Combinatorial optimization in networks with Shared Risk Link Groups

David Coudert, Stéphane Pérennes, Hervé Rivano, Marie-Emilie Voge

Combinatorial optimization in networks with Shared Risk Link Groups

David Coudert[1,2], Stéphane Perennes[1,2], Hervé Rivano[1,3] and Marie-Emilie Voge[4]

[1]Inria, France
[2]Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, 06900 Sophia Antipolis, France
[3]UrbaNet, INRIA/CITI, INSA-Lyon F-69621, France
[4]Univ. Lille, CRIStAL, CNRS UMR 9189, 59650 Villeneuve d'Ascq, France

March 13, 2016

Abstract

The notion of Shared Risk Link Groups (SRLG) captures survivability issues when a set of links of a network may fail simultaneously. The theory of survivable network design relies on basic combinatorial objects that are rather easy to compute in the classical graph models: shortest paths, minimum cuts, or pairs of disjoint paths. In the SRLG context, the optimization criterion for these objects is no longer the number of edges they use, but the number of SRLGs involved. Unfortunately, computing these combinatorial objects is NP-hard and hard to approximate with this objective in general. Nevertheless some objects can be computed in polynomial time when the SRLGs satisfy certain structural properties of locality which correspond to practical ones, namely the star property (all links affected by a given SRLG are incident to a unique node) and the span 1 property (the links affected by a given SRLG form a connected component of the network). The star property is defined in a multi-colored model where a link can be affected by several SRLGs while the span property is defined only in a mono-colored model where a link can be affected by at most one SRLG. In this paper, we extend these notions to characterize new cases in which these optimization problems can be solved in polynomial time. We also investigate the computational impact of the transformation from the multi-colored model to the mono-colored one. Experimental results are presented to validate the proposed algorithms and principles.

Keywords: Multi-layer networks, Shared Risk Link Group, colored graphs, labeled graphs, complexity, algorithms.

1 Introduction

The motivation of this work stems from the concept of Shared Risk Link Groups (SRLG) that captures wide-spreading faults in networks. More precisely, SRLGs are sets of links that may fail simultaneously if a given event (risk) occurs. The scope of this concept is very broad. For instance, it can be a set of fiber links of an optical backbone network that are physically buried at the same location and so that could be cut simultaneously, or cards failures arising in router nodes. It can also represent radio links in access and backhaul networks subject to environmental conditions affecting signal transmission, or traffic jam in road networks.

Optimization problems under SRLG constraints have received a large attention in the recent years. Most of the research effort was focussed on integrating SRLG constraints into classical

survivability problems such as shared or dedicated protection mechanisms. Several Integer Linear Program (ILP) formulations and heuristic algorithms for survivable routing under SRLG constraints [14,15,18,24,29–31,34–37] or duct failures [38] (a particular kind of SRLG associating all fiber links laid down in the same duct with a failure) have been proposed. Other works have focussed on optimizing the local and global [25], and average [9] reliability of connections, or determining the best set of p-cycles [22].

All these studies rely on basic problems like path, cuts, or pairs of disjoint paths. But their definitions have to be revisited in the SRLG context. Indeed, the reliability of a path is measured by the number of SRLGs it contains since the failure of a single SRLG along that path is sufficient to interrupt a connection. Similarly, a cut is expressed by the number of SRLGs whose simultaneous failures disconnect the graph. All these basic problems are known to be NP-hard in general (see e.g. [8, 12, 13, 18, 19, 23, 24, 37]). However, some problems can be solved in polynomial time when the topological structure of the SRLGs satisfy particular properties [1, 8, 11]. In particular, two notions have been introduced to characterize polynomial cases: the *span of a SRLG* [8], counting the number of connected components induced by this SRLG, and the *star property* [23], assessing whether all links impacted by a given SRLG are incident to a unique node or not. Such failure scenario corresponds to risks like the cut of a conduit containing links issued from a node, or card failures in a router node.

In this paper, we propose a deeper analysis of the impact of these topological properties on the computational complexity of optimization problems in this context. This enables us to exhibit more polynomial cases.

For that, we use the graph theoretic framework proposed in [8, 11–13, 23, 37]. The network is modeled by an undirected graph and each SRLG is modeled by a color assigned to some of the edges of this graph. Such a model is called a *multi-colored graph* since several colors may be assigned to a given edge, if the corresponding link of the network is subject to several SRLGs. The specific case where each edge is assigned a unique color (that is, each link belongs to only one SRLG) is strictly equivalent to the *Labeled Graph* model [3–7, 10, 16, 17, 19, 20, 27, 33, 40]. The term *label* is used in these papers instead of *color*, which was prefered in the context of SRLGs, hence the two equivalent terminologies of labeled and colored graphs.

Our results In this paper, we establish the following results:

- The minimum colored path and the minimum colored cut problems can be solved in time $2^k \cdot n^{O(1)}$ in colored graphs, where k is the number of colors with span larger than one (colors inducing more than one connected component).
- The problem of determining two color disjoint paths is NP-hard in colored graphs as soon as one color has span larger than one.
- The minimum colored path problem can be solved in time $2^k \cdot n^{O(1)}$ in multi-colored graphs with k non-star colors.
- The minimum colored cut problem is NP-hard in multi-colored graphs with the star property.
- There is an algorithm deciding if a multi-colored graph can be transformed into a colored graph with at most k colors of span more than one in time $3^k \cdot n^{O(1)}$. However, the problem of maximizing the number of colors of span one in the transformation is NP-hard.

Organization of the paper. We first present in Section 2 the network model and the optimization problems studied in this paper. In Sections 3, 4, and 5 we prove our results mentionned above. Section 6 is devoted to an experimental assessment of the algorithms and principles presented in this paper. In particular, we show that the exact transformation algorithm proposed in Section 5 is fast in practice. Furthermore, our experiments show that the algorithms proposed for minimum colored path and cut problems are competitive with respect to other exact methods.

2 Preliminaries

In this section, we fix the notations that will be used throughout this paper. We also present formally the models and the optimization problems under investigation.

2.1 Notation and definitions

The network is modeled by an undirected graph $G = (V, E)$, where V is the set of vertices (one vertex per node of the network) and E is the set of edges (one edge per link of the network). As said in the introduction, we model a SRLG by a color, and we will abuse the notations denoting $\mathcal{C}$ both the set of all colors and the coloring function associating an edge with the colors (SRLGs) it belongs to. So we denote by:

- $\mathcal{C}$: set of colors (set of SRLGs).
- $\mathcal{C}(e)$: set of colors assigned to edge e.
- $E(c)$: set of edges colored with color $c \in \mathcal{C}$ (set of links sharing risk c).

The formal definition of a (multi-)colored graph is as follows.

Definition 1 (Multi-colored graph)**.** *A multi-colored graph is a pair $(G, \mathcal{C})$, where $G = (V, E)$ is an undirected graph and $\mathcal{C}$ is a coloring of E.*

In the specific case where each edge has at most one color (each link is subject to a unique risk of failure), we will use the terminology of mono-colored graph which is equivalent to that of labeled graphs.

Definition 2 (Mono-colored graph)**.** *A mono-colored graph is a pair $(G, \mathcal{C})$ where each edge has at most one color.*

We will say that a color c is *incident* to a vertex u if that color is assigned to an edge incident to vertex u, and we say that two colors are *adjacent* if they are incident to the same vertex. The *colored degree* of a given vertex u, denoted $\delta_{\text{col}}(u)$, is its number of incident colors. The *maximum colored degree* of a (multi-)colored graph $(G, \mathcal{C})$ is $\Delta_{\text{col}}(G) = \max_{u \in V} \delta_{\text{col}}(u)$.

Clearly, a multi-colored graph can be transformed into a mono-colored graph (replacing each edge e with a path P of length $|\mathcal{C}(e)|$ and mapping the colors to the edges of P), and conversely. However, as will be explained in Section 5, such transformations may change drastically the computational complexity of the optimization problems.

2.2 Optimization problems

We now present the optimization problems studied in this paper.

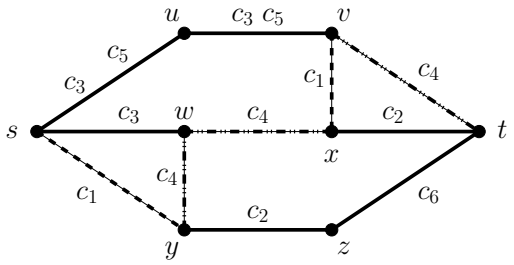


Figure 1: The MC-sw-Path is not part of the MC-st-Path.

2.2.1 Minimum colored st-path, MC-st-Path

As already said, in the SRLG context the reliability of a link is measured by the SRLGs it belongs to, that is the number of colors associated to the edge representing the link. For example, in Fig. 1, edge (u, v) is subject to both risks c_3 and c_5. By extension, one can define a colored st-path as follows.

Definition 3 (Colored st-path)**.** *A colored st-path in a multi-colored graph $(G, \mathcal{C})$, is a path P between s and t in G. Its cost is the number of colors appearing in the path, that is $|\cup_{e\in P} \mathcal{C}(e)|$.*

Finding the most reliable path from vertex s to vertex t corresponds to finding the colored st-path of minimum cost.

Problem 1 (MC-st-Path)**.** *Given a multi-colored graph $(G, \mathcal{C})$ and two vertices s and t in G, the minimum-colored st-path problem consists in finding a colored st-path P with minimum cost.*

In Fig. 1, we observe that the colored st-paths (s, w, x, t) and (s, u, v, t) have both 3 edges and 3 colors, and the colored st-path (s, y, w, x, v, t) has 5 edges and only 2 colors. This shows that a shortest path in terms of colors is different from a shortest path in terms of edges. Furthermore, we observe that the path from s to w followed by the minimum colored st-path is (s, y, w) and has 2 colors, while the minimum colored sw-path is the edge (s, w) with color c_3. This shows that in a multi-colored graph, the sub-path of a (color wise) shortest path is not a (color wise) shortest path itself, while this property is fundamental in (hop wise) shortest paths algorithms. It is therefore not surprizing that the MC-st-Path problem has been proved NP-hard [2, 24] and hard to approximate [8, 16].

2.2.2 Minimum colored st-cut, MC-st-Cut and MC-Cut

A link of the network may be broken by the failure of one of the SRLGs it belongs to. In the example of Fig. 1, any of the risks c_3 and c_5 is sufficient to break link (u, v). A colored st-cut is therefore defined as follows.

