Exploiting the diversity of shortest pairs of edge-disjoint paths

Marina Girolimetto[1], **Rodrigo S. Tessinari**[1],
Fabio O. Lima[2], **Claunir Pavan**[3], **Marcia H. M. Paiva**[1]

[1]Laboratório de Telecomunicações - LabTel, Universidade Federal do Espírito Santo (UFES), 29075-910 – Vitória – ES – Brazil

[2]Instituto Federal do Espírito Santo (IFES), 29173-087 – Serra – ES – Brazil

[3]Universidade Federal da Fronteira Sul (UFFS), 89814-470 – Chapecó – SC – Brazil

`marina.girolimetto@aluno.ufes.br`[1]

Abstract. *In an optical network, given a pair of source and destination nodes, some algorithm can be used to find shortest pairs of edge-disjoint paths to be used as working and backup paths. The Suurballe and Tarjan's algorithm is a solution, but it can found different shortest pairs of pathways interconnecting the same pair of source and destination nodes. In this paper, two versions of the Suurballe and Tarjan's algorithm is proposed to deal with that diversity. For each node pair of a given network topology, these versions find the most balanced shortest pair of working and backup paths and the least balanced one. Both algorithms are tested and analyzed in a set of 40 2-edge-connected topologies of real-world optical telecommunication networks. A difference of up to 29% was found between the two strategies.*

1. Introduction

In telecommunication networks, even a single node or link failure may significantly impair the data traffic. As a result, it causes communication problems in several pairs of source and destination nodes. Therefore, the topological design must guarantee that the network survives, i.e., that it remains connected in failure scenarios.

For instance, a necessary and sufficient condition for a network to survive any single link failure is that each pair of source and destination nodes is interconnected by at least two paths that do not share links. Noticing, a network that satisfies this condition is called *2-edge-connected*. Analogously, *2-node-connected* networks survive any single node failure since each pair of source and destination nodes is interconnected by at least two paths that do not share nodes [Whitney 1932].

Thus, it is possible to define two edge-disjoint or node-disjoint working and backup paths for each pair of source and destination nodes of a survivable network. Nevertheless, these paths can be used in different ways by different protection schemes. Usually, the working path carries the traffic load whereas the backup path carries the traffic in the case of a failure [Ramaswami et al. 2009]. However, in some protection schemes, both pathways are used simultaneously.

In protection schemes, working and backup paths must be defined previously to ensure instant recovery of any single link or node failure. The most straightforward protection scheme uses a shortest path as working path. Then, a second shortest path that does not share any node or link with the first one is sought to be the backup path.

The issue of this approach is that, in some network topologies, when fixing the shortest path as the working path, no disjoint path can be found for the backup path, even when they are *2-node-connected*. Figure 1 presents a *2-connected* network that exemplifies this situation, where for a pair of source node S and destination node D, the shortest path is found, but a disjoint backup path will not be possible without sharing a link. Techniques where the backup path is found only in case of failure also exist, but they fit into restoration techniques rather than protection.

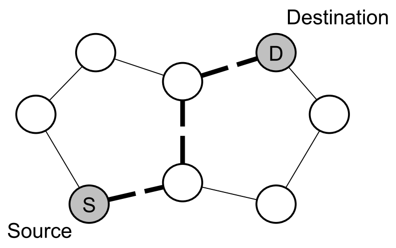


Figure 1. A *2-node-connected* network for which, if the shortest path between nodes S and D (the 3-hop path in dashed line) is chosen as working path, there is no disjoint path to be used when failures occur.

Therefore, to guarantee that two disjoint paths can be found, it is used an algorithm to find the shortest pair of node or link-disjoint paths for each pair of nodes. For instance, the *Suurballe and Tarjan's* (ST) algorithm [Suurballe and Tarjan 1984] can be used for that purpose. Usually, in that approach, the shortest path of the pair is used as working path and the other one as the backup path.

It happens that, in some network topologies, there may exist different shortest pairs of paths interconnecting the same pair of source and destination nodes. For instance, Figure 2 illustrates this case for the pair of source node S and destination node D of "Arnes" network. In that network, the shortest pair of paths from S to D of length 16, can correspond to (i) a working path length 3 and a backup path length 13 or (ii) working and backup paths length 8 each.

