Minimum Cost Survivable Routing Algorithms for Generalized Diversity Coding

Alija Pašić, Péter Babarczi, *Member, IEEE*, János Tapolcai, Erika R. Bérczi-Kovács, Zoltán Király, and Lajos Rónyai

***Abstract*— Generalized diversity coding is a promising proactive recovery scheme against single edge failures for unicast connections in transport networks. At the source node, the user data is split into two parts, and their bitwise XOR is computed as a third redundancy sub-flow. In order to guarantee instantaneous failure recovery without costly node upgrades, the network must ensure that any two of the three sub-flows reach the destination node in case of a single edge failure only by allowing flow duplication or merging identical flows, and avoiding any coding operation in the core network. In this paper, we investigate the corresponding routing problem to calculate capacity-efficient routes for these sub-flows. We propose a polynomial-time algorithm for topologies without capacity constraints on the links and without capability limitations of the nodes. We show that with node limitations the presented algorithm (as well as a minimum cost disjoint path-pair) provides a $4/3$-approximation for the routing problem. Furthermore, we formulate an integer linear program to provide a minimum cost solution with arbitrary constraints in general graphs and we propose a polynomial-time algorithm in directed acyclic graphs. Our simulation results suggest that with upgrading only a small set of core network nodes with flow duplication and merging capabilities most of the benefits of generalized diversity coding can be achieved.**

***Index Terms*— Survivable routing, incremental deployment, diversity coding, instantaneous recovery, transport networks.**

Manuscript received July 23, 2019; revised November 18, 2019; accepted December 7, 2019; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor P. P. C. Lee. Date of publication January 23, 2020; date of current version February 14, 2020. This work was supported in part by the High Speed Networks Laboratory (HSNLab), through the National Research, Development, and Innovation Fund of Hungary, under Project K124171, Project K128062, Project K115288, Project KH129589, and TUDFO/51757/2019-ITM Thematic Excellence Program, in part by the BME-Artificial Intelligence FIKP of EMMI under Grant BME FIKP-MI/SC, in part by the Industry and Digitization Subprogramme, NRDI Office, in 2019, in part by the National Development Agency of Hungary based on a source from the Research and Technology Innovation Fund under Grant FK 132524, and in part Project no. ED_18-1-2019-0030 (Application-specific highly reliable IT solutions) that has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Thematic Excellence Programme funding scheme. This article was presented at the IFIP Networking Conference, Toulouse, France, May 2015. *(Corresponding author: Alija Pašić.)*

Alija Pašić, Péter Babarczi, and János Tapolcai are with the MTA-BME Future Internet Research Group, Budapest University of Technology and Economics, 1111 Budapest, Hungary, and also with the MTA-BME Information Systems Research Group, Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics (BME), 1111 Budapest, Hungary (e-mail: pasic@tmit.bme.hu; babarczi@tmit.bme.hu; tapolcai@tmit.bme.hu).

Erika R. Bérczi-Kovács is with the Department of Operations Research, Eötvös Loránd University, 1053 Budapest, Hungary, and also with the MTA-ELTE Egerváry Research Group on Combinatorial Optimization (EGRES), Eötvös Loránd University, 1053 Budapest, Hungary (e-mail: koverika@cs.elte.hu).

Zoltán Király is with the Department of Computer Science, Eötvös Loránd University, 1053 Budapest, Hungary (e-mail: kiraly@cs.elte.hu).

Lajos Rónyai is with the Institute for Computer Science and Control, 1111 Budapest, Hungary, and also with the Department of Algebra, Budapest University of Technology and Economics (BME), 1111 Budapest, Hungary (e-mail: ronyai@sztaki.hu).

Digital Object Identifier 10.1109/TNET.2019.2963574

I. INTRODUCTION

DESPITE extensive research effort focused on developing capacity-efficient survivable routing schemes in the last decades *dedicated* $1+1$ *path protection* is still the most commonly used scheme of the current communication networks [1]. Dedicated path protection is appealing because of its ultrafast recovery time combined with the robust and straightforward operation: it sends the user data along two disjoint paths to instantaneously recover from single edge failures [2]. Although it consumes at least twice as much capacity as a single path, there are efficient algorithms to calculate a $1+1$ routing solution [3] and its operation does not require to modify the operation of core network nodes.

Several survivable routing schemes were introduced in the past decades which could significantly reduce its bandwidth utilization [4]–[11]. Network coding-based approaches perform algebraic operations on the data at core network nodes [4]–[6], partial path protection methods guarantee a minimum grade of service after failure using multi-path routing strategies [7], [8], and shared protection approaches pre-compute backup paths but signal them only after a failure occurs [9]–[11]. Although they are capacity efficient, these methods did not reach the phase of widespread deployment. We argue that this is because they sacrifice either the ultrafast recovery time, the low computational complexity, or the simple operation of $1+1$, each of which is a desired property for network operators.

Although optimal capacity efficiency can be achieved with network coding [4], [12]–[14] while the 50 ms recovery time constraint of carrier-grade networks is maintained, with the current technology it requires extensive data processing at core network nodes. *Diversity coding* (DC) [15], [16] provides a solution for this problem, where the user data is split at the source node into two parts A and B, and a third sub-flow with the redundancy data $A \oplus B$ is created, too ($\oplus$ denotes the exclusive OR (XOR) operation). These three sub-flows are forwarded along three edge-disjoint paths to the destination, which can decode the sent data from arbitrary two of the three with a simple XOR operation. Therefore, DC maintains all the desired properties of $1+1$ (*i.e.*, instantaneous failure recovery, simple operation, and low complexity). However,

DC requires the existence of three edge-disjoint paths between the communication endpoints, which is rarely present in transport networks.

In order to tackle the connectivity issue recent works [17], [18] generalized diversity coding and provided polynomial-time network coding algorithms to route the three sub-flows on minimum cost survivable subgraphs instead of disjoint simple paths. While [17] focused on algebraic properties such as the necessary field size for coding, in [18] we revisited the problem with a pure graph theoretical mindset, and demonstrated that no in-network coding is necessary at all. Although these works assumed that a minimum cost subgraph is given for coding, finding such subgraphs (survivable routing) was first discussed in [19].[1] In this paper we extend [19] in order to make the *generalized diversity coding* concept a viable alternative for $1+1$ in transport networks. To be more specific, from a practical perspective we introduce an approximation algorithm for networks with limited node capabilities, and discuss incremental network node upgrade strategies to deploy our method into real networks. Furthermore, from a theoretical perspective, we propose a polynomial-time survivable routing algorithm in directed acyclic graphs.

The rest of the paper is organized as follows. In Section II we formulate our problem, and reveal important structural properties of the minimum cost survivable routings. In Section III a polynomial-time algorithm is presented in fully upgraded networks without capacity constraints. In Section IV we prove that $1+1$ approximates our routing problem in partially upgraded networks, and provide a $4/3$-approximation algorithm for this scenario. As the routing problem is NP-complete with scarce bandwidth resources in partially upgraded networks [19], in Section V we present an integer linear program for general topologies and a polynomial-time algorithm in directed acyclic graphs. In Section VI we show our simulation results, which reveals the network scenarios where the generalized diversity coding approach can be a real alternative of $1+1$ with a minimal (or even without) network upgrade. Finally, Section VII concludes the paper.

II. Background

A. Problem Formulation

A transport network is a collection of routers, switches (referred to as nodes) and high bandwidth communication channels (referred to as edges) between them. It may be represented by a directed graph $G=(V,E,k,c)$ with node set V and edge set E. Each $e\in E$ edge has two attributes, namely its capacity $k(e)\in\mathbb{N}$, *i.e.*, number of bandwidth units available for data transmission, and its cost $c(e)\in\mathbb{R}^+$, which is defined as the cost of using one unit of bandwidth along edge e. Given a connection request $D=(s,t,d)$, with information source $s\in V$, with information sink $t\in V$, and the number of data units d requested for transmission.[2] Our goal is to allocate non-negative bandwidth $f(e)$ for each edge e which is resilient against single edge failures. This goal can be achieved either with applying three link-disjoint paths (Fig. 1a), or using three directed acyclic graphs which might share common edges (Fig. 1b), but even upon the failure of these edges all data units are received at the sink without any network reconfiguration, formally:

Definition 1: The allocated bandwidth $f(e)$ for each edge e implements a ***survivable routing*** *of connection request $D=(s,t,d)$ in G, if $\forall e\in E: f(e)\le k(e)$, and there is an $s-t$ flow of value at least d in G with edge capacities f, even if we delete any single edge of G.*

Our goal is to find a survivable routing f for connection D with minimum *bandwidth cost*, formally:

$$\min\sum_{e\in E} c(e)\cdot f(e). \tag{1}$$

We say that routing is **vulnerable** if it is not survivable. Furthermore, a survivable routing is **critical**, if we cannot further decrease the bandwidth value $f(e)$ along any edge in $e\in E$ without making the routing vulnerable. Intuitively speaking, critical means the routing is a local minimum. The rest of the paper is devoted to finding the global minimum.

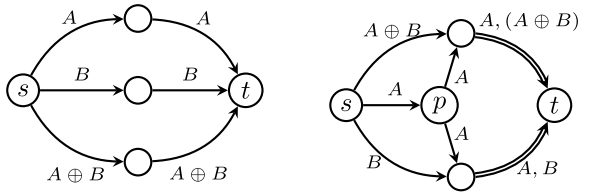
This optimization problem has been investigated for decades in the literature, and it was shown that finding the optimal survivable routing for a connection with $d>2$ data parts,

[1]The survivable routing problem was later extended to include different delay requirements of the applications [20]. However, in the current paper we deal with the original problem [19] without any delay constraint.