Definition 4 (Colored st-cut)**.** *A colored st-cut in a multi-colored graph $(G, \mathcal{C})$ is a set $C \subseteq \mathcal{C}$ of colors such that s and t are in distinct connected components of $G \setminus S$, where $S = \cup_{c\in C} E(c)$.*

Finding the weakness of the network between vertices s and t hence corresponds to finding the colored st-cut which contains the minimum number of SRLGs.

Problem 2 (MC-st-Cut)**.** *Given a multi-colored graph $(G, \mathcal{C})$ and two vertices s and t, the MC-st-Cut problem consists in finding a colored st-cut C such that $|C|$ is minimized.*

In Fig. 1, the set of colors $\{c_1, c_4\}$ constitute a MC-st-Cut although it has 5 edges, and the set $\{c_1, c_3\}$ is a MC-st-Cut with only 3 edges. The MC-st-Cut problem has been proved NP-complete and hard to approximate in general in [8, 19, 27, 39, 40].

The reliability of a network is associated with its minimum cut. So we have:

Definition 5 (Colored cut)**.** *A colored cut in a multi-colored graph $(G, \mathcal{C})$ is a set $C \subseteq \mathcal{C}$ of colors such that $G \setminus S$, with $S = \cup_{c \in C} E(c)$, has at least two connected components.*

Problem 3 (MC-Cut)**.** *Given a multi-colored graph $(G, \mathcal{C})$, the MC-Cut problem consists in finding a minimum size colored cut.*

In other words, a MC-Cut is the minimum MC-st-Cut over all pairs of vertices s, t. In Fig. 1, color c_3 is a MC-Cut since it disconnects vertex u from the rest of the graph. The MC-Cut problem is NP-complete in general in multi-colored graphs [13]. However, the complexity of the problem in colored graphs is still an open question. Yet, polynomial-time solvable cases have recently been identified [39]. More precisely, the MC-Cut problem can be solved in polynomial time when the graph has bounded treewidth, or is planar, or the maximum number of edges with the same color is bounded by a constant $f_{\max}$ (e.g., $\max_{c \in \mathcal{C}} E(c) \leq f_{\max}$).

2.2.3 Color disjoint paths problem, CDP

The most common mechanism to protect a connection against any single failure is to assign to it a protection path that is disjoint from its working path. The diverse routing problem is thus to determine (at least) two disjoint st-paths [26]. In the SRLG context, the working and protection paths must not only be edge disjoint, but also SRLG disjoint. For stronger survivability requirement (i.e. protection against multiple simultaneous failure events), more than two SRLG disjoint paths may be required. It is thus important to determine the maximum number of diversely routed colored st-paths. More formally, we have:

Definition 6 (Color disjoint st-paths)**.** *In a multi-colored graph $(G, \mathcal{C})$, k colored st-paths $P_1, P_2, \ldots, P_k$ are color disjoint if $(\cup_{e \in P_i} \mathcal{C}(e)) \cap (\cup_{e \in P_j} \mathcal{C}(e)) = \emptyset$ for all $1 \leq i < j \leq k$.*

Problem 4 (k-CDP)**.** *Given a multi-colored graph $(G, \mathcal{C})$, and two vertices s and t, do there exist k color disjoint st-paths?*

Problem 5 (Max diverse colored st-path)**.** *Given a multi-colored graph $(G, \mathcal{C})$, and two vertices s and t, find the maximum number k of color disjoint st-paths.*

Problem 4 has been proved NP-complete for $k \geq 2$ in [18] and later in [37]. Problem 5 has also been proved NP-complete in [18].

Notice that in the SRLG context, the max flow - min cut relation (i.e., the maximum number of edge disjoint st-paths equals the minimum st-cut) does not hold. To see that, take a graph with two vertices s and t and three parallel edges e_1, e_2, and e_3, respectively colored with $\{c_1, c_2\}$, $\{c_1, c_3\}$, and $\{c_2, c_3\}$. The MC-st-Cut of this graph has two colors (e.g., c_1 and c_2), but it is not possible to find two color disjoint st-paths.

3 Span in mono-colored graphs

In this section, we concentrate on optimization problems in mono-colored graphs (an edge has a unique color or SRLG). In this context, an important notion for studying the complexity of colored problems is the *span* of a color. This notion, introduced in [8], allows the characterization of cases in which optimization problems in mono-colored graphs can be solved in polynomial time.

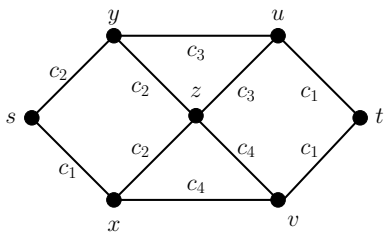


Figure 2: A colored graph. The minimum colored st-path is s, x, v, t with colors c_1, c_4. The minimum colored st-cut is induced by color c_1.

Definition 7 (Span of a color)**.** *In a mono-colored graph $(G, \mathcal{C})$, the span of color $c \in \mathcal{C}$, denoted* span(c)*, is the number of connected components induced by the edges with color c.*

For example in Fig. 2 we have span$(c_1) = 2$ and span$(c_2) = 1$.

A thorough analysis of the complexity and hardness of the problems presented in Section 2.2 with respect to the maximum value of the span of the colors, that is $\max_{c \in \mathcal{C}}$ span(c), has been conducted in [8]. In particular, it has been proved that the MC-st-Path, MC-st-Cut, MC-Cut, and 2-CDP problems can be solved in polynomial time in colored graphs in which all colors have span 1. However, except may be for the MC-Cut problem which is still open, all these problems are NP-hard and difficult to approximate when the maximum span of the colors is bounded by a constant $k > 1$, and become even harder to approximate when there is no assumption on the maximum span of the colors.

In this section, we consider the number of colors of span greater than 1 instead of the maximum value of the span of the colors. This allows us to prove that the MC-st-Path, MC-st-Cut, and MC-Cut problems can be solved in polynomial time when the number of colors of span > 1 is bounded by a constant. Concerning the color disjoint paths problem, we strengthen previous hardness results, proving that the 2-CDP problem is NP-hard when exactly one color has span > 1.

3.1 MC-st-Path and MC-st-Cut

Theorem 1. *The MC-st-Path problem (Problem 1) can be solved in time $2^k n^{\mathcal{O}(1)}$ in mono-colored graphs with k colors of span greater than 1.*

Proof. The proof given in [8] that the MC-st-Path problem can be solved in polynomial time in colored graphs when all colors have span one assumes that all edges have a color. The proof builds an auxiliary graph H_{st} with one vertex per color and an edge between two vertices if the corresponding colors are incidents in $(G, \mathcal{C})$. It also adds vertices s and t and edges between s (resp., t) and vertices associated with colors incident to s (resp., t) in $(G, \mathcal{C})$. Then, it deduces a minimum colored st-path P_c in $(G, \mathcal{C})$ from a shortest st-path P in H_{st} . In fact, P_c is a st-path in the subgraph of G induced by the colors associated with the vertices of P.

We can extend the proof of [8] to the case where some edges of $(G, \mathcal{C})$ have no color (the creation and usefulness of colorless edges will be explained in the remainder of the proof) as follows. We build an instance $(G, \mathcal{C}')$ where $\mathcal{C}'$ contains all colors in $\mathcal{C}$ plus one new color per non-colored edge in $(G, \mathcal{C})$. Then, we build the auxiliary graph H'_{st} of $(G, \mathcal{C}')$, assign weight 1 to vertices of H'_{st} associated with colors in $\mathcal{C}$ and weight 0 to the other vertices, compute a minimum cost st-path P' in H'_{st}, and deduce a minimum colored st-path P'_c in $(G, \mathcal{C}')$ as previously. Obviously, P'_c has as few vertices associated with colors in $\mathcal{C}$ as possible, otherwise contradicting the fact that it is a shortest st-path in H'_{st}. Consequently, a minimum colored

st-path P_c in $(G, \mathcal{C})$ has the same set of edges as P'_c and for cost the number of vertices of P' that are associated with colors in $\mathcal{C}'$.

Now, we proceed by enumerating over all colored graphs $(G^i, \mathcal{C}_{=1})$, where $\mathcal{C}_{=1}$ is the set of colors of span one, $\mathcal{C}_{>1} \subseteq \mathcal{C}$ is the set of colors of span greater than 1, $\left\{\mathcal{C}^i_{>1},\ 0 \leq i \leq 2^{|\mathcal{C}_{>1}|} - 1\right\}$ is the set of subsets of $\mathcal{C}_{>1}$ ($\mathcal{C}^0_{>1} = \emptyset$, and $\mathcal{C}^{2^{|\mathcal{C}_{>1}|}-1}_{>1} = \mathcal{C}_{>1}$), and $G^i = (V, E^i)$ is the subgraph of G in which all edges colored with colors in $\mathcal{C}_{>1} \setminus \mathcal{C}^i_{>1}$ are removed (i.e., $E^i = E \setminus \cup_{c \in \mathcal{C}_{>1} \setminus \mathcal{C}^i_{>1}} E(c)$). In other words, we ensure that the computed paths do not use any color from the set $\mathcal{C}_{>1} \setminus \mathcal{C}^i_{>1}$. So, $(G^i, \mathcal{C}_{=1})$ is a colored graph where some edges have no color: the edges with a color in $\mathcal{C}^i_{>1}$ in $(G, \mathcal{C})$. We then compute a minimum colored st-path $P^i_{=1}$ in $(G^i, \mathcal{C}_{=1})$ and deduce the colored st-path P^i in $(G, \mathcal{C}_{=1} \cup \mathcal{C}^i_{>1})$ with cost $cost(P^i) = cost(P^i_{=1}) + |\mathcal{C}^i_{>1}|$. In other words, we assume that P^i uses all colors in $\mathcal{C}^i_{>1}$ even though it is not necessary. If there is no st-path $P^i_{=1}$ in $(G^i, \mathcal{C}_{=1})$, we set $cost(P^i) = +\infty$.

