Survivable Routing Meets Diversity Coding

Alija Pašić*, János Tapolcai*, Péter Babarczi*, Erika R. Bérczi-Kovács†, Zoltán Király‡, Lajos Rónyai§
*MTA-BME Future Internet Research Group, Budapest University of Technology and Economics (BME)
†Department of Operations Research, Eötvös University, Budapest, Hungary
‡Department of Computer Science, Eötvös University, Budapest, Hungary and MTA-ELTE Egerváry Research Group
§Computer and Automation Research Institute Hungarian Academy of Sciences and BME
*{pasic, tapolcai, babarczi}@tmit.bme.hu, †koverika@cs.elte.hu, ‡kiraly@cs.elte.hu, §ronyai@sztaki.hu

***Abstract*—Survivable routing methods have been thoroughly investigated in the past decades in transport networks. However, the proposed approaches suffered either from slow recovery time, poor bandwidth utilization, high computational or operational complexity, and could not really provide an alternative to the widely deployed single edge failure resilient dedicated $1+1$ protection approach. Diversity coding is a candidate to overcome these difficulties with a relatively simple technique: dividing the connection data into two parts, and adding some redundancy at the source node. However, a missing link to make diversity coding a real alternative to $1+1$ in transport networks is finding its minimum cost survivable routing, even in sparse topologies, where previous approaches may fail. In this paper we propose a polynomial-time algorithm with $O(|V||E|\log|V|)$ complexity for this routing problem. On the other hand, we show that the same routing problem turns to be NP-hard as soon as we limit the forwarding capabilities of some nodes and the capacities of some links of the network.**

***Index Terms*—survivable routing, diversity coding, fast recovery, transport networks**

I. Introduction

A survivable routing scheme in transport networks has three utmost important requirements: *low recovery time, simplicity (i.e., low computational and operational complexity) and efficient capacity allocation*. But which one is the most important for service providers, and what are they willing to sacrifice in order to reach that? To answer this, we have to look what is used in practice. In networks, the most commonly used survivable routing scheme is the so called *dedicated* $1+1$ *path protection*, which sends the user data along two disjoint paths (primary and backup). Although it consumes twice as much capacity as the primary capacity, there are efficient algorithms to calculate $1+1$ routing (i.e., disjoint path-pair [1]), while it provides instantaneous recovery from any single edge failure.

Although $1+1$ is still the most commonly used protection scheme, in bandwidth utilization there is still room for improvement. On the other hand, based on the previous observation, simplicity and ultra fast recovery time seem to be the most important features, and efficient capacity allocation comes only after them. Several survivable routing schemes were introduced [2], [3] in the past decades which could significantly reduce the bandwidth utilization [3], [4], but they sacrifice either *the ultra fast recovery time, the low computational complexity, or – most importantly – the simple operation*. Following this argument we show that with a careful design we can keep these merits while near optimal capacity allocation can be reached.

ISBN 978-3-901882-68-5

In transport networks, optimal capacity efficiency can be achieved with in-network modification of data (called *network coding*) [4]–[6] while *low recovery time (i.e.,* $<$ 40-50 *ms) is maintained*. However, the application of complex network coding operations sacrifices simplicity. On the other hand, there are some simple special cases of coding which could satisfy all three requirements in transport networks, e.g., *diversity coding* (DC) [7], which splits the data at the source node into two parts A and B, and creates redundancy data $A \oplus B$, too ($\oplus$ denotes the exclusive OR (XOR) operation on the data), then sends these on three edge-disjoint paths. Diversity coding can reduce the capacity consumption of $1+1$ (from 2 to 1.5 unit), while its complexity and recovery time are the same. On the other hand, this method is applicable only in 3 edge-connected networks which is a crucial drawback.

Recently, the papers [8] and [9] made important steps to remedy the connectivity problem of diversity coding, and enable it to be an alternative of $1+1$ in transport network protection. They formulate the survivable routing problem of $1+1$ and diversity coding more generally, while considering the same routing scenario: single edge failure resilience [10], while the connection data is divided into two parts at the source node. Although they provide polynomial-time network (and diversity) coding algorithms in a minimum cost survivable routing (called coding subgraph), the issue of finding a minimum cost survivable routing is not addressed in these works. However, this would be the last step towards the practical implementation of a capacity efficient, simple and low-complexity diversity coding based survivable routing method with ultra-fast recovery time as a counterpart of $1+1$ protection; which will be made in this paper.

Although the general versions of minimum cost survivable routing have shown to be NP-complete [11], [12], surprisingly, the computational complexity of optimal capacity allocation for this practically relevant special case is an open question. However, using the state-of-the-art results [9] it is known that a minimum cost survivable routing with diversity coding can be decomposed into three directed acyclic graphs (DAG), *which preserves one of the most important features of* $1+1$*, i.e., simplicity*. Thus, in order to solve the optimal capacity allocation of diversity coding, we formulate the minimum cost survivable routing problem as finding three appropriate DAGs.

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 with source node s, target node t with bandwidth d unit demand
$R = (V^R, E^R, f)$	survivable routing of a connection with node set $V^R \subseteq V$, edge set $E^R \subseteq E$, and flow values $\forall e \in E^R : f(e) \leq k(e)$
$G^* = (V, E^*, c)$	directed multi-graph with edge set E^*, where all edge 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 $A \oplus B$

We will demonstrate that with the help of this formulation the problem is polynomial time solvable, which result makes diversity coding an alternative to $1 + 1$ in 2 edge-connected networks as well.

The rest of the paper is organized as follows. In Section II we formulate the minimum cost survivable routing problem with diversity coding (SRDC), and reveal some structural properties of the three DAGs, which will be used in the algorithms. As the main contribution of the paper, in Section III a polynomial-time algorithm with $O(|V||E| \log |V|)$ complexity is presented for the SRDC problem. In Section IV we show that the SRDC problem is NP-hard with some bottleneck links and with limited node capabilities. Finally, in Section V we show some simulation results, which reveals some network scenarios where SRDC can be a real alternative of $1 + 1$ protection, and in Section VI we conclude the paper.