[2]The notation is summarized in Table I.

TABLE I
NOTATION LIST FOR THE SURVIVABLE ROUTING PROBLEM

Notations	Description
$G=(V,E,k,c)$	directed graph with node set V, edge set E, edge costs $c(e)\in\mathbb{R}^+$, capacities $k(e)\in\mathbb{N}$
$D=(s,t,d)$	connection request between source node s, target node t with d unit(s) of data
$R=(V^R,E^R,f)$	survivable routing of a connection with node set $V^R\subseteq V$, edge set $E^R\subseteq E$, and bandwidth values $\forall e\in E^R: f(e)\le k(e)$
$G^*=(V,E^*,c)$	directed multi-graph with edge set E^*, where all edges in $G=(V,E,k,c)$ are replaced by $k(e)$ parallel edges each with cost $c(e)$
$R^*=(V^{R^*},E^{R^*})$	survivable routing of a connection in $G^*=(V,E^*,c)$, which is a DAG
A,B	two parts in which the connection's data is decomposed
$A\oplus B$	redundancy XOR data created at source s from connection's data parts
$E_A,E_B,E_{A\oplus B}$	routing DAGs for A, B and $A\oplus B$, where $E_A\cup E_B\cup E_{A\oplus B}=E^{R^*}$, $E_A\cap E_B=\emptyset$, $E_A\cap E_{A\oplus B}=\emptyset$, $E_B\cap E_{A\oplus B}=\emptyset$



(a) Diversity coding: using three link-disjoint paths.

(b) Generalized diversity coding: using three directed acyclic graphs.

Fig. 1. Different options to route $d=2$ data parts on three sub-flows.

or finding the optimal survivable routing for multiple edge failures are NP-complete problems [21]–[23]. However, in current transport networks, single edge failures are the most relevant failure scenarios [2], while dividing user data into more than two parts is impractical from an operational point of view. Furthermore, the minimum cost routing solution in most real-world networks can be reached by dividing the input flow into 2 sub-flows [9]. The first results on the complexity of this practically relevant special case of single edge failure minimum cost survivable routing when $d = 2$ was presented in [19], and it was shown that the problem is NP-complete with topological constraints, while polynomial-time solvable in the unconstrained case.

In this paper we will focus on the algorithmic techniques solving the survivable routing problem for this practical scenario, *i.e.*, the connection can be routed as two parts of equal (unit) size,[3] denoted by A and B; considering multiple constrained scenarios. We are searching a survivable routing for a single demand $D = (s, t, 2)$ at a time. Our algorithms exploit the special structural property of critical survivable routing solutions, which is detailed in Section II-B.

B. Structure of Critical Survivable Routing Solutions

First we define a couple of auxiliary graphs for simple arguments. Let $R = (V^R, E^R, f)$ denote the survivable routing graph, which is a subgraph of G (*i.e.*, $V^R \subseteq V$, $E^R \subseteq E$) with positive bandwidth f, (*i.e.*, $\forall e \in E^R : 0 < f(e) \leq k(e)$). In [17], [18] R was called "coding graph", and several properties have been proved which we will overview in this subsection. For the sake of easier presentation of our results, we introduce auxiliary graph $G^* = (V, E^*, c)$. The node set of G^* is the same as the node set of G, and each $e \in E$ is replaced by $k(e)$ *parallel edges* (*i.e.*, edges which have the same tail and head node as e), each with cost $c(e)$. Note that $k(e)$ is a non-negative integer, and a single edge failure e in G corresponds to the failure of all $k(e)$ edges in G^*. A *critical survivable routing*[4] $R^* = (V^{R^*}, E^{R^*})$ forms in G^* a Directed Acyclic Graph (DAG) according to Lemma 4 of [18]. It represents the routing of the connection, where $V^{R^*} \subseteq V$, $E^{R^*} \subseteq E^*$, while the objective function in Eq. (1) can be rewritten as:

$$\min \sum_{e \in E^{R^*}} c(e). \tag{2}$$

Definition 2: A ***routing DAG*** *$H \subset G^*$ is a subgraph of G^*, which is a DAG connecting s to t in such a way that there exist a positive integer l and different nodes $s = v_0, v_1, \ldots, v_l = t$ of H, such that in H for every i with $0 \leq i < l$ node v_{i-1} is connected to v_i by a directed path or two fully-edge-disjoint directed paths,*[5] *and H is the edge-disjoint union of these segments. If the segment from v_{i-1} to v_i consits of two directed paths, then v_{i-1} is called a* ***splitter node*** *and v_i a* ***merger node*** *(for obvious reasons). The edge set between a splitter node and the corresponding merger node is called an* ***island****.*

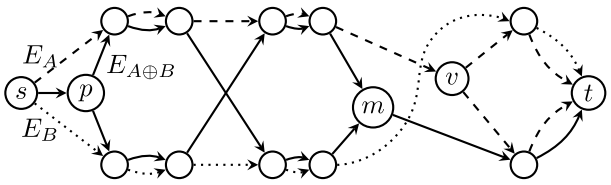


Fig. 2. A survivable routing $R^* = (V^{R^*}, E^{R^*})$ for connection $D = (s, t, 2)$ with the corresponding routing DAGs E_A, E_B and $E_{A\oplus B}$ denoted with dashed, dotted and solid edges, respectively.

A critical survivable routing R^* of G^* is the edge-disjoint union of three routing DAGs H_1, H_2, H_3. Moreover, for any edge $e \in E$ at most two corresponding parallel edges are in R^*, and if two such edges appear, then one of them is part of an island (*e.g.*, in Fig. 1b the routing DAG corresponding to sub-flow A has an island between splitter node p and merger node t, and this island contains parallel edges with the other two routing DAGs). Therefore, if we delete from all the H_i DAGs the edges corresponding to an edge $e \in E$, then at least two of the resulting DAGs $H_i \setminus \{e\}$[6] will still include directed paths from s to t and implement a survivable routing. Please note that, in the routing problem under investigation (*i.e.*, $d = 2$) the routing DAGs of the sum are denoted as $E_A, E_B, E_{A\oplus B}$, indicating that on the first DAG we send data part A, on the second one data part B, and on the third one $A \oplus B$. We have the following facts about diversity coding:

Theorem 1: If G contains a survivable routing then it contains a critical routing R as well.

If R is critical, then it is a DAG. Also, then R^* can be obtained as the union of edge-disjoint routing DAGs $E_A, E_B, E_{A\oplus B}$ of G^*.

Any node of a critical R can be splitter (or merger) in at most one of the three routing DAGs.

The proof of the claims of Theorem 1 are included in [17], [18]. To be more specific, in [17] the authors proved that a critical survivable routing is a DAG, while in [18] it was shown that its corresponding R^* can be decomposed into three edge-disjoint routing DAGs with disjoint set of splitter and merger nodes. As a corollary, R^* can be obtained as the union of three *appropriately selected* routing DAGs, which gives the basic concept of our routing algorithms proposed in this paper.

We will refer to the routings satisfying Theorem 1 as **Survivable Routing with Diversity Coding (SRDC)**. Note that, in an arbitrary SRDC solution (one is shown in Figure 2) the three routing DAGs carry, the same data part respectively (either A, B or $A \oplus B$), regardless of the failure (*i.e.*, no data retransmission or flow rerouting is necessary). Hence, if two routing DAGs remain $s - t$ connected, the source data parts A and B can be reconstructed at the destination node with an XOR operation (if necessary at all). In diversity coding all of $E_A, E_B, E_{A\oplus B}$ are $s \to t$ paths. However, the deployment of an SRDC solution might require splitting and merging of

[3]Input parameters (*e.g.*, edge capacity) can be scaled accordingly. Note that, $k(e)$ can be arbitrary real value in practice, however for this granularity we only need to know whether the edge can carry $0, 1$ or 2 data parts.

[4]Note that in [18] the term *minimum coding graph* is used instead of critical survivable routing.

[5]We call edge-disjoint paths in G^* "fully-edge-disjoint", if we explicitly require that their corresponding edges form edge-disjoint paths in G as well.

[6]We will use notation $G \setminus \{e\}$ to denote if a given edge or edge set is failed or removed from graph G.

the routing DAGs at the core nodes (*e.g.*, nodes p and m in Figure 2). In Figure 2 E_A consists of an $s \to v$ path and a $v \to t$ island, E_B is an $s \to t$ path, while $E_{A\oplus B}$ consists of a path $s \to p$, an island $p \to m$, and a path $m \to t$.

C. Incremental Upgrade of Node Capabilities

In [24]–[26] the possible extension of node capabilities in Software Defined Networking (SDN) is discussed. Several implementations of network coding are presented, where besides merging and splitting also the much more complex NC capability is implemented. In [24], [25] Multiprotocol Label Switching (MPLS) labels are utilized to distribute sequence numbers [27]. With the sequence numbers, we are able to identify, duplicate (split) or merge given flows. Therefore, a splitter can be deployed by applying regular flow rules, while a merger functionality can be implemented as a network function [22], [28]. Thus, we believe that implementing splitting and merging operations are reasonably simple in SDN; however, a software update is still necessary, which might be performed incrementally in the network.