We claim that the path P with minimum cost over all paths P^i is a minimum colored st-path in $(G, \mathcal{C})$. Firstly, since we enumerate over all colored graphs $(G^i, \mathcal{C}_{=1})$, at least one of them results in a path P^i with non infinite cost (in the worse case $(G^{2^{|\mathcal{C}_{>1}|}-1}, \mathcal{C}_{=1})$), and so we have a valid colored st-path. Now, consider a minimum colored st-path P of $(G, \mathcal{C})$ with cost γ and suppose it uses the colors of some set $\mathcal{C}^i_{>1}$. The corresponding path $P^i_{=1}$ in $(G^i, \mathcal{C}_{=1})$ is a minimum colored st-path with cost $w = \gamma - |\mathcal{C}^i_{>1}|$. Indeed, suppose there exists a path with cost $w' < w$ in $(G^i, \mathcal{C}_{=1})$, then the corresponding path in $(G, \mathcal{C})$ will have weight $w' + |\mathcal{C}^i_{>1}| < \gamma$, a contradiction.

Summarizing, the minimum colored st-path P in $(G, \mathcal{C})$ is associated with a minimum colored st-path P^i of some $(G, \mathcal{C}_{=1} \cup \mathcal{C}^i_{>1})$, and so with a minimum colored st-path $P^i_{=1}$ in $(G^i, \mathcal{C}_{=1})$. The construction of $(G^i, \mathcal{C}_{=1})$ can be done in polynomial time, the computation of P^i can also be done in polynomial time, and there are $2^{|\mathcal{C}_{>1}|}$ graphs $(G^i, \mathcal{C}_{=1})$. When $|\mathcal{C}_{>1}|$ is bounded by a constant k, the overall computation time is $2^k n^{\mathcal{O}(1)}$. □

In the example of Fig. 2, we have $\mathcal{C}_{>1} = \{c_1\}$. Also, there is no st-path from s to t in $(G^0, \mathcal{C}_{=1})$, or said equivalently, the minimum cost is $+\infty$. In $(G^1, \mathcal{C}_{=1})$, the minimum cost st-path $P^1_{=1}$ is s, x, v, t of cost 1 to which we associate the path s, x, v, t in $(G, \mathcal{C})$ of minimum cost $1 + 1 = 2$ and using colors c_1 and c_4.

Theorem 2. *The MC-st-Cut problem (Problem 2) can be solved in time $2^k n^{\mathcal{O}(1)}$ in mono-colored graphs with k colors of span greater than 1.*

Proof. The proof is similar to the proof of Theorem 1, except that instead of computing a path in $(G^i, \mathcal{C}_{=1})$, we compute a minimum colored st-cut. So, we simply need to extend the proof of [8] in order to compute a minimum colored st-cut in a colored graph in which some edges have no color, which is done computing a minimum *weighted* vertex st-cut in the auxiliary graph build in the proof of [8] instead of a minimum vertex st-cut. □

Remark that the cost of a cut is independent of the span (number of connected components) of the colors. The same observation holds for MC-st-Path and MC-Cut.

In the example of Fig. 2, we compute the minimum colored st-cuts $S^0_{=1}$ and $S^1_{=1}$ respectively in $(G^0, \mathcal{C}_{=1})$ and $(G^1, \mathcal{C}_{=1})$ to determine the MC-st-Cut. $S^0_{=1}$ has cost $+\infty$ since s and t are disconnected when removing edges with color c_1, and so $S^0 = S^0_{=1} \cup \{c_1\}$ has also cost $+\infty$. $S^1_{=1}$ has cost 0 since the edges in $E(c_1)$ have no color in $(G^1, \mathcal{C}_{=1})$, and so $S^1 = \{c_1\}$ has cost 1. The MC-st-Cut is induced by color c_1.

Corollary 1. *The MC-Cut problem (Problem 3) can be solved in time $2^k n^{\mathcal{O}(1)}$ in mono-colored graphs with k colors of span greater than 1.*

Proof. The MC-Cut is the minimum over all pairs of vertices $\{s, t\}$ of the MC-st-Cut. □

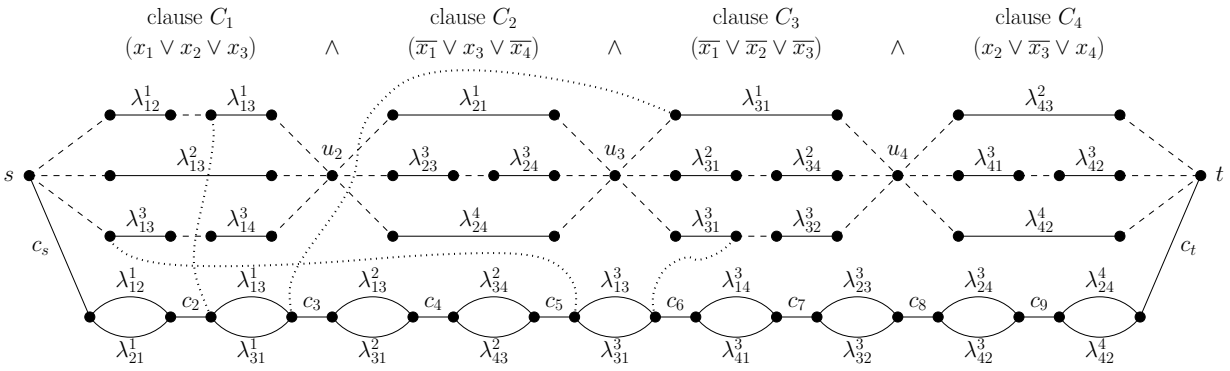


Figure 3: Reduction from 3SAT to 2-CDP. Only four crossing edges are represented (colors λ^1_{13}, λ^1_{31}, λ^3_{13} and λ^3_{31}). All edges drawn with line style "- - - -" are assigned color c_1.

3.2 Color Disjoint Paths

The 2-CDP problem can be solved in polynomial time in colored graphs in which all colors have span 1 [8, 11]. However, as soon as one color has span > 1, the 2-CDP problem is NP-hard.

Theorem 3. *There is a polynomial time reduction from 3SAT to the 2-CDP problem (Problem 4 for $k = 2$) in mono-colored graphs with exactly one color of span > 1. Consequently, finding two color disjoint paths in such a colored graph is NP-hard.*

Proof. Consider an instance of 3SAT with n variables x_i, $1 \leq i \leq n$, and m clauses C_j, $1 \leq j \leq m$, each being a disjunction of at most three literals. The 3SAT problem consists in finding a truth assignment satisfying the m clauses.

From this instance we construct an instance of the 2-CDP problem as follows. Let us first define the color set $\mathcal{C}$. It contains a color c_1, which will be the only one of span > 1 and two colors, λ^i_{jk} and λ^i_{kj}, for each pair of clauses (C_j, C_k) such that x_i belongs to C_j and $\overline{x_i}$ belongs to C_k. We will add some other colors later to complete the construction.

The mono-colored graph $(G, \mathcal{C})$ is divided into two parts, each one connecting the vertices s and t of G. The first part represents the clauses and is thus divided into m sections delimited by $m+1$ vertices $u_1 = s, u_2, \ldots, u_{m+1} = t$. For each $j \in \{1, \ldots, m\}$ and for each literal x_i (resp. $\overline{x_i}$) belonging to C_j there is a path connecting u_j to u_{j+1} containing one edge of each color λ^i_{jk} (resp. λ^i_{kj}) such that C_k contains $\overline{x_i}$ (resp. x_i). We insert an edge of color c_1 at the beginning and end of each $u_j u_{j+1}$-paths, and between the consecutive edges of each color λ^i_{jk} (resp. λ^i_{kj}) such that C_k contains $\overline{x_i}$ (resp. x_i), as shown on Fig. 3.

The second part of the graph is a kind of path composed of an alternating succession of simple and multiple edges. This path is long enough to contain a multiple edge for each pair of clauses (C_j, C_k) such that x_i belongs to C_j and $\overline{x_i}$ belongs to C_k, for each variable x_i. Such a multiple edge consists of two parallel edges of colors respectively λ^i_{jk} and λ^i_{kj}. The simple edges have all a distinct and new color reduced to one edge, and the path begins with a simple edge with color denoted c_s and ends with a simple edge with color denoted c_t.

The two parts are connected in s and t but also through each color of the λ^i kind as follows. Indeed, note that till now each of these colors has span 2 with exactly one edge in the first part of the graph and one edge in the second. Since we want c_1 to be the only color of span > 1, these colors have to be of span 1: we add an edge of color λ^i_{jk} between one extremity of the first occurrence of the color λ^i_{jk} and one extremity of its second occurrence. In the following we refer to such edges as *crossing edges*.

Before proving the equivalence in terme of positiveness of the two instances, let us show that if there are two color disjoint paths in the colored graph $(G,\mathcal{C})$, then one of them belongs entirely to the first part of the graph and the other to the second part. The first remark is that s and t both have colored degree two with several edges of color c_1 in the first part and one edge in the second part of the graph with a color different from c_1. Thus if there are two color disjoint st-paths, there is necessarily one (*the first path*) that begins and ends with color c_1 in the first part of the graph, and an other one (*the second path*) which begins with the edge of color c_s and ends with the edge of color c_t in the second part. Then, since the first path has to use color c_1 and since each edge of the λ^i color kind is inserted between two edges of color c_1, the second path cannot use any crossing edge in order to go on in the first part as it will use as next edge one of color c_1 which is forbidden. Consequently, according to the topology of the second part, the second path has to use all the simple edges of the second part of the graph and exactly one color from each pair $(\lambda^i_{jk}, \lambda^i_{kj})$. This also implies that the first path cannot use any crossing edge because then it has to use either a simple edge of the second part or an other crossing edge of color λ^i_{jk} associated with the color λ^i_{kj} of the crossing edge it has used to come in the second part. Note that there is only one crossing edge of a given color λ^i_{jk}. So, to go from one part of the graph to the other and back, two distinct λ kind colors are necessarily used.

Therefore, we have established that if two color disjoint st-paths exist in the graph we have constructed, one of them is contained in the first part of the graph, and the other in the second part. We now need to prove that if there are two color disjoint st-paths in the graph, there is a truth assignment satisfying all clauses of the instance of 3SAT.

Assume that the first path uses the subpath corresponding to literal x_i (resp. $\overline{x_i}$) between u_j and u_{j+1}. Then it uses all the color λ^i_{jk} (resp. λ^i_{kj}) such that x_i (resp. $\overline{x_i}$) belongs to C_j and $\overline{x_i}$ (resp. x_i) belongs to C_k. This implies that the second path has to use all the color λ^i_{kj} (resp. λ^i_{jk}) to reach t and be color disjoint from the first path. Consequently, the first path cannot use these last colors and thus cannot use the subpath corresponding to $\overline{x_i}$ (resp. x_i) between u_k and u_{k+1} for all the k concerned.

