

Efficient Algorithms for Shared Backup Allocation in Networks with Partial Information

Yigal Bejerano¹, Joseph Naor^{2,*}, and Alexander Sprintson³

¹ Bell Labs, Lucent Technologies
bej@research.bell-labs.com

² Department of Computer Science, Technion-Israel Institute of Technology
naor@cs.technion.ac.il

³ Department of Electrical Engineering, California Institute of Technology
spalex@caltech.edu

Abstract. We study efficient algorithms for establishing reliable connections with bandwidth guarantees in communication networks. In the normal mode of operation, each connection uses a *primary* path to deliver packets from the source to the destination. To ensure continuous operation in the event of an edge failure, each connection uses a set of backup *bridges*, each bridge protecting a portion of the primary path. To meet the bandwidth requirement of the connection, a certain amount of bandwidth must be allocated on the edges of primary path, as well as on the backup edges. In this paper, we focus on minimizing the amount of required backup allocation by *sharing* backup bandwidth among different connections. We consider efficient sharing schemes that require only *partial* information about the current state of the network. In particular, the only information available for each edge is the total amount of primary allocation and the cost of allocating backup bandwidth on this edge. We consider the problem of finding a minimum cost backup allocation together with a set of bridges for a given primary path. We prove that this problem is \mathcal{NP} -hard and present an approximation algorithm whose performance is within $\mathcal{O}(\log n)$ of the optimum, where n is the number of edges in the primary path.

1 Introduction

Modern communication networks are expected to provide a certain level of *Quality of Service* (QoS) guarantees and also be resilient to failures. A widely used approach to achieve this goal is to provision for each connection a *primary* path and a set of backup paths. The primary QoS path is used during normal network operation; upon failure of a network element (node or edge) in the primary path, the traffic is immediately switched to a backup path. To provide QoS guarantees, a certain amount of bandwidth must be reserved on each edge of the primary path, as well as on the backup edges.

* This research is supported in part by a foundational and strategic research grant from the Israeli Ministry of Science, and by a US-Israel BSF Grant 2002276.

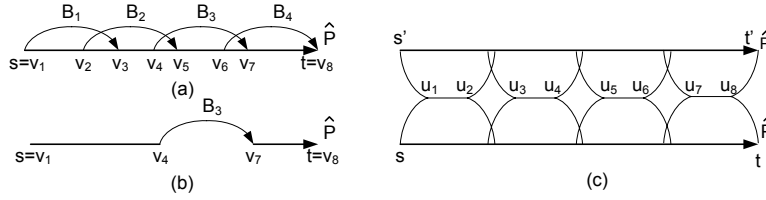


Fig. 1. (a) A primary path and a set of bridges $\{B_1, \dots, B_4\}$; (b) Activation of bridge B_3 upon failure of an edge (v_5, v_6) (c) Sharing of backup edges by two connections

In this paper we employ a *local restoration* method in order to facilitate resilience to edge failures. In this method we provision a set of *bridges*, each bridge protecting a portion of the primary path. A bridge is a path between two nodes of the primary path that shares no common edges with it. Upon failure of an edge in the primary path, the traffic is switched to one of the bridges. Fig. 1(a) depicts an example of a primary path \hat{P} and a set of bridges $\{B_1, \dots, B_4\}$ that protect edges of \hat{P} . Fig. 1(b) depicts the backup path used upon failure of edge (v_5, v_6) of \hat{P} . The backup path is formed by substituting the subpath $\{v_4, \dots, v_7\}$ of \hat{P} by bridge B_3 .

Due to budget constraints, resilience and survivability must be built into a network in an efficient manner, that is, the amount of bandwidth dedicated for this purpose must be minimized. An attractive way to achieve this goal is by *sharing* backup bandwidth among multiple connections. Sharing of backup bandwidth is possible due to low probability of simultaneous failures of multiple edges in the network. Fig. 1(c) shows an example of two connections that share several backup edges. The connections use \hat{P} and \hat{P}' as primary paths and share backup edges (u_1, u_2) , (u_3, u_4) , (u_5, u_6) , and (u_7, u_8) .

We consider a practical network setting, in which the source node of the connection computes both the primary path and a set of bridges. The source node has only *partial* information about the current state of the network. In particular, for each edge e , the following two parameters are given:

1. f_e - The total amount of bandwidth provisioned by the primary paths that use edge e ;
2. $c_e(b)$ - The cost of allocating b units of backup bandwidth on edge e .

We use the first parameter, f_e , to ensure that upon failure of edge e , all primary paths that use e are protected. To this end, we reserve along each edge of the bridge B that protects e at least f_e units of bandwidth. Indeed, in the worst case scenario all traffic that uses a failed edge e may be re-routed via bridge B .

The function $c_e(b)$ specifies the cost of allocating b units of backup bandwidth on edge e . This function may depend, for example, on the amount of backup bandwidth provisioned on e by prior connections. Indeed, in the case when edge e has a large amount of bandwidth provisioned by other connections, the cost of

allocating b units of backup bandwidth for the current connection may be small or even zero.