Hence, in our model the set of the currently available splitter and merger nodes are given as the input of the problem and are denoted as $\mathcal{P} \subseteq V$ and $\mathcal{M} \subseteq V$, respectively. If all nodes are capable of performing the splitting and merging operation, *i.e.*, $\mathcal{P} = V$ and $\mathcal{M} = V$, then we say that the network is *fully upgraded*. If only a given set of nodes is capable to perform the actions, then we deal with the *partially upgraded network* scenario. Note that, *for a given connection request* $D = (s, t, 2)$ *we always assume that* $s \in \mathcal{P}$ *and* $t \in \mathcal{M}$, as these operations can be done by the application instead of network node upgrades.

III. Polynomial-Time Survivable Routing Algorithm in Fully Upgraded Networks

In this section we show that the minimum cost survivable routing problem for $d = 2$ with diversity coding is solvable in polynomial time if $\mathcal{P} = V$, $\mathcal{M} = V$ and there are no capacity constraints on the edges, meaning that $f(e)$ can be an arbitrary large positive integer. We shall see later, that large capacities are not really necessary in this setting, and in fact, $\forall e \in E : k(e) = 2$ is equivalent to the no capacity constraint scenario.

Suppose that we have a critical survivable routing R such that R^* is the sum of three routing DAGs E_A, E_B, and $E_{A\oplus B}$. We show here an important property of the islands of these DAGs:

Lemma 1: Let R^* be a critical survivable routing, which is a subgraph of G^* corresponding to network G that has no capacity constraints. Let R^* be the union of 3 routing DAGs E_A, E_B, and $E_{A\oplus B}$. Assume $\mathcal{E}^{R^*}_{p,m}$ is an island for a given splitter (p) and merger (m) node in E_A. Let $\mathcal{E}^{G}_{p,m}$ denote an arbitrary edge-disjoint dipath-pair[7] connecting p to m in G, with the corresponding fully-edge-disjoint dipath-pair $\mathcal{E}^{G^*}_{p,m}$ in G^*.

Then the routing $R' = (R^* \setminus \mathcal{E}^{R^*}_{p,m}) \cup \mathcal{E}^{G^*}_{p,m}$ is also survivable.

Proof: Since we have no capacity constraints, we can select the edges for the new island in G^* to be different from the edges used in E_B and $E_{A\oplus B}$. The survival property of routing R^* implies that no edge e of G appears in two routing DAGs, unless e appears in an island of one of the DAGs. This holds also in R' as the non-island edges of R^* and R' are the same, hence the deletion of all edges corresponding to e can disconnect at most one of the 3 routing graphs.[8] As a consequence, after the deletion of e we still have two $s - t$ dipaths in $R' \setminus \{e\}$. ∎

Corollary 1: Let R^* be a minimum cost survivable routing and $\mathcal{E}^{R^*}_{p,m}$ an island for a given splitter (p) and merger (m) node. If the network has no capacity constraints, then $\mathcal{E}^{R^*}_{p,m}$ is a minimum cost fully-edge-disjoint dipath-pair from node p to node m in G^*.

Proof: R^* is a minimum cost survivable routing, hence it is also critical. This implies that it is the union of three routing DAGs, and these may have islands. Now if $\mathcal{E}^{R^*}_{p,m}$ is not a minimum cost dipath-pair for a splitter-merger pair p, m, then with an optimal dipath-pair the construction of Lemma 1 would give a survivable routing R' with cost lower than R^*, which is a contradiction. ∎

An optimal dipath-pair for p, m can be calculated with Suurballe's algorithm in $O(|E| + |V| \log_2 |V|)$ steps [3]. Note that $\mathcal{E}^{G^*}_{p,m}$ survives a single edge failure, as it corresponds to a disjoint path-pair in G. Thus, we can substitute it with a fail-safe edge between p and m in E_A. This gives the basic idea for the algorithm, searching for a survivable routing in a tractable form.

Claim 1: Let R^* be a critical *survivable routing, decomposed into* 3 *routing DAGs E_A, E_B, and $E_{A\oplus B}$. Replacing every island $\mathcal{E}^{G^*}_{p,m}$ with an edge (p, m) results in three edge-disjoint $s \to t$ paths.*

Now we are ready to present our constructive proof, which gives a polynomial-time algorithm to find an optimal survivable routing. Let T denote the set of node-pairs that have an edge-disjoint dipath-pair between them in G. For each node-pair $(u, v) \in T$ we compute the minimum cost disjoint dipath-pair and save the total cost as $\mathrm{cost}(u, v)$. We construct the following auxiliary (multi-)graph $\widehat{G} = (V, \widehat{E}, \widehat{c})$. The node set of $\widehat{G}$ is the same as the node set of G, and we will have $|E| + |T|$ edges. The edges of $\widehat{G}$ are the edges of E with cost $\widehat{c}(e) = c(e)$ for every $e \in E$, and we add an edge $e_n = (u, v)$ for every $(u, v) \in T$ with cost $\widehat{c}(e_n) = \mathrm{cost}(u, v)$. We refer to the newly added edges as *virtual edges*.

Theorem 2: If the network has no capacity constraints on the edges, the minimum cost survivable routing R^* can be computed in $O(|V||E| \log_{1+|E|/|V|} |V|)$ steps.

Proof: We start with a lemma about edge-disjoint dipaths in $\widehat{G}$.

Lemma 2: Let $\pi_A, \pi_B, \pi_{A\oplus B}$ be three edge-disjoint $s \to t$ dipaths in $\widehat{G}$. By replacing every virtual edge (p, m) with an island $\mathcal{E}^{G^*}_{p,m}$ of minimum cost we get edge-sets E_A, E_B, and $E_{A\oplus B}$ in G^* that form a survivable routing. Moreover, the cost of these edge-sets in G^* equals the cost of the paths in $\widehat{G}$, and vice versa.

[7] For brevity, we use "dipath" instead of directed path in the proofs.

[8] Please note that the modified E_A may no longer be a DAG.

Proof: Equality of costs is straightforward. Since $\pi_A, \pi_B, \pi_{A\oplus B}$ are edge-disjoint in $\widehat{G}$, every edge e in E is contained in at most one path as a non-virtual edge, and may be contained in other island(s) used for substituting virtual edges. In case of a failure of an $e \in E$, the latter remain connected, hence at most one of the edge-sets corresponding to $\pi_A, \pi_B, \pi_{A\oplus B}$ can be disconnected, which proves the claim. ∎

The Lemma above implies that any three edge-disjoint $s \to t$ dipaths in $\widehat{G}$ can be transformed into a feasible survivable routing in G with the same total cost as the three dipaths. To complete the proof of correctness, we need to show that a minimum cost survivable routing R is mapped to a union of three edge-disjoint $s \to t$ dipaths in $\widehat{G}$ with minimal cost. Theorem 1 implies that R must be critical. Now according to Claim 1, R corresponds to three edge-disjoint $s \to t$ dipaths in $\widehat{G}$. Moreover, the cost of the three edge-disjoint $s \to t$ dipaths equals to the bandwidth cost of the survivable routing. This cost must be minimal for the three $s-t$ dipaths in $\widehat{G}$ according to Lemma 2.

Finally, finding the minimum cost of three edge-disjoint paths could be done in $O(|E|\log_{1+|E|/|V|}|V|)$ time [3]. In the construction of $\widehat{G}$, finding the pair of shortest edge-disjoint path from a *single source to every destination* is $O(|E|\log_{1+|E|/|V|}|V|)$ time [29], which *should be launched for every source node*, resulting $O(|V||E|\log_{1+|E|/|V|}|V|)$ steps, which proves the theorem. ∎

It was shown in [18], as a consequence of Theorem 1, that in a *critical survivable routing* for a connection with $d = 2$ the bandwidth values are $f(e) \leq 2$ for every $e \in E$. Thus, without loss of generality, we may build the auxiliary graph G^* with $k(e) = 2$ (*i.e.*, at most $2|E|$ edges) when searching for a solution in the no capacity constraint scenario.

IV. Approximation Survivable Routing Algorithm in Partially Upgraded Networks

In this section we present an approximation algorithm to solve the SRDC problem in partially upgraded networks with no capacity constraints on the edges. First, we show that the algorithm provided by Theorem 2 cannot solve the survivable routing problem when not all nodes are capable to perform the splitting and merging action. In Figure 3 only node m is upgraded, *i.e.*, only node m can be a splitter or merger in addition to the source (s) and the destination (t) node. If diversity coding is used, the total cost of the solution is 22, since the user data is sent along three edge-disjoint paths (*i.e.*, $\pi_1 = s \to v_1 \to v_2 \to v_3 \to v_{12} \to t$ (cost 5), $\pi_2 = s \to v_7 \to v_8 \to v_9 \to v_{13} \to t$ (cost 5) and $\pi_3 = s \to v_{10} \to v_{11} \to t$ (cost 12)). If $1 + 1$ is used the cost of the solution is 20 (twice the cost of the π_1 and π_2 paths). The optimal survivable routing is 19 (given by the dotted, dashed and densely dotted edges in Figure 3). Note that, between nodes v_4 and m two copies of the same data is transferred in order to get to merger node m in the network. However, using the polynomial time algorithm provided by Theorem 2 to find the three routing DAGs between s and t would use v_4 as a merger node to remove the duplicate copies

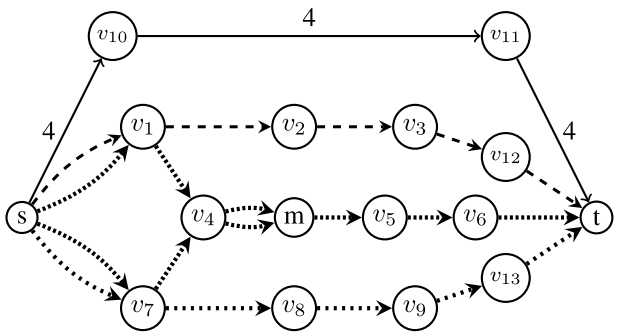


Fig. 3. The optimal survivable routing solution in partially upgraded networks ($\mathcal{P} = \{s, m\}$ and $\mathcal{M} = \{t, m\}$) with cost 19 is not critical. Edge capacities are $\forall e \in E : k(e) = 2$ and edge costs are unit (otherwise written next to the edge).

from edge (v_4, m), which would result in an invalid solution with cost 18.