Therefore, when there are two color disjoint st-paths in the graph we can deduce a truth assignment by setting to true any literal corresponding to a subpath used by the first path between s and t. From the preceding paragraph we are sure that no variable can be assigned to true (literal x_i true) and false (literal $\overline{x_i}$ true) at the same time. In addition, since the first path has to go through every section representing a clause, at least one literal per clause is set to true, which means that all the clauses are satisfied. If some variables are not assigned any value, then they are not necessary to satisfy any clause, and we can assign them any value without changing the answer to the 3SAT instance. Consequently, if there are two color disjoint st-paths connecting s and t, there is a truth assignment satisfying all the clauses of the 3SAT instance.

We now only need to prove that when there is a truth assignment satisfying all the clauses of the 3SAT instance, there are also two color disjoint st-paths connecting s and t. So we consider a truth assignment satisfying the instance of 3SAT. We define the first path to be composed of all the subpaths associated with literal set to true by this assignment. Since the assignment satisfies all clauses, the path connects s and t. In addition, by construction of the graph, the first path can use only one color from each pair $(\lambda^i_{jk}, \lambda^i_{kj})$, the other color of each pair is then available for the second path which thus exists and that concludes the proof. □

Corollary 2. *The k-CDP (Problem 4) and max diverse colored st-path (Problem 5) problems are NP-hard in colored graphs with at least one color of span > 1.*

4 Star property

We now consider the case where all the links affected by a risk share an end-vertex. Such failure scenario corresponds to risks like the cut of a conduit containing links issued from a node, or card failures in a router node. It leads to optimization problems in multi-colored graphs with a peculiar property called the *star property* [23]. In this section we show how to exploit this locality property to solve some problems in polynomial time.

Definition 8 (Star color)**.** *In a multi-colored graph $(G, \mathcal{C})$, a color $c \in \mathcal{C}$ is called a* star color *if all edges of $E(c)$ are incident to the same vertex, called the* center of the color *(i.e., they form a star). When $|E(c)| = 1$, one can choose any of the end-vertices of edge $e \in E(c)$ as the center of color c. Other colors are called* non star colors*.*

Definition 9 (Star property)**.** *A multi-colored graph has the* star property *if it has only star colors.*

The multi-colored graph of Fig. 4a has the star property, which is not the case for the example of Fig. 1 where colors c_1, c_2, and c_4 are not stars.

Notice that the star property in mono-colored graphs implies that all colors have span 1, but the converse is false (see Section 5).

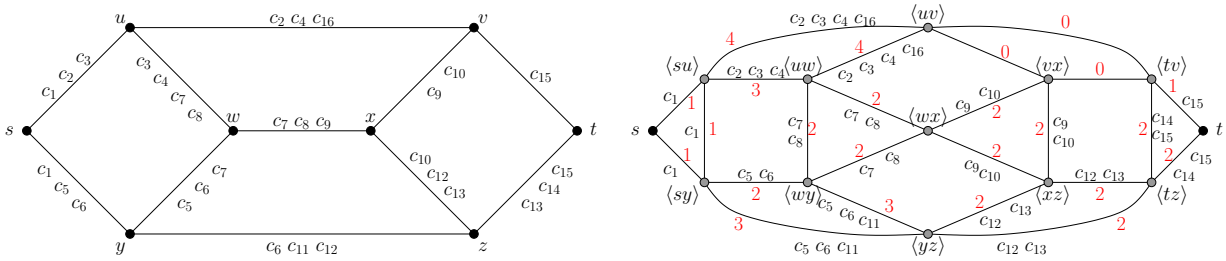
4.1 Graphs with the star property

It has been claimed in [23] that the 2-CDP problem can be solved in polynomial time in multi-colored graphs with the star property. However, it has later been proved in [1] that in fact the 2-CDP problem is NP-Hard in this context. We now show that the star property can be exploited to construct an auxilliary multi-colored graph in which shortest path algorithms such as Dijkstra can be used without any modification as explained after.

Theorem 4. *The MC-st-Path problem can be solved in polynomial time in multi-colored graphs with the star property.*

Proof. We use an auxilliary multi-colored graph $(G_{st}, \mathcal{C}_{st})$ that depends on the vertices s and t needing to be connected by a minimum color path in the initial multi-colored graph $(G, \mathcal{C})$. Each st-path in the auxilliary graph G_{st} corresponds to an st-path of G of the same cost. This auxilliary graph is constructed as follows. We assume here that all colors have a center, even if they appear on a single edge of G in which case we can choose as center one of its end-vertices. We associate with each edge $\{u, v\}$ of G a vertex $\langle uv \rangle$ in G_{st}, and there is an edge $\{\langle uv \rangle, \langle vw \rangle\}$ in G_{st} if G has both edges $\{u, v\}$ and $\{v, w\}$. Then we assign color $c \in \mathcal{C}$ to edge $\{\langle uv \rangle, \langle vw \rangle\}$ if $c \in \mathcal{C}(\{u, v\}) \cup \mathcal{C}(\{v, w\})$ is centered in v. Next, we add vertex s to G_{st}, edges between s and $\langle su \rangle$ for all neighbors u of s in G, and we assign color c to edge $\{s, \langle su \rangle\}$ if $c \in \mathcal{C}(\{s, u\})$ is centered in s. We also add a vertex t to G_{st}, edges between t and $\langle tu \rangle$ for all neighbors u of t in G, and we assign color c to edge $\{t, \langle tu \rangle\}$ if $c \in \mathcal{C}(\{t, u\})$ is centered in t. Finally, we set the weight of edge e in G_{st} to $|\mathcal{C}_{st}(e)|$ (edges such that $\mathcal{C}_{st}(e) = \emptyset$ have weight 0).

The next step is to prove that Dijkstra's algorithm computes an optimal colored st-path in G_{st}. To do so, observe that there exists a shortest path which does not use two edges of the form $\{\langle uv \rangle, \langle vw \rangle\}$ and $\{\langle vw \rangle, \langle vx \rangle\}$ (i.e., three vertices with a common symbol v). Indeed, by construction the edge costs satisfy the triangular inequality (colors of $\{u, v\}$ and $\{v, x\}$ centered in v appear either on $\{\langle uv \rangle, \langle vw \rangle\}$ or on $\{\langle vw \rangle, \langle vx \rangle\}$). Therefore, a path given by the algorithm uses at most one edge containing a given color, and so the cost of a shortest path in G_{st} is exactly the number of colors used by the corresponding path in $(G, \mathcal{C})$ by construction. □



(a) $(G, \mathcal{C})$ has the star property.

(b) $(G_{st}, \mathcal{C}_{st})$ with edge weights.

Figure 4: A a multi-colored graph with the star property and the auxilliary graph G_{st} as constructed in the proof of Theorem 4.

In the example of Fig. 4, the path $s, \langle su \rangle, \langle uv \rangle, \langle tv \rangle, t$ is a shortest path in G_{st} between s and t and it has cost 6. We deduce that the MC-st-Path uses in $(G, \mathcal{C})$ the 6 colors $c_1, c_2, c_3, c_4, c_{15}, c_{16}$ and follows the path s, u, v, t.

The complexity of the MC-st-Cut and MC-Cut problems have not yet been formally investigated with the star property. However, it happens that the multi-colored graphs constructed in the NP-completeness proofs of the MC-Cut and MC-st-Cut problems given in [8, 13] have the star property. More precisely, using these results we get:

Theorem 5. *The MC-Cut problem (Problem 3) is NP-hard in multi-colored graphs with the star property. Furthermore, unless* $NP \subset TIME(n^{\mathcal{O}(\log \log n)})$*, it is hard to approximate within a factor* $(1 - \varepsilon) \ln m$*,* $\forall \varepsilon > 0$*, where* m *is the number of edges of the multi-colored graph.*

Theorem 6. *The MC-st-Cut problem (Problem 2) is NP-hard in multi-colored graphs with the star property. Furthermore , unless* $P = NP$*, it is hard to approximate within a factor* $2^{\log^{1-\delta} |\mathcal{C}|^{\frac{1}{2}}}$*, where* $\mathcal{C}$ *is the set of colors and* $\delta = (\log \log |\mathcal{C}^{\frac{1}{2}}|)^{-\varepsilon}$ *for all* $\varepsilon < \frac{1}{2}$*.*

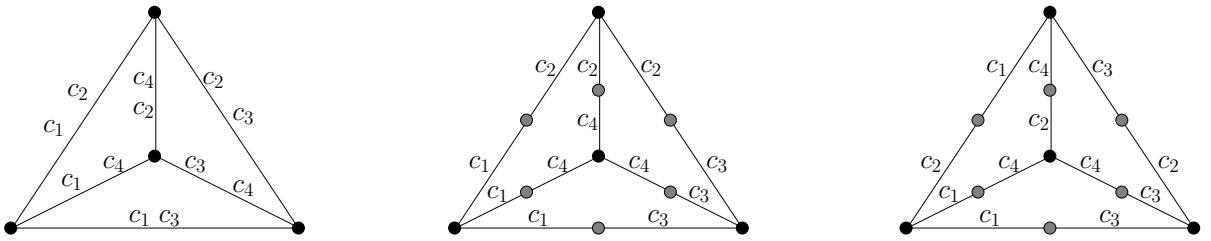
4.2 Graphs with a bounded number of non star colors

Theorem 7. *The MC-st-Path problem (Problem 1) can be solved in time* $2^k n^{\mathcal{O}(1)}$ *in multi-colored graphs with* k *non star colors.*

Proof. We use a similar auxiliary multi-colored graph as in the proof of Theorem 4. Let $\mathcal{NS} \subseteq \mathcal{C}$ be the set of non star colors of G and let $k = |\mathcal{NS}|$. For each color $c \in \mathcal{NS}$ and for each edge $\{u, v\}$ of G such that $c \in \mathcal{C}(\{u, v\})$, we arbitrarily set the center of c on edge $\{u, v\}$ to either u or v. Then, in the construction of G_{st}, edge $\{\langle uv \rangle, \langle vw \rangle\}$ is assigned the colors of $\mathcal{C}(\{u, v\}) \cup \mathcal{C}(\{v, w\})$ that are centered in v.

Now, for each subset S of the 2^k subsets of $\mathcal{NS}$ we construct a new graph G_S from G_{st} as follows. All the edges containing the non star colors of $\mathcal{NS} \setminus S$ are removed and the cost of the colors belonging to S is set to zero when computing the integer cost of an edge, unlike other remaining colors whose cost is 1 on each edge where they appear. Then a shortest path is computed in G_S. We add $|S|$ to the cost of the obtained path to account for the colors from S whose cost was null and which have thus potentially been used freely by the path.

