Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

Sebastian Orlowski
Michał Pióro

On the complexity of column generation in survivable network design with path-based survivability mechanisms

ZIB-Report 08-51 (December 2008)

On the complexity of column generation in survivable network design with path-based survivability mechanisms

Sebastian Orlowski, Zuse Institute Berlin, orlowski@zib.de
Michał Pióro, Warsaw University of Technology, mpp@tele.pw.edu.pl

December 2008

Abstract

This survey concerns optimization problems arising in the design of survivable communication networks. It turns out that such problems can be modeled in a natural way as non-compact linear programming formulations based on multicommodity flow network models. These non-compact formulations involve an exponential number of path flow variables, and therefore require column generation to be solved to optimality. We consider several path-based survivability mechanisms and present results, both known and new, on the complexity of the corresponding column generation problems (called the pricing problems). We discuss results for the case of the single link (or node) failures scenarios, and extend the considerations to multiple link failures. Further, we classify the design problems corresponding to different survivability mechanisms according to the structure of their pricing problem. Finally, we show that almost all encountered pricing problems are hard to solve for scenarios admitting multiple failures.

1 Introduction

In the literature on communication network design and traffic engineering, a variety of linear programming (LP) formulations have been developed to incorporate various protection and restoration mechanisms against node and link failures into network optimization models. Many of them employ non-compact link-path formulations where each flow variable represents end-to-end demand flow routed on a path in the underlying communication network. In such formulations the number of routing paths, and thus the number of flow variables, grows exponentially with the network size. To solve link-path formulations with respect to all possible routing paths in networks of practical sizes, a common approach is to use column (path) generation, i.e., to start with a small set of flow variables corresponding to an initial set of routing paths, and to generate further variables only when needed to improve the current solution.

For a given optimal LP solution with respect to a restricted set of variables, the *pricing problem* or *column generation problem* is to identify further columns that could improve the LP value, or to discover that no such columns exist. A column can improve the current optimal solution if it has a negative reduced cost, i.e., if it violates a dual constraint. If new columns are found, the LP is resolved with the new variables. This process is repeated until no improving variables are found.

To be sure that in the end the LP is optimally solved with respect to *all* variables, the pricing algorithm has to be exact, i.e., if there exists an improving variable, the algorithm must be able to identify it. With exponentially many variables, computing the reduced cost for every potential variable individually is not feasible. It is thus of interest whether the pricing problem can be solved in polynomial time with respect to the size of the underlying network.

The primary goal of this survey is to systematically summarize the results (otherwise spread over the literature) on the complexity of column generation for various network survivability mechanisms and failure scenarios. The survivability mechanisms covered in this paper include path diversity, unrestricted reconfiguration, and several ways of end-to-end path restoration. We distinguish between single and multiple link failure scenarios, whether the selection of backup paths for working paths is *failure state-dependent* or *failure state-independent*, and whether the capacity of surviving links of a failing path can be released and reused for backup flows (this is known as *stub release*) or not. Note that in the latter case (*no stub release*) working and backup link capacity are separated. In the case of failure-independent restoration without stub release, we distinguish whether backup capacity is shared between demands or dedicated to each demand individually.

It has been known that for survivability mechanisms without stub release, under a single link or single node failure scenario the pricing problem reduces to a (polynomial) shortest-path or a shortest-pair of disjoint paths problem with respect to link weights derived from the dual LP solution. For other mechanisms, the pricing problem has been shown to be $\mathcal{NP}$-hard already under a single link failure scenario. When it comes to multiple failures (also referred to as *shared risk link groups*), little has been done so far for most of the considered survivability concepts—this issue is discussed in the balance of this survey.

It turns out that from the complexity viewpoint, the pricing problems for all the considered survivability concepts are composed of only few types of minimization problems: (a) a classical shortest-path problem, (b) a classical shortest-cycle problem (more precisely, a shortest failure-disjoint pair of paths problem), (c) a shortest path problem where the path length is the sum of given non-negative prices of the failure states in which the path fails, (d) a shortest path problem with link weights depending on the set of the failure states in which the path survives, and (e) a shortest path pair problem with link weights of the backup path depending on the set of failure states in which the primary path fails. In the single failure case, the complexity of the pricing problem depends on these structural classes; when multiple (even only double) link failures are taken into account, the pricing problem for most of the survivability mechanisms turns out to be $\mathcal{NP}$-hard. Furthermore, for most of the considered concepts, we are able to find a compact LP formulation if, and only if, the pricing problem is polynomial. Whether or not this is true in general is discussed in Section 9.4.

This report is organized as follows. In the next section, we will introduce the notation used in the rest of the paper. Section 3 describes in detail an LP formulation, its dualization, and the corresponding pricing problem for a particular problem called PD (path diversity); other survivability mechanisms are discussed (deliberately in less detail) in Sections 4–7. After summarizing the results in Section 8, we discuss their consequences and possible extensions in Section 9.

2 Notation

We will now introduce the notation needed to discuss the link-path formulations and pricing problems for various survivability concepts and failures scenarios.

Network. The considered network is modeled using a undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ composed of a set $\mathcal{V}$ of nodes and a set of links $\mathcal{E}$ between the nodes. For ease of exposition, we assume that the graph does not contain loops nor parallel links, i.e., $\mathcal{E} \subseteq \mathcal{V}^{|2|}$ where $\mathcal{V}^{|2|}$ is the set of all two-element subsets of the set of nodes $\mathcal{V}$. The end nodes of link $e \in \mathcal{E}$ are denoted by u_e and v_e, so $e = \{u_e, v_e\}$. The cost of realizing one unit of demand flow on link $e \in \mathcal{E}$ is denoted by ξ_e, and its capacity (which will serve as a variable) by y_e. Finally, $\delta(v) := \{e \in \mathcal{E} \mid e \ni v\}$ denotes the set of all links incident to node $v \in \mathcal{V}$.

Demands. The set $\mathcal{D} \subseteq \mathcal{V}^{|2|}$ represents undirected point-to-point demands. For notational convenience, at most one demand between each pair of nodes is assumed. The source and target of demand $d \in \mathcal{D}$ are denoted by u_d and v_d and assumed to be different from each other (the choice of the source and the target out of the two end nodes of a demand is arbitrary). The demand value (volume) of demand $d \in \mathcal{D}$ is given by $h_d > 0$.

Network states. All survivability concepts discussed in this paper are defined with respect to a given failure scenario. For this, we introduce a set $\mathcal{S} \subseteq 2^{\mathcal{E}}$ of *network states* each of which corresponds to a subset of failing links. Set $\mathcal{S}$ is called the *failure scenario*. It is assumed that $\mathcal{S}$ contains the normal, failure-less state $\emptyset$ in which all links are operational. The set $\mathcal{S}^* := \mathcal{S} \setminus \{\emptyset\}$ contains the *failure states* in which at least one link fails. The notation $\mathcal{S}_e = \{s \in \mathcal{S} \mid e \notin s\}$ will denote the set of all states $s \in \mathcal{S}$ in which link $e \in \mathcal{E}$ is available, and $\bar{\mathcal{S}}_e = \mathcal{S} \setminus \mathcal{S}_e$ will be the set of all failure states in which it fails. Throughout the paper, we will assume that the number of states in set $\mathcal{S}$ is polynomial with respect to the size of the network; otherwise the pricing problems must be exponential. Node failures are discussed in Subsection 9.3.

Routing paths. Each demand $d \in \mathcal{D}$ has a set $\mathcal{P}_d$ of undirected candidate paths that can be used for realizing the demand flow. Unless stated otherwise, $\mathcal{P}_d$ is a subset of all the elementary paths from u_d to v_d, i.e., the candidate paths do not traverse any node more than once. The set of all candidate paths is denoted by $\mathcal{P} := \bigcup_{d \in \mathcal{D}} \mathcal{P}_d$. As we have assumed at most one demand per a pair of nodes, the sets $\mathcal{P}_d, d \in \mathcal{D}$ are mutually disjoint; hence, each path $p \in \mathcal{P}$ can be simply identified with the set of the links it traverses, so that $p \subseteq \mathcal{E}$. The flow realizing the volume of demand $d \in \mathcal{D}$ on path $p \in \mathcal{P}_d$ will usually be denoted by variable x_p.

The set $\mathcal{P}_d^s$ of candidate paths for demand $d \in \mathcal{D}$ available in state $s \in \mathcal{S}$ is defined as

$$\mathcal{P}_d^s = \{p \in \mathcal{P}_d \mid p \cap s = \emptyset\} \subseteq \mathcal{P}_d;$$

similarly, $\bar{\mathcal{P}}_d^s := \mathcal{P}_d \setminus \mathcal{P}_d^s$ denotes the complementary set of all candidate paths failing in state $s \in \mathcal{S}$. Furthermore, $\mathcal{P}_e \subseteq \mathcal{P}$ is the set of all paths containing link $e \in \mathcal{E}$, and $\mathcal{P}_e^s := \mathcal{P}_e \cap \left(\bigcup_{d \in \mathcal{D}} \mathcal{P}_d^s\right)$ denotes the set of all paths containing link $e \in \mathcal{E}$ that are available in state $s \in \mathcal{S}$. Observe that by the definition of the normal state, $\mathcal{P}_d^\emptyset = \mathcal{P}_d$ and $\mathcal{P}_e^\emptyset = \mathcal{P}_e$.

The notation

$$\mathcal{S}_p = \{s \in \mathcal{S} \mid p \cap s = \emptyset\}$$

and $\bar{\mathcal{S}}_p := \mathcal{S} \setminus \mathcal{S}_p$ refers to the sets of states $s \in \mathcal{S}$ in which path $p \in \mathcal{P}$ is available or unavailable, respectively. For the path protection/restoration mechanisms considered in the sequel we will use the following notation. The notation $\mathcal{Q}_p \subseteq \mathcal{P}$ refers to all candidate backup paths for protecting a particular path $p \in \mathcal{P}$. These are paths with the same end nodes as path p that never fail together with p (in other words, p and q are *failure-disjoint*). Hence, if $q \in \mathcal{Q}_p$ then for all $s \in \mathcal{S}$,

$$p \cap s \neq \emptyset \quad \Rightarrow \quad q \cap s = \emptyset.$$

Also, $\mathcal{Q}_{pe} := \{q \in \mathcal{Q}_p \mid q \ni e\}$ denotes the set of all paths protecting path p that contain a particular link $e \in \mathcal{E}$. The set of all (failure-disjoint) primary-backup path pairs $c = (p, q)$ for demand $d \in \mathcal{D}$

will be denoted by $\mathcal{C}_d := \{c = (p, q) \mid p \in \mathcal{P}_d, q \in \mathcal{Q}_p\}$. Primary-backup path pairs will be denoted by $c \in \mathcal{C} := \bigcup_{d \in \mathcal{D}} \mathcal{C}_d$. Also, for each link $e \in \mathcal{E}$, the set of all pairs $c \in \mathcal{C}$ such that $e \in p$ will be denoted by $\mathcal{C}_e^1$, and the set of all pairs c such that $e \in q$ will be denoted by $\mathcal{C}_e^2$. Variable x_c denotes the flow realizing the volume of demand $d \in \mathcal{D}$ on path-pair $c \in \mathcal{C}_d$.