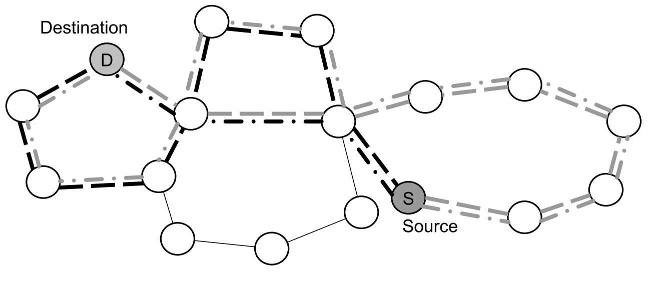


Figure 2. Two different shortest pairs of edge-disjoint paths interconnecting the source node S to the destination node D of "Arnes" network. In dotted and dashed lines: working path length 3 in black color and backup path length 13 in gray color; in dashed lines: working and backup paths length 8 each.

The existence of more than one shortest pair of paths associated with the same pair of source and destination nodes allows us to explore different strategies for choosing the working and backup paths. For example, in the case of latency-sensitive service, the working and backup paths should ideally have the same length, or at least their lengths should be as close as possible. In this case, one must adopt the strategy of seeking what is termed as the pair of *most balanced* paths. In other scenarios, the working path is ideally as short as possible since the signal is only released on it and the backup path is activated only if necessary. In this case, the most appropriate strategy would be to seek what is called a *least balanced* pair of paths. Both protection strategies are observed in practical cases such as in optical transport networks [Manchester et al. 1999].

It is important to notice that, when running the standard ST algorithm no assurance can be given concerning which shortest pair of paths will be found. Thus, the considerable path length difference found in the above example cannot be used for the benefit of any desired protection scheme.

From this observation and to explore different strategies for the choice of working and backup paths, we propose two versions of the ST algorithm, as detailed in Section 4, to be used in the context of optical networks under protection schemes. These versions find, for a given topology, the most balanced and the least balanced shortest pairs of working and backup paths for each pair of nodes and are called: *Suurballe and Tarjan Most Balanced* ($STMB$) and *Suurballe and Tarjan Least Balanced* ($STLB$) algorithms.

We tested and analyzed how both $STMB$ and $STLB$ algorithms perform in a set of 40 real-world optical telecommunication networks. The impact of these algorithms is investigated through (i) the number of transponders required to route a uniform traffic demand and (ii) the protection coefficient, i.e., the fractional amount of additional capacity required to implement dedicated-path protection scheme [Korotky 2004, Labourdette et al. 2005].

This paper is organized as follows. Section 2 comments briefly on protection schemes focusing on dedicated-path protection. Section 3 describes the *Suurballe and Tarjan's* algorithm, and the most balanced and least balanced algorithms are described in Section 4. The results are presented in Section 5 whereas Section 6 concludes this paper and points out future research directions.

2. Dedicated-Path Protection

Protection schemes are a type of fault-recovery mechanism for optical networks, usually classified as dedicated protection and shared protection. Each category can also be divided into link protection and path protection. Protection techniques can be found in more details in [Grover 2003, Zang 2012], whereas [Shen et al. 2016] presents a recent survey for elastic optical networks.

Dedicated protection schemes usually require more resource but present shorter recovery time than shared protection ones. On the other hand, path protection usually requires fewer resources and presents lower end-to-end propagation delay compared to link protection [Zang 2012]. This work focus on dedicated-path protection schemes.

In dedicated-path protection schemes, a working and backup edge-disjoint paths are defined for each node pair of the network and the resources (e.g., transponders) cannot be shared with other pairs. Dedicated protection can be implemented in two different ways, known as 1:1 and 1+1 [Zang 2012, Lang et al. 2007, Berger et al. 2007].

In 1:1 protection, resources for the backup path are reserved only to ensure recovery when a failure occurs. During normal network operation conditions, traffic is transmitted through the working path only and the backup resources can be used to transport low-priority traffic. Whenever a failure is detected on the working path, that extra traffic is discarded, and the traffic affected by the fault is switched to the backup path.