Note that only a subset of S might have been actually used by the path and thus the cost that we associate with the path is not necessarily its actual number of colors. However, since we consider all subsets of non star colors, and then finally keep as solution the shortest path among all the computed ones, it is of no consequence. The same reasoning as in the proof of Theorem 4 permits to check that the obtained path is optimal.



(a) $(G, \mathcal{C})$.

(b) Associated Colored Graph with 4 colors of span 1.

(c) Associated Colored Graph with 4 colors of span > 1.

Figure 5: A multi-colored graph (Fig. 5a) and two possible Associated Colored Graphs. The mono-colored graph of Fig. 5b has span 1 and the star property.

To conclude, this solving method consists in computing a classical shortest path 2^k times, where $k = |\mathcal{NS}|$, on a graph obtained in polynomial time from the initial graph. Therefore the time complexity of the algorithm is $2^k n^{\mathcal{O}(1)}$. □

5 Transformations

In the previous sections, we have seen that some problems can be solved in polynomial time in either multi-colored graph with the star property or in mono-colored graphs with colors of span 1. In this section we present the basic transformations from one of these representations to the other. The objective of these transformations is to use the best combination of representations and algorithms to solve a given problem.

5.1 Associated colored graph

Definition 10 (AND transformation and Associated Colored Graph)**.** *An* AND *transformation consists in replacing each edge $e = \{u, v\}$ of the multi-colored graph (a* multi-colored edge*) by a subpath of length $k = |\mathcal{C}(e)|$, $P_e = \{e_i = (u_i, u_{i+1}),\ 0 \leq i \leq k\}$, with $u_0 = u$ and $u_k = v$, and then mapping the k colors of $\mathcal{C}(e)$ on the k edges of P_e. The resulting graph is mono-colored and called the Associated Colored Graph obtained by the* AND *transformation.*

The AND transformation corresponds to the natural interpretation of the colors as SRLGs: the failure of a single SRLG is sufficient to break a link. Since the mapping of colors to the edges is arbitrary, a multi-colored graph has several distinct Associated Colored Graphs. For instance, an AND transformation applied to the multi-colored graphs of Fig. 5a may result in the colored graphs of Fig. 5b with all colors of span 1, or in the colored graph of Fig. 5c with all colors of span > 1. In view of the preceding sections it is interesting to do an AND transformation which gives an Associated Colored Graph of span 1 if possible or with the minimum number of colors with span > 1.

In an AND transformation, when an edge $\{u, v\}$ is replaced by a subpath, the color c which appears on the first edge of the subpath will be said to be *attached to* u and the color c' which appears on the last edge attached to v. If the edge $\{u, v\}$ is mono-colored with color c, color c will be considered as attached to both end vertices.

5.2 Decision problem

We now show that deciding whether a multi-colored graph can be transformed into a colored graph of span 1 or not can be done in polynomial time. Then we will extend the algorithm to

obtain an algorithm finding whether, for a given k, a multi-colored graph can be transformed into a colored graph with at most k colors of span >1. Once a bound K on the number of colors of span >1 is found with this algorithm, it will be possible to determine by sequential or binary search the maximum number of colors of span 1 in an associated colored graph of the initial multi-colored graph.

To simplify the presentation, we assume that the graph has neither multiple edges nor loops, but the result can easily be adapted to these cases. The algorithm proceeds in 3 steps that we first describe informally while giving claims showing when a color cannot be of span 1.

Step 1: This step consists in determining for each edge whether its color set contains colors whose repartition in the graph implies that they cannot all be of span 1. We first split the colors into 2 classes: U (like Unique), the set of colors c such that $|E(c)| = 1$; and $\bar{U}$, the set of other colors that is the colors c such that $|E(c)| \geq 2$.

The colors of U are trivially of span 1 and can be put in an AND transformation on any edge of the subpath associated with the unique edge on which they appear. However we cannot completely ignore them. As an example suppose a color c appears on three consecutive edges $\{u, v\}$, $\{v, w\}$ and $\{w, x\}$; if there is no other color on $\{v, w\}$, then c is of span 1, by attaching c to v in the AND transformation of $\{u, v\}$ and to w in the AND transformation of $\{w, x\}$. But if there is another color c' in U on the edge $\{v, w\}$, c will no more be of span 1 in any AND transformation.

If we want a color of $\bar{U}$ appearing on an edge $\{u, v\}$ to be of span 1 after an AND transformation, then this color cannot be put in the middle of the subpath associated with the edge. Indeed it will be disconnected from any other occurrence of the color. So it has necessarily to be attached to u or v. As an edge has only two end vertices and at most one color can be attached to one end vertex, we get the following claims:

Claim 1. *If there is an edge e with $p \geq 3$ colors of $\bar{U}$ on it, then after any* AND *transformation at least $p - 2$ colors have span > 1.*

Claim 2. *If there is an edge e with 2 colors of $\bar{U}$ on it, then there can exist an* AND *transformation where these two colors are of span 1 only if they can be attached to different end vertices of e.*

So we could immediately rule out the case where 3 or more colors of $\bar{U}$ appear on the same edge. If we have 2 colors c and c' of $\bar{U}$ appearing on an edge $\{u, v\}$, then they can be of span 1 only if there exists an AND transformation such that c is attached to an end vertex of $\{u, v\}$ and c' to the other. In that case if there are other colors of U on this edge they will be put arbitrarily on the intermediate edges of the associated subpath. If there is only one color c of $\bar{U}$ on an edge $\{u, v\}$, and if there are other colors of U appearing also on $\{u, v\}$, c will have to be attached to one end vertex of $\{u, v\}$ and the other colors of U will have to be put arbitrarily on the remaining edges of the associated path. Otherwise if c is the unique color of $\bar{U}$ appearing on $\{u, v\}$ (edge mono-colored) nothing has to be done. These mono-colored edges will play a special role as we will see in Claims 4 and 6. Finally if we have no color of $\bar{U}$ on the edge $\{u, v\}$, we can put the colors of U on this edge in an arbitrary order on the subpath associated.

Step 2: The next step of the algorithm will consist in examining independently each color of $\bar{U}$. At the end of this step, we will have decided for each color either that the color considered cannot be of span 1, or that the color has a given status relatively to each edge on which it appears of one of the 3 kinds:

- c is the unique color appearing on the edge e.
- c is forced on the edge e to be attached to one specific end vertex of these edge.

- c is free on the edge e (i.e. it can be attached to any one of the end vertices).

Let c be a given color of $\bar{U}$ and let $G[E(c)]$ be the subgraph generated by the edges $E(c)$ of color c. We have the following immediate claim.

Claim 3. *If $G[E(c)]$ is not connected, then color c cannot be of span 1.*

So, in what follows, **let us suppose that $G[E(c)]$ is connected**. Let $F(c)$ be the set of edges mono-colored with color c, and let $G[F(c)]$ be the corresponding graph.

Claim 4. *If $G[F(c)]$ is not connected, then color c cannot be of span 1.*

Proof. Consider two connected components $G[F_1]$ and $G[F_2]$ of $G[F(c)]$, with $F_1, F_2 \subset F(c)$. As $G[E(c)]$ is connected, they are connected by a path between $G[F_1]$ and $G[F_2]$: $e_1, \ldots, e_i, \ldots e_k$ with $e_i = \{u_i, u_{i+1}\}$ and $u_1 \in G[F_1]$, $u_{k+1} \in G[F_2]$; the other vertices are neither in $G[F_1]$ nor in $G[F_2]$ and the edges are not mono-colored. But if color c is of span 1, it should, in any AND transformation, be attached on e_1 to u_1 and on e_k to u_k; but that is impossible if $k = 1$ as e_1 is not mono-colored and impossible if $k > 1$ as color c on e_1 will be disconnected from color c on e_k. □

Claim 5. *If $F(c)$ is empty, color c can be of span 1 only if it is a star color. In that case let u be the central vertex of the star, then color c is forced to be attached to u on all its edges which are of the form $\{u, v_j\}$.*

Proof. If c is not a star color there exists in $G[E(c)]$ a path of length 3: $\{u, v\}$, $\{v, w\}$, $\{w, x\}$ (x can be equal to u). As $F(c)$ is empty, to ensure a span 1 after an AND transformation color c should be attached to v on $\{u, v\}$ and $\{v, w\}$ and to w on $\{v, w\}$ and $\{w, x\}$ a contradiction. □

Claim 6. *If $G[F(c)]$ is connected and not empty and color c is of span 1, then any edge $e = \{u, v\}$ of $E(c) \setminus F(c)$ has at least one end vertex in $G[F(c)]$. Furthermore, if the edge e has only the vertex u (resp v) in common with $G[F(c)]$, then c is forced on e to be attached to u (resp v). If it has its two end vertices in $G[F(c)]$ then c is free on e.*

Proof. Suppose an edge $e = \{u, v\}$ of $E(c) \setminus F(c)$ containing c has no end vertices in $G[F(c)]$. As $G[E(c)]$ is connected, there is a path joining a vertex u_0 of $G[F(c)]$ to u consisting of edges not in $F(c)$. Then to be of span 1 c should be joined in any AND transformation to u_0 on the first edge of this path, but it will be disconnected from the c on the edge e and so c cannot be of span 1. If e is connected to $G[F(c)]$ only by the vertex u to be of span 1, it is forced to be attached to u. □

Theorem 8. *Deciding whether an AND transformation on a multi-colored graph can result in a colored graph of span 1 or not can be done in time $\mathcal{O}\left(\sum_{e \in E} |\mathcal{C}(e)|\right)$.*

Proof. The proof follows from the complete description of the algorithm given in Algorithm 1.

In STEP 2 of Algorithm 1, we have to test for each color c if $G[E(c)]$ and $G[F(c)]$ are connected; then to test when $F(c)$ is empty if $E(c)$ is a star; and finally to determine for each edge in $E(c) \setminus F(c)$ if its end vertices are in $G[F(c)]$. All these tests can be done in $\mathcal{O}(|E(c)|)$ operations. Therefore STEP 2 can be done in at most $\mathcal{O}(|E_m|)$, where E_m is the set of edges of the resulting colored graph and $|E_m| \leq \sum_{e \in E} |\mathcal{C}(e)|$. In STEP 3, we have to consider for each edge the status of the colors appearing on it. That can be done in $\mathcal{O}(|\mathcal{C}(e)|)$ operations and so again STEP 3 can be done with $\mathcal{O}(|E_m|)$ operations. □

Algorithm 1 From multi-colored to colored graph of span 1.