Assumptions. In practical applications, network features different than described above could be required. First of all, we could consider directed links (arcs) or demands, and directed or bi-directed link capacity, as this may be necessary for certain communication network models. In the main body of the report we assume undirected links and demands and defer the discussion of other options to Subsection 9.3.

Another important cases arise when modularity of link capacities or node hardware is taken into account, as well as when demand flows are forced to be unsplittable (non-bifurcated). Still, as this paper focuses on column generation, we restrict ourselves to linear programs and assume all the primal variables (link capacity and flow variables) to be continuous, nonnegative, and unbounded from above. This assumption is satisfied in a natural way in the linear (LP) relaxations of most practical network planning problems. Moreover, only the linear relaxations are used in the branch-and-price algorithms for exact resolution of the mixed-integer programs (MIP) resulting from link modularity or unsplittable flows.

Notice that the dual LPs discussed in this paper are always feasible because the zero vector is a feasible dual solution. To ensure that in the considered pricing problems, all occurring dual LPs have a bounded optimal solution, we assume that the initial set of routing paths results in a feasible initial primal LP. When it is not clear how to compute such a set of paths, feasibility of the primal LP can always be achieved by introducing artificial slack variables. Alternatively, column generation can also be done using a dual ray instead of a dual optimal solution if the primal LP is infeasible.

3 Path diversity – PD

We start the main part of our considerations on the complexity of pricing problems with a network design model related to the path diversity concept. For this particular model we will illustrate in detail the basic ideas behind path-flow LP formulations, their dual (LP) problems and the related pricing tasks. Other survivability concepts presented in the following sections will be discussed in less detail. We assume that the reader has a basic knowledge of linear programming techniques; for details we refer to the linear and integer programming books [Min86, NW88].

3.1 Primal problem

Conceptually, the simplest way of protecting traffic against failing network components is by over-provisioning. Protection concepts based on path diversity follow this approach by routing more than the specified demand value h_d in the failure-less state $\emptyset$, and ensuring that at least a specified fraction of it survives each considered failure scenario without rerouting any flow.

Several such concepts have been presented in the literature, for example *diversification* [DS98] and its generalization *demand-wise shared protection (DSP)* [KZJH05, WOZ$^+$05, KZ07]. For the purpose of this paper, we subsume both concepts under the name *path diversity*. The related network design problem is given by the following LP formulation. In essence, the formulation simply states that for each demand enough flow must survive in every network state.

$$\begin{aligned}
&\text{minimize} && \sum_{e\in\mathcal{E}} \xi_e y_e && && \text{(1a)}\\
&[\lambda_d^s \geq 0] && \sum_{p\in\mathcal{P}_d^s} x_p \geq h_d^s && d\in\mathcal{D},\ s\in\mathcal{S} && \text{(1b)}\\
& && \sum_{p\in\mathcal{P}_e} x_p \leq y_e && e\in\mathcal{E} && \text{(1c)}\\
& && x, y \geq 0. && && \text{(1d)}
\end{aligned}$$

If any of the end-nodes of some demand $d \in \mathcal{D}$ fails in state $s \in \mathcal{S}$, we assume $h_d^s = 0$ because otherwise the above LP becomes infeasible due to $\mathcal{P}_d^s = \emptyset$. The symbol λ_d^s in brackets to the left of constraint (1b) denotes the corresponding dual variable which is used in the problem dual to PD derived in the next section.

Notice that with fractional (i.e., continuous) capacity variables y_e, each inequality in (1c) could be turned into an equation, and then the problem would decompose into a set of separate problems for each demand $d \in \mathcal{D}$:

$$\begin{aligned}
&\text{minimize} && \sum_{p\in\mathcal{P}_d} \xi_p x_p && && \text{(2a)}\\
&[\lambda_d^s \geq 0] && \sum_{p\in\mathcal{P}_d^s} x_p \geq h_d^s && s\in\mathcal{S} && \text{(2b)}\\
& && x \geq 0, && && \text{(2c)}
\end{aligned}$$

where $\xi_p := \sum_{e\in p} \xi_e$ is the cost of sending one unit of flow along path $p \in \mathcal{P}$. Nevertheless, we keep the problem in form (1) because this decomposition cannot be applied to other survivability concepts discussed in this paper, or if additional constraints, like cutting planes, are present.

3.2 Dual problem

We will now derive the dual formulation to problem PD (1) using the Lagrangean function. For the other survivability concepts discussed in the following sections, we will formulate the dual without showing its derivation. For ease of notation, we will use the general concept of duality for convex primal problems [Las70], which allows us to write down the constraints in a "natural" way and have non-negative dual variables nevertheless. This approach differs from the usual definition of LP duality only in the sign of the dual variables. We thus assume primal problems of the following form:

$$\begin{aligned}
&\text{minimize} && F(x) && && \text{(3a)}\\
&[\lambda_j] && f_j(x) = 0 && j = 1, 2, \ldots, J && \text{(3b)}\\
&[\pi_k \geq 0] && g_k(x) \leq 0 && k = 1, 2, \ldots, K && \text{(3c)}\\
& && x \in X. && && \text{(3d)}
\end{aligned}$$

The dual variables λ associated with equality constraints (3b) are unconstrained in sign, while the dual variables π associated with inequality constraints (3c) are non-negative. Then the dual function, to be maximized over λ and $\pi \geq 0$, is defined as:

$$W(\lambda, \pi) := \min_{x\in X} \mathcal{L}(x; \lambda, \pi), \tag{4}$$

where the Lagrangean function, $\mathcal{L}(x;\lambda,\pi)$, is given by:

$$\mathcal{L}(x;\lambda,\pi) := F(x) + \sum_{j=1}^{J} \lambda_j f_j(x) + \sum_{k=1}^{K} \pi_k g_k(x). \tag{5}$$

Formally, by definition, the dual problem is given by:

$$\max_{\pi \geq 0,\ \lambda} W(\lambda,\pi) = \max_{\pi \geq 0,\ \lambda} \min_{x \in X} \mathcal{L}(x;\lambda,\pi). \tag{6}$$

In fact, problem (6) can admit unbounded values of the dual function $W(\lambda,\pi)$ defined by (4). Therefore, to make the dual a proper optimization problem, all vectors λ and π for which $W(\lambda,\pi) = -\infty$ have to be eliminated. This in general leads to an extra set of constraints (on top of $\pi \geq 0$) to be added to the formulation of the dual problem (6).

Finally, we recall the so called strong duality theorem which states that when the primal problem (3) has a bounded optimal solution F^* then this solution is equal to the optimal objective W^* of the dual problem (6).

The problem dual to PD can be obtained by substituting the capacity y_e of link $e \in \mathcal{E}$ by its load $\sum_{p \in \mathcal{P}_e} x_p$ in (1a), and by relaxing the demand constraints (1b) into the objective function. This leads to the following Lagrangean function using the primal variables $x_p \geq 0$ and the dual variables $\lambda_d^s \geq 0$ of the demand constraints (1b):

$$\begin{aligned}\mathcal{L}(x;\lambda) &:= \sum_{e \in \mathcal{E}} \xi_e (\sum_{p \in \mathcal{P}_e} x_p) + \sum_{d \in \mathcal{D}} \sum_{s \in \mathcal{S}} \lambda_d^s (h_d^s - \sum_{p \in \mathcal{P}_d^s} x_p) \\ &= \sum_{d \in \mathcal{D}} \sum_{s \in \mathcal{S}} h_d^s \lambda_d^s + \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d} (\sum_{e \in p} \xi_e - \sum_{s \in \mathcal{S}_p} \lambda_d^s) x_p. \end{aligned} \tag{7}$$

Then the dual problem is given by:

$$\max_{\lambda \geq 0} W(\lambda) = \max_{\lambda \geq 0} \min_{x,y \geq 0} \mathcal{L}(x,y;\lambda). \tag{8}$$

Problem (8) admits unbounded values of the dual function $W(\lambda)$ so we have to get rid of all vectors λ for which $W(\lambda) = -\infty$. This leads to the following form of the dual:

$$\max_{\lambda \geq 0} \big\{ \sum_{d \in \mathcal{D}} \sum_{s \in \mathcal{S}} h_d^s \lambda_d^s \mid \sum_{s \in \mathcal{S}_p} \lambda_d^s \leq \sum_{e \in p} \xi_e, \ \ d \in \mathcal{D},\ p \in \mathcal{P}_d \big\}. \tag{9}$$

Notice that this dual problem always has the feasible solution $\lambda = 0$. By introducing auxiliary dual variables $\Lambda_d,\ d \in \mathcal{D}$, we obtain a more handy form of the dual:

$$\max \quad W(\lambda) = \sum_{d \in \mathcal{D}} \sum_{s \in \mathcal{S}} h_d^s \lambda_d^s \tag{10a}$$

$$\Lambda_d = \sum_{s \in \mathcal{S}} \lambda_d^s \qquad d \in \mathcal{D} \tag{10b}$$

$$\Lambda_d \leq \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^s \qquad d \in \mathcal{D},\ p \in \mathcal{P}_d \tag{10c}$$

$$\lambda \geq 0. \tag{10d}$$

Given an optimal dual solution (λ^*, Λ^*) with respect to the current set of candidate paths, the goal of the pricing problem for demand $d \in \mathcal{D}$ is to find a new path $p \in \mathcal{P}_d$ which violates its dual constraint (10c), i.e., which satisfies

$$\sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^{s*} < \Lambda_d^*. \tag{11}$$

If added to the current LP relaxation, such a path p may improve the primal objective function. The details of this pricing problem for single and multiple link failures are discussed in the following two sections.

Notice that if the capacities y_e have to satisfy additional integrality restrictions, the values ξ_e in the dual constraints (10c) have to be replaced by the nonnegative dual values of the capacity constraints (1c) (similarly for all other survivability discussed in this paper). This does not affect the complexity of the pricing problems.

3.3 Pricing problem for single failures

Under a single link failure scenario $\mathcal{S} \subseteq \{\{e\} \mid e \in \mathcal{E}\} \cup \{\emptyset\}$, the pricing problem for PD can be solved in polynomial time, as observed by Wessäly et al. [Wes00, WOZ+05]. To see this, note that with single link failures only, condition (11) can be rewritten as follows:

$$\sum_{e \in p} (\xi_e + \lambda_d^{\{e\}*}) < \Lambda_d^*. \tag{12}$$

The right-hand side depends only on the demand, and the link weights on the left-hand side are nonnegative. Hence, for each demand $d \in \mathcal{D}$, violation of dual constraint (10c) can be tested by searching for a shortest path between the end-nodes of d with respect to the demand-dependent link weights $\gamma_d^e := \xi_e + \lambda_d^{\{e\}*}$ using for example the Dijkstra algorithm, and comparing its length to the value of Λ_d^*. If a shortest path p' for demand d' fulfils condition (12), then adding path p' to $\mathcal{P}_{d'}$ (and thus the corresponding constraint $\Lambda_{d'} \leq \sum_{e \in p'} (\xi_e + \lambda_{d'}^{\{e\}})$ to the dual formulation (10)) can potentially improve the primal objective value. Otherwise, no path for this demand violates its dual constraint for the current set of optimal dual variables.