The partial information model used in this paper is inspired by the paper of Kodialam and Lakshman [1]. This paper provides an empirical evidence that partial information model facilitates efficient sharing of backup bandwidth, resulting in a reduction in the total amount of backup bandwidth reserved throughout the network. At the same time, the model requires only a small amount of routing information that needs to be disseminated in the network (only a few parameters per each edge in the network).

Our Results. In this study we consider the problem of finding a minimum cost backup allocation together with a set of bridges for a given primary path. We prove that this problem is \mathcal{NP} -hard and present an approximation algorithm whose performance is within $\mathcal{O}(\log n)$ of the optimum, where n is the number of edges in the primary path.

We prove that this problem is \mathcal{NP} -hard. However, we show that, by exploiting a certain combinatorial structure of the problem, we can devise an efficient approximation algorithm that achieves an approximation ratio of $8 \log n$. The computational complexity of our algorithm is $\mathcal{O}(n^3(|E| + |V| \log |V|))$, where n is the maximum number of edges in the primary path. Our algorithm can be easily extended for the problem of finding both a primary path and backup allocation. While the ideas that led to the development of our algorithm as well as the performance proofs are rather involved, the algorithm *per se* is relatively simple and easy to implement.

Proof Techniques. We analyze important properties of the problems related to bandwidth allocation with backup sharing. Specifically, we identify *internal edge sharing* as a major obstacle to finding an efficient solution. Internal edge sharing refers to the situation in which bridges that protect the same primary path share edges (see e.g. Fig. 2). We prove by employing involved combinatorial techniques that for any given primary path there is an optimal set of bridges such that each edge is included in at most $8 \log n$ bridges, where n is the number of edges in the primary path.

This property allows us to construct an approximation algorithm based on local optimization. Specifically, we define the *local cost* of a bridge to be the sum of the costs incurred at its edges. Our goal is then to find a set of bridges that protect all edges in the primary path such that the total local cost of all bridges is minimal. We show that such a solution has a certain hierarchical property and present an algorithm based on dynamic programming that identifies an optimal set of bridges with respect to local costs. We note that this approach effectively ignores internal sharing. Indeed, since the cost of each bridge is computed independently of other bridges, an edge can contribute to the cost of several bridges. However, since each edge appears in at most $8 \log n$ bridges, the cost of our solution is at most $8 \log n$ times higher than the optimum.

Related Work. Sharing of backup resources as a means for improving network performance was proposed by [1,2]. The network design problems in the context

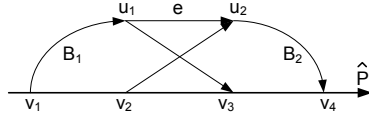


Fig. 2. Internal sharing. Bridges $B_1 = \{v_1, u_1, u_2, v_2\}$ and $B_2 = \{v_3, u_1, u_2, v_4\}$ share an edge (u_1, u_2) .

of backup sharing were investigated by [3,4]. In particular, [3] studied a simple model in which edges that belong to primary and backup paths are assigned different costs while [4] focused on the restoration problems in the *Hose* model. Papers [5–7] investigated practical aspects of path restoration. Several heuristic methods, based on linear programming for partial information model, were proposed in [1]. The current study is the first one to provide efficient approximation algorithms for computing shared backup allocation in the partial information model.

Due to space constraints, some proofs and technical details are omitted and can be found in the full version of this paper.

2 Model and Definitions

We represent the network by an undirected graph $G(V, E)$, where V is the set of nodes and E is the set of edges. An (s, t) -path is a sequence of distinct nodes $P = \{s = v_0, v_1, \dots, t = v_n\}$, such that, for $1 \leq i \leq n$, $(v_{i-1}, v_i) \in E$. Here, $n = |P|$ is the *hop count* of P . We denote by $E(P)$ the set of edges of P . The subpath of P that extends from v_i to v_j is denoted by $P_{(v_i, v_j)}$.

A unicast connection links a source node s with a destination node t . In a normal mode of operation, the packets are sent over the *primary* path $\hat{P} = \{s = v_0, v_1, \dots, t = v_n\}$. To ensure continuous operation in the event of an edge failure, we provision a set of *bridges* $\mathbb{B} = \{B_1, B_2, \dots, B_k\}$ that protect edges in the primary path. A bridge $B_i = \{s_i, \dots, t_i\}$ is a path between $s_i \in \hat{P}$ and $t_i \in \hat{P}$ that has no common edges with the subpath $\hat{P}_{(s_i, t_i)}$ of \hat{P} . We say that a bridge $B_i = \{s_i, \dots, t_i\}$ *protects* an edge $e \in \hat{P}_{(s_i, t_i)}$ if upon failure of edge e the traffic is switched from $\hat{P}_{(s_i, t_i)}$ to B_i . We denote by $\varphi(B_i)$ the set of edges in the primary path protected by bridge B_i . The set of edges that belong to bridges in \mathbb{B} is denoted by E^r , i.e., $E^r = \{e \in B_i \mid B_i \in \mathbb{B}\}$. For each edge $(v_{i-1}, v_i) \in \hat{P}$ we denote by $f_{(v_{i-1}, v_i)}$ the total amount of primary bandwidth reserved on (v_{i-1}, v_i) .