STEP 1: Split the colors in two classes, U the set of color c such that $|E(c)| = 1$, and $\bar{U}$ the set of other colors that is the colors c such that $|E(c)| \geq 2$.

STEP 2: For each color c in $\bar{U}$. Let $G[E(c)]$ be the subgraph generated by the edges $E(c)$ of color c and $G[F(c)]$ the subgraph consisting of the edges $F(c)$ mono-colored with c.

- If $G[E(c)]$ is not connected or $G[F(c)]$ is not connected, STOP; color c cannot be of span 1 in any AND transformation by Claims 3 and 4.
- If $G[E(c)]$ is connected and $F(c)$ is empty.

  Case 1: If c is not a star color ($E(c)$ not a star), STOP; color c cannot be of span 1 in any AND transformation by Claim 5.

  Case 2: If c is a star color ($E(c)$ is a star with central vertex u) then mark the color c as forced to be attached to u on all the edges of this star.
- If $G[E(c)]$ is connected and $G[F(c)] \neq \emptyset$ is connected.

  If there exists an edge e of $E(c) \setminus F(c)$ having none of its end vertices in $G[F(c)]$, STOP; color c cannot be of span 1 in any AND transformation by Claim 6.

  Otherwise, if $e = \{u, v\}$ of $E(c) \setminus F(c)$ intersects $G[F(c)]$ in exactly one end vertex, say u, then mark c as forced on the edge e to be attached to u. If e of $E(c) \setminus F(c)$ has its two end vertices in $G[F(c)]$, mark color c as free on e.

STEP 3: Examine the edges e of the graph in any order. Let $e = \{u, v\}$ be one edge. One of the following cases occur:

1) edge e contains two colors of $\bar{U}$ which are forced to be attached to the same end vertex. By Claim 2, one of these two colors cannot be of span 1. STOP; G cannot be transformed in a colored graph of span 1.

2) edge e contains $p \geq 3$ colors of $\bar{U}$ and by Claim 1, $p-2$ of these colors cannot be of span 1. STOP; G cannot be transformed in a colored graph of span 1.

END: If we have not reached a STOP with the impossibility to transform the graph in a colored graph of span 1, then we get a colored graph of span 1 by doing an AND transformation of the edges e as follows.

a) edge e contains two colors c and c' of $\bar{U}$ which are forced to be attached to the different end vertices of e (say c to u and c' to v), or one at least of the colors say c is free and the other c' is forced to be attached to v, or both c and c' are free. Then do an AND transformation, where c is put on the first edge of the subpath replacing e and c' on the last edge of this subpath and map the other colors of U appearing on e arbitrarily on the middle edges of the path.

b) edge e contains one color c of $\bar{U}$. If c is forced to be attached to u (resp v) or free then do an AND transformation which puts color c on the first (resp. last) edge of the path replacing e and put the other colors of U if any on the remaining edges of the path.

c) edge e contains no color of $\bar{U}$. Then do an AND transformation putting the other colors of U arbitrarily on the edges of the path.

Note that in fact when Algorithm 1 reaches a STOP, we have more information than only the fact that the graph cannot be transformed in a graph of span 1. Indeed the transformation is not possible for different reasons.

If we stop in STEP 2, then we know that the color c cannot be of span 1 in any transformation. So let us call REJ (like rejected) the set of colors which lead to a STOP in STEP 2. Then $|\texttt{REJ}|$ is a lower bound on the number of colors which cannot be of span >1 in any AND transformation.

If we stop in STEP 3, we know that some colors cannot be together of span 1, but we do not know exactly what colors. If we stop because two colors c and c' are forced to be attached to the same end vertex of e we know that at least one cannot be of span 1 and we can decide to sacrifice one of them by putting it in the middle of the subpath replacing the edge or attaching it at the other end-vertex. If the algorithm stops because there are strictly $p > 2$ colors of $\bar{U}$ on e, then we can decide to sacrifice $p-2$ of them.

With these remarks we can slightly modify the algorithm to decide if there are for a fixed integer k at most k colors of span >1.

Do firstly the preprocessing of STEPs 1 and 2 where U is the set of colors with $|E(c)| = 1$ and by replacing in STEP 2: "STOP; color c cannot be of span 1 in any AND transformation" with: "Put color c in a set REJ (like rejected)".

At the end of STEP 2 add the line "if $|\texttt{REJ}| > k$ then STOP; there are more than k colors of span >1".

Now in the new STEP 3, we will have 3 kind of colors: S (like Sacrificed) initialized to $S = \texttt{REJ}$ (where REJ is the set of rejected colors during STEP 2), $U' = U \cup S$, and $\bar{U} = \mathcal{C} \setminus U'$. Intuitively colors of S will not be of span 1 and so can, like those of U, be mapped freely in an AND transformation. Colors of $\bar{U}$ are still candidate to be of span 1.

The new STEP 3 will depend on $S, \bar{U}$, and $k' = k - |S|$ and is describe in Algorithm 2, that we call ALGO$(S, \bar{U}, k')$ and which returns one of the two answers:

YES: there exists an AND transformation which creates at most k' colors of span >1 in $\bar{U}$ and so at most k colors of span >1 in the Associated Colored Graph, or

NO: in any AND transformation there are more than k' colors of span >1 and so any Associated Colored Graph has more than k colors of span >1.

Note that ALGO$(\emptyset, \bar{U} = \mathcal{C} \setminus U, 0)$ corresponds exactly to the STEP 3 of Algorithm 1.

Theorem 9. *Deciding whether an* AND *transformation on a multi-colored graph can result in a colored graph with at most k colors of span >1 or not can be done in time $\mathcal{O}\left(3^k \sum_{e \in E} |\mathcal{C}(e)|\right)$.*

Proof. The complexity of the new STEP 3 for a given k' is in STEP 3-2 at most twice the complexity of the algorithm for $k'-1$. In STEP 3-3 it is at most $\binom{p}{2}$ times the complexity of the algorithm for $k'+2-p$; that is either at most 3 times the complexity for $k'-1$ when $p=3$ or at most 3^{p-2} the complexity for $k'+2-p$ when $p>3$. Recall that $3^{p-2} > \binom{p}{2}$ as soon as $p>3$, so the complexity of the algorithm ALGO$(S, \bar{U}, k')$ is at most $3^{k'}$ times the complexity of the algorithm for $k'=0$ which is $\mathcal{O}(|E_m|)$, and we have $|E_m| \leq \sum_{e \in E} |\mathcal{C}(e)|$. The worst case appears when there are exactly 3 colors of $\bar{U}$ on each edge e. □

Finally, if the number of colors of span >1 is bounded by some constant K with $|\texttt{REJ}| \leq K \leq |\mathcal{C}|$, we can apply the algorithm for $k = |\texttt{REJ}|, \ldots, K$ to determine in polynomial time the minimum number of colors of span >1 (the first value of k for which the answer YES is returned).

Algorithm 2 ALGO$(S, \bar{U}, k')$

STEP 3-1: split the edges in 3 categories

A: those containing two colors c and c' of $\bar{U}$ which are forced to be attached to the same end vertex and so one color has to be sacrificed;

B: those containing $p \geq 3$ colors of $\bar{U}$ and so $p-2$ colors have to be sacrificed;

C: the other edges.

STEP 3-2: If A is not empty, choose one edge e of category A and let c and c' be the two colors of $\bar{U}$ which are forced to be attached to the same end vertex of this edge. Then, branch on two sub-cases. More precisely, apply ALGO$(S \cup \{c\}, \bar{U} \setminus \{c\}, k'-1)$.

(i) If the answer for this algorithm is YES then return YES.

(ii) If the answer for this algorithm is NO, apply ALGO$(S \cup \{c'\}, \bar{U} \setminus \{c'\}, k'-1)$. If the answer for this second algorithm is YES, then return YES. Otherwise, return NO.

STEP 3-3: If there is no edge of category A, but some edge e of category B. Let $c_1, c_2, \ldots, c_p$ be the p colors of $\bar{U}$ present on this edge.

- If $k' + p - 2 > k$ return NO.
- Otherwise branch on $\binom{p}{2}$ sub-cases. More precisely, for each pair of color $\{c_i, c_j\}$, let $S_{i,j}$ be the set of colors on the edge e different from c_i and c_j. Apply successively ALGO$(S \cup S_{i,j}, \bar{U} \setminus S_{i,j}, k' + 2 - p)$. If one of these algorithms return YES, then return YES. If all the $\binom{p}{2}$ algorithms return NO, return NO.

5.3 NP-hardness

The problem of maximizing the number of colors of span 1 in an AND transformation takes its interest from the results of Section 3: it comes down to minimizing the number of colors of span > 1. We use the analysis of the preceeding section to give two different ways to prove that the problem is NP-Hard in general.

If we consider edges of category A (see step 3.1 of Algorithm 2) forced to be attached to the central vertex of a star, we get the following reduction.

Proposition 1. *The two following problems are equivalent:*

- VERTEX COVER*: find a minimum vertex cover (of the edges) of a graph G with c vertices and m edges;*
- *Find the minimum number of colors (among a set $\mathcal{C}$ of colors), with span > 1 in a star with m edges.*

Proof. Associate with each vertex i of G a color c_i and to each edge $e = \{i, j\}$ of G an edge $\{x, v_e\}$ of the star with the two colors c_i and c_j (the sacrificed colors will correspond to the vertices of a vertex cover as the colors are all star colors forced in x and conflicting). □

Similarly, considering the edges of category B gives the following reduction.

Proposition 2. *The two following problems are equivalent:*

- SET COVER: *given a family of m triples on $|\mathcal{C}|$ elements find a minimum number of elements covering the triples;*
- *Find the minimum number of colors with span > 1 in the graph consisting of a star plus for each edge $e = \{x, v\}$ of the star we add 3 paths of length 2 between x and v.*

Proof. Associate with each vertex i a color c_i and to each triple $T = \{i, j, k\}$ associate the edge of the star $\{x, v_T\}$ with the 3 colors c_i, c_j, c_k plus 3 paths between x and v_T uniquely colored each with color c_i, resp. c_j, resp. c_k.

The set cover corresponds to the edges to be sacrificed (the 3 colors are free by construction, the edge having its two end vertices in $F(c)$. □

Last we can use an approximability preserving reduction from Maximum Set Packing (MSP) (a formal proof can be found in [32]) to establish inapproximability result.