3.4 Pricing problem for multiple failures

In a multiple failure state a group of links fails simultaneously. Such a group of links that fail together is sometimes called a *shared risk link group* (SRLG) [SCT01]. In this section, we show by reduction to the *minimum-color shortest-path problem* (MC-PATH) that path generation for PD is in general difficult if multiple link failure scenarios are considered.

In the MC-PATH problem, also known as *minimum label shortest-path problem*, every link $e \in \mathcal{E}$ is assigned a set $\mathcal{C}_e$ of colors (labels) out of the set of colors $\mathcal{C}$ with given weights $w_c > 0$. The length of a path p in this colored network is defined as the total weight of the colors traversed by p. In contrast to the classical shortest path problem, the weight of a used color is counted only once even if the path contains several links with that color. Given two nodes $u, v \in \mathcal{V}$, the goal of MC-PATH is to find a u-v-path with minimum length. This problem has been shown to be $\mathcal{NP}$-hard for a general color setting [YVJ05, CDP+06]. Also, various inapproximability results for MC-PATH are known [YVJ05, CDP+06, HMS07].

The pricing problem for PD, PRICE-PD, is defined separately for each demand $d \in \mathcal{D}$ and consists of finding a path p from u_d to v_d minimizing the generalized path length

$$\langle p \rangle = \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^{s*}. \tag{13}$$

The second sum contains the dual values of those network states $s \in \mathcal{S}$ in which the path fails, i.e., when the path contains at least one link from s. Notice that if the path contains several failing links from s, the weight λ_d^{s*} is counted only once, as in the MC-PATH problem. By identifying failure states with colors and assuming all costs ξ_e to be zero, it is easy to see that the pricing problem (13) contains MC-PATH as a special case, which shows that PRICE-PD is $\mathcal{NP}$-hard in general.

If the considered set of multiple failures is exponential in the number of network nodes, $\mathcal{NP}$-hardness of PRICE-PD is not surprising. Coudert et al. [CDP+06] have shown, however, that MC-PATH is $\mathcal{NP}$-hard already for the polynomially bounded set of all double link failures, i.e., $\mathcal{S} = \mathcal{E}^{|2|} \cup \{\emptyset\}$. Some special cases, however, are known to be polynomial, e.g., link failures induced by single node failures. Such cases are discussed in Section 9.

The pricing problem can be formulated as a mixed-binary programming problem (MIP) using the node-link notation. In this notation the link flows must be directed. In case of an undirected network, for each link $e \in \mathcal{E}$ ($e = \{v, w\}$) we must use two directed link flow variables: $x_{e,vw}$ denoting the flow in direction from v to w, and $x_{e,wv}$ denoting the flow in the opposite direction.

$$\min \sum_{e \in \mathcal{E}} \xi_e x_e + \sum_{s \in \mathcal{S}} \lambda_d^{s*} Y^s \tag{14a}$$

$$\text{s.t.} \sum_{\substack{e \in \delta(v) \\ e = \{v,w\}}} (x_{e,vw} - x_{e,wv}) = \begin{cases} 0, & v \in \mathcal{V} \setminus \{u_d, v_d\} \\ 1, & v = u_d \\ -1, & v = v_d \end{cases} \tag{14b}$$

$$1 \geq Y^s \geq x_{e,vw} + x_{e,wv} \qquad s \in \mathcal{S},\ e \in s,\ e = \{v, w\} \tag{14c}$$

$$x_{e,vw}, x_{e,wv} \in \{0, 1\} \qquad e \in \mathcal{E},\ e = \{v, w\}. \tag{14d}$$

Let (x, Y) be an optimal solution of (14). Then constraints (14b) together with the binary requirement (14d) will ensure that the the flows $x_{e,vw}$, $x_{e,wv}$ equal to 1 will specify a directed single-path flow of value 1 from u_d to v_d. Moreover, assuming positive link unit costs ξ, at most one of the variables $x_{e,vw}$ and $x_{e,wv}$ will be nonzero for any link $e \in \mathcal{E}$, and the links $e \in \mathcal{E}$ with $x_{e,vw}$ or $x_{e,wv}$ equal to 1 will form a single path from u_d to v_d. Finally, the variables Y^s (identifying the failure states in which the so specified flow fails) will also be binary because they are minimized and restricted from below by binary values.

Alternatively, the pricing problem can be converted into an instance of the *shortest-path problem with resource constraints* (SPPRC) [ID05]. For a given instance of the pricing problem, the corresponding instance of SPPRC is constructed as in Figure 1. The network graph of the original problem is preserved and is presented as a cloud in the figure. Each link in this graph has a primal cost ξ_e, as in the pricing problem. For the SPPRC problem instance we introduce $|\mathcal{S}|$ resources (corresponding to the failure states), and assume that link $e \in \mathcal{E}$ consumes one unit of resource $s \in \mathcal{S}$ if e fails in s, and nothing otherwise. The original network graph is extended by $|\mathcal{S}|$ additional nodes and $2 \cdot |\mathcal{S}|$ links denoted by a^s and b^s, $s \in \mathcal{S}$, as shown in Figure 1.

Links a^s have cost $\xi_{a^s} := 0$; these links consume a large amount $M > \sum_{e \in \mathcal{E}} \xi_e$ of resource s and nothing of all other resources. The links b^s have cost $\xi_{b^s} := (\lambda_d^s)^*$ and do not consume any resources. Let v'_d denote the last node as depicted in Figure 1. The objective in the resulting instance

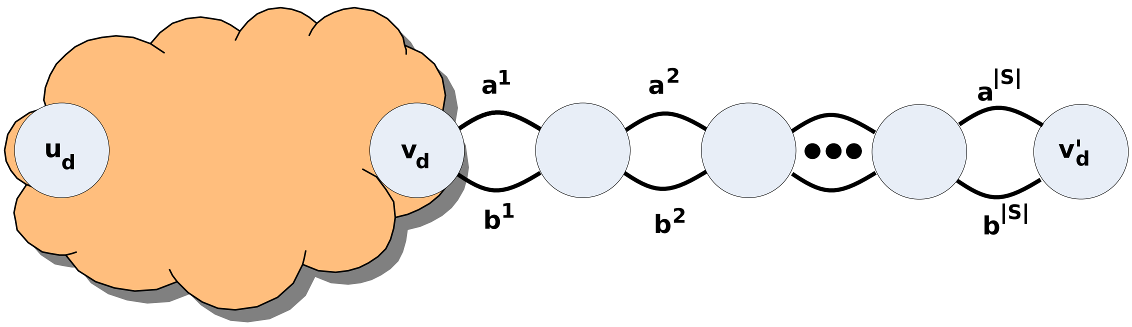

Figure 1: Transformation of the PD pricing problem to SPPRC

of SPPRC is to find a shortest path from u_d to v'_d in the transformed network with respect to the primal link costs which consumes at most M of any resource. This leads to a routing path with the same cost as described in the MIP objective (14b): the contribution of the found path to the first sum comes from the cloud, and the resource constraints of links a^s make sure that if a path uses state s somewhere in the cloud, link b^s must be used, which contributes a value of $(\lambda_d^s)^*$ to the path cost. Using this transformation, we can solve the pricing problem using algorithms developed for SPPRC, for example based on dynamic programming (see [ID05]).

Tomaszewski et al. [TPŻ08] have recently proved (by reduction to the fractional graph coloring problem FRACTIONAL-COLORING [GLS81]) that for multiple failures problem PD (1) is $\mathcal{NP}$-hard itself. They showed that PD is $\mathcal{NP}$-hard already for the single-demand version (2a) for a certain multiple link failure scenario containing $|\mathcal{V}|$ failure states. This yields an alternative proof that the pricing problem PRICE-PD is $\mathcal{NP}$-hard for a polynomial number of failure states.

4 Unrestricted restoration of flows – UR

Contrary to PD where path flows are fixed and cannot be changed, the *unrestricted restoration* (UR) concept allows all flows to be freely rearranged in a failure situation, so that in effect all the path flows are established from scratch using link capacity that survives the failure (i.e., stub release is assumed). In other words, the flow patterns in different failure states are completely decoupled. UR is also known as *global rerouting*. Using variables x_p^s to denote the flow on path p in state s, UR can be formulated as a non-compact linear program as follows:

$$\min \sum_{e\in\mathcal{E}} \xi_e y_e \tag{15a}$$

$$[\lambda_d^s \geq 0] \qquad \sum_{p\in\mathcal{P}_d^s} x_p^s \geq h_d^s \qquad d\in\mathcal{D},\ s\in\mathcal{S} \tag{15b}$$

$$[\pi_e^s \geq 0] \qquad \sum_{p\in\mathcal{P}_e^s} x_p^s \leq y_e \qquad e\in\mathcal{E},\ s\in\mathcal{S}_e \tag{15c}$$

$$x, y \geq 0. \tag{15d}$$

The problem dual to UR, which can be easily derived as in (3)–(5), is as follows:

$$\max \sum_{d\in\mathcal{D}}\sum_{s\in\mathcal{S}} h_d^s\lambda_d^s \tag{16a}$$

$$\sum_{s\in\mathcal{S}_e}\pi_e^s \le \xi_e \qquad e\in\mathcal{E} \tag{16b}$$

$$\lambda_d^s \le \sum_{e\in p}\pi_e^s \qquad d\in\mathcal{D},\ s\in\mathcal{S},\ p\in\mathcal{P}_d^s \tag{16c}$$

$$\lambda,\ \pi \ge 0. \tag{16d}$$

In any optimal solution (λ^*, π^*) of the dual, the value λ_d^{s*} is the length of a shortest path $p \in \mathcal{P}_d^s$ with respect to link metrics $\pi_e^{s*}, e \in \mathcal{E}$. Hence, even with multiple link failures, the pricing problem reduces to a shortest path problem for every demand $d \in \mathcal{D}$ and every network state $s \in \mathcal{S}$ separately with respect to the link metrics π_e^{s*} in the network surviving in state s. The whole pricing problem is thus solvable in polynomial time with respect to the number of nodes, links, and network states.

5 Situation-dependent restoration of affected flows – FD

From now on we assume that non-failing path flows must be preserved so that only failing flows are restored. This natural feature is referred to as *restricted reconfiguration*. In this section we assume failure-dependent restoration, i.e., the backup routing of a demand depends on the particular failure state. The failure-independent case is discussed in the next section.