In order to solve this issue, Algorithm 1 is based on finding 3-edge-disjoint paths in an auxiliary graph $\widetilde{G}$, which is constructed in the same way as $\widehat{G}$ in Section III, with the exception that virtual edges are added only between upgraded nodes where a disjoint path-pair exist ($\forall u \in \mathcal{P}, v \in \mathcal{M} : u \neq v$) instead of every pair of distinct node-pairs where a disjoint path-pair exist. Obviously, if $\mathcal{P} = V,\ \mathcal{M} = V$ we get back $\widehat{G}$ and the constructive algorithm of Theorem 2. The computational complexity of Algorithm 1 is dominated by the creation of the auxiliary graph resulting $O(|V||E|\log_{1+|E|/|V|}|V|)$ steps.

A. Algorithm 1 Approximates SRDC

$1 + 1$ was proved to be a 2-approximation [23] for the general survivable routing problem. However, our evaluations and simulations on hundreds of graphs showed that the ratio between the cost of the optimal SRDC solution and $1 + 1$ is below $4/3$ in all investigated topologies. Thus, it led us to the conjecture that $1 + 1$ is a $4/3$-approximation for the special case of $d = 2$ data units.

Claim 2: $1+1$ *is a* $4/3$*-approximation algorithm for SRDC when* $\forall e \in E : k(e) = 2$.

Proof: Let the two edge-disjoint paths of the $1+1$ solution be denoted by π_1, π_2 and denote their cost[9] as $|\pi_1|$ and $|\pi_2|$, respectively. Furthermore, the paths of the SRDC solution are denoted with $\pi_{E_a}, \pi_{E_b}, \pi_{E_c}$ in $\widehat{G}$. We know that the cost of the paths for the $1 + 1$ solution *i.e.*, $|\pi_1| + |\pi_2|$ is lower than the cost of each path-pair form the SRDC solution, since we would utilize the lower cost paths for the $1+1$. Hence we know that: $|\pi_1| + |\pi_2| \leq |\pi_{E_a}| + |\pi_{E_b}|$, $|\pi_1| + |\pi_2| \leq |\pi_{E_a}| + |\pi_{E_c}|$, and $|\pi_1| + |\pi_2| \leq |\pi_{E_c}| + |\pi_{E_b}|$.

We have to show that the following inequality always holds:

$$2(|\pi_1| + |\pi_2|) \leq \frac{4}{3}(|\pi_{E_a}| + |\pi_{E_b}| + |\pi_{E_c}|), \tag{3}$$

We emphasize that the $1 + 1$ sends both data parts (A and B) on both paths resulting in cost of $2(|\pi_1| + |\pi_2|)$, while SRDC transfers A, B, and $A \oplus B$ on three disjoint paths resulting in the overall cost of $|\pi_{E_a}| + |\pi_{E_b}| + |\pi_{E_c}|$.

[9]If $\forall e \in E : c(e) = 1$ then the cost denotes the length of each path.

Algorithm 1 Survivable Routing With Diversity Coding in Partially Upgraded Networks

Input: $G^* = (V, E^*, c)$, $D = (s, t, 2)$
Result: $R^* = (V^{R^*}, E^{R^*})$, in specific, routing DAGs E_A, E_B, and $E_{A\oplus B}$
begin
  Define cost $\widehat{c} : E \to \mathbb{R}^+$ and edge set $\widetilde{E} = \emptyset$, $E_s = \emptyset$;
  `// Create graph` $\widetilde{G} = (V, \widetilde{E}, \widetilde{c})$.
  Add $\forall e \in E$ to $\widetilde{E}$ with $\widetilde{c}(e) = c(e)$;
  for $u \in \mathcal{P}$**:** **do**
    Find the pair of shortest edge-disjoint paths from source u to all other nodes $v \in \mathcal{M}, u \neq v$ in G with Suurballe's algorithm (denote their cost with $\text{cost}(u, v)$);
    Add virtual edge between the splitter merger node-pairs where a disjoint path-pair exist $e_n = (u, v)$ to $\widetilde{E}$ with $\widetilde{c}(e_n) = \text{cost}(u, v)$;
  `// Find 3 edge-disjoint paths in` $\widetilde{G}$.
  Find minimum cost 3 edge-disjoint paths between s and t in $\widetilde{G}$ with Suurballe's algorithm;
  Add the traversed edges (*i.e.*, their corresponding edges in G^*) to E_s;
  for $e = (u, v) \in E_s$ **do**
    if e *is a virtual edge* **then**
      Replace virtual edge e with minimum cost island $\mathcal{E}^{G^*}_{u,v}$ in E_s;
  `// Save optimal survivable routing` R^*.
  for $e = (u, v) \in E_s$ **do**
    Add nodes u, v to V^{R^*} (if $u, v \notin V^{R^*}$);
    Add edge e to E^{R^*};

If we add the three inequalities and multiply by 2 we get that:

$$6(|\pi_1| + |\pi_2|) \leq 4(|\pi_{E_a}| + |\pi_{E_b}| + |\pi_{E_c}|), \tag{4}$$

From here it follows trivially that the inequality in Eq. (3) is always satisfied. ∎

Built on this fact, the following theorem can be stated.

Theorem 3: Algorithm 1 is a $4/3$*-approximation algorithm for SRDC when* $\forall e \in E : k(e) = 2$.

Proof: Since the source s and target node t are always allowed to be splitter and merger, Algorithm 1 can return $1+1$ as a worst case solution,[10] for every possible input. As $1+1$ is a 4/3-approximation algorithm for SRDC according to Claim 2, Algorithm 1 provides a 4/3-approximation as well. ∎

V. Survivable Routing With Limited Free Capacities

In practice some edges might have limited capacities (*i.e.*, $k(e) = 1$, referred to as "bottleneck edges" in the rest of the paper), depending on the previously allocated demands. It was previously shown that with capacity constraints in partially upgraded networks the SRDC problem becomes NP-complete [19]. Hence, in Section V-A we present an Integer Linear Program (ILP) in general network topologies. On the other hand, in Section V-B we give a polynomial-time algorithm in directed acyclic graphs.

First, we show that the algorithm presented in Theorem 2 cannot cope with networks with some edge capacities $k(e) = 1$. The problem is that in such a capacity constrained case $\mathcal{E}^G_{p,m}$ depends on the route of the other two routing DAGs, *i.e.*, another routing DAG may use the single available capacity unit along an edge $e \in \mathcal{E}^G_{p,m}$ of the minimum cost disjoint path-pair. For example, Figure 4(a) shows a network with an optimal survivable routing of cost 20. Note that, the virtual edge $e_n = (v_1, t)$ has cost $\widehat{c}(e_n) = 5$ because $\text{cost}(v_1, t)$ is the cost of the shortest path-pair $v_1 \to v_2 \to v_3 \to t$ and $v_1 \to v_5 \to t$ is $3 + 2 = 5$. The minimum cost 3 edge-disjoint paths in $\widehat{G}$ are shown in Figure 4(b). Clearly, this is not a valid solution in the capacity constrained case, as edge $e = (v_2, v_3)$ has only $k(e) = 1$ available capacity in G, while two routing DAGs should use it in the optimal solution.

A next attempt for solution would be to modify the algorithm to find the minimum cost 3 edge-disjoint paths with Suurballe's algorithm using the augmenting path technique. Applying this technique to SRDC, the virtual edges are only traversed by the 3^{rd} augmenting path, only after 2 edge-disjoint paths were already found. A natural extension of the polynomial time algorithm provided in Section III may be to run the disjoint path search for each virtual edge (*e.g.*, to (v_1, t)) as a disjoint path-pair between nodes v_1 and t. During this search the reverse edges of the already found 2 edge-disjoint paths can be used (shown in Figure 4(c)) similarly as in Suurballe's algorithm, and additionally it can use the reverse edges of the third edge-disjoint path's segment between s and v_1 (which is $s \to v_7 \to v_2 \to v_3 \to v_6 \to v_5 \to v_1$). This could result in an augmenting path between splitter v_1 and merger t of $v_1 \to v_5 \to v_6 \to v_3 \to t$. In this case the second augmenting path between splitter v_1 and merger t would be $v_1 \to v_4 \to v_5 \to t$. This in fact results in a vulnerable routing shown in Figure 4(d) with cost 16.