In 1+1 protection, the resources reserved for the backup path are used all the time since the traffic is transmitted through the working and backup paths simultaneously. Consequently, the destination node can select the best signal received, keeping the connection alive even in case of a single failure.

To implement dedicated-path protection in optical networks operating in opaque transport mode, every optical channel spanning a link requires a pair of transponders. Therefore, the average number of optical transponders per link for the working paths is given by [Pavan et al. 2015]:

$$\langle t^w \rangle = 2\frac{\langle d \rangle \langle h^w \rangle}{\langle \delta \rangle}, \tag{1}$$

where $\langle d \rangle$ is the average number of traffic demands, $\langle h^w \rangle$ is the average number of hops considering only the working paths, and $\langle \delta \rangle$ is the average nodal degree of the network. Similarly, the average number of optical transponders per link for the backup paths is given by:

$$\langle t^b \rangle = 2\frac{\langle d \rangle \langle h^b \rangle}{\langle \delta \rangle}, \tag{2}$$

where $\langle h^b \rangle$ is the average number of hops considering only the backup paths.

The efficiency of protection schemes can be measured by the ratio between the total amount of resources reserved for backup paths and the total amount of resources reserved for working paths [Korotky 2004, Labourdette et al. 2005]. This ratio is referred to as protection coefficient [Pavan et al. 2015] or as spare capacity redundancy [Shen et al. 2016].

The dedicated-path protection coefficient is calculated as [Pavan et al. 2015]:

$$\langle k_p \rangle = \frac{\langle h^b \rangle}{\langle h^w \rangle}. \tag{3}$$

Notice that, since $\langle h^b \rangle$ is always greater than or equal to $\langle h^w \rangle$, $\langle k_p \rangle$ is always greater than or equal to one. Therefore, the higher the $\langle k_p \rangle$, the higher the cost to protect the network from failure.

As shown in Eqs. (1), (2), and (3), both the number of transponders and the protection coefficient directly depend on the working and backup path lengths. Thus, different dedicated-path protection schemes can benefit from different ways to define these paths.

3. Suurballe and Tarjan's algorithm

This section describes the *Suurballe and Tarjan's* (ST) algorithm [Suurballe and Tarjan 1984].

Let $G = (V, E)$ be a *2-edge-connected* graph of order N where $V = \{v_i, v_j, ...v_N\}$ is the set of vertices and E is the set of edges of unit weights. For each pair of source node, v_s, and destination node, v_d, of a *2-edge-connected* network topology, the ST algorithm finds the shortest pair of edge-disjoint paths to be used as working and backup paths. According to Whitney's version of the Menger's Theorem [Whitney 1932], it is possible to find the two paths whenever the network is *2-edge-connected.* Also, the running time complexity of the ST algorithm is $O(|E| + |V|\ log\ |V|))$. The ST algorithm works according to the following steps [Oliveira 2010]:

1. Run *Dijkstra's* algorithm and finds the shortest path P_1 connecting the source node, v_s, and the destination node, v_d;
2. Change the weights of P_1 edges. This Step is known as "cost transformation" and occurs as follows:

$$c_{ij} = \begin{cases} N & ,if\ (v_i, v_j)\ \in\ P_1 \\ c_{ij}\ +\ c_i\ -\ c_j & ,if\ (v_i, v_j)\ \notin\ P_1 \\ & and\ (v_j, v_i)\ \notin\ P_1 \\ 0 & ,if\ (v_j, v_i)\ \in\ P_1, \end{cases}$$

   where c_i is the cost of path of node v_s to node v_i and c_{ij} is the cost of the edge v_i, v_j;
3. Run *Dijkstra's* algorithm once again with the weights of the edges of P_1 modified and finds the second shortest path P_2;
4. Check if P_1 and P_2 share any edge. If they do, P_1 and P_2 are redefined to P_1' and P_2' where the path P_1 will be formed by the initial edges of P_1 that precede the shared edge, followed by the end of the edges of P_2 that follow the shared edge. The common edge is not added to any of the new paths. For P_2 the method is the same, beginning with P_2 and ending with the edges of P_1. This procedure will be repeated if there is more than one common edge in the paths. If there is no shared edge in P_1 and P_2, then they remain.