With stub release, the capacity on surviving parts (stubs) of a failing routing path can be reused for backup paths; without stub release, it is reserved for the normal network state.

5.1 Situation-dependent restoration without stub release – FD-nSR

We first consider the case without stub release, i.e., capacity cannot be reused for backup flow. The primal problem FD-nSR can be stated in the following way (alternative formulations can be found in [PM04]).

$$\min \sum_{e\in\mathcal{E}}\xi_e y_e \tag{17a}$$

$$[\lambda_d \ge 0] \qquad \sum_{p\in\mathcal{P}_d} x_p \ge h_d \qquad d\in\mathcal{D} \tag{17b}$$

$$[\lambda_d^s \ge 0] \qquad \sum_{p\in\bar{\mathcal{P}}_d^s} x_p \le \sum_{q\in\mathcal{P}_d^s} x_q^s \qquad d\in\mathcal{D},\ s\in\mathcal{S}^* \tag{17c}$$

$$[\pi_e^s \ge 0] \qquad \sum_{p\in\mathcal{P}_e} x_p + \sum_{q\in\mathcal{P}_e^s} x_q^s \le y_e \qquad e\in\mathcal{E},\ s\in\mathcal{S}_e \tag{17d}$$

$$x, y \ge 0. \tag{17e}$$

The dual problem to FD-nSR derived according to (3)–(5) is as follows:

$$\max \sum_{d \in \mathcal{D}} h_d \lambda_d \tag{18a}$$

$$\sum_{s \in \mathcal{S}_e} \pi_e^s = \xi_e \qquad e \in \mathcal{E} \tag{18b}$$

$$\lambda_d^s \le \sum_{e \in q} \pi_e^s \qquad d \in \mathcal{D},\ s \in \mathcal{S}^*,\ q \in \mathcal{P}_d^s \tag{18c}$$

$$\lambda_d \le \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^s \qquad d \in \mathcal{D},\ p \in \mathcal{P}_d \tag{18d}$$

$$\lambda, \pi \ge 0. \tag{18e}$$

Denote the optimal dual variables by (λ^*, π^*). Similarly to unrestricted reconfiguration, it is easy to find improving protection paths $q \in \mathcal{P}_d^s$ for demand $d \in \mathcal{D}$ in any failure situation $s \in \mathcal{S}^*$ by solving a shortest path problem between the end-nodes of d in the surviving network with respect to link weights π_e^{s*}, and comparing the path length to λ_d^{s*}. To generate a primary path for demand $d \in \mathcal{D}$, on the other hand, a path minimizing $\sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^s$ has to be found. Thus, the pricing problem for FD-nSR is essentially the same as for PD and is therefore polynomial for single link failures [Wes00, Orl03, BNGK07]) and $\mathcal{NP}$-hard for multiple failures. As for PD, path generation for FD-nSR in the multiple link failure case can be approached by the SPPRC approach. An SPPRC algorithm specialized for FD-nSR is discussed in [DZP+08]. The algorithm shows very good effectiveness even for large networks.

In fact, for multiple failures problem FD-nSR is $\mathcal{NP}$-hard itself already in the single-demand version, as demonstrated in [TPŻ08] through a reduction to FRACTIONAL-COLORING.

5.2 Situation-dependent restoration using stub release – FD-SR

In problem FD-SR, we consider situation-dependent restoration of failed flows with stub release, i.e., capacity can be reused for backup flow. The corresponding LP formulation FD-SR reads:

$$\min \sum_{e \in \mathcal{E}} \xi_e y_e \tag{19a}$$

$$[\lambda_d \ge 0] \qquad \sum_{p \in \mathcal{P}_d} x_p \ge h_d \qquad d \in \mathcal{D} \tag{19b}$$

$$[\lambda_d^s \ge 0] \qquad \sum_{p \in \bar{\mathcal{P}}_d^s} x_p \le \sum_{q \in \mathcal{P}_d^s} x_q^s \qquad d \in \mathcal{D},\ s \in \mathcal{S}^* \tag{19c}$$

$$[\pi_e^s \ge 0] \qquad \sum_{p \in \mathcal{P}_e^s} x_p + \sum_{q \in \mathcal{P}_e^s} x_q^s \le y_e \qquad e \in \mathcal{E},\ s \in \mathcal{S}_e \tag{19d}$$

$$x, y \ge 0. \tag{19e}$$

The problem dual to FD-SR is as follows:

$$\max \sum_{d\in\mathcal{D}} h_d \lambda_d \tag{20a}$$

$$\sum_{s\in\mathcal{S}_e} \pi_e^s = \xi_e \qquad e\in\mathcal{E} \tag{20b}$$

$$\lambda_d^s \le \sum_{e\in q} \pi_e^s \qquad d\in\mathcal{D},\ s\in\mathcal{S}^*,\ q\in\mathcal{P}_d^s \tag{20c}$$

$$\lambda_d \le \sum_{e\in p}\Big(\sum_{s\in\mathcal{S}_p} \pi_e^s\Big) + \sum_{s\in\bar{\mathcal{S}}_p} \lambda_d^s \qquad d\in\mathcal{D},\ p\in\mathcal{P}_d \tag{20d}$$

$$\lambda,\pi \ge 0. \tag{20e}$$

Denote the optimal dual variables by (λ^*,π^*). Similarly to the previous case FD-nSR, it is easy to find improving protection paths $q\in\mathcal{P}_d^s$ for demand $d\in\mathcal{D}$ in any failure situation $s\in\mathcal{S}^*$ by solving the shortest path problem between the end-nodes of d in the surviving network with respect to link weights π_e^{s*}.

On the other hand, finding improving primary paths is $\mathcal{NP}$-hard already in the case of single link failures, as suggested in [Wes00, pp. 44 and 113], and shown later in [Orl03] and [MV04]. To solve the pricing problem for a fixed demand $d\in\mathcal{D}$ we have to find a path p from u_d to v_d minimizing

$$\langle p\rangle = \sum_{e\in p}\Big(\sum_{s\in\mathcal{S}_p} \pi_e^{s*}\Big) + \sum_{s\in\bar{\mathcal{S}}_p} \lambda_d^{s*}. \tag{21}$$

The difficulty in minimizing this sum stems from the term $\sum_{e\in p}\sum_{s\in\mathcal{S}_p}\pi_e^{s*}$. It makes that the weight of a path in the pricing problem is not merely the sum of independent link weights; instead, the contribution $\sum_{s\in\mathcal{S}_p}\pi_e^{s*}$ of a link to the path weight depends on the set of failure situations in which the path survives, and thus on the whole path. Under a full single link failure scenario, this path weight reduces to

$$\langle p\rangle = \sum_{e\in p}\Big(\sum_{f\notin p}\pi_e^{f*} + \lambda_d^{e*}\Big). \tag{22}$$

Maurras and Vanier [MV04] and Orlowski [Orl03] showed by reduction to the Hamilton path problem and to the max-cut problem, respectively, that already minimizing $\sum_{e\in p}\sum_{f\notin p}\pi_e^{f*}$ is $\mathcal{NP}$-hard, which immediately implies the $\mathcal{NP}$-hardness of minimizing (22).

To solve the pricing problem (21) exactly in practice, it can be formulated as a mixed-binary problem. The binary flow variables $X = (x_{e,vw}, x_{e,wv} : e\in\mathcal{E})$ describe a path from u_d to v_d; the capacity variables $Y = (Y^s : s\in\mathcal{S}^*)$ indicate in which states the path fails, and the coupling variables $Z = (Z_e^s : s \in \mathcal{S}, e \in \mathcal{E})$ indicate combinations of used edges with unused states.

$$\min \sum_{e\in\mathcal{E}} \sum_{s\in\mathcal{S}} \pi_e^{s*} Z_e^s + \sum_{s\in\mathcal{S}^*} \lambda_d^{s*} Y^s \tag{23a}$$

$$\text{s.t.} \quad \sum_{\substack{e\in\delta(v)\\ e=\{v,w\}}} (x_{e,vw} - x_{e,wv}) = \begin{cases} 0, & v \in \mathcal{V} \setminus \{u_d, v_d\} \\ 1, & v = u_d \\ -1, & v = v_d \end{cases} \tag{23b}$$

$$Y^s \le \sum_{\substack{e\in s\\ e=\{v,w\}}} (x_{e,vw} + x_{e,wv}), \quad s \in \mathcal{S} \tag{23c}$$

$$Y^s \ge x_{e,vw} + x_{e,wv}, \quad s \in \mathcal{S},\ e \in s,\ e = \{v,w\} \tag{23d}$$

$$Y^s + Z_e^s \ge x_{e,vw} + x_{e,wv}, \quad s \in \mathcal{S},\ e \in \mathcal{E},\ e = \{v,w\} \tag{23e}$$

$$X \in \{0,1\}^{2|\mathcal{E}|},\ 1 \ge Y \ge 0, Z \ge 0 \tag{23f}$$

Although Y and Z are binary indicator variables, their integrality conditions can be relaxed similarly as in (14) because they are minimized and bounded from below by integer values.

Notice that the MIP formulation (23) may have optimal solutions where the primary path determined by flow variables X visits a node more than once. In fact, adding a loop to an elementary path may augment the second sum in the objective function (23a) but reduce the first sum. This effect is due to the special structure of the pricing problem for primary paths with the considered mechanism FD-SR. Elementary paths, i.e., paths visiting each node at most once, can be enforced by additional constraints

$$\sum_{e\in\delta(v),\ e=\{v,w\}} x_{e,vw} \le 1, \qquad v \in \mathcal{V} \setminus \{u_d\}. \tag{24}$$

The authors of [BNGK07] (where a similar problem is considered) make a remark that already for the failure scenario containing just one single link failure, i.e., when $\mathcal{S} = \{\emptyset, \{e\}\}$ for some link $e \in \mathcal{E}$, the pricing problem limited to elementary paths, although polynomial, is non-trivial and must be solved by generating a shortest pair of node-disjoint paths.

Fadejeva [Fad03] investigated problem FD-SR with the additional restriction that every demand should be routed on a single path (which does not affect the pricing complexity). She showed that already for single link failures, the pricing problem is an $\mathcal{NP}$-hard shortest-path problem with negative weights if elementary paths are required, and solves it exactly using a quadratic shortest paths algorithm. As an alternative, she proposes a slight relaxation of the problem where routing paths with loops but with at most $|\mathcal{V}| - 1$ links are allowed. The pricing problem of this relaxation can be solved in polynomial time by searching for a shortest path in a layered auxiliary graph based on $|\mathcal{V}|$ copies of $\mathcal{G}$.

6 Situation-independent restoration of affected flows – FI

In this section we still assume that flows not affected by a failure situation are not rearranged. This time, however, the failed flows are restored in a failure-independent fashion, i.e., the restoration flow pattern is always the same and does not depend on the particular failure state.