II. PROBLEM FORMULATION AND RELATED WORK

A. Survivable Routing

A transport network is a collection of routers, switches (referred to as nodes) and high bandwidth communication links (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 $D = (s, t, d)$, with information source $s \in V$, with information sink $t \in V$, and the number of bandwidth units d requested for data transmission.

Definition 1. *We say that $R = (V^R, E^R, f)$ is a **survivable routing** of a connection $D = (s, t, d)$ in G (where $V^R \subseteq V$, $E^R \subseteq E$, and $\forall e \in E^R : f(e) \leq k(e)$), if there is an $s - t$ flow of value $F \geq d$ in R, even if we delete any single edge of R. On the other hand, a routing is **vulnerable** if it is not survivable.*

Note that, after the edge failure is identified any routing method could be adopted and resend the flows on the intact edges of a survivable routing R, clearly resulting in slow recovery. However, several network coding theorems [4], [5], [13] ensure that a sufficient amount of information reaches the destination after a failure occurred without failure identification; but for the price of complex in-network operations. We will demonstrate that we can bring together these merits (i.e., fast recovery and simplicity) of survivable routing approaches with the help of current research results on network coding [8] and on diversity coding [9]. This is discussed in Section II-B.

We say that a **survivable routing R is critical**, if we can not further decrease the flow value $f(e)$ along any edge in $e \in E^R$ without making routing R vulnerable. Our goal is to find a survivable routing R for connection D with minimum *bandwidth cost* from the possible set of survivable routings $\mathcal{R}^D$, formally:

$$\min_{R \in \mathcal{R}^D} \sum_{e \in E^R} c(e) \cdot f(e). \tag{1}$$

Claim 1. *The minimum cost survivable routing in terms of Eq. (1) is critical.*

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, or finding the optimal survivable routing for multiple edge failures are NP-complete problems [11], [12], [14]. However, in current transport networks single edge failures are the most relevant failure scenarios [10], while dividing user data into more than two parts is impractical from an operational point of view. Surprisingly, the complexity of this practically relevant special case of single edge failure minimum cost survivable routing when $d = 2$ is an open question.

Thus, in this paper we will investigate the survivable routing problem when $d = 2$, i.e., the connection can be routed as two parts of equal size, denoted by A and B (for the sake of simplicity, the problem can be scaled to have both with rate 1). This restriction is motivated by the fact that the minimum cost routing solution in most real word networks can be reached by dividing the input flow into 2 subflows [2]. Furthermore, assuming two data parts a survivable routing solution preserves the simplicity of $1 + 1$ protection.

For the sake of easier presentation of our results, the auxiliary graph $G^* = (V, E^*, c)$ is introduced. 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 which have the same tail and head node as e, each with cost $c(e)$. *Note that single edge failure e in G corresponds to the failure of all $k(e)$ edges in G^*.* A critical survivable routing $R^* = (V^{R^*}, E^{R^*})$ forms a directed acyclic graph (DAG) in G^* (discussed in Section II-B), representing 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:

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

The notation is summarized in Table I.

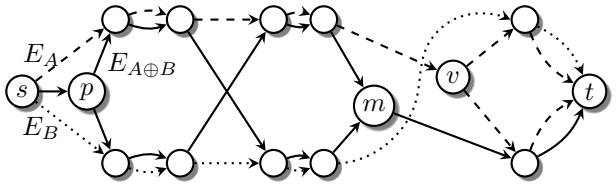


Fig. 1. A survivable routing $R^* = (V^{R^*}, E^{R^*})$ for connection $D = (s, t, 2)$ with the corresponding routing DAGs E_A, E_B and $E_{A\oplus B}$.

B. Diversity Coding

In survivable routing besides the theoretically good properties like low bandwidth utilization and fast recovery, from a practical point of view simplicity and easy deployment are essential as well. Thus, complex data processing at core nodes (i.e., other than s and t) like network coding is not desired. Thus, all complex operations have to be moved to the edge of the network (i.e., to nodes s and t). We will refer to the survivable routing approaches which satisfy this property as diversity coding based methods, or simply *diversity coding*. Luckily, for single edge failures and two data parts, the range of survivable routings providing this simplicity is quite wide:

Theorem 1. *[9, Theorem 2] Suppose that survivable routing R is critical. Then there are disjoint edge sets $E_A, E_B, E_{A\oplus B}$ of R^*, called **routing DAGs**, such that for an arbitrary edge $e \in E^R$, after removing the corresponding edge(s) from E^{R^*} at least two of the routing DAGs connect s to t.*

For example, in "traditional" diversity coding, for a connection $D = (s, t, 2)$ the redundancy data $A \oplus B$ is calculated at the source s, and A, B and $A \oplus B$ is sent along three disjoint end-to-end paths between s and t. The edge sets used by the disjoint paths are denoted as $E_A, E_B, E_{A\oplus B}$, respectively. Also $1 + 1$ protection could be treated as an implementation of diversity coding: A and B are sent along two disjoint paths E_A and E_B, while the redundancy data $A \oplus B$ is sent along both paths. Note that both of these routings are survivable.

We will refer to routings satisfying Theorem 1 as **Survivable Routing with Diversity Coding (SRDC)** throughout this paper. Note that with the help of diversity coding the three routing DAGs can be operated independently from each other ($E_A \cup E_B \cup E_{A\oplus B} = E^{R^*}$, $E_A \cap E_B = \emptyset$, $E_A \cap E_{A\oplus B} = \emptyset$, $E_B \cap E_{A\oplus B} = \emptyset$), i.e., they carry the same data part regardless of the failure, while involved operations are performed only at the end nodes of the connection. However, this general implementation of diversity coding requires splitting and merging of the paths at the core nodes. Let $\delta^-(v)$ and $\delta^+(v)$ denote the in-degree and the out-degree of a node, respectively.

Definition 2. *Node $p \in V^{R^*}$ is called **splitter**, if $\delta^-(p) = 1$ and $\delta^+(p) = 2$, i.e., it receives data on a single edge, while forwards the same copy on two outgoing edges. Similarly, node $m \in V^{R^*}$ is called **merger**, if $\delta^-(m) = 2$ and $\delta^+(m) = 1$, i.e., it receives* the same data *on two incoming edges, while forwards one of them (or upon failure the intact one) on its single outgoing edge. The set of available splitters and mergers are denoted as $\mathcal{P} \subseteq V$ and $\mathcal{M} \subseteq V$, respectively.*

We believe that splitting and merging operations are simple enough in the sense that every node in the network can perform them without any complicated software update. Furthermore, in current networking paradigm, such as Software Defined Networking (SDN), a splitter can be easily deployed by applying simple flow rules, while a merger functionality can be implemented as a simple network function as well [14].