The resulting pair of paths, which can be P_1 and P_2 or P_1' and P_2' is used to define working and backup paths.

To exemplify this procedure, consider the network shown in Figure 3 [a] where $v_s = v_1$ and $v_d = v_8$ and the edges have unitary weight. The shortest path P_1 obtained is $v_1 \leftrightarrow v_5 \leftrightarrow v_4 \leftrightarrow v_8$, with three hops. The weights of all the edges of P_1 are changed; thus, it is found a second shortest path P_2, $v_1 \leftrightarrow v_2 \leftrightarrow v_3 \leftrightarrow v_4 \leftrightarrow v_5 \leftrightarrow v_6 \leftrightarrow v_7 \leftrightarrow v_8$. Next, it is verified if P_1 and P_2 share edges, in this example, $v_4 \leftrightarrow v_5$. Therefore, P_1 and P_2 are redefined as P_1' and P_2', respectively. Illustrated in Figure 3 [b] both end paths: the four-hops P_1' is given by $v_1 \leftrightarrow v_5 \leftrightarrow v_6 \leftrightarrow v_7 \leftrightarrow v_8$ and also the four-hops, P_2' is given by $v_1 \leftrightarrow v_2 \leftrightarrow v_3 \leftrightarrow v_4 \leftrightarrow v_8$.

4. Proposed versions of *Suurballe and Tarjan's* algorithm

As observed in Section 1, the *Suurballe and Tarjan's* algorithm can found different shortest pairs of paths interconnecting the same pair of source and destination nodes. This section describes the two proposed versions of this algorithm called: *Suurballe and Tarjan Most Balanced* ($STMB$), and *Suurballe and Tarjan Least Balanced* ($STLB$).