To ensure continuous operation in the event of an edge failure, we reserve a certain amount of bandwidth on each edge in E^r . In the partial information model, the backup reservation $\omega(e)$ on each edge $e \in B_j$ must satisfy:

$$\omega(e) \geq \max_{(v_{i-1}, v_i) \in \varphi(B_j)} f_{(v_{i-1}, v_i)}. \tag{1}$$

The reason for such a conservative approach is to guarantee that all primary paths that use edge (v_{i-1}, v_i) can be restored in the event of a failure of (v_{i-1}, v_i) .

Indeed, since we have no information about the backup paths used by other connections, we make a worst-case assumption that all backup traffic triggered by a failure of (v_{i-1}, v_i) is routed via edges of bridge B_j .

Since an edge $e \in E^r$ can belong to multiple bridges, the backup $\omega(e)$ reservation on edge e must satisfy:

$$\omega(e) \geq \max_{B_j, e \in B_j} \max_{(v_{i-1}, v_i) \in \varphi(B_j)} f_{(v_{i-1}, v_i)} \quad (2)$$

For each edge $e \in E$ we are given a function $c_e(b)$ that specifies, for any $b \geq 0$, the cost of reserving b units of backup bandwidth on edge e . We assume that functions $c_e(b)$ are monotonically increasing and can be computed in constant time.

Backup Allocation Problem

In the backup allocation problem, our goal is to find a *restoration topology* $\hat{\mathcal{R}}$ for a given primary path $\hat{P} = \{s = v_0, v_1, \dots, t = v_n\}$. A restoration topology $\hat{\mathcal{R}}$ is specified by a 4-tuple $\{\mathbb{B}, E^r, \varphi, \omega\}$, where \mathbb{B} is a set of bridges $\{B_1, B_2, \dots, B_k\}$; $E^r = \{e \in B_i \mid B_i \in \mathbb{B}\}$ is the set of edges that belong to bridges in \mathbb{B} ; $\varphi: \mathbb{B} \rightarrow 2^{E(\hat{P})}$ is a function that specifies, for each bridge $B_j \in \mathbb{B}$, the set of edges in the primary path protected by B_j ; and $\omega: E^r \rightarrow Z$ is a function that specifies the amount of backup bandwidth we need to allocate on each edge of E^r .

A *feasible restoration topology* must satisfy the following conditions:

1. Each edge in the primary path \hat{P} must be protected by a bridge in \mathbb{B} , i.e., for each $(v_{i-1}, v_i) \in \hat{P}$ there is a bridge $B_j \in \mathbb{B}$, such that $(v_{i-1}, v_i) \in \varphi(B_j)$.
2. The backup allocation $\omega(e)$ on each edge $e \in E^r$ must satisfy Equation (2).

The cost $C(\hat{\mathcal{R}})$ of the restoration topology is defined to be the cost of its edges, i.e., $C(\hat{\mathcal{R}}) = \sum_{e \in E^r} c_e(\omega(e))$, where c_e is the cost function associated with edge e . Our goal is to find a restoration topology of minimal cost. We refer to this problem as Problem BA (Backup Allocation) and denote the minimal cost of a solution to Problem BA by OPT. In the full version of this paper we show that Problem BA is \mathcal{NP} -hard.

3 Properties of the Optimal Backup Allocation

The main obstacle in finding an optimal solution to Problem BA is the fact that different bridges in $\hat{\mathcal{R}}$ can share edges. Such *internal sharing* is one of the reasons of the \mathcal{NP} -hardness of the problem. In this section we prove that there exists a restoration topology $\hat{\mathcal{R}} = \{\mathbb{B}, E^r, \varphi, \omega\}$ such that $C(\hat{\mathcal{R}}) = \text{OPT}$ and each edge $e \in E^r$ belongs to at most $8 \log n$ bridges of \mathbb{B} , where n is the number of edges in the primary path.

Theorem 1. *Given a primary path \hat{P} , there exists a restoration topology $\hat{\mathcal{R}} = \{\mathbb{B}, E^r, \varphi, \omega\}$ for \hat{P} such that $C(\hat{\mathcal{R}}) = \text{OPT}$ and each edge $e \in E^r$ belongs to at most $8 \log n$ bridges of $\hat{\mathbb{B}}$, where $n = |\hat{P}|$.*

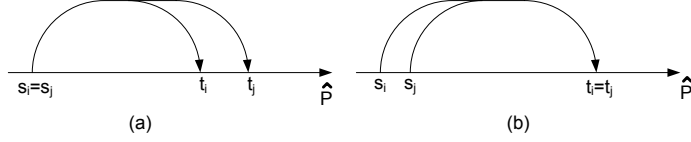


Fig. 3. Bridges with common prefix (a) and suffix (b)

In Section 4, we use Theorem 1 in order to devise an approximation algorithm for Problem BA. The algorithm finds a restoration topology $\hat{\mathcal{R}}$ whose cost is at most $8 \log n$ times more than OPT. The rest of this section is devoted to the proof of Theorem 1.