Theorem 10 ([32])**.** *The problem of maximizing the number of colors of span 1 (or minimizing the number of colors of span > 1) in an* AND *transformation is NP-Hard and is not approximable within a factor $|\mathcal{C}|^{\frac{1}{2}-\varepsilon}$, $\forall \varepsilon > 0$ unless P=NP.*

6 Experimentations

This section is dedicated to an experimental assessment of the algorithms and principles presented above. Our results are reported in Table 1 and discussed in Section 6.3.

For conducting these experiments, we have exploited a large set of instances presented in the literature and detailed in Section 6.1. One subset is based on real-world instances, U.S., European and Indian networks, while the other subset is a randomly generated benchmark which highlights the influence of the different parameters (size of the network, number of colors, number of colors of span more than 1) on the complexity of the problems.

We assess the performances of the algorithms by a comparison with the performances of Integer Linear Programming (ILP) formulations. It is important to stress that our implementations, described in Section 6.2 are not thoroughly optimized and use the SageMath open-source mathematical system [28] for the sake of efficient development and interactions with ILP solvers (we use IBM ILOG Cplex version 12.6). Nevertheless, the results highlight the exponential dependency to the number of colors of span more than 1 while the actual size of the networks is of lesser impact.

6.1 Instances

The benchmark instances used for testing our algorithms are based on instances proposed in [15, 21, 24, 41]. Their characteristics are reported in Table 1.

All the SRLGs used in [15] on the U.S. network and ARPANET topologies are stars, and each edge is assigned a single SRLG. Furthermore, both instances have span 1.

The US-NET and Italia topologies used in [24] have both 11 SRLGs; among them 2 (resp. 3) have span > 1, and 2 (resp. 3) are not stars. Since some edges were not assigned a SRLG, we added 21 (resp. 17) SRLGs, each impacting a single edge.

In [21], two topologies (U.S. network and India) have been proposed. Nodes correspond to major cities, and links were chosen using information about national highways and major railway lines. Then SRLGs were specified using seismic (hazard) maps. Also, some links belong to several SRLGs (up to 9 for U.S. network), each involving at least two edges, thus leading to many SRLGs of span > 1 after an AND transformation. Since some links were not assigned SRLGs, we also added 13 (resp. 1) SRLGs to the U.S. network (resp. India) topology.

Instances					non star colors	AND transf. span >1	AND transf. time (in sec.)	Aux. graph $\lvert V_{st}\rvert$	Aux. graph $\lvert E_{st}\rvert$	Running time (in sec.) MC-st-Path P	MC-st-Path ILP	2-CDP P	2-CDP ILP	MC-st-Cut P	MC-st-Cut ILP
Name		$\lvert V\rvert$	$\lvert E\rvert$	$\lvert\mathcal{C}\rvert$											
U.S. network	[15]	15	27	19	0	0	0.01	19	49	0.01	0.71	0.17	1.56	0.84	0.73
ARPANET	[15]	20	32	18	0	0	0.01	18	38	0.01	1.54	0.27	3.54	1.22	1.54
Italia	[24]	21	36	28	3	3	0.05	31	70	0.09	2.03	–	4.86	7.67	1.95
US-NET	[24]	24	43	32	2	2	0.05	34	81	0.08	3.03	–	7.64	9.96	2.94
U.S. network	[21]	26	42	40-8	2	5	0.40	45	81	0.17	4.56	–	78.82	7.26	3.54
India	[21]	28	40	32	11	13	0.88	56	72	36.06	6.66	–	41.03	2.85	4.12
$T_1'-1$	[41]	19	56	31-2	5	1	0.01	30	139	0.03	2.24	–	5.31	7.67	2.13
$T_1'-2$	[41]	19	58	29	4	1	0.06	30	147	0.03	2.34	–	6.32	7.43	2.22
$T_1'-3$	[41]	19	53	29-1	7	0	0.05	28	108	0.01	2.28	0.50	5.54	2.69	2.03
$T_1'-4$	[41]	24	82	43-1	7	0	0.06	42	235	0.01	5.51	1.57	14.86	9.06	4.95
$T_1'-5$	[41]	24	80	41-1	8	1	0.12	41	225	0.07	5.42	–	14.35	146.91	4.90
$T_1'-6$	[41]	24	82	40-2	9	1	0.10	39	224	0.07	5.56	–	15.21	19.40	4.97
$T_1'-7$	[41]	24	82	38	8	0	0.06	38	212	0.01	5.52	1.45	14.78	8.13	4.88
$T_1'-8$	[41]	31	91	47	9	1	0.14	48	248	0.12	10.32	–	26.87	36.76	9.43
$T_1'-9$	[41]	50	196	85	16	1	0.53	86	620	0.57	73.38	–	268.77	240.09	52.15
$T_1'-10$	[41]	50	186	85-1	19	7	1.10	93	612	1.46	76.60	–	326.93	1767.34	47.48
$T_1'-11$	[41]	50	191	85	19	4	0.88	89	621	0.98	82.03	–	355.31	527.80	49.07
$T_1'-12$	[41]	60	198	90-1	18	0	0.24	89	568	0.11	112.53	23.68	479.51	140.69	77.35
$T_1'-13$	[41]	65	197	100	18	0	0.42	100	586	0.14	120.61	28.40	425.49	170.40	92.20
$T_1'-14$	[41]	65	197	100	18	0	0.35	100	588	0.14	121.47	28.59	429.50	169.02	91.64
$T_1'-15$	[41]	100	335	150-1	28	1	1.39	150	1041	3.83	669.57	–	3721.78	1649.21	348.51

Table 1: Characteristics of benchmark instances and run time of experiments. Running times are sums over all pairs of vertices.

In [41], a large set of benchmark instances were randomly generated (topology and SRLGs): the topologies are random directed graphs; each arc is assigned a single SRLG; in these instances, denoted $T_1 - i$, each SRLG corresponds to an adjacent subset of arcs (i.e. of span 1 if the directions of arcs are not considered). Since our algorithms are designed for undirected graphs, we have projected these instances on their underlying undirected graphs, each edge $\{u, v\}$ being assigned the union of the SRLGs of both arcs uv and vu. The resulting instances are reported as $T_1' - i$ in Table 1.

For some of these instances, we noted that some SRLGs involve the same sets of edges. Since our algorithms can easily be adapted to weighted SRLGs, we decided to remove duplicated SRLGs. Also, we first set the weight of each SRLG to one, and then, if $E(c) = E(c')$, we remove SRLG c' and increase the weight of c by one. The number of removed SRLGs is indicated in column $|\mathcal{C}|$ of Table 1 with a -i.

Most of these instances, real-world or randomly generated, are multi-colored graphs. For applying our algorithms, we need to transform them into associated colored graphs. Also, we have used the exact algorithm presented in Section 5 to perform the AND transformation of all aforesaid multi-colored instances. The resulting associated colored graphs are those minimizing the number of colors of span > 1.

6.2 Implementation

We have implemented the algorithms presented in this paper using SageMath [28]. The source code, as well as the instances used to evaluate our algorithms, can be found at `http://www-sop.inria.fr/members/David.Coudert/code/srlg.shtml`.

For the implementation of 2-CDP we have used the Suurballe-Tarjan algorithm [26]. This implementation is used only on networks for which the AND transformation results in a graph of span 1. When no two SRLG disjoint paths exists between s and t, we compute the pair of paths minimizing the number of common SRLGs (see e.g. [37]).

As explained before, we use weights to model duplicated SRLGs. We also use weights in the algorithms explained in the proof of Theorems 1 and 2: given the set $\mathcal{C}_{>1}$ of colors of span >1, we assign weight 0 to all colors in the subset $\mathcal{C}^i_{>1}$, and weight $+\infty$ to all colors in the subset $\mathcal{C}_{>1} \setminus \mathcal{C}^i_{>1}$ (this is equivalent as removing edges from the graph).

For determining the MC-st-Cut when some colors have span >1, we use a *minimum weight vertex st-cut* that returns sets of vertices of minimum weight whose removal disconnects s from t. We solve it in polynomial time using a standard integer linear programming formulation.

For the implementation of the algorithms presented in Theorems 1, 2, and 7, we use a simple branch-and-bound formulation to iterate over the subsets of non star colors or of colors of span >1. We thus avoid visiting branches that could not lead to better solutions than current bound. This allows for significant reductions of the search space and running time.

6.3 Discussion

The results reported in Table 1 confirm several theoretical facts established in the first part of this article and validate our approach. Indeed, a first look at the results shows that whenever the Associated Colored Graph is of span 1 (all colors have span 1), our algorithms for MC-st-Path and 2-CDP are very efficient, despite the non-optimized code, always beating the ILP formulations. However, for MC-st-Cut the ILP formulation is always the fastest. Moreover, except for the India network which has a large number of colors of span >1 compared to the network size, we are able to solve MC-st-Path much faster than using the ILP formulation. Observe that the reported running times are the minimum between the algorithm for graphs with few colors of span >1 presented in Theorem 1 and the algorithm for graphs with few non star colors presented in Theorem 7. In general, the algorithm for graphs with few colors of span >1 is the fastest since the considered instances have more non star colors than colors with span >1. The algorithm for graphs with few non star colors remains competitive with respect to the ILP formulation as long as the number of non star colors is less than 15.

When the number of colors of span >1 increases, the competitiveness of the algorithm presented in Theorem 2 for the MC-st-Cut problem drops. The comparison between the instances $T'_1 - 9$, 10 and 11 highlights the strong impact of the number of colors of span >1 for our algorithm. While the size of the initial networks, the number of initial SRLGs, and the size of the auxiliary graphs are quite similar, the running time of the algorithm explodes when the number of color of span >1 is 7 (instance $T'_1 - 10$) and 4 (instance $T'_1 - 11$) compared to the instance $T'_1 - 9$ with only 1 such color.

Conversely, comparing the running time of the algorithms among instances with span 1 (after transformation) shows a polynomial dependency to the size of the instances, and more precisely to the size of the auxiliary graphs. For example, instances $T'_1 - 5$ and $T'_1 - 6$ have similar size of auxiliary graphs and running times, and $T'_1 - 8$ which has a larger auxiliary graph has also larger running time.

On the other hand, one can see that the strong dependency of the running time with the number of color of span >1 does not apply to the ILP formulations. The same comparison as

above (instances $T'_1 - 9$, 10 and 11) is, in contrast, very relevant, but the same phenomenon appears all over the results.