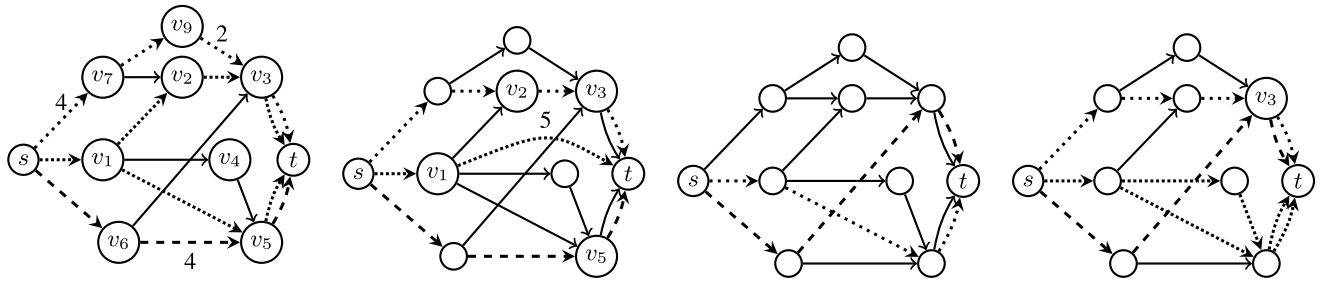
A. Optimal Solution in General Graphs

In this section we present an ILP to obtain an optimal survivable routing R in terms of bandwidth cost. The ILP formulation provides the three routing DAGs for SRDC even with capacity constraints and node limitations. To do so, we need to introduce the so called *reduced capacity function* [17] (see Theorem 4):

$$k'(e) = \begin{cases} 1.5 & \text{if } k(e) \geq 2 \\ 1 & \text{if } k(e) = 1. \\ 0 & \text{otherwise} \end{cases}$$

Theorem 4: [17, Theorem 2] A survivable routing exists in a given graph $G = (V, E, k, c)$ *if and only if there is a flow of value three in* $G = (V, E, k', c)$.

[10] Note that the $1 + 1$ can be considered as sending $A \oplus B$ along two edge-disjoint paths *i.e.* on the island between the source s and target node t.



(a) The optimal survivable routing with the routing DAGs. The edge costs other than 1 are shown.

(b) The 3 paths in $\widehat{G}$ of Theorem 2. For the sake of simplicity, only virtual edge (v_1, t) is shown.

(c) The 2 edge-disjoint paths with minimum cost.

(d) Solution of the augmenting path technique: A non-critical vulnerable routing, where the failure of (v_3, t) disrupts two routing DAGs.

Fig. 4. An example network $G^* = (V, E^*, c)$ with capacity constraint on the edges (remember from the construction of G^* that $k(e) = 2$ edges in G are parallel edges in G^*), where $c(e) = 1$, or written next to the edge otherwise. The edges of the routing DAGs E_A, E_B and $E_{A\oplus B}$ are denoted as dashed, dotted and densely dotted lines, respectively. Here Algorithm 1. presented in [19] fails for connection $D = (s, t, 2)$.

Theorem 4 will be used in our ILP formulation. Note that, given a routing DAG E_A in a critical survivable routing, a variable x^A which is half on the edges of an island and 1 on all other (path) edges in E_A, forms an $s - t$ flow of value 1, according to Theorem 1. Armed with this fact, we investigate the benefits which diversity coding can provide for survivable routing. Our goal is to obtain the (critical) bandwidth values $f(e)$ in the arbitrary directed input graph $G = (V, E, k, c)$ which minimize the bandwidth cost in terms of Equation 1 for the connection $D = (s, t, 2)$. The three flows are denoted as $w \in \{A, B, A \oplus B\} = \mathcal{W}$, respectively, with corresponding (real) flow variables $x^w(e)$ and indicator variables $f^w(e)$. We have $f^w(e) = 1$ if and only if there is in w a positive flow through e, otherwise $f^w(e) = 0$. Based on Theorem 4 the reduced capacity values $k'(e)$ ensure that the failure of an arbitrary edge e disconnects at most one routing DAG, thus, at least two routing DAGs remain which connect s and t, *i.e.*, the data can be decoded at the destination. Our objective is to minimize the bandwidth cost of the SRDC problem in terms of Equation 1:

$$\min \sum_{e \in E} c(e) \cdot f(e).$$

The following constraints are required:

$$\forall w \in \mathcal{W}, \forall i \in V\text{:} \quad \sum_{(i,j)\in E} x^w(i,j) - \sum_{(j,i)\in E} x^w(j,i) = \begin{cases} 1 & \text{, if } i = s \\ -1, & \text{if } i = t \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

$$\forall w \in \mathcal{W}, \forall i \in \mathcal{P} \setminus \mathcal{M}\text{:} \quad \sum_{(i,j)\in E} f^w(i,j) \geq \sum_{(j,i)\in E} f^w(j,i), \tag{6}$$

$$\forall w \in \mathcal{W}, \forall i \in \mathcal{M} \setminus \mathcal{P}\text{:} \quad \sum_{(i,j)\in E} f^w(i,j) \leq \sum_{(j,i)\in E} f^w(j,i), \tag{7}$$

$$\forall w \in \mathcal{W}, \forall i \in V \setminus \{P \cup \mathcal{M}\}\text{:} \quad \sum_{(i,j)\in E} f^w(i,j) = \sum_{(j,i)\in E} f^w(j,i), \tag{8}$$

$$\forall e \in E\text{:} \sum_{w \in \mathcal{W}} x^w(e) \leq k'(e), \tag{9}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{:}\ x^w(e) \leq f^w(e), \tag{10}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{:}\ 2x^w(e) \geq f^w(e), \tag{11}$$

$$\forall e \in E\text{:} \sum_{w \in \mathcal{W}} f^w(e) = f(e), \tag{12}$$

$$\forall e \in E\text{:}\ f(e) \leq k(e), \tag{13}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{:}\ 0 \leq x^w(e) \leq 1, \tag{14}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{:}\ 0 \leq f^w(e) \leq 1 \text{ are integers.} \tag{15}$$

The constraint in Eq. (5) formulates the flow conservation for each routing DAG w. Additionally, Eq. (6)-(8) formulate the constraints needed for representing the different node capabilities. Namely, Eq. (6) represents the set of nodes that can only perform the splitting operation ($\mathcal{P} \setminus \mathcal{M}$). Eq. (7) is needed for the nodes that are only capable of merging the data stream ($\mathcal{M} \setminus \mathcal{P}$) and Eq. (8) is for non-upgraded nodes. Note that we do not need extra constraints for the nodes ($\mathcal{P} \cup \mathcal{M}$) that can both split and merge the data. Eq. (9) sets the maximal flow value based on the reduced capacity function, while Constraints (10)-(11) sets the indicator variables $f^w(e)$ of edge usage for the routing DAGs in G. Eq. (12) sets the bandwidth value in $G = (V, E, k, c)$, *i.e.*, if edge e was used in an arbitrary routing DAG w, we have to include it in the final solution with value $f(e) = \sum_{\mathcal{W}} f^w(e)$. Constraints (14)-(15) set the bounds for the flow variables, and set the integer constraint for the indicator variables $f^w(e)$. Note that the $f^w(e)$ variables correspond to the edge set $w \in \mathcal{W}$ in the solution, *i.e.*, provide the three DAGs. Since Constraint (5) ensures that $x^A + x^B + x^{A\oplus B}$ gives an $s - t$ flow of value 3 in G, from Theorem 4 we get that $f(e)$ is indeed survivable.

In order to analyze the complexity of the ILP we have to assess the number of constraints and variables necessary to formulate the problem. For the formulation of the flow and node capability constraints, *i.e.*, for Eq. (5)-(8) $O(|V|)$ constraints are necessary. The rest of the equations, *i.e*, Eq. (9)-(15) are formulated for the links; thus, $O(|E|)$ constraints are needed. Regarding the variables, we have flow and indicator variables (*i.e*, $f^w(e)$ and x^A) defined to each

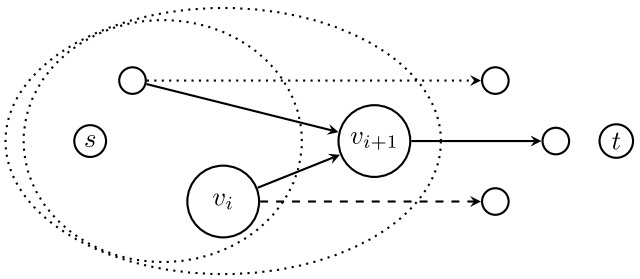


Fig. 5. Edges in L_i, P_i, Y_i and $L_{i+1}, P_{i+1}, Y_{i+1}$ denoted with dashed, dotted and solid edges, respectively.

edge, which results in $O(|E|)$ altogether. Therefore, the ILP has $O(|E|)$ variables and $O(|V| + |E|)$ constraints.

B. Polynomial-Time Algorithm in Directed Acyclic Graphs

Although finding the optimal SRDC solution is NP-complete in general graphs, here we give a polynomial-time algorithm for the special case when the input topology is a DAG. Given a DAG G, let $v_1, v_2, \ldots, v_n$ be a fixed topological order of the nodes in G, that is, for every edge $e = (v_i, v_j)$, $i < j$ holds. For capacities $k(e)$ and cost $c(e)$ we are going to give an algorithm to find a minimum cost survivable routing solution for demand $D = (s, t, 2)$. We can assume that $s = v_1$ and $t = v_n$.