We begin by introducing the following notation. Let $B_i = \{s_i, \dots, t_i\}$ be a bridge in \mathbb{B} . A subpath $\{s_i, \dots, v\}$ of B_i is referred to as a *prefix* of B_i . Similarly, a subpath $\{v, \dots, t_i\}$ is referred to as a *suffix* of B_i . We say that two bridges $B_i = \{s_i, \dots, t_i\}$ and $B_j = \{s_j, \dots, t_j\}$ *share a prefix* if there exists a node $v \in B_i$ such that the prefix $\{s_i, \dots, v\}$ of B_i is identical to the prefix $\{s_j, \dots, v\}$ of B_j and the suffixes $\{v, \dots, t_i\}$ and $\{v, \dots, t_j\}$ of B_i and B_j are mutually edge-disjoint. Similarly, we say that two bridges $B_i = \{s_i, \dots, t_i\}$ and $B_j = \{s_j, \dots, t_j\}$ *share a suffix* if there exists a node $v \in B_i$ such that the suffix $\{v, \dots, t_i\}$ of B_i is identical to the suffix $\{v, \dots, t_j\}$ of B_j and the prefixes $\{s_i, \dots, v\}$ and $\{s_j, \dots, v\}$ of B_i and B_j are mutually edge-disjoint. Fig. 3 depicts examples of bridges that share a prefix and a suffix.

3.1 Outline of the Proof

Let $\hat{P} = \{s = v_0, v_1, \dots, t = v_n\}$ be a primary path and let $\mathcal{R} = \{\mathbb{B}, E^r, \varphi, \omega\}$ be a restoration topology for \hat{P} such that $C(\mathcal{R}) = \text{OPT}$ and the number of edges in E^r is minimal. We construct a restoration topology $\hat{\mathcal{R}} = \{\hat{\mathbb{B}}, E^r, \hat{\varphi}, \omega\}$ that satisfies condition of the theorem. The restoration topology $\hat{\mathcal{R}}$ is formed from \mathcal{R} by modifying the set of bridges \mathbb{B} and the bridge assignment function φ . The set of backup edges E^r and backup allocation ω of $\hat{\mathcal{R}}$ is identical to that of \mathcal{R} .

Our proof includes the following steps:

1. First, we construct a restoration topology $\bar{\mathcal{R}} = \{\bar{\mathbb{B}}, E^r, \bar{\varphi}, \omega\}$ that satisfies the following property. Let $\bar{B}_i = \{s_i, \dots, t_i\}$ and $\bar{B}_j = \{s_j, \dots, t_j\}$ be two bridges of $\bar{\mathbb{B}}$ such that there exists an edge (x, y) that appears in both bridges (in the same direction). Then, bridges \bar{B}_i and \bar{B}_j either share a suffix or share a prefix (see Fig. 4).
2. Then, we show that there exist a partition $\pi = \{S_i\}$ of $\bar{\mathbb{B}} = \{\bar{B}_1, \bar{B}_2, \dots, \bar{B}_k\}$ such that: (i) For each subset S_i of π it holds that either all bridges in S_i share a prefix or all bridges in S_i share a suffix. (ii) Each edge $e \in E^r$ belongs to bridges of at most four different subsets of π .
3. Finally, we prove the following assertion. Let S_i be a subset of bridges in $\bar{\mathbb{B}}$ such that either all bridges in S_i share a prefix or all bridges share a suffix. Also, let $E(S_i) \subseteq E^r$ be the set of edges that belong to bridges in S_i , i.e., $E(S_i) = \{e \in E^r \mid B \in S_i\}$. Then, there exists a set of bridges \hat{S}_i such that

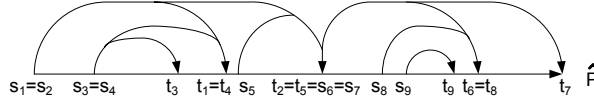


Fig. 4. An example of a restoration topology that satisfies the condition of Step 1

- (i) $E(\hat{S}_i) \subseteq E(S_i)$, where $E(\hat{S}_i) = \{e \in \hat{P} \mid \hat{B} \in \hat{S}_i\}$; (ii) Set \hat{S}_i protects the same set of edges as S_i , i.e., each edge $(v_{i-1}, v_i) \in \hat{P}$ protected by a bridge in S_i is protected by a bridge in \hat{S}_i ; (iii) Each edge $e \in E(\hat{S}_i)$ belongs to at most $2 \log n$ bridges of \hat{S}_i .

It is easy to verify that the union of all sets \hat{S}_i that correspond to sets in π satisfies the requirement of the theorem.

Step 1. Let $\hat{P} = \{s = v_0, v_1, \dots, t = v_n\}$ be a primary path and let $\mathcal{R} = \{\mathbb{B}, E^r, \varphi, \omega\}$ be an optimal restoration topology for \hat{P} . We show how to construct a set of bridges $\bar{\mathbb{B}}$ and the corresponding bridge assignment function $\bar{\varphi}$ such that any two bridges in $\bar{\mathbb{B}}$ that have an edge in common either share a prefix or share a suffix.

