FRANCE	WDM planning data provided by a network operator.
GIUL39	Derived from a VPN planning problem in a two-layer metropolitan network with SDH as the underlying transport technology. Provided by an equipment vendor. The installable capacities have been constructed for SNDlib.
PDH	PDH planning data provided by a network operator.
POLSKA	SDH transport network of the Polish Telecom from the early 1990's.
TA1, TA2	SDH planning problems provided by the network operator Telekom Austria.
ZIB54	Slightly modified real-world instance from a telecom network operator.

Instances from international research projects The network instances in this group have been defined as reference networks in large research projects funded by the European Commission. These projects involved major telecom network operators and equipment manufacturers, as well as academic partners. Hence, even if the reference networks defined in the projects do not represent real networks, they were carefully constructed to correspond to realistic planning scenarios.

COST266	Originating from the project *COST266–Advanced Infrastructure for Photonic Networks* [7] of the European Cooperation in the Field of Scientific and Technical Research (COST).
NOBEL-US, NOBEL-EU, NOBEL-GERMANY, GERMANY50	The NOBEL networks are reference networks originating from the European project NOBEL [8], where a detailed cost model was developed for various kinds of SDH and WDM equipment, like fibers, transponders, repeaters, multiplexers, and line cards, see for instance [9]. The network GERMANY50, provided by T-Systems International AG, is an extension of the NOBEL-GERMANY network.

Other instances The following network instances either do not fit into one of the above categories, or we do not know about their origin. Several of these network instances had been used in the network design literature before they were sent to us; we give the reference to their first appearance in the literature if available. For a number of old data sets, solving a linear problem was nearly sufficient to solve the integer problem because the capacity granularity was small in comparison to the demands. In such cases, we have scaled the demands or capacities to make the problem instances more challenging.

ATLANTA, NEWYORK, NORWAY	The ATLANTA network represents an ATM network in Los Angeles, the NEWYORK network a telecommunication network in the greater New York area, and the NORWAY network a backbone network in Norway. These networks have been used in [10], among others. Demands and installable capacities have been scaled for SNDlib.
DI-YUAN	An optical city network with unknown background.
JANOS-US, JANOS-US-CA	The JANOS-US network was first presented in [11]. The JANOS-US-CA instance is an extension of that network published in [12].
PIORO40	An unpublished randomly generated network with an SDH cost structure.
SUN	The network, first presented in [10], have been constructed from NEWYORK by bidirecting all links and changing the demand pattern. We have scaled all demands by 10 to make them integer.

5 Final remarks

The purpose of the SNDlib is to provide realistic network planning data for the network design community, which can be used to compare different mathematical models and optimization algorithms. At the same time, we wanted to create a platform for sharing not only network planning data but also other information required by the community. To this end, SNDlib provides

- a collection of bibliography entries related to design of fixed networks,
- a list of upcoming conferences in the field,
- a list of relevant journals, and
- a mailing list[2] related to fixed network design.

In order to keep this information up to date, everybody is cordially invited to contribute to SNDlib by any of the following means:

- Visit the website `http://sndlib.zib.de`.
- Use and reference test instances from SNDlib in your papers (a suitable BIBTEX entry can be found in the bibliography on the SNDlib website).
- Submit new solutions or dual bounds for existing test instances.
- Submit new test instances for upcoming versions.
- Send us references to your papers to include them in the SNDlib bibliography.
- Submit new entries to the conference list.
- Send tools to us to share them with others, such as visualization scripts or converters between different network formats.
- Use the SNDlib mailing list to make relevant information related to network design available to others.
- Send suggestions and questions to `sndlib-webmaster@zib.de`.

We hope that SNDlib will be helpful in developing network planning and design methodologies, and in consequence will contribute to improving the quality of future communication networks.

6 Acknowledgement

Special thanks to the contributors of the SNDlib who have helped building up the library either by providing network instances, problem solutions and bibliography information, or by doing a lot of programming work. In particular, this includes Edoardo Amaldi, Pieter Audenaert, Dan Bienstock, Andreas Bley, Giuliana Carello, Tibor Cinkler, Mateusz Dzida, Claus Gruber, Oktay Günlük, Ralf Hülsermann, Peter Jonas, Roman Klähne, Mariusz Mycek, Mats Petter Petterson, Christian Raack, Franz Serajnik, Janos Tapolcai, Di Yuan, Michał Zagożdżon, and Adrian Zymolka. We would also like to thank Marc Pfetsch for his invaluable comments.

[2]The list is moderated to avoid spam mail.

References

[1] Tobias Achterberg, Thorsten Koch, and Alexander Martin. MIPLIB 2003. *Operations Research Letters*, 34(4):1–12, 2006. `http://miplib.zib.de`.

[2] G. Reinelt. TSPLIB–A Traveling Salesman Problem library. *ORSA Journal on Computing*, 3:376–384, 1991. `http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95`.

[3] T. Koch, A. Martin, and S. Voß. SteinLib: An updated library on Steiner tree problems in graphs. Technical Report ZR-00-37, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2000. `http://elib.zib.de/steinlib`.

[4] FAP web–A website devoted to frequency assignment. `http://fap.zib.de`, 2000–2007.

[5] T. Koch. Zimpl user guide. ZIB Report ZR-01-20, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2001. `http://zimpl.zib.de`.

[6] R. Fourer, D. M. Gay, and B. W. Kernighan. A modeling language for mathematical programming. *Management Science*, (36):519–554, 1990. `http://www.ampl.com`.

[7] S. De Maesschalck, D. Colle, I. Lievens, M. Pickavet, P. Demeester, C. Mauz, M. Jäger, R. Inkret, B. Mikac, and J. Derkacz. Pan-European optical transport networks: An availability-based comparison. *Photonic Network Communications*, 5(3):203–225, 2003.

[8] NOBEL–Next generation Optical networks for Broadband European Leadership. `http://www.ist-nobel.org`, 2004–2007.

[9] M. Gunkel, R. Leppla, M. Wade, A. Lord, D. Schupke, G. Lehmann, C. Fürst, S. Bodamer, B. Bollenz, H. Haunstein, H. Nakajima, and J. Martensson. A cost model for the WDM layer. In *International Conference on Photonics in Switching (PS 2006), Herakleion, Crete, Greece*, October 2006.

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

[11] M. De, V. Mariappan, V. Chandramouli, and S. Kuppusamy. US national network design. Presentation held at CReWMaN, University of Texas, Arlington, 2002.

[12] J. Tapolcai. *Routing Algorithms In Survivable Telecommunication Networks*. PhD thesis, Budapest University of Technology and Economics, March 2005.