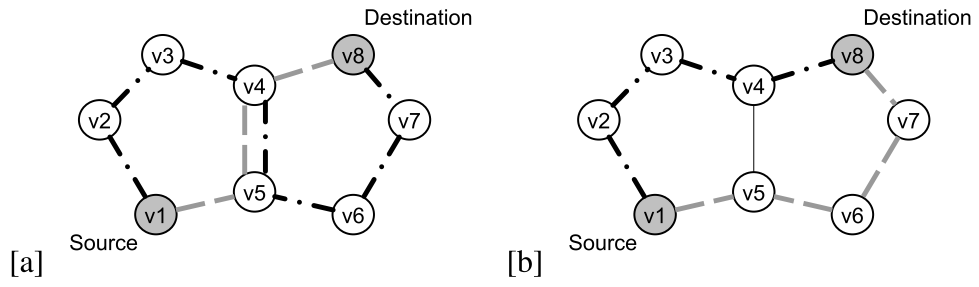


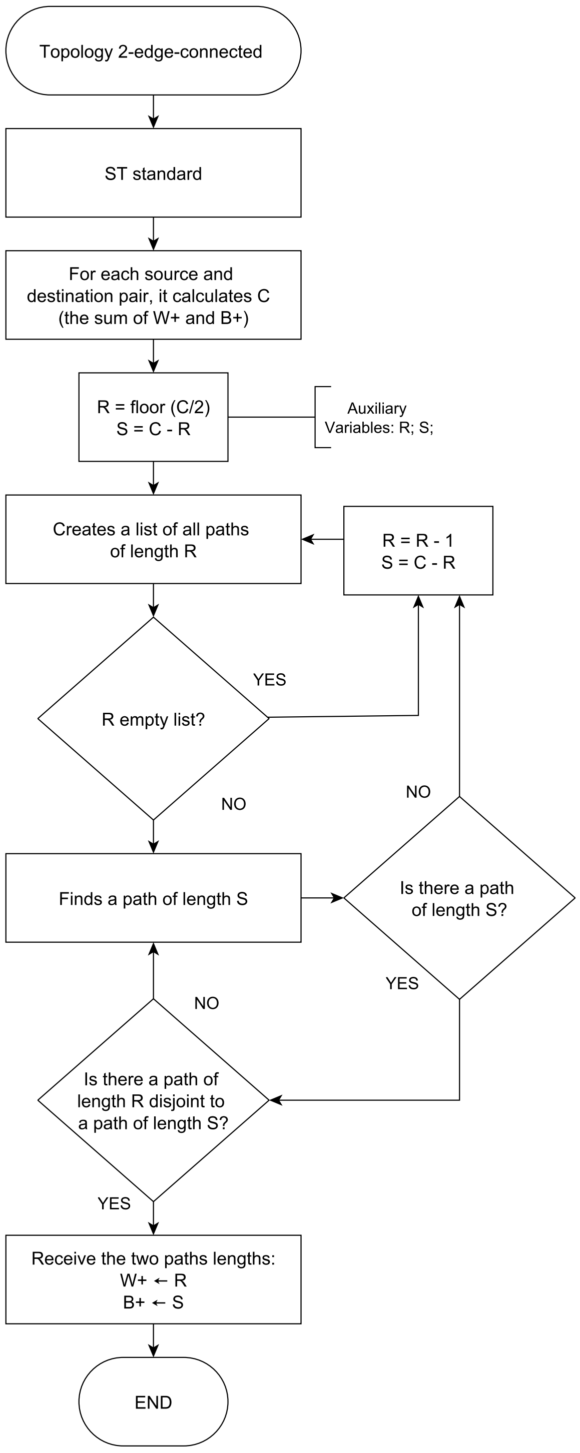
Figure 3. From $v_s = v_1$ to $v_d = v_8$, [a] shortest path (in dashed gray line) and second shortest path (in black line with dot and dash), obtained with the *Dijkstra's* algorithm. [b] Working and backup disjoint paths obtained with the ST algorithm.

4.1. Suurballe and Tarjan Most Balanced (STMB)

The aim of $STMB$ algorithm is to find the shortest pair of edge-disjoint paths for each pair of source and destination nodes, keeping the lengths of the working and backup paths as equal as possible. Hereafter, we refer to these lengths as W^+ ("*Length of the Most Balanced Working Path*") and B^+ ("*Length of the Most Balanced Backup Path*"). Also, we refer to the sum of these lengths as C.

The flowchart of the algorithm is shown in Figure 4 and follows this way:

1. Run the standard ST algorithm for a *2-edge-connected* network. For each pair of source and destination nodes, two edge-disjoint paths are found. Then C is stored;
2. Define R and S variables for each pair, based on C. The R will define the new length of the W^+ and its value will be the rounded down of half of C. The S will define the new length of the B^+ and its value will be the result of C minus R. This Step will always be possible because C is a positive integer measured in number of hops;
3. Create a list to store all paths length R for a given pair of source and destination nodes, using *Yen's* algorithm [Yen 1970], which determines K-shortest paths. For this Step, the paths in ascending order are presented and, when the first path length R appears, the list will no longer be empty and will store the path. The next paths length R will be added as well. When the *Yen's* algorithm returns the first path length greater than R, or all possible paths are over, this Step is ended. Because it is an exhaustive algorithm, the *Yen's* algorithm will always be feasible;
4. Check if the R list is empty. If it is, no path length R has been found and then the R will be decremented and return to Step 3. If it is not, follow to the next Step;
5. Find one and only one path length S, also using the *Yen's* algorithm;
6. Check for a path length S. If it does not exist, R is decremented and returns to Item 3. If it exists, compare the path length S with the first path from the list of paths length R. At this point, it is checked if there is any shared edge between them. If there is not a shared edge, the values of R and S are assigned to W^+ and B^+, respectively. If there is a shared edge in these paths, the next path length R in the list is tested. If no path length R is disjoint to path length S, then returns to Step 5 to test another path length S, if it exists. If all paths length S are tested, and no path disjoint R is found, then R is decremented, and the algorithm returns to Step 3. This procedure will be performed until two edge-disjoint paths are found.



Topology 2-edge-connected
ST standard
For each source and destination pair, it calculates C (the sum of W+ and B+)
R = floor (C/2)
S = C - R
Auxiliary Variables: R; S;
Creates a list of all paths of length R
R = R - 1
S = C - R
YES
R empty list?
NO
NO
Finds a path of length S
Is there a path of length S?
YES
NO
Is there a path of length R disjoint to a path of length S?
YES
Receive the two paths lengths:
W+ ← R
B+ ← S
END

Figure 4. Flowchart of the *Suurballe and Tarjan Most Balanced* algorithm ($STMB$).