Let G^r be a subgraph of G induced by edges in E^r . Each edge $e \in G^r$ has a backup reservation $\omega(e)$ in \mathcal{R} . We introduce a new bottleneck metric for paths in G^r with respect $\omega(e)$. Specifically, given a path P in G^r we define the weight of the path $B(P)$ to be the smallest amount of backup reservation of an edge in P , i.e., $B(P) = \min_{e \in P} \omega(e)$.

Next, we define two functions, $\tau(v)$ and $\gamma(v)$, for each node $v \in G^r$. Function $\tau(v)$ maps a node $v \in G^r$ to a node $v_i \in \hat{P}$, while function $\gamma(v)$ maps a node $v \in G^r$ to a path between $\tau(v)$ and v in G^r . In order to define $\tau(v)$ and $\gamma(v)$, we identify a tree \mathcal{T}_{v_i} that connects $v_i \in \hat{P}$ with the rest of the nodes in G^r such that each path $P = \{v_i, \dots, u\} \in \mathcal{T}_{v_i}$ has the maximum bottleneck weight among all paths that connect v_i and u in G^r .

The function $\tau(v)$ is defined as follows. If v is a node in the primary path \hat{P} , then $\tau(v) = v$. Otherwise, $\tau(v)$ is equal to the node $v_i \in \hat{P}$ that satisfies the following conditions (i) The path between v_i and v in \mathcal{T}_{v_i} has more bandwidth than any other path between $v_j \in \hat{P}$ and v . (ii) The distance (in hops) between s and v_i in \hat{P} is smaller than that of any other node $v_j \in \hat{P}$ that satisfy condition (i).

The second function, $\gamma(v)$, is defined as follows. If v belongs to the primary path \hat{P} , then $\gamma(v)$ is an empty path. Otherwise, $\gamma(v)$ is a path between $v_i = \tau(v)$ and v in \mathcal{T}_{v_i} .

We are ready to describe the construction of the set of bridges $\bar{\mathbb{B}}$. For each edge $(v_{i-1}, v_i) \in \hat{P}$, we identify a bridge \bar{B}_i as follows. Let $B_j = \{s_j, \dots, t_j\} \in \mathbb{B}$ be a bridge in \mathcal{R} that protects edge (v_{i-1}, v_i) . Let (x, y) be an edge of B_j for which it holds that $\tau(x)$ is a predecessor of v_{i-1} in \hat{P} and $\tau(y)$ is a successor of v_i in \hat{P} . Note that such an edge must exist because $\tau(s_j)$ is a predecessor of v_{i-1} and $\tau(t_j)$ is a successor of v_i . Then, we set \bar{B}_i to be a concatenation of path $\gamma(x)$, edge (x, y) , and a path $\gamma(y)$ (in a reverse direction). Note that for any

node $v \in \gamma(x)$ it holds that $\tau(v) = \tau(x)$ and for each node $v \in \gamma(y)$ it holds that $\tau(v) = \tau(y)$. This implies that paths $\gamma(x)$ and $\gamma(y)$ are mutually node-disjoint. This fact, in turn, implies that \bar{B}_i is a simple path (i.e., does not include a cycle).

In the following lemma we prove that bridges in $\bar{\mathbb{B}}$ satisfy the required property.

Lemma 2. *Let \bar{B}_i and \bar{B}_j be two bridges in $\bar{\mathbb{B}}$ that share an edge (x, y) , i.e., $(x, y) \in \bar{B}_i$ and $(x, y) \in \bar{B}_j$. Then, \bar{B}_i and \bar{B}_j share either a prefix or a suffix.*

Proof. We consider three cases. In the first case it holds that $\tau(x) = \tau(y)$ and x is an ancestor of y in \mathcal{T}_{v_i} . In this case $\gamma(y)$ is a prefix of both bridges \bar{B}_i and \bar{B}_j . Indeed, if $\gamma(y)$ is a suffix of \bar{B}_i then \bar{B}_i contains a loop $\{x, y, x\}$, resulting in a contradiction. In the second case $\tau(x) = \tau(y)$ and y is an ancestor of x in \mathcal{T}_{v_i} . In this case $\gamma(x)$ is a suffix of both bridges \bar{B}_i and \bar{B}_j . Finally, in the third case $\tau(x) \neq \tau(y)$. In this bridge \bar{B}_i is identical to \bar{B}_j . Indeed, in this case $\gamma(x)$ is a prefix of both \bar{B}_i and \bar{B}_j , while $\gamma(y)$ is a suffix of both \bar{B}_i and \bar{B}_j .

Step 2. We show that $\bar{\mathbb{B}} = \{\bar{B}_1, \bar{B}_2, \dots, \bar{B}_k\}$ can be partitioned into subsets $\pi = \{S_i\}$ such that: (i) For each subset S_i it either holds that any two bridges in S_i share a prefix or any two bridges in S_i share a suffix. Further, any edge $e \in E^r$ belongs to bridges of at most four different subsets of π .