In [8], [9] it was shown that a critical survivable routing has a well defined structure (created by a maximal series of $s-t$ cuts or by a block decomposition). In a critical solution, we have an alternating series of splitter and merger nodes (if there are any at all). An example of a survivable routing with diversity coding is presented in Figure 1. For routing DAG $E_{A\oplus B}$ node p is a splitter and node m is a merger, while for routing DAG E_A node v and node t act as a splitter and merger, respectively.

Definition 3. *Nodes p and m are splitting and merging node-pair (**or** pm**-pair** shortly) of a survivable routing R^* (given with its routing DAGs $E_A, E_B, E_{A\oplus B}$), if there exists a routing DAG, without loss of generality, E_A, which splits at node p and merges back at m. The edge set of the corresponding disjoint path-pair between p and m in the survivable routing is denoted by $\mathcal{E}^{R^*}_{p,m}$, and referred to as an **island** in DAG E_A.*

Note that, there always exists an edge disjoint pair of paths between a splitter and its corresponding merger in a critical survivable routing R^* [8], [9].

Claim 2. *Each routing DAG in a critical solution consists of a series of paths and islands from s to t. Furthermore, a pm-pair could be part of at most one routing DAG [8], [9].*

For example, in "traditional" diversity coding all of $E_A, E_B, E_{A\oplus B}$ are $s \to t$ paths. In Figure 1 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 $m \to t$. The algorithms to find the optimal routing DAGs with their computational complexity are shown in Table II. Note that, diversity coding corresponds to the case when only the source and destination node can be a pm-pair, which can be solved by Suurballe's algorithm[1]. We are interested in the general cases as well, when more pm-pairs exists till the other extreme, i.e., all node-pairs can act as a pm-pair. In the rest of the paper, we introduce these capacity allocation approaches, and conduct simulations to demonstrate their benefits.

III. Survivable Routing with Infinite Capacities

In this section we show that the minimum cost survivable routing problem with diversity coding is solvable in polynomial time, if the edge capacities $k(e)$ are "infinite", i.e., there are no bottleneck links in the network. Note that for the demand $D = (s, t, 2)$ in survivable routing with diversity coding (SRDC) we are searching three routing DAGs, each

[1]Note that the augmenting path technique of Suurballe's algorithm can be used to find 3 edge-disjoint paths.

TABLE II
OPTIMAL SURVIVABLE ROUTING ALGORITHMS PROPOSED IN THIS PAPER (INFINITE: $\forall e \in E : k(e) = 2$, CONSTRAINED: $\exists e \in E : k(e) = 1$)

	Infinite Capacities	Capacity Constrained
$\mathcal{P} = \{s\}$, $\mathcal{M} = \{t\}$	Suurballe $O(\|V\| \log_2 \|V\| + \|E\|)$	Suurballe $O(\|V\| \log_2 \|V\| + \|E\|)$
$\mathcal{P} \subset V$, $\mathcal{M} \subset V$	-	- NP-complete
$\mathcal{P} = V$, $\mathcal{M} = V$	SRDC-I $O(\|V\|\|E\| \log_{1+\|E\|/\|V\|} \|V\|)$	Integer Linear Program -

forwarding one unit of capacity (either A, B or $A \oplus B$). Thus, in a critical survivable routing one would think that infinite edge capacity means $\forall e \in E : k(e) = 3$, i.e., the case when all DAGs may use the same edge. However, it was shown in [9] (as a consequence of Theorem 1) that with the application of diversity coding (called resilient flow decomposition) in a *critical survivable routing* for a connection with $d = 2$ the flow values are $\forall e \in E^R : f(e) \leq 2$. Thus, without loss of generality, we can restrict the available capacities to $k(e) = 2$ for every edge e, without ruling out the minimum cost survivable routing (which is critical, obviously). Hence, in SRDC infinite edge capacities mean $\forall e \in E : k(e) = 2$ (instead of 3). Further note that, in this case the auxiliary graph G^* has $2|E|$ edges, as each edge of G is duplicated.

Claim 3. *Let R^* be a critical survivable routing, decomposed into 3 routing DAGs E_A, E_B, and $E_{A\oplus B}$ according to Theorem 1 and let $\mathcal{E}^{R^*}_{p,m}$ be an island for a given pm-pair in E_A. Let $\mathcal{E}^{G}_{p,m}$ denote an arbitrary edge-disjoint path-pair between nodes p and m in G, with the corresponding edges $\mathcal{E}^{G^*}_{p,m}$ in G^*. If the network has infinite edge capacities, the routing $R' = R^* \setminus \mathcal{E}^{R^*}_{p,m} \cup \mathcal{E}^{G^*}_{p,m}$ is also survivable.*

Proof: Note that in $R' = R^* \setminus \mathcal{E}^{R^*}_{p,m} \cup \mathcal{E}^{G^*}_{p,m}$ the routing DAG E_A has island $\mathcal{E}^{G^*}_{p,m}$ instead of island $\mathcal{E}^{R^*}_{p,m}$. In order to demonstrate that R' is survivable against single edge failures of G, we have to show that at least two routing DAGs connect s to t upon these failures. For an edge failure not in $\mathcal{E}^{G^*}_{p,m}$, DAGs that remained connected in R^* will be also connected in R'. If the failed edge $e \in \mathcal{E}^{G^*}_{p,m}$, then we know that in R^* there were two routing DAGs which connected s to t. The only DAG that changed is E_A, but note that it also remains connected, because the failure is in an island. This proves the claim. ∎

Corollary 1. *Let R^* be a minimum cost survivable routing and $\mathcal{E}^{R^*}_{p,m}$ an island for a given pm-pair. If the network has infinite edge capacities, then $\mathcal{E}^{R^*}_{p,m}$ is a minimum cost disjoint path-pair between nodes p and m in G.*

Such a pair can be calculated with Suurballe's algorithm in $O(|E| + |V| \log_2 |V|)$ steps [1]. 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 4. *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 st-paths.*

Now, we are ready to present the main result of the paper.