6.1 Situation-independent restoration without stub release – FI-nSR

We first consider the case without stub release, where surviving but unused working capacity cannot be reused for backup flows. Using variables z_{pq} to denote the flow on backup path q of working path

p, problem FI-nSR can be formulated as follows:

$$\min \sum_{e\in\mathcal{E}} \xi_e y_e \tag{25a}$$

$$[\lambda_d \geq 0] \qquad \sum_{p\in\mathcal{P}_d} x_p \geq h_d \qquad d\in\mathcal{D} \tag{25b}$$

$$x_p \leq \sum_{q\in\mathcal{Q}_p} z_{pq} \qquad p\in\mathcal{P} \tag{25c}$$

$$[\pi_e^s \geq 0] \qquad \sum_{p\in\mathcal{P}_e} x_p + \sum_{d\in\mathcal{D}}\sum_{p\in\bar{\mathcal{P}}_d^s}\sum_{q\in\mathcal{Q}_{pe}} z_{pq} \leq y_e \qquad e\in\mathcal{E},\ s\in\mathcal{S}_e\setminus\{\emptyset\} \tag{25d}$$

$$x, y, z \geq 0. \tag{25e}$$

The problem dual to FI-nSR reads as follows:

$$\max \sum_{d\in\mathcal{D}} h_d\lambda_d \tag{26a}$$

$$\sum_{s\in\mathcal{S}_e\setminus\{\emptyset\}} \pi_e^s = \xi_e \qquad e\in\mathcal{E} \tag{26b}$$

$$\lambda_d \leq \sum_{e\in p}\xi_e + \sum_{e\in q}\Big(\sum_{s\in\bar{\mathcal{S}}_p}\pi_e^s\Big) \qquad d\in\mathcal{D},\ p\in\mathcal{P}_d,\ q\in\mathcal{Q}_p \tag{26c}$$

$$\lambda, \pi \geq 0. \tag{26d}$$

Note that dual variables corresponding to primal constraints (25c) do not appear in the dual formulation (26). The reason is that these constraints can be written as equalities without changing the optimal solution value; hence, they can be used to eliminate variables x_p from the formulation, and can be removed from the formulation themselves.

As will be explained in Section 7.2, the pricing problem for (26) is always difficult, i.e., also in the single link failure case. In fact, the pricing problem for FI-nSR consists in finding, for each demand $d\in\mathcal{D}$, a pair of failure-disjoint paths $c=(p,q)$ from u_d to v_d minimizing

$$\langle c\rangle = \sum_{e\in p}\xi_e + \sum_{e\in q}\Big(\sum_{s\in\bar{\mathcal{S}}_p}\pi_e^{s*}\Big). \tag{27}$$

Note that the link metrics for calculating the length of the primary path $p\in\mathcal{P}_d$ are equal to the true unit link costs ξ_e, while the link metrics for calculating the length of the backup path $q\in\mathcal{Q}_p$ are given by the dual cost $\sum_{s\in\bar{\mathcal{S}}_p}\pi_e^{s*}$. If a pair $c=(p,q)$ violates its dual constraint, then the corresponding column generation adds either one variable z_{pq} or two variables x_p and z_{pq} to the primal problem, depending on whether x_p is already included in the LP or not.

Again, the problem of minimizing (27) can be formulated as a mixed-binary problem. Alternatively, the pricing problem can be solved using the SPPRC method, as proposed in [SPR+07]. An SPPRC algorithm specialized for FI-nSR is also described in [DZP+08]. According to the authors of [DZP+08] their algorithm is more time-efficient than that of [SPR+07] and performs very well even on large networks.

Recently, Żotkiewicz et al. [ŻPT08] have directly shown that problem FI-nSR is $\mathcal{NP}$-hard, using a reduction from 2DIV-PATH [FHW78]. The latter problem is a special case of the $\mathcal{NP}$-hard Disjoint Connecting Paths problem [GJ79, problem ND40], and consists in finding two link-disjoint paths between two distinct pairs of nodes.

6.2 Situation-independent restoration with stub-release – FI-SR

The next case assumes situation-independent flow restoration using stub release. This restoration concept is perhaps not too practical — we discuss it here for completeness. The corresponding primal problem is as follows.

$$\min \sum_{e\in\mathcal{E}} \xi_e y_e \tag{28a}$$

$$[\lambda \geq 0] \qquad \sum_{p\in\mathcal{P}_d} x_p \geq h_d \qquad d \in \mathcal{D} \tag{28b}$$

$$[\pi_e^{\emptyset} \geq 0] \qquad \sum_{p\in\mathcal{P}_e} x_p \leq y_e \qquad e \in \mathcal{E} \tag{28c}$$

$$x_p \leq \sum_{q\in\mathcal{Q}_p} z_{pq} \qquad p \in \mathcal{P} \tag{28d}$$

$$[\pi_e^s \geq 0] \qquad \sum_{p\in\mathcal{P}_e^s} x_p + \sum_{d\in\mathcal{D}} \sum_{p\in\bar{\mathcal{P}}_d^s} \sum_{q\in\mathcal{Q}_{pe}} z_{pq} \leq y_e \qquad e \in \mathcal{E},\ s \in \mathcal{S}_e \setminus \{\emptyset\} \tag{28e}$$

$$x, y, z \geq 0. \tag{28f}$$

The problem dual to FI-SR takes the following form (as for the dual to FI-nSR, dual variables corresponding to primal constraints (28d) do not appear in the dual formulation):

$$\text{maximize} \sum_{d\in\mathcal{D}} h_d \lambda_d \tag{29a}$$

$$\sum_{s\in\mathcal{S}_e} \pi_e^s = \xi_e \qquad e \in \mathcal{E} \tag{29b}$$

$$\lambda_d \leq \sum_{e\in p} (\sum_{s\in\mathcal{S}_p} \pi_e^s) + \sum_{e\in q} (\sum_{s\in\bar{\mathcal{S}}_p} \pi_e^s) \qquad d \in \mathcal{D},\ p \in \mathcal{P}_d,\ q \in \mathcal{Q}_p \tag{29c}$$

$$\lambda, \pi \geq 0. \tag{29d}$$

For each demand $d \in \mathcal{D}$, the pricing problem consists of finding a pair of failure-disjoint paths $c = (p, q)$ from u_d to v_d minimizing

$$\langle c \rangle = \sum_{e\in p} (\sum_{s\in\mathcal{S}_p} \pi_e^{s*}) + \sum_{e\in q} (\sum_{s\in\bar{\mathcal{S}}_p} \pi_e^{s*}). \tag{30}$$

As discussed in Section 5.2, already minimizing the first sum under a single link failure scenario is $\mathcal{NP}$-hard, and hence also solving the whole pricing problem. Still, as we already know, this is not enough to be sure that the original problem FI-SR is $\mathcal{NP}$-hard itself. This fact has been recently proven in [PTZ09] by means of the construction used in [ŻPT08] for FI-nSR. As before, the problem of minimizing (30) can be formulated as a mixed-binary problem.

7 Single backup path protection and restoration – SB

Single backup path protection further restricts the restoration flow pattern by assuming that the entire primary flow is restored on one single backup path, used in all failure situations when the primary paths fails. In fact, this kind of restoration usually assumes that for each demand $d \in \mathcal{D}$, the entire demand volume is allocated to a single primary path. We note that in contrast to the previous problems, which can be modeled by linear programs, the single-path assumption requires a mixed-integer programming formulation.

Recall that pairs of primary-backup path are denoted by c, where $c \in \mathcal{C} := \bigcup_{d \in \mathcal{D}} \mathcal{C}_d$, and $\mathcal{C}_d$ is the set all candidate pairs for demand $d \in \mathcal{D}$. For each link $e \in \mathcal{E}$, the set of all pairs $c = (p, q)$ such that $e \in p$ is written as $\mathcal{C}_e^1$, and the set of all pairs c such that $e \in q$ as $\mathcal{C}_e^2$. The flow allocated to pair $c \in \mathcal{C}$ will be denoted by x_c.

7.1 SB with dedicated protection capacity – SB-D

In this section we consider the case with dedicated protection capacity, i.e., we assume that the protection capacity is not shared between different demands in different failure situations (as in the previous sections) but is reserved for each particular primary flow. This type of protection is commonly known as *1+1 protection*. The primal problem SB-D reads:

$$\min \sum_{e \in \mathcal{E}} \xi_e y_e \tag{31a}$$

$$[\lambda_d \geq 0] \qquad \sum_{c \in \mathcal{C}_d} x_c \geq 1 \qquad d \in \mathcal{D} \tag{31b}$$

$$[\pi_e \geq 0] \qquad \sum_{d \in \mathcal{D}} h_d \sum_{c \in \mathcal{C}_d \cap \mathcal{C}_e} x_c \leq y_e \qquad e \in \mathcal{E} \tag{31c}$$

$$y_e \geq 0,\, x_c \in \{0, 1\} \tag{31d}$$

As in problem PD discussed in Section 3, link capacities are dedicated to each particular demand. Therefore, the problem can be split and solved separately for each demand. Every such separate formulation is strongly unimodular and will yield a binary optimal vertex solution even if the integrality condition (31d) is relaxed to $0 \leq x_c \leq 1$. It is a straightforward exercise to derive the dual problem to the LP relaxation of SB-D, and to show that column generation consists in finding, for each demand $d \in \mathcal{D}$, a minimum cost cycle through its end-nodes with respect to the dual capacity cost π_e. This can be done by solving a min-cost-flow problem with capacities 1 and value 2 [Suu74]. This is the survivable analogon of a simple multi-commodity flow, where the pricing problem searches for minimum-cost paths with respect to the dual capacity cost.

If the total capacity y_e is replaced by working capacity y_e^1 for the primary flows and backup capacity y_e^2 for the backup flow with corresponding dual variables π'_e, π''_e, the pricing problem for each demand is to find a disjoint path pair $c = (p, q)$ minimizing $\langle c \rangle = \sum_{e \in p} \pi'_e + \sum_{e \in q} \pi''_e$. This problem is $\mathcal{NP}$-hard already for single link failures [XCX$^+$06]. If, however, the capacity of a link is defined exactly by its flow, as in model (31), both dual values can be replaced by the original cost ξ_e of link $e \in \mathcal{E}$, and both the pricing problem and the original problem reduce to finding a cycle $c \in \mathcal{C}_d$ minimizing $\sum_{e \in c} \xi_e$.

For the general case of multiple resource failures, finding a minimum-cost disjoint pair of paths is $\mathcal{NP}$-hard [Hu03], and so is the primal problem SB-D.