We construct the $\pi = \{S_i\}$ through the following *partitioning* procedure. The procedure uses an undirected auxiliary graph $G'(V', E')$, described below. The vertices V' of G' correspond to the nodes of the primary path \hat{P} and the edges E' of G' correspond to the bridges in $\bar{\mathbb{B}}$. Specifically, for each bridge $\bar{B}_i = \{s_i, \dots, t_i\} \in \bar{\mathbb{B}}$ we add an edge between s_i and t_i in G' . In the full version of this paper we prove that the subgraph G^r induced by edges in E^r , and, in turn, the auxiliary graph G' do not contain cycles.

The procedure includes the following steps for each connected component G'_i of G' :

1. Select a node $v \in G'_i$ and find an orientation of edges in G'_i such that the resulting graph \vec{G}'_i is a directed tree rooted at v ;
2. For each node $u \in \vec{G}'_i$ create two subsets S_u^1 and S_u^2 in the partition π . Both subsets include bridges of $\bar{\mathbb{B}}$ that correspond to edges incident to u . The first subset includes bridges for which u is the starting node, while the second set includes bridges for which u is the end node.

Lemma 3. *Let e be an edge in E^r and let S be a set of bridges in $\bar{\mathbb{B}}$ that include e . Then, the bridges of S belong to at most four different subsets of π .*

Step 3. Let $\pi = \{S_i\}$ be a partition of $\bar{\mathbb{B}}$ that satisfies the conditions of Step 2. Also, let $S_i \in \pi$ be a set of bridges such that all bridges in S_i share a prefix. We begin by removing from S_i all *redundant* bridges. A bridge $\bar{B} \in S_i$ is said to be redundant if all edges in $\varphi(\bar{B})$ can be protected by other bridges in S_i .

We denote by $E(S_i) = \{e \in \hat{B} \mid \hat{B} \in S_i\}$ the set of edges that belong to bridges in S_i and show that there exists a set of bridges \hat{S}_i that satisfies the

following conditions: (i) $E(\hat{S}_i) \subseteq E(S_i)$, where $E(\hat{S}_i) = \{e \in \hat{B} \mid \hat{B} \in \hat{S}_i\}$; (ii) Each edge $(v_{i-1}, v_i) \in \hat{P}$ protected by a bridge in S_i is protected by a bridge in \hat{S}_i ; (iii) Each edge $e \in E(\hat{S}_i)$ belongs to at most $2 \log n$ bridges of \hat{S}_i .

We denote by s' the starting node of all bridges in S_i and by $T = \{t^j\}$ the set of end nodes of bridges in S_i , such that t^{j-1} is a predecessor of node t^j in \hat{P} . For each $j, 1 \leq j \leq T$, we denote by B^j the bridge in S_i with end node t^j . We also denote by G' the subgraph of G induced by $E(S_i)$. Note that G' is a subgraph of G^r (recall that G^r is a subgraph of G induced by edges in E^r). Since G^r does not contain a cycle, G' is a tree. In addition, we denote by Γ the set of edges in \hat{P} protected by bridges in S_i , i.e., $\Gamma = \cup_{\hat{B} \in S_i} \varphi(\hat{B})$. We partition Γ to $|T|$ subsets $\Gamma_1, \dots, \Gamma_{|T|}$ such that subset Γ_1 include edges in Γ located between nodes s' and t^1 and a subset Γ_j includes edges of Γ located between nodes t^{j-1} and t^j in T .

Lemma 4.

1. For each $j, 1 \leq j \leq T$, it holds that bridge B^j can protect all edges in Γ_j , i.e., $\min_{e \in B^j} \omega(e) \geq \max_{(v_{x-1}, v_x) \in \Gamma_j} f_{(v_{x-1}, v_x)}$
2. Let P be a path in G' between t^k and t^j , where t^k is a predecessor of t^j in \hat{P} . Then, P is a bridge that can protect all edges in Γ_j , i.e., $\min_{e \in P} \omega(e) \geq \max_{(v_{x-1}, v_x) \in \Gamma_j} f_{(v_{x-1}, v_x)}$.

We are ready to describe a procedure that constructs the set of bridges \hat{S}_i . The procedure includes the following steps.

1. For each $B_j \in S_i$ set $\varphi(B_j) = \Gamma_j$.
2. While S_i is not empty, perform the following operations:
 - (a) Denote by $E(S_i)$ the set of edges that belong to bridges in S_i , i.e., $E(S_i) = \{e \in \hat{B} \mid \hat{B} \in S_i\}$. Denote by G' the subgraph of G^r induced by edges in $E(S_i)$.
 - (b) Identify an Euler tour $W = \{s', \dots, s'\}$ in G' .
 - (c) Break W into paths $\{P_k\}$ by cutting W at nodes s and $t_k \in T$.
 - (d) Denote by s_k the starting node of P_k and by t_k the end node of P_k , such that s_k is a predecessor of t_k in \hat{P} .
 - (e) For each path $P_k = \{s_k, \dots, t_k\}$ for which there is a bridge $B^j = \{s', \dots, t^j\} \in S_i$ with the same end node (i.e., $t_k = t^j$), perform the following operations:
 - i. Add P_k into \hat{S}_i and set $\varphi(P_k) = \varphi(B^j)$.
 - ii. Remove B^j from S_i .