Another experimental remark is that the 2-CDP is the most difficult problem to solve. It is a known fact that for colored graph, the link between the *st*-Cut and the ability to find "independent" paths is broken [8]. This result illustrates again that fault tolerant network design is a major issue in these settings.

Last, observe that the running time of the exact algorithm presented in Section 5 to perform the AND transformation of all instances is quite small. As expected, it increases with the final number of colors of span > 1, and also with the size of the input graph.

7 Conclusion

In this paper, we have pointed out the importance of the topological structure of the SRLGs to solve the MC-*st*-Path and MC-*st*-Cut problems. In particular, we have designed algorithms whose time complexity depend on either the number of SRLGs of span > 1 or the number of non-stars SRLGs. Therefore, when all the SRLGs have span 1 or are stars, we get polynomial time algorithms.

Besides, we have assessed our approach through experiments on benchmark instances from the literature, both built randomly or on the basis of real-world topologies. Our experimental results show the exponential dependency on these two parameters on the computational time. They also highlight again the difficulty of finding color disjoint paths, which is a major issue for survivable network design. Indeed, the 2-CDP problem is NP-Complete as soon as one SRLG has span > 1 (Theorem 3) or when all SRLGs are stars [1].

Acknowledgment

We would like to thank the anonymous referees for their very helpful comments. We are also very thankful to Jean-Claude Bermond for his help in improving the readability of the manuscript. This work has been partially supported by the ANR project Stint under reference ANR-13-BS02-0007 and the French Government "Investments for the Future" Program under reference ANR-11-LABX-0031-01.

References

[1] J.-C. Bermond, D. Coudert, G. D'Angelo, and F. Z. Moataz. Finding disjoint paths in networks with star shared risk link groups. *Theoretical Computer Science*, 579:74–87, 2015.

[2] H. Broersma, X. Li, G. Woeginger, and S. Zhang. Paths and cycles in colored graphs. *Australasian Journal of Combinatorics*, 31:299–311, 2005.

[3] F. Carrabs, R. Cerulli, and M. Gentili. The labeled maximum matching problem. *Computers & Operations Research*, 36(6):1859–1871, 2009.

[4] R. Cerulli, P. Dell'Olmo, M. Gentili, and A. Raiconi. Heuristic approaches for the minimum labelling hamiltonian cycle problem. *Electronics Notes in Discrete Mathematics*, 25:131–138, 2006.

[5] R. Cerulli, A. Fink, M. Gentili, and S. Voß. *The Next Wave on Computing, Optimization, and Decision Technologies*, chapter Metaheuristics comparison for the minimum labelling spanning tree problem, pages 93–106. Springer, New York, 2005.

[6] R. Cerulli, A. Fink, M. Gentili, and S. Voß. Extensions of the minimum labelling spanning tree problem. *Journal of Telecommunications and Information Technology*, 4:39–45, 2006.

[7] R. Chang and S. Leu. The minimum labeling spanning trees. *Information Processing Letters*, 63(5):277–282, 1997.

[8] D. Coudert, P. Datta, S. Perennes, H. Rivano, and M.-E. Voge. Shared risk resource group: Complexity and approximability issues. *Parallel Processing Letters*, 17(2):169–184, June 2007.

[9] D. Coudert, F. Huc, F. Peix, and M.-E. Voge. Reliability of connections in multilayer networks under shared risk groups and costs constraints. In *IEEE International Conference on Communications (ICC)*, pages 5170 – 5174, Beijing, China, May 2008.

[10] B. Couëtoux, L. Gourvès, J. Monnot, and O. A. Telelis. Labeled traveling salesman problems: Complexity and approximation. *Discrete Optimization*, 7(1-2):74–85, Feb. 2010.

[11] P. Datta and A. Somani. Graph transformation approaches for diverse routing in shared risk resource group (SRRG) failures. *Computer Networks*, 52(12):2381–2394, Aug. 2008.

[12] J. Doucette and W. Grover. Shared-risk logical span groups in span-restorable optical networks: Analysis and capacity planning model. *Photonic Network Communications*, 9(1):35–53, 2005.

[13] A. Faragó. A graph theoretic model for complex network failure scenarios. In *INFORMS Telecommunications Conference*, Dallas, Texas, Mar. 2006.

[14] L. Guo, J. Cao, H. Yu, and L. Li. A new shared-risk link groups (SRLG)-disjoint path provisioning with shared protection in WDM optical networks. *Journal of Network and Computer Applications*, 30(2):650–661, 2007.

[15] L. Guo and L. Li. A novel survivable routing algorithm with partial shared-risk link groups (SRLG)-disjoint protection based on differentiated reliability constraints in WDM optical mesh networks. *IEEE/OSA Journal of Lightwave technology*, 25(6):1410–1415, June 2007.

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

[17] R. Hassin, J. Monnot, and D. Segev. The complexity of bottleneck labeled graph problems. In *International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 4769 of *Lecture Notes in Computer Science*, pages 328–340, Dornburg near Jena, Germany, June 2007. Springer.

[18] J. Hu. Diverse routing in mesh optical networks. *IEEE Transactions on Communications*, 51:489–494, Mar. 2003.

[19] S. Jha, O. Sheyner, and J. M. Wing. Two formal analyses of attack graphs. In *IEEE Computer Security Foundations Workshop (CSFW)*, pages 49–63, Cape Breton, Nova Scotia, Canada, June 2002. IEEE Computer Society.

[20] Q. Jiang, D. Reeves, and P. Ning. Improving robustness of PGP keyrings by conflict detection. In *RSA Conference Cryptographers' Track (CT-RSA)*, volume 2964 of *Lecture Notes in Computer Science*, pages 194–207. Springer, Feb. 2004.

[21] M. Kiese, V. Marcheva, J. Eberspacher, and D. Schupke. Diverse routing based on shared risk link groups. In *International Conference on Design of Reliable Communication Networks (DRCN)*, pages 153–159, Washington, DC, USA, Oct. 2009. IEEE.

[22] C. Liu and L. Ruan. p-cycle design in survivable WDM networks with shared risk link groups (SRLGs). In *International Conference on Design of Reliable Communication Networks (DRCN)*, pages 207–212, Ischia, Italy, Oct. 2005. IEEE.

[23] X. Luo and B. Wang. Diverse routing in WDM optical networks with shared risk link group (SRLG) failures. In *International Conference on Design of Reliable Communication Networks (DRCN)*, pages 445–452, Ischia, Italy, Oct. 2005. IEEE.

[24] L. Shen, X. Yang, and B. Ramamurthy. Shared risk link group (SRLG)-diverse path provisioning under hybrid service level agreements in wavelength-routed optical mesh networks. *IEEE/ACM Transactions on Networking*, 13:918– 931, Aug. 2005.

[25] S. Stefanakos. Reliable routings in networks with generalized link failure events. *IEEE/ACM Transactions on Networking*, 16:1331–1339, Dec. 2008.

[26] J. Suurballe and R. Tarjan. A quick Method for Finding Shortest Pairs of Disjoint Paths. *Networks*, 14:325–336, 1984.

[27] L. Tang and P. Zhang. Approximating minimum label s-t cut via linear programming. In *Latin American Symposium on Theoretical Informatics (LATIN)*, volume 7256 of *Lecture Notes in Computer Science*, pages 655–666. Springer, 2012.

[28] The Sage Developers. *Sage Mathematics Software (Version 6.8)*, 2015. `http://www.sagemath.org`.

[29] A. Todimala and B. Ramamurthy. IMSH: an iterative heuristic for SRLG diverse routing in WDM mesh networks. In *IEEE International Conference on Computer Communications and Networks (ICCCN)*, pages 199 – 204. IEEE, 2004.

[30] A. Todimala and B. Ramamurthy. Survivable virtual topology routing under shared risk link groups in WDM networks. In *International Conference on Broadband Communications, Networks and Systems (BroadNets)*, pages 130–139, San Jose, CA, USA, Oct. 2004. IEEE.

[31] A. Urra, E. Calle, and J. Marzo. Partial disjoint path for multi-layer protection in GMPLS networks. In *International Conference on Design of Reliable Communication Networks (DRCN)*, pages 165–170, Ischia, Italy, Oct. 2005. IEEE.

[32] M.-E. Voge. *Optimisation des réseaux de télécommunication : Réseaux multi-niveaux, Tolérance aux pannes et Surveillance du trafic.* PhD thesis, Ecole doctorale STIC, Université Nice-Sophia Antipolis, Nov. 2006.

[33] Y. Wan, G. Chen, and Y. Xu. A note on the minimum label spanning tree. *Information Processing Letters*, 84(2):99–101, 2002.

[34] D. Xu, Y. Xiong, and C. Qiao. Novel algorithms for shared segment protection. *IEEE Journal on Selected Areas in Communications*, 21(8):1320–1331, Oct. 2003.

[35] D. Xu, Y. Xiong, C. Qiao, and C. Li. Trap avoidance and protection schemes in networks with shared risk link groups. *IEEE/OSA Journal of Lightwave technology*, 21(11):2683 – 2693, Nov. 2003.

[36] D. Xu, Y. Xiong, C. Qiao, and C. Li. Failure protection in layered networks with shared risk link groups. *IEEE Network*, 18(3):36–41, June 2004.

[37] S. Yuan, S. Varma, and J. Jue. Minimum-color path problems for reliability in mesh networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, volume 4, pages 2658–2669, Houston, TX, USA, Mar. 2005.

[38] H. Zang, C. Ou, and B. Mukherjee. Path-protection routing and wavelength assignment (RWA) in WDM mesh networks under duct-layer constraints. *IEEE/ACM Transactions on Networking*, 11:248–258, Apr. 2003.

[39] P. Zhang. Efficient algorithms for the label cut problems. In *Theory and Applications of Models of Computation (TAMC)*, volume 8402 of *Lecture Notes in Computer Science*, pages 259–270. Springer International Publishing, 2014.

[40] P. Zhang, J.-Y. Cai, L.-Q. Tang, and W.-B. Zhao. Approximation and hardness results for label cut and related problems. *Journal of Combinatorial Optimization*, 21(2):192–208, Feb. 2011.

[41] Q. Zhang, J. Sun, G. Xiao, and E. Tsang. Evolutionary algorithms refining a heuristic: Hyper-heuristic for shared-path protections in WDM networks under SRLG constraints. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 37(1):51–61, Feb. 2007.