7.2 SB with shared protection capacity – SB-S

Now we assume that the pool of protection capacity is shared between the demands and situations, and arrive at the following relaxation of the primal problem SB-S.

$$\min \sum_{e\in\mathcal{E}} \xi_e(y_e^1 + y_e^2) \tag{32a}$$

$$[\lambda_d \geq 0] \qquad \sum_{r\in\mathcal{C}_d} x_r \geq h_d \qquad d \in \mathcal{D} \tag{32b}$$

$$[\pi_e \geq 0] \qquad \sum_{c\in\mathcal{C}_e^1} x_c \leq y_e^1 \qquad e \in \mathcal{E} \tag{32c}$$

$$[\pi_e^s \geq 0] \qquad \sum_{c\in\mathcal{C}_e^2,\ s\in\bar{\mathcal{S}}_p} x_c \leq y_e^2 \qquad e \in \mathcal{E},\ s \in \mathcal{S}_e \setminus \{\emptyset\} \tag{32d}$$

$$x, y \geq 0. \tag{32e}$$

Notice that the summation in constraint (32d) is taken over all path pairs whose backup path contains the considered link e and whose primary path fails in the considered state s.

Although looking quite different at first glance, the LP relaxation (32) is equivalent to problem FI-nSR of Section 6.1. The reason is that several pairs $c = (p, q) \in \mathcal{C}_d$ can have the same primary path $p \in \mathcal{P}_d$; all the corresponding paths q are used to protect path p in a bifurcated way. Also, it is not difficult to see that the dual problem of (32) is identical to the dual of FI-nSR.

$\mathcal{NP}$-hardness of path generation for SB-S under a multiple failure scenario follows from the result of [Hu03] for problem SB-D (see the previous section). $\mathcal{NP}$-hardness of path generation for (32) (and hence for FI-nSR) in the single-link failure case was demonstrated in [SPR+07]. In fact, it was shown there that for single-link failure scenarios the pricing problem for SB-S is pseudo-polynomial: polynomial with respect to the size of the network graph, and exponential with the number of failure states $|\mathcal{S}|$, see discussion about SPPRC at the end of Section 3.4.

In fact, in the literature the non-relaxed version of formulation (32) is more common (see for example the single backup path problem in [PM04]). In such a non-bifurcated version the entire demand volume is allocated to one single primary-backup path-pair. For this we have to use binary variables $u_c,\ c \in \mathcal{C}_d,\ d \in \mathcal{D}$, to indicate whether cycle c is used or not, constraints

$$\sum_{c\in\mathcal{C}_d} u_c = 1, \quad d \in \mathcal{D} \tag{33}$$

instead of (32b), and the substitution

$$x_c = u_c h_d, \quad d \in \mathcal{D},\ c \in \mathcal{C}_d. \tag{34}$$

We finally note that this non-bifurcated version of SB-S is almost always $\mathcal{NP}$-hard (see [ŻPT08]; it is polynomial only in the case of one single demand and a single link-failure scenario.

8 Summary of pricing problems

This section summarizes the essence of the pricing problems described in sections 3–7.

PD – path diversity

Parameters: $\lambda_d^s \geq 0$: dual cost of realizing demand $d \in \mathcal{D}$ in state $s \in \mathcal{S}$.
Pricing problem: For each demand $d \in \mathcal{D}$ find a u_d-v_d-path p minimizing

$$\langle p \rangle = \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^s. \tag{35}$$

Remarks: The pricing problem is $\mathcal{NP}$-hard for multiple failures. For single link failures it is polynomially solvable by searching for a shortest path for demand $d \in \mathcal{D}$ with respect to demand-dependent link weights $\gamma_d^e = \xi_e + \lambda_d^{\{e\}}$.

UR – unrestricted reconfiguration

Parameters: $\pi_e^s \geq 0$: dual cost of link $e \in \mathcal{E}$ in state $s \in \mathcal{S}$
Pricing problem: For each demand $d \in \mathcal{D}$ and each state $s \in \mathcal{S}$ find a u_d-v_d-path p minimizing

$$\langle p \rangle = \sum_{e \in p} \pi_e^s. \tag{36}$$

Remarks: The states are completely independent of each other. Hence, for any set of failure situations, improving paths for demand $d \in \mathcal{D}$ in state $s \in \mathcal{S}$ can be found in polynomial time by searching for a shortest path with respect to demand-independent link weights π_e^s .

FD-nSR – situation-dependent restricted restoration without stub release

Parameters: $\pi_e^s \geq 0$: dual cost of protection capacity of link $e \in \mathcal{E}$ in failure state $s \in \mathcal{S}^*$.
Pricing problem:

1. For each demand $d \in \mathcal{D}$ and for each failure state $s \in \mathcal{S}^*$ find a shortest backup u_d-v_d-path with respect to demand-independent link weights π_e^s . Denote the lengths of the resulting shortest paths by r_d^s ($d \in \mathcal{D}$, $s \in \mathcal{S}^*$).

2. For each demand $d \in \mathcal{D}$ find a working u_d-v_d-path p minimizing

$$\langle p \rangle = \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} r_d^s. \tag{37}$$

Remarks: The pricing problem is $\mathcal{NP}$-hard for multiple failures. For single link failures it reduces to a shortest-path problem path for each demand $d \in \mathcal{D}$ with respect to demand-dependent link weights $\gamma_d^e = \xi_e + r_d^{\{e\}}$. Observe that the pricing problem for working paths in FD-nSR has the same structure as problem (14) when λ_d^s is substituted with r_d^s.

FD-SR – situation-dependent restricted restoration with stub release

Parameters: $\pi_e^s \geq 0$: dual cost of link $e \in \mathcal{E}$ in state $s \in \mathcal{S}$.
Pricing problem:

1. For each demand $d \in \mathcal{D}$ and each failure state $s \in \mathcal{S}^*$ find a shortest backup u_d-v_d-path with respect to demand-independent link weights π_e^s ; denote its length by r_d^s.

2. For each demand $d \in \mathcal{D}$ find a working u_d-v_d-path p minimizing

$$\langle p \rangle = \sum_{e \in p} \sum_{s \in \mathcal{S}_p} \pi_e^s + \sum_{s \in \bar{\mathcal{S}}_p} r_s^d. \tag{38}$$

Remarks: The pricing problem for working paths is $\mathcal{NP}$-hard even for single link failures.

FI-nSR – situation-independent restricted restoration without stub release

Parameters: $\pi_e^s \geq 0$: dual cost of protection capacity of link $e \in \mathcal{E}$ in state $s \in \mathcal{S}^*$.
Pricing problem: For each demand $d \in \mathcal{D}$ find a pair of failure-disjoint paths $c = (p, q)$ from u_d to v_d minimizing

$$\langle c \rangle = \sum_{e \in p} \xi_e + \sum_{e \in q} \sum_{s \in \bar{\mathcal{S}}_p} \pi_e^s. \tag{39}$$

Remarks: The pricing problem is $\mathcal{NP}$-hard even for single link failures.

FI-SR – situation-independent restricted restoration with stub release

Parameters: $\pi_e^s \geq 0$: dual cost of link $e \in \mathcal{E}$ in state $s \in \mathcal{S}$.
Pricing problem: For each demand $d \in \mathcal{D}$ find a pair of failure-disjoint paths $c = (p, q)$ from u_d to v_d minimizing

$$\langle c \rangle = \sum_{e \in p} \sum_{s \in \mathcal{S}_p} \pi_e^s + \sum_{e \in q} \sum_{s \in \bar{\mathcal{S}}_p} \pi_e^s. \tag{40}$$

Remarks: The pricing problem is $\mathcal{NP}$-hard even for single link failures.

SB-D - single backup path restoration with dedicated protection capacity

Parameters: $\xi_e \geq 0$: primal unit cost of link $e \in \mathcal{E}$.
Pricing problem: For each demand $d \in \mathcal{D}$ find a pair of failure-disjoint paths $c = (p, q)$ from u_d to v_d minimizing

$$\langle c \rangle = \sum_{e \in p} \xi_e + \sum_{e \in q} \xi_e. \tag{41}$$

Remarks: The pricing problem is polynomial for a any single link failure scenario but $\mathcal{NP}$-hard for a general failure scenario, including the scenario containing the failures of all pairs of links.

SB-S - single backup path restoration with shared separated protection capacity

Remarks: The pricing problem for the relaxation of SB-S is the same as for FI-nSR, and hence $\mathcal{NP}$-hard already for single link failures.

Overview table

Table 1 summarizes the complexity of the considered pricing problems in a single and multiple link failure setting together with references where this complexity has been shown. The letters (a) to (e) refer to the five classes of pricing problems defined in the introduction. For some survivability concepts, the $\mathcal{NP}$-hardness for multiple failures follows from the $\mathcal{NP}$-hardness for single failures. The second column states whether non-failing but unused capacity is released in a failure situation (stub release) or reserved for the normal network state, and whether capacity is dedicated to a particular demand or shared between demands.

problem	release, sharing	restoration type	path/pair length minimize for each $d \in \mathcal{D}$	dual constraint	failure type/complexity single	failure type/complexity multiple
PD	release	none	$\langle p \rangle = \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \lambda_d^s$	$\sum_{s \in \mathcal{S}_p} \lambda_d^s \leq \sum_{e \in p} \xi_e$	polynomial (a)	$\mathcal{NP}$-hard (c)
	dedicated				[Wes00, WOZ+05]	Sec. 3.4
UR	release	unrestricted	$\langle p \rangle = \sum_{e \in p} \pi_e^s, \; s \in \mathcal{S}$	$\sum_{s \in \mathcal{S}_e} \pi_e^s = \xi_e$	polynomial (a)	polynomial (a)
	shared	(from scratch)			Sec. 4	Sec. 4
FD-nSR	no release	restricted	$\langle q^s \rangle = \sum_{e \in q} \pi_e^s, \; s \in \mathcal{S}^*$	$\sum_{s \in \mathcal{S}_e \setminus \{\emptyset\}} \pi_e^s = \xi_e$	polynomial (a)	$\mathcal{NP}$-hard (c)
	shared	failure-dependent	$\langle p \rangle = \sum_{e \in p} \xi_e + \sum_{s \in \bar{\mathcal{S}}_p} \langle q^s \rangle$		[Orl03]	Sec. 5.1
FD-SR	release	restricted	$\langle q^s \rangle = \sum_{e \in q} \pi_e^s, \; s \in \mathcal{S}^*$	$\sum_{s \in \mathcal{S}_e} \pi_e^s = \xi_e$	$\mathcal{NP}$-hard (d)	$\mathcal{NP}$-hard (d)
	shared	failure-dependent	$\langle p \rangle = \sum_{e \in p} \sum_{s \in \mathcal{S}_p} \pi_e^s + \sum_{s \in \bar{\mathcal{S}}_p} \langle q^s \rangle$		[Orl03, MV04]	
FI-nSR	no release	restricted	$\langle c \rangle = \sum_{e \in p} \xi_e + \sum_{e \in q} \sum_{s \in \bar{\mathcal{S}}_p} \pi_e^s$	$\sum_{s \in \mathcal{S}_e \setminus \{\emptyset\}} \pi_e^s = \xi_e$	$\mathcal{NP}$-hard (e)	$\mathcal{NP}$-hard (e)
	shared	failure-independent			[SPR+07]	
FI-SR	release	restricted	$\langle c \rangle = \sum_{e \in p} \sum_{s \in \mathcal{S}_p} \pi_e^s + \sum_{e \in q} \sum_{s \in \bar{\mathcal{S}}_p} \pi_e^s$	$\sum_{s \in \mathcal{S}_e} \pi_e^s = \xi_e$	$\mathcal{NP}$-hard (e)	$\mathcal{NP}$-hard (e)
	shared	failure-independent			Sec. 6.2	
SB-D	no release	single-path	$\langle c \rangle = \sum_{e \in p} \xi_e + \sum_{e \in q} \xi_e$	—	polynomial (b)	$\mathcal{NP}$-hard (b)
	dedicated	failure-independent	$\bar{\mathcal{S}}_p \cap \bar{\mathcal{S}}_q = \emptyset$		[Suu74]	[Hu03]
SB-S	no release	single-path	$\langle c \rangle = \sum_{e \in p} \xi_e + \sum_{e \in q} \sum_{s \in \bar{\mathcal{S}}_p} \pi_e^s$	$\sum_{s \in \mathcal{S}_e \setminus \{\emptyset\}} \pi_e^s = \xi_e$	$\mathcal{NP}$-hard (e)	$\mathcal{NP}$-hard (e)
	shared	failure-independent			[SPR+07]	

Table 1: Complexity of pricing problems for different survivability concepts

9 Concluding remarks

In the previous sections we have investigated the complexity of the path generation problem under single and multiple link failure scenarios for various survivability mechanisms. This section provides a classification of the results and discusses some possible extensions.