Fig. 5 demonstrates a single iteration of the procedure. A set of bridges that share a common prefix is depicted in Fig. 5(a). An Euler tour W on G' is depicted in Fig. 5(b). We break W into four paths, two of which are added to \hat{S}_i , as depicted in Fig. 5(c).

Theorem 5. *The set of bridges \hat{S}_i satisfies the following conditions: (i) $E(\hat{S}_i) \subseteq E(S_i)$, where $E(\hat{S}_i) = \{e \in \hat{B} \mid \hat{B} \in \hat{S}_i\}$; (ii) Each edge $(v_{i-1}, v_i) \in \hat{P}$ protected by a bridge in S_i is protected by a bridge in \hat{S}_i ; (iii) Each edge $e \in E(\hat{S}_i)$ belongs to at most $2 \log n$ bridges of \hat{S}_i .*

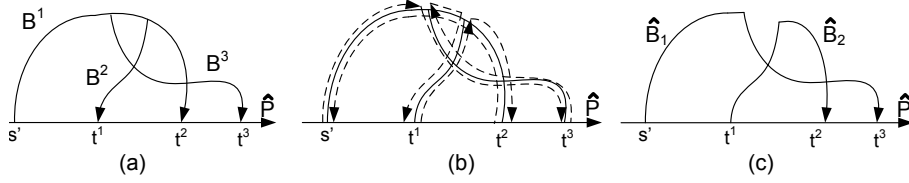


Fig. 5. (a) A set S_i of bridges that share a common prefix (b) An Euler tour on G' (c) Two new bridges are added to \hat{S}_i

The case in which all bridges in S_i have a common suffix can be proven by using similar arguments.

4 Approximation Algorithm for Problem BA

4.1 Locally Optimal Restoration Topologies

In this section we present an approximation algorithm for Problem BA. We begin by defining the notion of the *local cost* of a restoration topology.

Definition 1 (Local Cost). Let $\mathcal{R} = \{\mathbb{B}, E^r, \varphi, \omega\}$ be a restoration topology for \hat{P} . The local cost $C^*(B)$ of a bridge $B \in \mathbb{B}$ is defined to be the sum of local costs incurred at its edges, i.e., $C^*(B) = \sum_{e \in B} c_e(\omega(e))$. The local cost of a restoration topology is defined to be the sum of the local costs of its bridges:

$$C^*(\mathcal{R}) = \sum_{B \in \mathbb{B}} C^*(B) = \sum_{B \in \mathbb{B}} \sum_{e \in B} c_e(\omega(e)). \quad (3)$$

Definition 2 (Partial Restoration Topology). Let P be a subpath of the primary path \hat{P} . A restoration topology $\mathcal{R} = \{\mathbb{B}, E^r, \varphi, \omega\}$ is referred to as a partial restoration topology for P if each edge in P is protected by a bridge $B \in \mathcal{R}$. The backup allocations $\omega(e)$ on each edge $e \in E^r$ must satisfy the conditions of (2).

We say that a partial restoration topology \mathcal{R} for P is *locally optimal* if its local cost is less than or equal to the local cost of any other partial restoration topology \mathcal{R} for P . The minimum cost of a locally optimal restoration topology for P is denoted by $\text{OPT}^*(P)$. From Theorem 1 it follows that $\text{OPT}^*(\hat{P}) \leq 8 \cdot \text{OPT} \log n$, where $n = |\hat{P}|$.

The next lemma establishes a hierarchical property of restoration topologies with minimal local cost. This lemma allows us to use the methods of dynamic programming in order to find restoration topologies of minimal local cost.

Lemma 6. Let $P' = \{s', \dots, t'\}$ be a subpath of the primary path \hat{P} . Then one of the following conditions hold:

1. Path P can be partitioned into edge-disjoint subpaths P^1, \dots, P^k such that $\text{OPT}^*(P') = \sum_{i=1}^k \text{OPT}^*(P^i)$;

2. There exist edge-disjoint subpaths P^1, \dots, P^k of P' , a bridge $B = \{s', \dots, t'\} \in G \setminus P'$, and a value ζ , such that:
- (a) ζ is the maximum value $f_{(v_{i-1}, v_i)}$ of an edge $(v_{i-1}, v_i) \in P'$ that does not belong to paths P^1, \dots, P^k ;
 - (b) $B = \{s', \dots, t'\}$ is a minimum cost path between s' and t' with respect to the cost $\sum_{e \in B} c_e(\zeta)$;
 - (c) $\text{OPT}^*(P') = \sum_{i=1}^k \text{OPT}^*(P^i) + \sum_{e \in B} c_e(\zeta)$.