If there are two edge-disjoint paths in the topology for each pair of source and destination nodes, the ST algorithm will be successful, and consequently, the $STMB$ will also be.

4.2. Suurballe and Tarjan Least Balanced (STLB)

Different from $STMB$, the least balanced version of the ST algorithm, $STLB$, maximizes the differences between W^- ("*Length of the Least Balanced Working Path*") and B^- ("*Length of the Least Balanced Backup Path*").

The $STLB$ algorithm acts similarly to $STMB$ but with two differences: $i)$ to obtain the shortest working path possible, R starts with the shortest path length found by *Dijkstra's* algorithm; and $ii)$ when two edge-disjoint paths length R and S are not found, R is incremented. As in $STMB$, if there are two edge-disjoint paths in the topology for each pair of source and destination nodes, the ST algorithm will be successful, and consequently, the $STLB$ will also be.

4.3. STMB's and STLB's proof

Given a pair of source and destination nodes, the $STMB$ and $STLB$ algorithms will not necessarily fetch all possible pairs of paths length C. Once the disjoint paths lengths R and S are found, even if they are modified by increasing or decreasing, the algorithm stops. These paths can be the same as defined by ST or not.

The number of pairs of paths length C influences the stopping criterion of the algorithm. When there are not many pairs of paths length C, the algorithm will execute quickly, since there are fewer options for checking. This is the best case. On the other hand, if there are many paths and R and S are incremented or decremented several times, finding the new W^+ or W^- and B^+ or B^- in the last possible path length options, then the algorithm can take a long time in operations. So, this is the worst case.

The problem of finding a path of a given length without affecting edges would solve the *Hamiltonian Path Problem* by requiring a path length $N - 1$. Since the *Hamiltonian Path Problem* is NP-Complete [Schrijver 2003], the problem in question will also be. Efficient algorithms known for this problem have an exponential cost such as the *Yen* [Yen 1970] algorithm used in Steps 3 and 5 of the $STMB$ and $STLB$ algorithms, for which the running time complexity is $O(KV(E + V\ log\ V))$.

For finding simple paths length k in a graph, the best known asymptotic complexity is currently $O(2^k)$ time [Williams 2009]. This problem has been proved to be NP-Complete [Nykänen and Ukkonen 2002].

5. Results

To test the algorithms, we used a set of 40 real-world telecommunications networks available in [Routray et al. 2013, Pinto 2014]. These networks are *2-edge-connected*, containing between 9 and 100 nodes and 12 to 171 links each. The proposed algorithms were implemented using $C++$ language and the *ElasticO++ framework* [Tessinari et al. 2016]. The total processing time to compute the paths using $STMB$ algorithm in all 40 networks was approximately one hour, from which the biggest network (USA with 100 nodes) took about 57 minutes. The processing time obtained using $STLB$ was higher, approximately seven hours for the USA topology and three minutes in total for the other 39 networks. In our tests, we used a PC desktop equipped with an Intel Core i5-6600K CPU @3.50GHz and 8GB of RAM.

In both $STMB$ and $STLB$, the lengths of the working and backup paths were obtained for each pair of source and destination nodes of each network topology. Hereafter, those lengths are referred to as W^+ and B^+ for $STMB$ algorithm and W^- and B^- for the $STLB$ algorithm. First, we explore in each network the differences between working and backup paths of $STMB$ and $STLB$ for each pair of source and destination nodes. Subsequently, we consider the sum of those differences and compare the results obtained for the 40 networks.