9.1 Classification of pricing problems

It can be seen in Table 1 that the pricing problems of all the considered survivability mechanisms have certain regularities. They are all composed of five types of minimization problems that determine the complexity of the pricing problem:

1. classical shortest-path problem for each network state (polynomial for both single and multiple failure scenarios)
2. classical shortest failure-disjoint pair of paths problem (polynomial for single failures, $\mathcal{NP}$-hard for multiple failures)
3. shortest path problem where the path length is the sum of given prices of the failure states in which the path fails (polynomial for single failures, $\mathcal{NP}$-hard for multiple failures)
4. shortest path problem with link weights depending on the set of the failure states in which the path survives ($\mathcal{NP}$-hard already for single failures), and
5. shortest path pair problem with link weights of the backup path depending on the set of failure states in which the primary path fails ($\mathcal{NP}$-hard already for single failures).

Notice that some pricing problems contain two of these subproblems; in this case, the overall complexity is determined by more difficult subproblem. If the formulation contains separate flow variables for each failure state, the pricing problem for backup paths is always easy because a flow variable for a particular failure state is affected only by the constraints of that state. The hard part, if any, is finding improving working paths or disjoint pairs of working and backup paths because the variables for working paths are coupled by conditions corresponding to all the network states.

9.2 Flow cost

In the previous discussions, we have always assumed that there is no flow cost but only capacity cost. The effect of adding flow cost depends on the cost structure: if all routing paths for a specific demand have the same cost c per flow unit, the parameter c only appears as a constant in the pricing problem and does not affect its complexity. If, on the other hand, the flow cost depends on the specific routing path, even the pricing problem of a simple multi-commodity flow routing without survivability restrictions is a weighted longest-path problem, which is in general $\mathcal{NP}$-hard [GJ79, problem ND29].

9.3 Directed graphs, node failures, and connected components

Directed graphs Up to now we have assumed undirected graphs. In fact, in most cases switching to directed links or bi-directed links does not affect the complexity of the pricing problems. The reader is advised to consider each case one by one and confirm this statement.

Node failures In practical applications, node failures may have to be considered in addition to link failures. We have not discussed node failures in the main part of this paper because they can be simulated by link failures in an auxiliary graph $\mathcal{G}'$, which is constructed from the original graph $\mathcal{G}$ as follows.

First, each original undirected link $e \in \mathcal{E}$ is replaced by a pair of antiparallel directed arcs e' and e'' with $\xi'_e = \xi''_e = \xi_e$. In a second step, every node $v \in \mathcal{V}$ is split into two nodes v' and v'' connected by a single (dummy) directed arc e_v of cost $\xi_{e_v} = 0$, as shown in Figure 2.

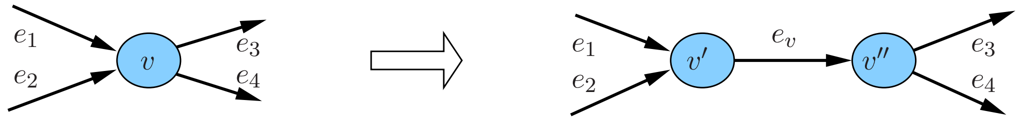

Figure 2: Node splitting transformation

Any path $p = \{e_1, e_2, \ldots, e_n\}$ from node s to node t in the original graph $\mathcal{G}$ corresponds to a directed path p' from s to node t in the new graph $\mathcal{G}'$ traversing arcs e'_i or e''_i ($i = 1, 2, \ldots, n$, whichever is consistent with the direction of the path) and the dummy arcs e_v for those nodes $v \in \mathcal{V}$ in $\mathcal{G}$ which belong to p (and vice versa). With this construction, the failure of link e_v in $\mathcal{G}'$ corresponds precisely to the failure of node $v \in \mathcal{V}$ in $\mathcal{G}$, and the failures of a single link with antiparallel counterpart in $\mathcal{G}'$ correspond to the single link failures in $\mathcal{G}$. Thus, any failure state in the original graph involving a certain number of links *and* nodes can be modeled by means of a failure involving only links in the transformed graph. Obviously, the transformation is polynomial in the size of the graph. It follows that pricing for single node failures *and* single link failures has the same complexity as pricing for single link failures.

Alternatively, assuming only elementary paths, single-node failures can be directly taken into account in the pricing problem by distributing half of the dual node weight among the incident edges, as described for the path diversity problem in [Wes00, WOZ+05].

Certainly, when considering node failures we should bear in mind that the volumes of all demands $d \in \mathcal{D}$ incident with a given node $v \in \mathcal{V}$ cannot be realized at all in a failure state $s \in \mathcal{S}$ involving node v. This can be reflected by setting $h_d^s = 0$ for such pairs (d, s) in PD and UR, and by skipping the constraints that force restoring affected flows for such demands and states. In the node-splitting transformation described above, it can be reflected by mapping a u-v-demand in $\mathcal{G}$ to a u''-v'-demand in $\mathcal{G}'$.

Failures of span 1 In Section 3.4, we have shown the $\mathcal{NP}$-hardness of PD for general multiple failures by reduction to MC-PATH, identifying colors with shared risk link groups. Coudert et al. [CDP+06] showed that MC-PATH is still polynomial if (1) every edge has only one color (such a graph is called *monochromatic*), and (2) the edges of each color form a connected component of the graph. By defining the span of a color c as the number of connected components of the subgraph induced by the edges of c, the second condition means that every color has span 1. As a special case, this condition holds if all links with the same color are incident to the same node. Recall that also the pricing problem for PD can be solved in polynomial time for single node failures. This raises the question whether it is also polynomial if the failing edges form a connected component of the graph, i.e., every shared risk link group is of span 1.

Unfortunately, this is not the case in general. The reason is that in our setting, a link may be affected by several failure situations, i.e., it has several colors. Although such a multichromatic graph can be transformed into a monochromatic graph by replacing each multi-colored edge by a series of single-colored edges, this transformation does not necessarily preserve the span of a color because its span in the resulting graph depends on the ordering of the single-colored links. Even if the edges of each color form a connected component in the original graph, it can not always be transformed into a monochromatic graph where each color still has span 1.

But even if every SRLG has span 1, i.e., every link is affected by at most one failure state, there cannot be a polynomial algorithm which solves all pricing problems (13) for PD. The reason is that if such an algorithm existed, we could also use it to solve any pricing problem for PD in networks with failures of larger span, which contradicts its $\mathcal{NP}$-hardness in the general case [CDP$^+$06]. In fact, any network with colors of span larger than 1 can be transformed into a span-1 network by connecting the components of each color with additional links with cost $\xi_e = \infty$. Such an artificial link will never be used by a shortest path unless the original graph is disconnected. With a polynomial number of colors (e.g., corresponding to all dual link failures), this transformation is polynomial, and the shortest paths in the original and modified graphs are the same. Hence, solving the pricing problem for PD (as for FD-nSR and FD-SR) is $\mathcal{NP}$-hard for general multiple link failure scenarios even if all of them correspond to connected components.

9.4 Complexity of the primal vs. complexity of the pricing problem

The $\mathcal{NP}$-hardness of the pricing problem associated with a specific non-compact linear programming formulation of a problem $\mathbb{P}$ does not necessarily mean that $\mathbb{P}$ is $\mathcal{NP}$-hard itself. For example, there may exist a compact LP formulation of $\mathbb{P}$. What we only know for sure is that if the pricing problem for some non-compact LP formulation of $\mathbb{P}$ is polynomial, then $\mathbb{P}$ is polynomial. This follows from the so called *separation theorem* (see [GLS88]). It can be shown that for each non-compact LP formulation considered in this report, an optimal solution of the pricing problem provides an optimal solution of the separation problem for the dual problem, i.e., provides the best (in the sense of the separation theorem) inequality separating the current optimal dual solution from the polyhedron of the dual problem corresponding to the full primal problem. Vice versa, if we can show that $\mathbb{P}$ is $\mathcal{NP}$-hard, then any LP formulation for it must be non-compact with a non-polynomial pricing problem (unless $\mathcal{P} = \mathcal{NP}$).

However, for each problem $\mathbb{P}$ considered in this report, except for FD-SR (whose $\mathcal{NP}$-hardness has not been proven as yet), a stronger result than the above holds: $\mathbb{P}$ is $\mathcal{NP}$-hard if, and only if, the pricing problem related to the considered non-compact formulation of $\mathbb{P}$ is $\mathcal{NP}$-hard. Let us recall that, as discussed in the previous sections, the pricing problem is $\mathcal{NP}$-hard for the following cases:

- PD, FD-nSR, SB-D for multiple-link failure scenarios
- FD-SR, FI-nSR, FI-SR (and linear relaxation of SB-S which is equivalent to FI-nSR) for both single or multiple failure scenarios.

For the remaining cases, i.e., for

- PD, FD-nSR, SB-D for single-link failure scenarios
- UR for single or multiple failure scenarios

the pricing problem is polynomial.

Since $\mathcal{NP}$-hardness has been proved for PD and FD-nSR [TPŻ08], SB-D [Hu03] (for multiple failures), and for FI-nSR/SB-S [ŻPT08] and FI-SR [PTZ09] (already for single link failure scenarios), the above statement is valid.