Theorem 2. *If the network has infinite edge capacities on all edges, the minimum cost survivable routing R^* can be computed in $O(|V||E| \log_{1+|E|/|V|} |V|)$ steps.*

Proof: Let $\mathrm{cost}(u, v)$ denote the sum of the edge costs of a minimum cost disjoint path-pair between nodes u and v in G. 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 the edges of $\widehat{G}$ are the edges of G with cost $\widehat{c}(e) = c(e)$ and we add an edge $e_n = (u, v)$ for every pair of distinct node-pairs with cost $\widehat{c}(e_n) = \mathrm{cost}(u, v)$. We refer to the newly added edges as *virtual edges*.

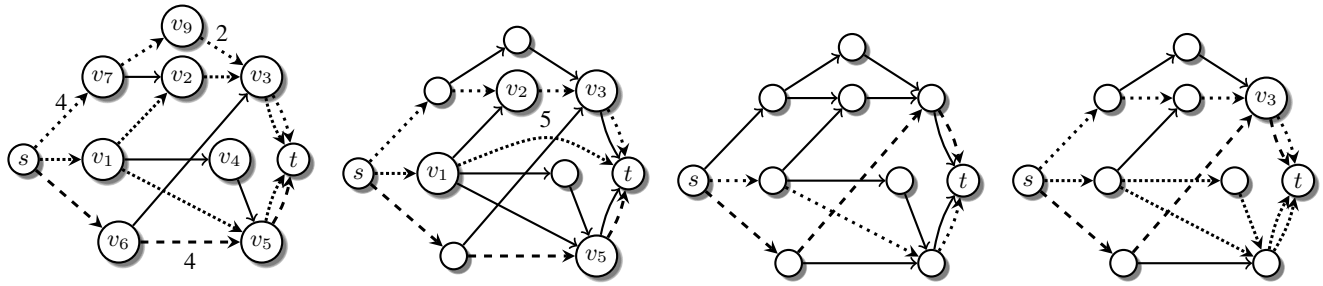
Claim 5. *Let $P_A, P_B, P_{A\oplus B}$ be three edge-disjoint st-paths in $\widehat{G}$. By replacing every virtual edge (p, m) with an island $\mathcal{E}^{G^*}_{p,m}$ of minimum cost we get DAGs E_A, E_B, and $E_{A\oplus B}$ in G^* that form a survivable routing. Moreover, the cost of these DAGs in G^* equals the cost of the paths in $\widehat{G}$, and vice versa.*

Proof: Equality of costs is straightforward. Since $P_A, P_B, P_{A\oplus B}$ 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 the failure of e, the latter ones remain connected, so at most one of the DAGs can be disconnected, which proves the claim. ∎

Since a minimum cost survivable routing is critical, from Claim 2 and Claim 4 we get that it corresponds to three edge-disjoint st-paths in $\widehat{G}$. In our algorithm, we search for 3 edge-disjoint paths with minimum bandwidth cost in $\widehat{G}$, which in turn corresponds to three DAGs in G^*. According to Claim 5, the obtained routing is survivable, and the cost of the minimum cost 3 edge-disjoint paths equals to the cost of the minimum cost survivable routing R^*. Finding minimum cost 3 edge-disjoint paths could be done in $O(|E|)$.

In the construction, finding the pair of shortest edge-disjoint path from a single source to every destination is $O(|E| \log_{1+|E|/|V|} |V|)$ time [1], which should be launched for every source node, resulting $O(|V||E| \log_{1+|E|/|V|} |V|)$ steps, which proves the theorem. ∎

The proof is constructive, and gives the polynomial-time algorithm to find an optimal survivable routing, detailed in Algorithm 1. A natural question is why the algorithm 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 2(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 $\mathrm{cost}(v_1, t)$ is



(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) A possible vulnerable routing, where the failure of (v_3, t) disrupts two routing DAGs.

Fig. 2. 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 fails for connection $D = (s, t, 2)$.

Algorithm 1: Survivable Routing with Diversity Coding - Infinite Capacities (SRDC-I)

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 $\widehat{E} = \emptyset$, $E_s = \emptyset$;
  `// Create graph` $\widehat{G} = (V, \widehat{E}, \widehat{c})$.
  Add $\forall e \in E$ to $\widehat{E}$ with $\widehat{c}(e) = c(e)$;
  for $u \in V$: **do**
    Find the pair of shortest edge-disjoint path from source u to all other nodes $v \in V, u \neq v$ in G with Suurballe's algorithm (denote their cost with $\text{cost}(u, v)$);
    Add virtual edge $e_n = (u, v)$ to $\widehat{E}$ with $\widehat{c}(e_n) = \text{cost}(u, v)$;
  `// Find 3 edge-disjoint paths in` $\widehat{G}$.
  Find minimum cost 3 edge-disjoint paths between s and t in $\widehat{G}$ 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^*};

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 2(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.

Another direction is 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 Algorithm 1 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 2(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 2(d) with cost 16. Thus, Algorithm 1 is not applicable to the capacity constrained case.

IV. Survivable Routing with Capacity Constraints

In this section, we will investigate the capacity constrained scenario, i.e., when there are some bottleneck links with $k(e) = 1$. Although a polynomial time algorithm exists for the infinite capacity case (Section III), the survivable routing with diversity coding (SRDC) problem is more complex in this scenario. In Section IV-A we present an Integer Linear Program (ILP) for finding the three routing DAGs for SRDC, while Section IV-B presents our efficient heuristic solution for the problem when all nodes can act as a pm-pair. Furthermore, we show that if the splitter and merger nodes are restricted in the topology from e.g., technological considerations, the SRDC problem turns to be NP-hard (Section IV-C).

A. Integer Linear Program for Minimum Cost SRDC

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.

To do so, we need to to introduce the so called *reduced capacity function* [8]: $k'(e) = \begin{cases} 1.5 & \text{if } k(e) \geq 2 \\ 1 & \text{if } k(e) = 1. \\ 0 & \text{otherwise} \end{cases}$

Theorem 3. *In [8, Theorem 2] the authors show that 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 $G = (V, E, k', c)$.*

Theorem 3 will be used both in our ILP formulation and in the heuristic approach described in Section IV-B. Note that, given a routing DAG E_A in a critical survivable routing, a function 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 Claim 2.