It should be noticed that for any given pair of nodes, W^- will always be smaller than or equal to W^+, and B^- will always be greater than or equal to B^+. Also, since both $STMB$ and $STLB$ define pairs of paths of same total length C, we have:

$$W^+ + B^+ = W^- + B^- = C. \tag{4}$$

The difference between the two strategies can be measured by the variation in the lengths of workings paths (or backup paths). Therefore, for each pair of source and destination nodes (s, d):

$$\Delta_{s,d} = W^+ - W^- = B^- - B^+. \tag{5}$$

In each algorithm, the difference between the working path and the backup path for each pair of nodes (s, d) is given by:

$$\begin{aligned} \Theta^+_{s,d} &= B^+ - W^+; \\ \Theta^-_{s,d} &= B^- - W^-. \end{aligned} \tag{6}$$

This difference between the working path and the backup path is related to the difference between the two strategies. This relationship is given by:

$$\Delta_{s,d} = \frac{\Theta^-_{s,d} - \Theta^+_{s,d}}{2}. \tag{7}$$

Therefore, the higher the difference between $\Theta^-_{s,d}$ and $\Theta^+_{s,d}$, the more significant the difference between the two algorithms.

From the networks tested, the one presenting the largest difference between the two strategies is the “Arnes” network (17 nodes and 20 edges) shown in Fig. 5 (this figure is repeated here for convenience).

One of the pairs of nodes reaching the largest difference between the $STMB$ and the $STLB$ is the pair (S, D) highlighted in Fig. 5, for which $C = 16$ edges. In $STMB$, both working and backup paths obtained have 8 hops, that is, $W^+ = B^+ = 8$. In $STLB$, the working path obtained has 3 hops, i.e., $W^- = 3$, whereas the backup path has 13 hops, i.e., $B^- = 13$. Therefore, $\Theta^+_{S,D} = 0$ and $\Theta^-_{S,D} = 10$ for this pair. Also, as the working path of the $STMB$ has 5 hops more than the working path of the $STLB$ (consequently, the backup path of the $STLB$ has 5 hops more than the backup path of $STMB$), then $\Delta_{S,D} = 5$.

The “Arnes” network has 136 pair of source and destination nodes. Of these, 60% obtained $\Theta^+_{s,d} = \Theta^-_{s,d}$ and $\Delta_{s,d} = 0$, that is, the working and backup paths defined in $STMB$ have the same length as the working and backup paths defined in $STLB$.

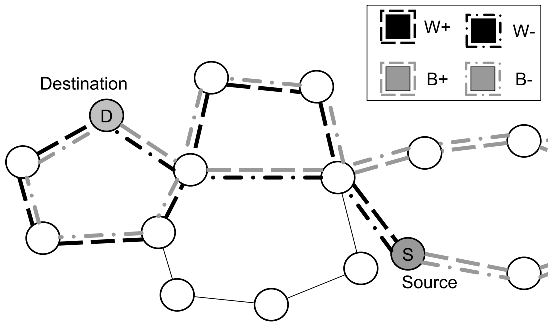


Figure 5. Working and backup paths defined by the $STMB$ and $STLB$ algorithms in the "Arnes" network for the node pair of source node S and destination node D.

Figure 6 shows the results of $\Theta^{+}_{s,d}$ and $\Theta^{-}_{s,d}$ for the other 40% of pair of source and destination nodes of "Arnes" network. These results are normalized according to C, i.e., for each pair (s, d), 100% corresponds to the sum of the respective working and backup path lengths. For instance, the $\Theta^{-}_{S,D} = 10$ mentioned earlier corresponds to 63% relatively to $C = 16$. Large differences between working and backup path lengths lead to large values of those variables. On the other hand, those variables are minimized when the working and backup paths have the same length. Also, $\Theta^{-}_{s,d} \geq \Theta^{+}_{s,d}$, as expected.