The hierarchical property allows us to efficiently find an optimal restoration topology $\hat{\mathcal{R}}$ by using the methods and tools of dynamic programming.

4.2 Dynamic Programming Algorithm

The algorithm exploits the hierarchical property of locally optimal restoration topologies, established by Lemma 6. The algorithm begins with the subpaths of \hat{P} that include a single edge and computes for each subpath an optimal restoration topology that protects it. Then, it computes locally optimal restoration topologies for all subpaths of length 2, 3 and so on, until the locally optimal restoration topology for the entire primary path is found. The order of processing the subpaths of P by the algorithm ensures that when the algorithm computes an optimal restoration topology for a subpath of length i , it already has available the optimal restoration topologies for all subpaths of length smaller than i .

We observe that optimal restoration topology for a subpath of length 1 includes a single bridge, which is easy to identify. Indeed, let P be a subpath that includes a single edge (v_{i-1}, v_i) . Note that each edge e of bridge B must be allocated $f_{(v_{i-1}, v_i)}$ units of backup bandwidth. Thus, to find a locally optimal bridge that protects (v_{i-1}, v_i) we compute a shortest path in $G \setminus \hat{P}$ with respect to edge costs $c_e(f_{(v_{i-1}, v_i)})$ between s_i and t_i , where s_i is a predecessor of v_{i-1} in \hat{P} and t_i is a successor of node v_i in \hat{P} . Such a path can be computed by a single invocation of a shortest path algorithm such as Dijkstra's algorithm.

For a subpath P of \hat{P} of length more than 1, we need to consider two possible cases, described in Lemma 6. For the first case, we need to determine a partition P^1, \dots, P^k of P such that total cost of restoration topologies $\mathcal{R}^1, \dots, \mathcal{R}^k$ is minimal, where \mathcal{R}^i is a locally optimal restoration topology that protect P^i . Note that the optimal restoration topologies $\mathcal{R}^1, \dots, \mathcal{R}^k$ are already computed by the algorithm. Such an optimal partition can be determined by using the auxiliary graph \hat{G} , that includes, for each subpath $P_{(v_i, v_j)}$ of P , an edge (v_i, v_j) , whose cost is equal to the cost of the locally optimal restoration topology that protects $P_{(v_i, v_j)}$. Then, we compute a shortest path between the source node and the destination node of P and identify an optimal restoration topology for P by taking the union of all restoration topologies that correspond to the edges of the shortest path.

The second case is more complicated, as we need to determine a bridge \hat{B} and several restoration topologies $\mathcal{R}_1, \dots, \mathcal{R}_k$ that protect the edges of P . The key step is to determine the amount of backup bandwidth ζ that must be reserved

on edges of \hat{B} . Indeed, if ζ is known, then we can easily determine \hat{B} (in a manner similar to the case in which the subpath P includes a single edge) and the edges of P that are protected by \hat{B} (edges (v_{i-1}, v_i) for which it holds that $f_{(v_{i-1}, v_i)} \leq \zeta$). Then, we can identify restoration topologies that protect all other edges of P (not protected by \hat{B}) by using the same technique as in the first case.

In order to determine ζ we use the following observation. We observe that ζ must be equal to $f_{(v_{i-1}, v_i)}$ for some of edges in the P . Thus, the optimal value of ζ can be found by performing the following exhaustive search: for each value of ζ from the set $\{f_{(v_{i-1}, v_i)} \mid (v_{i-1}, v_i) \in P\}$ we compute the optimal restoration topology and choose ζ for which the local cost of a restoration topology is minimal.

Theorem 7. *Given a primary path \hat{P} , the algorithm above identifies, in $\mathcal{O}(n^3(|E| + |V| \log |V|))$ time, a solution to Problem BA whose cost is at most $8 \log n$ times more than the optimum.*

References

1. Kodialam, M.S., Lakshman, T.V.: Dynamic routing of bandwidth guaranteed tunnels with restoration. In: Proceedings of IEEE INFOCOM'2000, Tel-Aviv, Israel (2000)
2. Hwang, H., Ahn, S., Choi, Y., Kim, C.: Backup path sharing for survivable ATM networks. In: Proceedings of ICOIN-12. (1998)
3. Chekuri, C., Gupta, A., Kumar, A., Naor, J., Raz, D.: Building edge-failure resilient networks. In: Proceedings of IPCO 2002. (2002)
4. Italiano, G., Rastogi, R., Yener, B.: Restoration algorithms for virtual private networks in the hose model. In: Proceedings of IEEE INFOCOM'02, New York, NY (2002)
5. Su, X., Su, C.F.: An online distributed protection algorithm in WDM networks. In: Proceedings of IEEE ICC'01. (2001)
6. Sengupta, S., Ramamurthy, R.: Capacity efficient distributed routing of mesh-restored lightpaths in optical networks. In: Proceedings of IEEE GLOBECOM '01. (2001)
7. Li, G., Wang, D., Kalmanek, C., Doverspike, R.: Efficient distributed path selection for shared restoration connections. In: Proceedings of IEEE INFOCOM'02, New York, NY (2002)