Definition 3: For any $1 \leq i < n$, let $S_i := \{v_1, \ldots, v_i\}$ and $T_i := \{v_{i+1}, \ldots, v_n\}$, finally let C_i denote the set of edges in G in the $S_i - T_i$-cut, that is those with tail in S_i and head in T_i. We call these cuts ***topological cuts****.*

Let C_i be a topological cut of G and L_i, P_i, Y_i three, not necessarily disjoint 1 or 2-element subsets of C_i. We call such an ordered *triplet τ a* ***coloring*** *of C_i, where $L_i \cup P_i \cup Y_i$ are the* ***colored*** *edges, and edges in L_i, P_i, Y_i are called* ***lime, purple, yellow****, respectively. We say that this coloring is* ***survivable****, if after the removal of any edge e in C_i, at least two of the sets L_i, P_i and Y_i remain non-empty. A coloring of cut C_i and a coloring of cut C_{i+1} are* ***compatible****, if they are the same on $C_i \cap C_{i+1}$ and for every colored edge in C_{i+1} with tail v_{i+1} there is an edge entering v_{i+1} with the same color (see Figure 5). A coloring τ_i of C_i is* ***feasible*** *for a capacity function k, if for every edge e in C_i, the number of colors containing e is at most $k(e)$. For a subset of edges $F \subseteq E$ let F^i denote $F \cap C_i$. We say that F is* ***s-reachable*** *if for every edge f in F there is a path from s to f through edges in F.*

Intuitively, a coloring of C_i intends to capture the parts of the survivable routing DAGs E_A, E_B, $E_{A \oplus B}$ which are subsets of C_i. As we seek a minimum cost solution, these parts cannot have more than two edges (according to Theorem 1).

Lemma 3: For minimum cost survivable routing R that decomposes into DAGs E_A, E_B and $E_{A \oplus B}$, for every topological cut C_i, the coloring $\tau_i = (E_A^i, E_B^i, E_{A \oplus B}^i)$ is survivable and feasible and consecutive colorings are compatible.

Proof: In a survivable routing every edge intersects at most two of the three routing DAGs, hence the removal of any edge from a cut C_i leaves at least one of the corresponding color classes untouched, which proves the survivability of the cuts. Since an edge of capacity 2 appears in at most two out of the three routing DAGs, the corresponding edge sets in a cut are also feasible. Finally compatibility of the cuts follows from the fact that the routing DAGs are s-reachable. ■

Lemma 4: If for three s-reachable subsets of edges L, P, Y for every topological cut C_i coloring $\tau_i = (L^i, P^i, Y^i)$ is survivable, then L, P and Y form survivable routing DAGs of G.

Proof: Assume indirectly that there is an edge $e = (v_i, v_j)$ the removal of which disconnects at least two DAGs. Then it is easy to check that cut C_i is not survivable. ■

Now we are ready to describe our algorithm, based on *dynamic programming*. For every $1 \leq i < n$, let G_i denote the graph obtained from G by the contraction of nodes in T_i. We are going to calculate the minimum cost of three survivable routing DAGs in G_i with a fixed survivable, feasible coloring τ_i on C_i. This value will be denoted by $\mathrm{opt}(\tau_i)$.

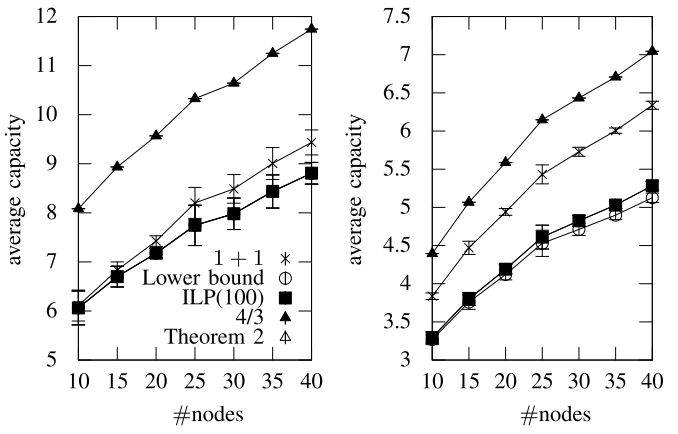
For $i = 1$, the cost of a survivable, feasible coloring of C_1 is just the sum of the costs of the colored edges with multiplicity (an edge may have multiple colors). For $1 < i < n$, let a survivable coloring $\tau_i = (L_i, P_i, Y_i)$ be given. Then

$$\begin{aligned}&\mathrm{opt}(\tau_i)\\ &= \sum_{e \in L_i \cap \delta^+(v_i)} c(e) + \sum_{e \in P_i \cap \delta^+(v_i)} c(e) + \sum_{e \in Y_i \cap \delta^+(v_i)} c(e)\\ &\quad + \min\left\{\mathrm{opt}(\tau_{i-1}) \,\middle|\, \begin{array}{l}\tau_{i-1} \text{ survivable, feasible coloring}\\ \text{of } C_{i-1} \text{ and compatible with } \tau_i\end{array}\right\}.\end{aligned}$$

From Lemma 3 and Lemma 4 the cost of a minimum cost survivable routing is $\min\{\mathrm{opt}(\tau_{n-1}) | \tau_{n-1}$ survivable, feasible coloring of $C_{n-1}\}$. Since edge capacities in a minimum cost survivable routing can be assumed to be 1 or 2, for every edge there are at most 6 possible colorings. Hence the number of survivable, feasible colorings of a topological cut C_i is $O(|C_i|^6)$, and the above recursion yields a polynomial-time algorithm. Note that the case of splitter and merger node sets (when $\mathcal{P}$ and $\mathcal{M}$ are given) can be easily integrated in the algorithm by the modification of compatibility, *e.g.*, only a merger node v_{i+1} can have two entering and one outgoing edges of the same color (see Figure 5).

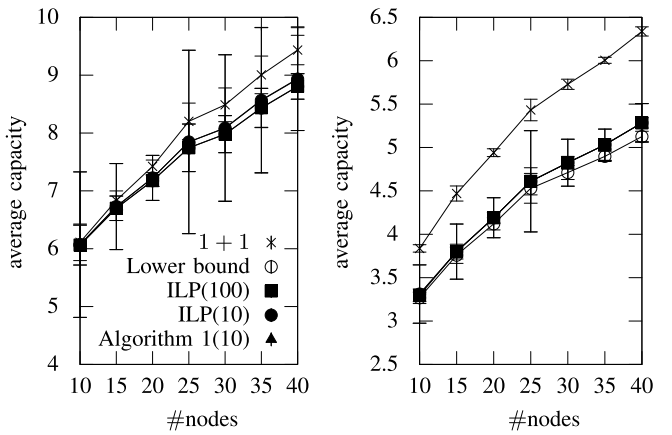
VI. Experimental Results

In our simulations we assume that a set of connection requests D is given between all possible source-target pairs and plot the *average capacity reserved per connection* for every survivable routing approach. We compare our methods to the theoretical lower bound [22] (data can be divided into an arbitrary number of parts) and to $1 + 1$ protection, which is a 2-approximation of the survivable routing problem against single edge failures in general [17], [23] and a 4/3-approximation of the SRDC problem with $d = 2$ data units. As a baseline, we also plot the optimal solution of the ILP(100) presented in Section V-A and the 4/3-approximation line of the optimal solution. The number in the parenthesis beside the algorithms refers to the percentage of upgraded nodes, *e.g.*, (10) means 10% of the nodes are upgraded with splitter/merger functionality. We investigate random generated real-like planar



(a) Optimality gap in sparse graphs (b) Optimality gap in maxplan graphs

Fig. 6. Bandwidth cost in sparse (average nodal degree between 2.4 and 3.2) and maximal planar (maxplan) graphs (average nodal degree between 4.2 and 5.7) with no capacity constraints in fully upgraded networks.



(a) Optimality gap in sparse graphs (b) Optimality gap in maxplan graphs

Fig. 7. Bandwidth cost in sparse (average nodal degree between 2.4 and 3.2) and maximal planar (maxplan) graphs (average nodal degree between 4.2 and 5.7) with no capacity constraints in partially upgraded networks.

$G = (V, E, k, c)$ topologies with different sizes and densities, and some real-world transport network topologies, too. By the real-like topologies, the simulation results are obtained by averaging several instances from the topologies with the same properties (95% confidence interval is plotted).

Note that we do not compare our method to the DC since the blocking probability of the DC is extremely high, due to the fact that it requires the existence of three edge-disjoint paths between the communication endpoints.

A. Fully Upgraded Networks Without Capacity Constraints

Here, we present the simulation results without capacity constraints in Figure 6. The x-axis represents the node numbers of the random networks, while the y-axis shows the average capacity reserved per connection. Our results in Figure 6a show why $1 + 1$ is still the most often deployed protection scheme, as the gap between the bandwidth cost of $1 + 1$ and the theoretical lower bound for survivable routing is small. However, our SRDC algorithm given by Theorem 2 outperforms $1 + 1$, and reaches the theoretical lower bound. This also demonstrates that the lower bound can be achieved by dividing the data into two parts in these topologies. On the other hand, in maximal planar graphs in Figure 6b the theoretical lower bound requires that connection data is divided into more than two data units. Although our algorithm still approaches the lower bound, $1+1$ *reserves one more edge (bandwidth unit) per connection* to provide the same simplicity as our SRDC method.

B. Partially Upgraded Networks Without Capacity Constraints

In Figure 7 we show a scenario where not all nodes are upgraded with the splitter/merger functionality. In particular, in Figure 7 we show that just by upgrading 10% of the nodes (which we consider as a typical scenario of incremental network upgrade) we can achieve significant improvement compared to the $1 + 1$, both in sparse (Figure 7a) and dense networks (Figure 7b). We can observe that Algorithm 1 provides results near to the optimal solution of the ILP(100), and demonstrates that even with 10% of upgraded nodes, our SRDC approach can bring real benefits. As the source and destination nodes are always considered to be splitter/merger for a given connection demand, in the denser networks (Figure 7b) almost always exist 3-disjoint paths, and no in-network splitting and merging is required. Hence, network node upgrades cannot bring huge capacity savings in this setting.