Armed with this fact, we investigate the benefits which diversity coding can provide for survivable routing. Our goal is to obtain the (critical) flow values $f(e)$ in the 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 $f^w(e)$. Based on Theorem 3 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:

$$\forall w \in \mathcal{W}, \forall i \in V\text{:}$$

$$\sum_{(i,j)\in E} x^w(i,j) - \sum_{(j,i)\in E} x^w(j,i) = \begin{cases} 1 & \text{, if } i = s \\ -1 & \text{, if } i = t \\ 0 & \text{, otherwise} \end{cases}, \tag{3}$$

$$\forall e \in E\text{: } \sum_{w \in \mathcal{W}} x^w(e) \leq k'(e), \tag{4}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{: } x^w(e) \leq f^w(e), \tag{5}$$

$$\forall e \in E\text{: } \sum_{w \in \mathcal{W}} f^w(e) \leq f(e) \leq k(e), \tag{6}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{: } 0 \leq x^w(e) \leq 1, \tag{7}$$

$$\forall w \in \mathcal{W},\ \forall e \in E\text{: } 0 \leq f^w(e) \text{ are integer variables.} \tag{8}$$

The constraint in Equation (3) formulates the flow conservation for each routing DAG w. Constraint (4) sets the maximal flow value based on the reduced capacity function, while Constraint (5) sets the indicator variables $f^w(e)$ of edge usage for the routing DAGs in G. Constraint (6) sets the flow 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 (7)-(8) sets the bounds for the flow variables, and sets the integer constraint for the indicator variables $f^w(e)$. Note that the $f^w(e)$ variables correspond to the edge set w in the solution, i.e., provide the three end-to-end DAGs. Since $x^A + x^B + x^{A\oplus B}$ gives an $s-t$ flow of value 3 in G, from Theorem 3 we get that f is indeed survivable.

B. Heuristic Approach for Finding the Optimal Coding Graph

As the ILP in Section IV-A is NP-hard, the running time of its solutions can be really long, especially in large networks. In this section, we present a fast heuristic approach for finding a survivable routing in the capacity constrained case.

Algorithm 2: Survivable Routing with Diversity Coding - Capacity Constrained (SRDC-C)

Input: $G = (V, E, k, c)$, $D = (s, t, 2)$, α
Result: $R = (V^R, E^R, f)$

1 **begin**
2 Define capacity $k', k'' : E \to \mathbb{R}^+$, cost $c' : E \to \mathbb{R}^+$ and edge sets $E_n = \emptyset$; $E' = \emptyset$; $E_s = \emptyset$;
// Create graph $G' = (V, E', k', c')$.
3 Add $e \in E$ to E' with $k'(e) = \min\{1, k(e)\}$ and $c'(e) = c(e)$;
4 **for** $e = (u, v) \in E : 2 \leq k(e)$ **do**
5 Add extra edge $e_n = (u, v)$ to E_n with $k'(e_n) := 0.5$ and $c'(e_n) := c(e) \cdot \alpha$;
6 $E' := E' \cup E_n$;
// Phase 1: Find flow in $G' = (V, E', k', c')$.
7 Find a minimum cost $s-t$ flow f' of value 3 in G' with respect to the reduced capacity function k';
8 **for** $e = (u, v) \in E$ **do**
9 Set $k''(e) := 0$;
10 Set $e_n = (u, v) \in E_n : k''(e_n) := 0$;
11 **if** $0 < f'(e)$ **then**
12 **if** $e_n = (u, v) \in E_n : 0 < f'(e_n)$ **then**
13 Add edge e to E_s and set $k''(e) := 1$;
14 **for** $e^1 = (u^1, v^1), e^2 = (u^2, v^2) \in E_s, e^1 \neq e^2$ **do**
15 **if** *no pair of edge-disjoint $s-t$ paths in $G \setminus (e^1 \cup e^2)$* **then**
16 $e_n^1 = (u^1, v^1), e_n^2 = (u^2, v^2) \in E_n : k''(e_n^1) = 0.5$; $k''(e_n^2) = 0.5$;
// Phase 2: Find flow in $G'' = (V, E', k'', c')$.
17 Find a minimum cost $s-t$ flow f'' of value 3 in G'' with respect to the capacity function k'';
// Save survivable routing R.
18 **for** $e = (u, v) \in E : 0 < f''(e)$ **do**
19 Add node u, v to V^R (if $u, v \notin V^R$);
20 Add edge e to E^R;
21 Set flow value $f(e)$ to 2 if $e_n = (u, v) \in E_n : 0 < f''(e_n)$, and to 1 otherwise;

The detailed description is given in Algorithm 2. The input of the algorithm is a graph $G = (V, E, k, c)$ and the connection request $D = (s, t, 2)$. The output is the survivable routing $R = (V^R, E^R, f)$. In Steps (4)-(6) the auxiliary graph $G' = (V, E', k', c')$ is created using the reduced capacities ($k'(e) = \min\{1, k(e)\}$). In the transformation, the $e \in E$ edges of G are added to E' with their original cost $c(e)$. Based on its capacity $k(e)$ of $e \in E$, we add a parallel edge $e_n = (u, v)$ (called *extra edge*) to E' if $2 \leq k(e)$ with reduced capacity $k'(e_n) = 0.5$ and with cost $c'(e_n) = c(e) \cdot \alpha$.

The main idea of the algorithm is to find a minimum

cost $s-t$ flow with value of 3 in the reduced capacity graph $G' = (V, E', k', c')$ in Steps (7)-(13)[2], in such way that "creating islands" (i.e., using the extra edge e_n with $k'(e_n) = 0.5$ capacity) is penalized via a scaling factor (i.e., the extra edges have a higher cost $c'(e_n) = c(e) \cdot \alpha, \alpha \geq 1$).