We have also found that in all the cases of non-compact LP formulations considered in this report the pricing problem is polynomial if, and only if, we can also establish a compact node-link LP formulation of the considered problem. The "only if" implication is easy to prove in general, since a compact node-link LP formulation shows that the problem is polynomial (as discussed above). However, the opposite implication is not that easy to prove in general. We have exhibited it for the discussed problems by providing explicit compact node-link formulations for all the relevant cases, i.e., for PD, FD-nSR, and SB-D (single link failures), and for UR (all failure scenarios). Below we present a selection of such formulations. For ease of notation we will assume directed links and demands; changing these formulations to undirected flows is a straightforward exercise.

For PD and a single link failure scenario $\mathcal{S} \subseteq \{\{e\} \mid e \in \mathcal{E}\} \cup \{\emptyset\}$, the compact node-link formulation is as follows.

$$\text{minimize} \quad \sum_{e\in\mathcal{E}} \xi_e \sum_{d\in\mathcal{D}} x_{ed} \tag{42a}$$

$$\text{s.t.} \quad \sum_{e\in\delta^+(v)} x_{ed} - \sum_{e\in\delta^-(v)} x_{ed} = \begin{cases} 0, & v \in \mathcal{V} \setminus \{u_d, v_d\} \\ X_d, & v = u_d \\ -X_d, & v = v_d \end{cases} \qquad d \in \mathcal{D},\ v \in \mathcal{V} \tag{42b}$$

$$X_d \geq h_d \qquad d \in \mathcal{D} \tag{42c}$$

$$X_d - x_{ed} \geq h_d^{\{e\}} \qquad d \in \mathcal{D},\ \{e\} \in \mathcal{S} \tag{42d}$$

$$x, X \geq 0. \tag{42e}$$

Above, variable x_{ed} denotes the flow realizing demand $d \in \mathcal{D}$ on link $e \in \mathcal{E}$, and variable X_d is the total flow realized for demand $d \in \mathcal{D}$. Also, $\delta^+(v)$ denotes the set of all links in $e \in \mathcal{E}$ outgoing from node $v \in \mathcal{V}$, and $\delta^-(v)$ the set of all links $e \in \mathcal{E}$ incoming to node $v \in \mathcal{V}$.

If we wished to take single node failures directly into account, i.e., without transforming the network graph as described in Section 9.3, then we would add to formulation (42) the following constraint for each failing node $v \in \mathcal{V}$:

$$X_d - \sum_{e\in\delta^+(v)} x_{ed} \geq h_d^{\{v\}}, \qquad d \in \mathcal{D},\ v \notin \{u_d, v_d\}. \tag{43}$$

Writing down compact node link formulations for UR (for an arbitrary failure scenario, using state-dependent link-flow variables x_{eds}) is a straightforward exercise. For FD-nSR and a single-link failure scenario $\mathcal{S} \subseteq \{\{e\} \mid e \in \mathcal{E}\} \cup \{\emptyset\}$ the relevant formulation reads:

$$\text{minimize} \quad \sum_{e\in\mathcal{E}} \xi_e y_e \tag{44a}$$

$$\text{s.t.} \quad \sum_{e\in\delta^+(v)} x_{ed} - \sum_{e\in\delta^-(v)} x_{ed} = \begin{cases} 0, & v \in \mathcal{V} \setminus \{u_d, v_d\} \\ h_d, & v = u_d \\ -h_d, & v = v_d \end{cases} \qquad d \in \mathcal{D},\ v \in \mathcal{V} \tag{44b}$$

$$\sum_{e\in\delta^+(v)\setminus s} z_{eds} - \sum_{e\in\delta^-(v)\setminus s} z_{eds} = \begin{cases} 0, & v \in \mathcal{V} \setminus \{u_d, v_d\} \\ x_{ed}, & v = u_d \\ -x_{ed}, & v = v_d \end{cases} \qquad d \in \mathcal{D},\ v \in \mathcal{V},\ s \in \mathcal{S} \tag{44c}$$

$$\sum_{d\in\mathcal{D}} (x_{ed} + z_{edf}) \leq y_e \qquad e \in \mathcal{E},\ s \in \mathcal{S} \setminus \{e\} \tag{44d}$$

$$x, z \geq 0. \tag{44e}$$

For SB-D and the single-link failure scenario $\mathcal{S} \subseteq \{\{e\} \mid e \in \mathcal{E}\} \cup \{\emptyset\}$ the relevant node-link formulation is as follows.

$$\text{minimize} \quad \sum_{e \in \mathcal{E}} \xi_e \sum_{d \in \mathcal{D}} h_d x_{ed} \tag{45a}$$

$$\text{s.t.} \quad \sum_{e \in \delta^+(v)} x_{ed} - \sum_{e \in \delta^-(v)} x_{ed} = \begin{cases} 0, & v \in \mathcal{V} \setminus \{u_d, v_d\} \\ 2, & v = u_d \\ -2, & v = v_d \end{cases} \qquad d \in \mathcal{D} \tag{45b}$$

$$1 \geq x \geq 0. \tag{45c}$$

Note that the above formulation is unimodular (see [AMO93]) and therefore its optimal vertex solutions are integer, as required for SB-D. The above formulations can be adapted to undirected links and demands using the notation already applied in the pricing problem formulations (14) and (23).

Finally, we note that all of the considered $\mathcal{NP}$-hard pricing problems (see Table 1) become polynomial (and treatable for example by the Dijkstra shortest path algorithm) when the lists of primary paths are fixed, so that the primary paths are given and are not subject to optimization/generation. This is a quite common situation in practical telecommunication network design.

Acknowledgment

The authors wish to express their gratitude to Andreas Bley, David Coudert, Arie Koster, Dritan Nace, Thomas Stidsen, Artur Tomaszewski, Marie-Émilie Voge, Roland Wessäly, and Mateusz Żotkiewicz for valuable discussions and helpful remarks on the presented subject.

This research has been supported by the DFG Research Center MATHEON, Polish Ministry of Science and Higher Education (grant N517 397334), Swedish Research Council (grant 621-2006-5509), and by the European project COST Action 293 "Graphs and Algorithms in Communication Networks".

References

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[BNGK07] A. Bashllari, D. Nace, E. Gourdin, and O. Klopfenstein. The MMF rerouting computation problem. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, 2007.

[CDP+06] D. Coudert, P. Datta, S. Perennes, H. Rivano, and M-E. Voge. Shared risk resource group: Complexity and approximability issues. *Parallel Processing Letters*, 2006. To appear. Preliminary version available as INRIA technical report 5859, 2006.

[DS98] G. Dahl and M. Stoer. A cutting plane algorithm for multicommodity survivable network design problems. *INFORMS Journal on Computing*, 10(1):1–11, 1998.

[DZP+08] M. Dzida, M. Zagożdżon, M. Pióro, T. Śliwinski, and W. Ogryczak. Path generation for a class of survivable network design problems. In *The 3rd NGI 2008 Conference on Next Generation Internet Networks, Cracow, Poland*, 28-30 April 2008.

[Fad03] L. Fadejeva. Ein Column-Generation-Ansatz zur Kostenoptimierung von ausfallsicheren Kommunikationsnetzen mit Single-Path-Routing. M.Sc. thesis, Technische Universität Berlin, August 2003.

[FHW78] S. Fortune, J.E. Hopcroft, and J.C. Wyllie. The directed subgraph homeomorphism problem. Technical report, Ithaca, NY, USA, 1978.

[GJ79] M. R. Garey and D. R. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co, New York, 1979.

[GLS81] Martin Grötschel, Lászlo Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[GLS88] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer Verlag, Berlin, 1988.

[HMS07] R. Hassin, J. Monnot, and D. Segev. Approximation algorithms and hardness results for labeled connectivity problems. *Journal of Combinatorial Optimization*, 14(4):437–453, November 2007.

[Hu03] J.Q. Hu. Diverse routing in optical mesh networks. *IEEE Trans. Com.*, 51(3):489–494, 2003.

[ID05] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosier, and M.M. Solomon, editors, *Column Generation*, pages 33–65. Springer, 2005.

[KZ07] A.M.C.A. Koster and A. Zymolka. Demand-wise shared protection and multiple failures. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, April 2007.

[KZJH05] A.M.C.A. Koster, A. Zymolka, M. Jäger, and R. Hülsermann. Demand-wise shared protection for meshed optical networks. *Journal of Network and Systems Management*, 13(1):35–55, March 2005.

[Las70] L. Lasdon. *Optimization Theory for Large Systems*. MacMillan, 1970.

[Min86] M. Minoux. *Mathematical Programming: Theory and Algorithms*. John Wiley & Sons, 1986.

[MV04] J-F. Maurras and S. Vanier. Network synthesis under survivability constraints. *4OR*, 2(1):53–67, March 2004.

[NW88] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

[Orl03] S. Orlowski. Local and global restoration of node and link failures in telecommunication networks. M.Sc. thesis, Technische Universität Berlin, February 2003. `http://www.zib.de/orlowski/`.

[PM04] M. Pióro and D. Medhi. *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Morgan Kaufman, 2004.

[PTZ09] M. Pióro, A. Tomaszewski, and M. Żotkiewicz. Computational complexity of optimization problems related to resilient networks with flow restoration. *Submitted to INFOCOM 2009*, 2009.

[SCT01] J. Strand, A. L. Chiu, and R. Tkach. Issues for routing in the optical layer. *IEEE Communications Magazine*, pages 81–87, 2001.

[SPR+07] T. Stidsen, B. Petersen, K.B. Rasmussen, S. Spoorendonk, M. Zachariasen, F. Rambach, and M. Kiese. Optimal routing with single backup path protection. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*, 2007.

[Suu74] J. W. Suurballe. Disjoint paths in a network. *Networks*, 4:125–145, 1974.

[TPŻ08] A. Tomaszewski, M. Pióro, and M. Żotkiewicz. On the complexity of resilient network design. *Submitted to Networks*, 2008.

[Wes00] R. Wessäly. *Dimensioning Survivable Capacitated NETworks*. PhD thesis, Technische Universität Berlin, April 2000.

[WOZ+05] R. Wessäly, S. Orlowski, A. Zymolka, A.M.C.A. Koster, and C. Gruber. Demand-wise shared protection revisited: A new model for survivable network design. In *Proceedings of the 2nd International Network Optimization Conference (INOC 2005), Lisbon, Portugal*, pages 100–105, March 2005.

[XCX+06] D. Xu, Y. Chen, Y. Xiong, C. Qiao, and X. He. On the complexity of and algorithms for finding the shortest path with a disjoint counterpart. *IEEE/ACM Transactions on Networking*, 14(1):147–158, 2006.

[YVJ05] S. Yuan, S. Varma, and J.P. Jue. Minimum-color path problems for reliability in mesh networks. In *Proceedings of the 24th IEEE Infocom 2005, Miami, USA*, volume 4, pages 2658–2669, March 2005.

[ŻPT08] M. Żotkiewicz, M. Pióro, and A. Tomaszewski. On the complexity of resilient network optimization. *Proc. of the 5th Polish-German Teletraffic Symposium, PGTS'2008, Berlin*, 2008.