C. Experimental Results With Capacity Constraints

In this subsection, we investigate the capacity constraint case through the performance of our methods in real network topologies (SNDLib [30] and Rocketfuel ASs [31]). In this scenario the network is heavily loaded, *i.e.*, due to the heavy traffic load some edges lack free capacity (*i.e.*, are considered as bottleneck edges). To achieve this, we continuously increase the traffic load and analyze the given state of the network. For a fair comparison, we only take into account the non-blocking scenarios, *i.e.*, where there is still a disjoint path-pair between all source and destination pairs even for $1+1$. Since no traffic matrix is given beforehand, we identified a certain number of edges which are most prone to congestion based on their betweenness centrality value. We considered these edges as bottlenecks in the simulations (*i.e.*, only a single capacity unit $k(e) = 1$ is available on them). Three traffic scenarios are distinguished:

- Light traffic load: no bottleneck edges in the network,
- Medium traffic load: maximum 10 bottleneck edges,
- Heavy traffic load: maximum 20 bottleneck edges.

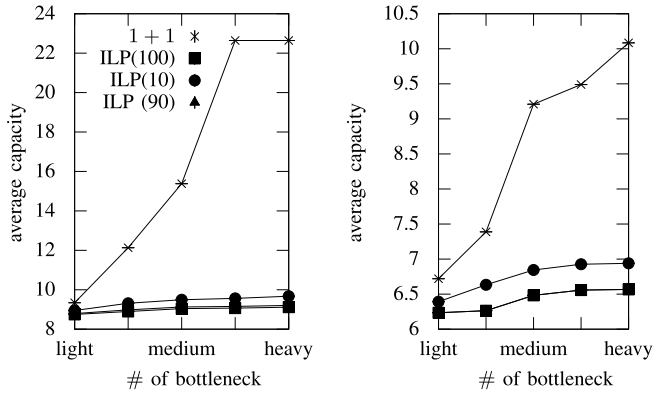
Note that in each scenario the maximum number of bottlenecks is chosen only if it does not violate the non-blocking condition.

In Figure 8 we show the results when both the capacity and the node capabilities are constrained which is the most challenging SRDC subproblem. One can observe that as the traffic load increases, the average bandwidth cost of $1 + 1$ increases dramatically (as the $1 + 1$ cannot use bottleneck edges), while the average bandwidth cost of the ILP(100), *i.e.*, optimal solution in fully upgraded networks remains low and

TABLE II

SIMULATION RESULTS ON REAL NETWORKS (UPPER PART: SNDLIB [30], LOWER PART: ROCKETFUEL ASS [31]) WITH HEAVY TRAFFIC LOAD

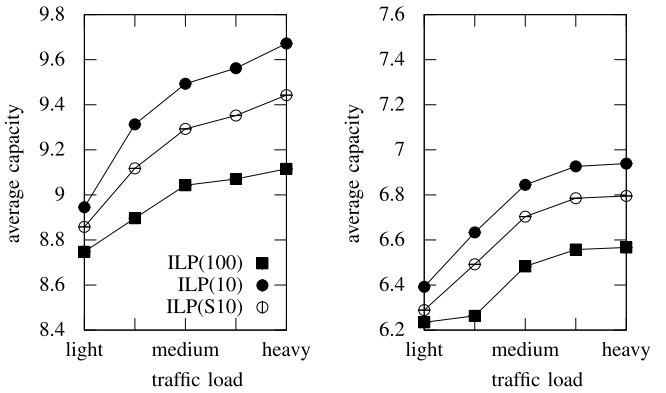
	Graph			Capacity consumption										
					ILP									
	$\|V\|$	$\|E\|$	diam.	1+1	All	4	S4	3	S3	2	S2	1	S1	0
Pan-European	16	22	6	12.4	7.64	7.8	7.81	7.84	7.81	7.97	7.81	8.01	7.95	8.16
Germany	17	26	6	11.92	7.51	7.78	7.51	7.81	7.58	7.82	7.69	7.90	7.80	7.98
European	22	45	5	12.02	5.43	5.55	5.65	5.58	5.65	5.59	5.65	5.62	5.66	5.74
Italian	33	56	9	21.04	9.30	9.89	9.46	9.97	9.53	9.97	9.58	9.99	9.75	10.07
Cost 266	37	57	8	22.64	9.11	9.52	9.24	9.54	9.25	9.61	9.33	9.67	9.44	9.77
AS6461 (Abovenet)	17	37	4	12.98	4.79	4.90	4.90	4.90	4.90	4.90	4.90	4.97	4.90	5.06
AS1755 (Ebone)	18	33	5	12.96	5.80	5.93	5.94	5.96	5.94	6.02	5.94	6.10	5.94	6.11
AS3967 (Exodus)	21	36	5	12.13	6.36	6.69	6.65	6.75	6.66	6.76	6.66	6.76	6.66	6.83
AS7018 (AT&T)	22	38	5	15.93	6.30	6.56	6.31	6.57	6.33	6.61	6.42	6.65	6.58	6.75
AS3257 (Tiscali)	27	64	4	7.45	4.71	4.91	4.84	4.93	4.84	4.95	4.84	4.96	4.84	4.96
AS1239 (Sprintlink)	30	69	6	10.08	6.56	6.83	6.71	6.85	6.71	6.91	6.71	6.94	6.79	6.99



(a) Bandwidth cost in Cost 266 (37 nodes, 57 edges with diameter 8) [30]

(b) Bandwidth cost in Sprintlink (AS1239) (30 nodes, 59 edges, with diameter 6) [31]

Fig. 8. The bandwidth cost in real-world topologies with capacity constraints in partially upgraded networks.



(a) Bandwidth cost in Cost 266 (37 nodes, 57 edges with diameter 8) [30]

(b) Bandwidth cost in Sprintlink (AS1239) (30 nodes, 59 edges, with diameter 6) [31]

Fig. 9. Comparison of the smart and the random node upgrade strategies with capacity constraints in partially upgraded real-world topologies.

scales well with the traffic load. Furthermore, with the increase of the percentage of the splitter/merger nodes the average capacity reserved per connection decreases, demonstrating the benefits of incremental deployment.

D. Incremental Deployment

In this subsection, we intend to give an insight for network operators how incremental deployment of SRDC improves the overall performance, and on the way the upgradeable nodes should be selected according to the budget. For selecting the upgradeable nodes, we compare two approaches:

- Random: nodes are selected uniformly random,
- Smart (S in the figures): in a pre-process phase we run the algorithm given by Theorem 2 for each source-target pair assuming there are no capacity constraints on the edges. We count how many times a given node was utilized as a splitter/merger in these solutions, and greedily upgrade the nodes with the highest values until the budget is reached.

In Figure 9 we show the effects of the traffic load increase. We see that the gap between the ILP(100), ILP(10) and ILP(S10) increases gradually as the traffic load increases. Furthermore, it also demonstrates that even with randomly upgrading 10% of the nodes, SRDC performs close to the optimal solution even in a heavy traffic scenario.

In Table II we demonstrate the benefits of a more fine-grained incremental deployment strategy on real network topologies in the heavily loaded network scenario. In particular Table II compares the results of $1+1$ and the presented ILP solution where the number in the table refers to the number of upgraded core nodes (besides the source and destination nodes of each connection request, which are always considered as merger/splitter), *e.g.*, S4 means that 4 of the core nodes are upgraded with splitter/merger functionality with the help of "smart" selection. Note that 0 refers to the case where only the source and destination nodes are capable of performing the splitting and merging operation, *i.e.*, the survivable routing is either $1+1$ or traditional diversity coding, whichever is better. Even in this case, we can achieve a significant gain, *i.e.*, the average capacity consumption can drop down to half from $1+1$ compared to the ILP. Furthermore, even with upgrading a small number of (random/cheap) core nodes we can further approach the optimal solution.

E. Run-Time Analysis

The simulations were performed on a computer running Debian Stretch Linux Version 9, with four 2.67 GHz Intel

Core2 Quad Processors with 12 GB RAM. To solve the ILPs, we used the Gurobi Optimizer version 6.0.4.

The running time of Theorem 2 and Algorithm 1 is dominated by the creation of the auxiliary graphs $\widehat{G}$ and $\widetilde{G}$, respectively. Hence, when just a few nodes are upgraded (the number of virtual edges between splitters and mergers is small) the computation time is around 80 ms in the 40 node random networks. However, if all the nodes are upgraded the time is about ten times higher, but always less than 1.5 s per demand.[11] In other words, the running time of Theorem 2 and Algorithm 1 is strongly influenced by the size of the network and the extent of the upgraded network nodes.

In the capacity constrained cases, the computation time of the ILPs is around 320 ms in the 40 node networks, and depends on the size of the network (increased number of variables and constraints), but it is independent of the number of bottleneck links. By incremental deployment, as we upgrade more nodes in the network the running time decreases slightly since fewer constraints (Eq. (6)-(7)) are needed.

VII. Conclusions

Generalized diversity coding is a novel, easily deployable routing scheme in transport networks which keeps the ultra-fast recovery and simplicity (both in computation and operation) of $1+1$. As a missing link of its practical implementation, we investigated the minimum cost survivable routing problem (SRDC), showed that a minimum cost subgraph can be computed in polynomial-time without capacity constraints on the edges (and in directed acyclic graphs), provided an approximation algorithm for partially upgraded networks, and proposed an integer linear program for the other scenarios. Our simulation results suggest that even with upgrading only a small set of network nodes, we can reduce the bandwidth cost of $1+1$ in most network scenarios and utilize up to three-four edges less per connection, which could lead to a significant capacity saving with an excessive number of connections. We argue that the novel method can provide a viable alternative for $1+1$ in transport network protection for the price of a minimal network upgrade.