In order to avoid "false" island creation (resulting from an imprecise selection of α), based on the results $f'(e)$ of the first phase, we prepare the input of the second phase of Algorithm 2. First, we initialize the reduced capacity for the next stage $\forall e \in E : k''(e) = 0$. We create an edge set E_s, so that the original edge $e = (u, v)$ is in E_s, if both $e = (u, v)$ and the corresponding extra edge $e_n = (u, v)$ is used in the solution of the minimum cost flow, in other words if $0 < f'(e)$ and $e_n = (u, v) \in E_n : 0 < f'(e_n)$. Additionally, we set the capacity of all used original edges ($e = (u, v) \in E_s : 0 < f'(e)$) to $k''(e) = 1$. In Iteration (14)-(16) we check for every edge-pair $e^1 = (u^1, v^1), e^2 = (u^2, v^2) \in E_s, e^1 \neq e^2$, whether there are two edge-disjoint $s-t$ paths in $G \setminus (e^1 \cup e^2)$ or not. If there is no such flow, then we consider the corresponding extra edges as potential islands in the survivable routing, and we set their capacity values to $k''(e_n^1) = 0.5$ and $k''(e_n^2) = 0.5$, respectively. Note that, the cost $c'(e)$ remains the same.

In Phase 2 in Step (17), we search again for a minimum cost flow with value of 3 between s and t, now in graph $G'' = (V, E', k'', c')$. Finally, the flow values $f''(e)$ of the solution give the survivable routing R. Hence, in Steps (18)-(21) the survivable routing is saved in a way that if both $e = (u, v)$ and $e_n = (u, v)$ were used in the solution ($f''(e) > 0$ and $f''(e_n) > 0$), then we set $f(e) = 2$. If $f''(e) > 0$ and $f''(e_n) = 0$, then $f(e) = 1$, and else $f(e) = 0$, which gives $R = (V^R, E^R, f)$. The routing DAGs in R can be constructed as shown in [9].

Selection of Scaling Factor α: We have seen that choosing a proper α for a network is essential. For this purpose, intuitively we used the ratio $\alpha = \frac{|E|}{|V|}$, which is corresponding to the density (to be specific, it is half of the average nodal degree) of the network. In a denser network it is more likely to have 3 edge-disjoint paths with a relatively short third path, which might be the optimal routing DAGs. Therefore, the cost of the extra edge is relatively high (α is high) in order to avoid creation of islands. Meanwhile in sparse networks, it is more likely that the third path (if it exists at all) is really long, and not beneficial to use. Note that, if any of the edges $e \in E'$ is used ($f(e) > 0$), we have to reserve $f(e) = 1$ flow on its corresponding edge in G (despite the fact that we used only 0.5 in the reduced capacity graph G'), as the flow values of the routing DAGs are integers in the optimal survivable routing R. Note that, theoretically this could result in a higher capacity consumption than $1 + 1$ for some connections in special topologies. However, as it is shown in Section V, in real-like networks with such a carefully chosen α it performs much better than $1 + 1$.

[2]Note that based on [8, Theorem 2] this is equivalent with finding a survivable routing in $G = (V, E, k, c)$.

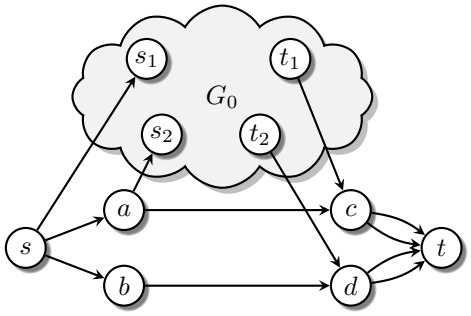


Fig. 3. The reduction of 2-DPP to the survivable routing problem (graph $G^* = (V, E^*, c)$ is shown).

C. Finding an SRDC in Directed Graphs with Restricted pm-Pairs is NP-Hard

Here, we show that if from technological considerations not all nodes can perform splitting and merging, the survivable routing problem turns to be NP-hard. In our proof we will use the well-known 2-Disjoint Path Problem (2-DPP), that has been proven to be NP-complete, both for the edge- and node-disjoint case by Fortune et al. [15].

Definition 4. *2-Disjoint Path Problem (2-DPP)*
Input: *Directed graph $G_0 = (V_0, E_0)$ and four distinct nodes $s_1, s_2, t_1, t_2 \in V_0$.*
Question: *Is there an edge-disjoint path-pair $P_1 : s_1 \rightsquigarrow t_1$ and $P_2 : s_2 \rightsquigarrow t_2$ in G_0?*

Theorem 4. *It is NP-complete to decide the existence of an SRDC in a directed graph $G = (V, E, k, c)$ if the candidate splitter ($\mathcal{P} \subset V$) and merger ($\mathcal{M} \subset V$) nodes are restricted to a given subset of the nodes.*

Proof: First, we give a polynomial-time reduction of the 2-DPP problem to our survivable routing problem. For this, we add six new nodes s, t, a, b, c, and d to G_0 and define the network $G = (V, E, k, c)$ as follows:

$$\begin{aligned} V &= V_0 \cup \{s, t, a, b, c, d\}, \\ E &= E_0 \cup \{(s, s_1), (s, a), (s, b), (a, s_2), \\ &\quad (a, c), (b, d), (t_1, c), (t_2, d), (c, t), (d, t)\}, \\ k(e) &= 2 \text{ if } e \in \{(c, t), (d, t)\}; \text{ and } 1 \text{ otherwise.} \end{aligned}$$

Besides the input graph, it is given, that $\mathcal{P} = \{a\}$ can only be a splitter node, and $\mathcal{M} = \{t\}$ can only be a merger. See also Figure 3 for the transformation of G_0 to $G^* = (V, E^*, c)$.

G^* was constructed in polynomial time. Therefore, it is sufficient to show that there exist two directed disjoint paths between nodes $s_1 \rightsquigarrow t_1$ and $s_2 \rightsquigarrow t_2$ if and only if there exist a survivable routing R^* for G^*.

$\Rightarrow$ If there exist two directed edge-disjoint paths between nodes $s_1 \rightsquigarrow t_1$ and $s_2 \rightsquigarrow t_2$, then the flowing three routing DAGs give a survivable routing:

- E_A: is the path $s \to b \to d \to t$,
- E_B: contains path $s \to s_1 \rightsquigarrow t_1 \to c \to t$, and
- $E_{A \oplus B}$: consists of path $s \to a$ and island $a \to t$ between pm-pair a and t with edge-disjoint paths $a \to s_2 \to t_2 \to d \to t$ and $a \to c \to t$.