References

[1] *Zayo Group-Communications Infrastructure Provider*. Accessed: Mar. 25, 2019. [Online]. Available: https://www.zayo.com/wavelengths/

[2] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in *Proc. IEEE INFOCOM*, vol. 4, Feb. 2005, pp. 2307–2317.

[3] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974.

[4] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[5] A. E. Kamal, "1+N network protection for mesh networks: Network coding–based protection using p-cycles," *IEEE/ACM Trans. Netw.*, vol. 18, no. 1, pp. 67–80, Feb. 2010.

[6] I. B. Barla, F. Rambach, D. A. Schupke, and G. Carle, "Efficient protection in single-domain networks using network coding," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–6.

[7] A. Das, C. U. Martel, and B. Mukherjee, "A partial-protection approach using multipath provisioning," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2009, pp. 1–5.

[11] If the auxiliary graph is already built, this time is a magnitude smaller.

[8] G. Kuperman, E. Modiano, and A. Narula-Tam, "Analysis and algorithms for partial protection in mesh networks," *J. Opt. Commun. Netw.*, vol. 6, no. 8, p. 730, Aug. 2014.

[9] K.-S. Sohn, S. Yeob Nam, and D. Sung, "A distributed LSP scheme to reduce spare bandwidth demand in MPLS networks," *IEEE Trans. Commun.*, vol. 54, no. 7, pp. 1277–1288, Jul. 2006.

[10] P.-H. Luo, J. Tapolcai, and T. Cinkler, "Comments on, "Segment shared protection in mesh communications networks with bandwidth guaranteed tunnels," *IEEE/ACM Trans. Netw.*, vol. 15, no. 6, p. 1616, Dec. 2007.

[11] A. Kodian and W. Grover, "Failure-independent path-protecting p-cycles: Efficient and simple fully preconnected optical-path protection," *J. Lightw. Technol.*, vol. 23, no. 10, pp. 3241–3259, Oct. 2005.

[12] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[13] S. Jaggi *et al.*, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[14] T. Ho *et al.*, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[15] E. Ayanoglu, C.-L. I, R. Gitlin, and J. Mazo, "Diversity coding for transparent self-healing and fault-tolerant communication networks," *IEEE Trans. Commun.*, vol. 41, no. 11, pp. 1677–1686, Nov. 1993.

[16] H. Øverby, G. Biczók, P. Babarczi, and J. Tapolcai, "Cost comparison of 1+1 path protection schemes: A case for coding," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 3067–3072.

[17] S. Rouayheb, A. Sprintson, and C. Georghiades, "Robust network codes for unicast connections: A case study," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 644–656, Jun. 2011.

[18] P. Babarczi, J. Tapolcai, A. Pasic, L. Ronyai, E. R. Berczi-Kovacs, and M. Medard, "Diversity coding in two–connected networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2308–2319, Aug. 2017.

[19] A. Pasic, J. Tapolcai, P. Babarczi, E. R. Berczi-Kovacs, Z. Kiraly, and L. Ronyai, "Survivable routing meets diversity coding," in *Proc. IFIP Netw. Conf. (IFIP Netw.)*, May 2015, pp. 1–9.

[20] A. Pašić, P. Babarczi, and A. Kőrösi, "Diversity coding-based survivable routing with QoS and differential delay bounds," *Opt. Switching Netw.*, vol. 23, pp. 118–128, Jan. 2017.

[21] G. Ellinas, E. Bouillet, R. Ramamurthy, J. Labourdette, S. Chaudhuri, and K. Bala, "Routing and restoration architectures in mesh optical networks," *Opt. Netw. Mag.*, vol. 4, no. 1, pp. 91–106, 2003.

[22] P. Babarczi, A. Pašić, J. Tapolcai, F. Németh, and B. Ladóczki, "Instantaneous recovery of unicast connections in transport networks: Routing versus coding," *Comput. Netw.*, vol. 82, pp. 68–80, May 2015.

[23] G. Brightwell, G. Oriolo, and F. B. Shepherd, "Reserving resilient capacity in a network," *SIAM J. Discrete Math.*, vol. 14, no. 4, pp. 524–539, Jan. 2001.

[24] F. Németh, A. Stipkovits, B. Sonkoly, and A. Gulyás, "Towards smartflow: Case studies on enhanced programmable forwarding in Openflow switches," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 85–86, 2012.

[25] J. Yang, B. Dai, L. Lv, and G. Xu, "Coding openflow: Enable network coding in SDN networks," *Int. J. Comput. Netw. Commun.*, vol. 7, no. 5, pp. 29–38, Sep. 2015.

[26] F. Gabriel, G. T. Nguyen, R.-S. Schmoll, J. A. Cabrera, M. Muehleisen, and F. H. Fitzek, "Practical deployment of network coding for real-time applications in 5G networks," in *Proc. 15th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2018, pp. 1–2.

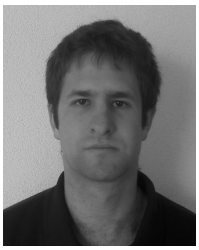
[27] T. Biermann, A. Schwabe, and H. Karl, "Creating butterflies in the core—A network coding extension for MPLS/RSVP-TE," in *NETWORKING* (Lecture Notes in Computer Science), vol. 5550, L. Fratta, H. Schulzrinne, Y. Takahashi, and O. Spaniol, Eds. Berlin, Germany: Springer, 2009.

[28] B. Ladoczki, C. Fernandez, O. Moya, P. Babarczi, J. Tapolcai, and D. Guija, "Robust network coding in transport networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2015, pp. 1–2.

[29] J. W. Suurballe and R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, no. 2, pp. 325–336, 1984.

[30] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0—Survivable network design library," in *Proc. INOC*, 2007.

[31] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with rocketfuel," *SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 4, p. 133, Oct. 2002.



Alija Pašić received the M.Sc. *(summa cum laude)* and Ph.D. *(summa cum laude)* degrees in electrical engineering from the Budapest University of Technology and Economics (BME), Hungary, in 2013 and 2019, respectively. He is currently an Assistant Professor with the High-Speed Networks Laboratory, Department of Telecommunications and Media Informatics, BME. His research interests include survivability in optical backbone networks, network coding, and artificial intelligence (AI).

Péter Babarczi (Member, IEEE) received the M.Sc. and Ph.D. *(summa cum laude)* degrees in computer science from the Budapest University of Technology and Economics (BME), Hungary, in 2008 and 2012, respectively. Since 2017, he has been an Alexander von Humboldt Post-Doctoral Research Fellow with the Chair of Communication Networks at the Technical University of Munich, Germany. He is currently an Assistant Professor with the Department of Telecommunications and Media Informatics, BME. His current research interests include multipath Internet routing, network coding in transport networks, and combinatorial optimization in softwarized networks. He received the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and the Post-Doctoral Research Fellowship of the Alexander von Humboldt Foundation.

János Tapolcai received the M.Sc. degree in technical informatics and the Ph.D. degree in computer science from the Budapest University of Technology and Economics (BME), Budapest, Hungary, in 2000 and 2005, respectively, and the D.Sc. degree in engineering science from the Hungarian Academy of Sciences (MTA) in 2013. He is currently a Full Professor with the High-Speed Networks Laboratory, Department of Telecommunications and Media Informatics, BME. He has authored over 150 scientific publications. His current research interests include applied mathematics, combinatorial optimization, optical networks and IP routing, addressing, and survivability. He is a Winner of the MTA Lendület Program and the Google Faculty Award in 2012 and the Microsoft Azure Research Award in 2018. He is a TPC member of leading conferences such as the IEEE INFOCOM from 2012 to 2017. He is also the General Chair of ACM SIGCOMM 2018. He was a recipient of several best paper awards, including ICC 2006, DRCN 2011, HPSR 2015, and NaNa 2016.

Erika R. Bérczi-Kovács received the M.Sc. degree in mathematics and the Ph.D. degree in applied mathematics from Eötvös Loránd University (ELTE), Budapest, Hungary, in 2007 and 2015, respectively. She is currently an Assistant Professor with the Department of Operations Research, ELTE. Her research interests include discrete mathematics, combinatorial optimization, and network coding. She is a member of the MTA-ELTE Egerváry Research Group on Combinatorial Optimization. She was a recipient of the NaNA 2016 Best Paper Award.

Zoltán Király received the M.Sc. and Ph.D. degrees in mathematics from Eötvös Loránd University (ELTE), Budapest, Hungary, in 1987 and 1997, respectively. He is currently an Associate Professor with the Department of Computer Science, Eötvös Loránd University. He is also a member of the MTA-ELTE Egerváry Research Group on Combinatorial Optimization. His research interests include graph theory, theory of algorithms, and combinatorial optimization. He received the Best Paper Award from ESA 2008.

Lajos Rónyai received the Ph.D. degree from Eötvös Loránd University, Budapest, Hungary, in 1987. He is a Research Professor with the Informatics Laboratory, Institute for Computer Science and Control. He leads a research group there which focuses on theoretical computer science and discrete mathematics. He is also a Full Professor with the Mathematics Institute, Budapest University of Technology and Economics. His research interests include efficient algorithms, complexity of computation, algebra, and discrete mathematics.