$\Leftarrow$ The proof is by contradiction. Assume there is an SRDC in G^*, but there is no directed edge-disjoint path-pair between nodes $s_1 \rightsquigarrow t_1$ and $s_2 \rightsquigarrow t_2$. Note that (c,t) and (d,t) forms a 2 link cut in G, while (s,s_1), (s,a) and (s,b) is a 3 edge cut, thus there should be a splitter node between these cuts; however a is the only node that can be a splitter. Thus a is a splitter node and two paths are traversing through s_1 and s_2. As the routing is survivable, two paths have to go from s_1 and s_2 to $t_1$and t_2 in G^*. Because every edge in G_0 has $k(e) = 1$, the two paths can not use the same (multi-)edge in G^*; thus, these two paths are edge-disjoint in G_0. However, owing to the indirect assumption, these are not 2 edge-disjoint paths $s_1 \rightsquigarrow t_1$ and $s_2 \rightsquigarrow t_2$. Therefore, these edge-disjoint paths could be $s_1 \rightsquigarrow t_2$ and $s_2 \rightsquigarrow t_1$ only. In this case edge (c,t) is traversed by two copies of the same data, one through $a \to c \to t$ and an other through $a \to s_2 \rightsquigarrow t_1 \to c \to t$. Thus the routing is vulnerable of edge failure (d,t) because after deleting this edge there will be a cut of capacity 1 in the SRDC (consisting of edge (s,a)). ■

V. Experimental Results

In this section we investigate the bandwidth cost in terms of Eq. (1) of the different survivable routing approaches through simulations. We compare our methods to the theoretical lower bound on the bandwidth cost of survivable routing methods (assuming the connection data can be split into arbitrary many parts), called "lower bound" [14] on the charts. We also compare our algorithms to the bandwidth cost of the most common deployed survivable routing scheme, the $1+1$ protection, which is the 2-approximation of the survivable routing problem [12] with assuming two data parts[3]. In the charts SRDC-I refers to Algorithm 1, SRDC-C refers to Algorithm 2, and ILP refers to the Integer Linear Program presented in Section IV-A. We investigated random generated real-like planar $G = (V,E,k,c)$ topologies with different sizes and densities, and some real world transport network topologies, too. The edge capacities were set high enough to ensure "infinite" edge capacities for all demands. This is important in the fair comparison of the bandwidth cost of different methods as we can get rid of blocking because of lack of resources. Note that, *traffic demands* $\mathcal{D} = (s,t,2)$ *were generated between all* $s-t$ *pairs*, and all the arcs have unit cost ($\forall e \in E : c(e) = 1$).

A. Gap to the Theoretical Lower Bound

The simulation results in sparse networks are presented in Figure 4a. This is an excellent example 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 methods outperform $1+1$ even in this scenario, when $1+1$ performs well. In fact, the SRDC-I and the ILP reach the theoretical lower bound, as optimal survivable routing can be reached with dividing connection data into two parts.

[3]Note that the 2-approximation algorithm for feasible coding graphs presented in [8] gives $1+1$ for most practical scenarios.

Figure 4b shows the simulation results in maximal planar graphs. It can be observed that all of our methods approaching the theoretical lower bound. However, in these topologies the theoretical lower bound requires that connection data is divided into more than two data parts, which is not always feasible from a practical point of view. The gap between $1+1$ and our SRDC methods is significant, especially if we take into consideration the fact, that the bandwidth cost reduction of 1 unit means that *every SRDC connection in the network uses one less bandwidth unit than* $1+1$.

B. Scalability in Terms of Network Size

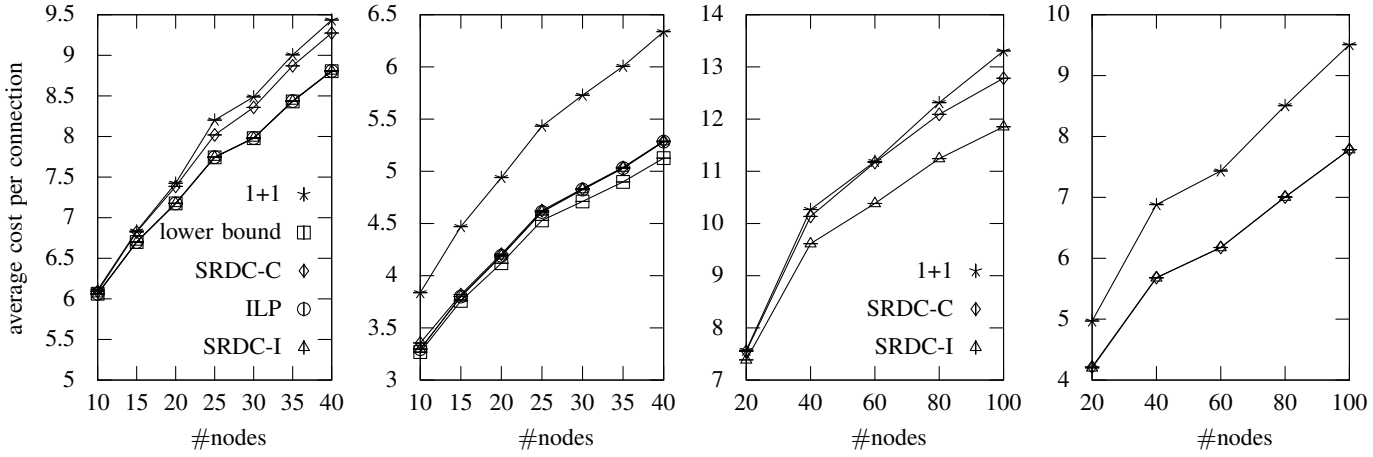
In Figure 4c and in Figure 4d, we investigated the performance of SRDC in larger networks. One can observe that the performance gap between the minimum cost SRDC-I solution and $1+1$ grows as the network size increases, which clearly shows the benefit of SRDC, while the simplicity of $1+1$ is maintained. Furthermore, the SRDC-C heuristic method outperforms $1+1$ in both scenarios. In specific, it performs close to the minimum cost SRDC-I in maximal planar graphs, while its running time is 10 times faster. This is because the maximal planar graphs are relatively dense networks. Thus, there are a number of disjoint paths between s and t, which likely results 3 edge-disjoint paths for both algorithms.

C. Performance Analysis in Real-World Topologies

Finally, in this section we investigated the performance of our methods in real network topologies (SNDLib [16] and Rocketfuel [17]). In this scenario the edge capacities are constrained, i.e., 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 these edges), thus, $1+1$ and SRDC-I cannot use them in the survivable routing. One can observe in Figure 5 that as the number of bottleneck edges increases, the average bandwidth cost of $1+1$ and SRDC-I increases dramatically, while the average bandwidth cost of SRDC-C scales much better in terms of the number of bottleneck links.

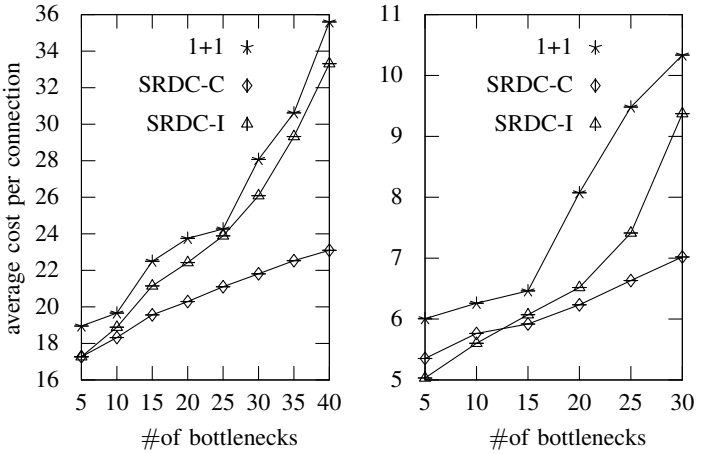
VI. Conclusions

Survivable routing with diversity coding (SRDC) is a novel, easily deployable routing scheme in transport networks which keeps the ultra-fast recovery and simplicity (both in computation and operation) of $1+1$. Furthermore, SRDC can reduce the bandwidth cost of $1+1$ in most network scenarios with 1 unit per connection, which could lead to a significant capacity saving in transport networks with excessive number of connections. This requires only some minor modification in the current operation of the widely deployed dedicated $1+1$ protection from the service providers. As a missing link of the practical implementation of SRDC, we investigated its optimal capacity allocation. We showed that a minimum cost routing for SRDC can be computed in polynomial time without capacity constraints on the links, while the problem turns to be complex when both bottleneck links and limited node capabilities coexist in the network.



(a) Optimality gap in sparse graphs (b) Optimality gap in maxplan graphs (c) Performance in sparse graphs (d) Performance in maxplan graphs

Fig. 4. 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 infinite capacities. The legends for all figures are shown in Fig. 4a and Fig. 4c.



(a) Bandwidth cost in NSFNET (79 nodes, 108 links with diameter 16) [16] (b) Bandwidth cost in Abovenet (17 nodes, 37 links, with diameter 4) [17]

Fig. 5. The bandwidth cost in real-world topologies in the capacity constrained scenario, depending on the number of bottleneck links.

ACKNOWLEDGMENTS

Research of J. Tapolcai and P. Babarczi was partially supported by the Hungarian Scientific Research Fund (OTKA grant K108947). P. Babarczi was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences (MTA). Z. Király and E. Bérczi-Kovács received grants no. CNK 77780 and no. K 109240 from the National Development Agency of Hungary, based on a source from the Research and Technology Innovation Fund. L. Rónyai was supported by the Hungarian Research Fund (OTKA grant NK105645). This document has been produced with the financial assistance of the European Union under the FP7 GÉANT project grant agreement number 605243 as part of the MINERVA Open Call project.

REFERENCES

[1] J. W. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, pp. 125–145, 1974.

[2] K.-S. Sohn, S. Y. Nam, and D. K. Sung, "A distributed LSP scheme to reduce spare bandwidth demand in MPLS networks," *IEEE Transactions on Communications*, vol. 54, no. 7, pp. 1277–1288, 2006.

[3] P.-H. Ho, J. Tapolcai, and T. Cinkler, "Segment shared protection in mesh communication networks with bandwidth guaranteed tunnels," *IEEE/ACM Trans. on Networking*, vol. 12, no. 6, pp. 1105–1118, 2004.

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

[5] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Trans. on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[6] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans on IT*, vol. 51, no. 6, pp. 1973–1982, 2005.

[7] E. Ayanoglu, I. Chih-Lin, R. Gitlin, and J. Mazo, "Diversity coding for transparent self-healing and fault-tolerant communication networks," *IEEE Trans. on Communications*, vol. 41, no. 11, pp. 1677–1686, 1993.

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

[9] P. Babarczi, J. Tapolcai, L. Rónyai, and M. Médard, "Resilient flow decomposition of unicast connections with network coding," in *Proc. IEEE Intl. Symp. on Information Theory (ISIT)*, 2014, pp. 116–120.

[10] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in *Proc. IEEE Infocom*, vol. 4. Citeseer, 2004, pp. 2307–2317.

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

[12] G. Brightwell, G. Oriolo, and F. B. Shepherd, "Reserving resilient capacity in a network," *SIAM journal on discrete mathematics*, vol. 14, no. 4, pp. 524–539, 2001.

[13] C. Fragouli and E. Soljanin, "Information flow decomposition for network coding," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 829–848, 2006.

[14] P. Babarczi, A. Pasic, J. Tapolcai, F. Németh, and B. Ladóczki, "Instantaneous recovery of unicast connections in transport networks: Routing versus coding," *accepted to Elsevier Computer Networks*, 2015.

[15] S. Fortune, J. Hopcroft, and J. Wyllie, "The directed subgraph homeomorphism problem," *Theoretical CS*, vol. 10, no. 2, pp. 111–121, 1980.

[16] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0–Survivable Network Design Library," in *Proc. INOC*, 2007.

[17] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with rocketfuel," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 133–145, 2002.