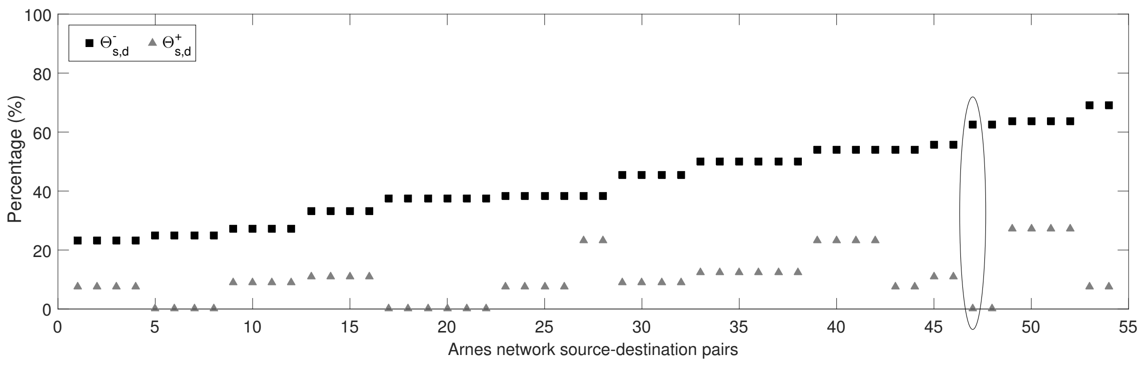


Figure 6. Difference between working and backup path lengths of $STLB$ ($\Theta^{-}_{s,d}$) and $STMB$ ($\Theta^{+}_{s,d}$) of the "Arnes" network. The horizontal axis represents pairs of source and destination nodes for which there is a difference in path lengths between the two strategies. The pair is shown in Figure 5 is circled. The pairs are organized according to $\Theta^{-}_{s,d}$ in ascending order.

The sum of all working path lengths of the $STMB$ and the $STLB$ for "Arnes" network give $\sum W^{+} = 530$ and $\sum W^{-} = 412$, respectively. It results in a total variation $\sum \Delta_{s,d} = 118$, i.e., the two algorithms differ by 29% relatively to the total working path length. Relatively to $\sum C$ for all node pairs, that difference reaches 9% (see Fig. 7).

To compare differences between working and backup paths of $STMB$ and $STLB$ among the 40 networks, first it was computed for each network: $\sum W^{+}$, $\sum W^{-}$, $\sum B^{+}$, $\sum B^{-}$. Then, $\sum \Theta^{+}_{s,d}$ and $\sum \Theta^{-}_{s,d}$ were obtained. The result is shown in Figure 7 where, for each network, 100% corresponds to $\sum C$ for all node pairs. Notice that $\sum \Theta^{-}_{s,d} \geq \sum \Theta^{+}_{s,d}$, as expected.

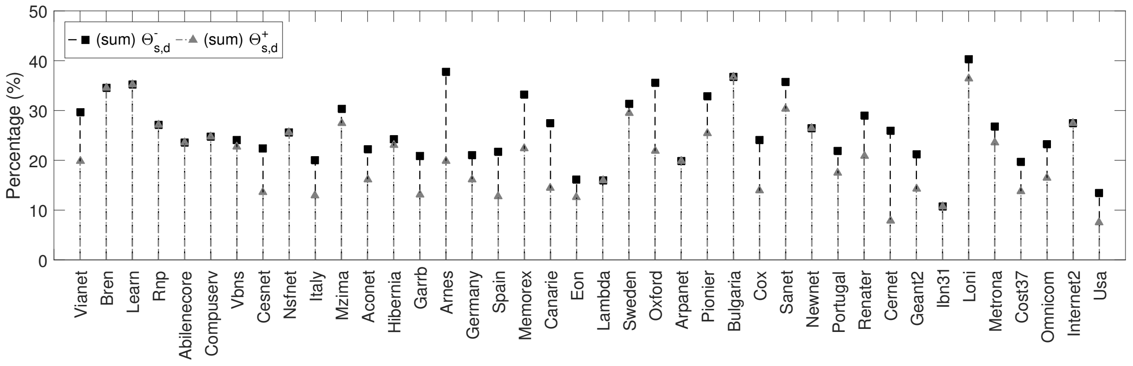


Figure 7. Difference between the sum of working path lengths and the sum of backup path lengths in $STLB$ ($\sum \Theta^{-}_{s,d}$) and $STMB$ ($\sum \Theta^{+}_{s,d}$). The horizontal axis represents 40 analyzed networks. The networks are organized by their number of nodes in ascending order.

In 12 networks, the working and backup paths defined for the $STLB$ were the same as for the $STMB$ for each pair of source and destination nodes. These networks have $\sum \Theta^{+}_{s,d} = \sum \Theta^{-}_{s,d}$. Therefore, there is no difference when using $STMB$ and $STLB$ for 30% of the analyzed networks.

Among networks for which $\sum \Theta^{+}_{s,d} \neq \sum \Theta^{-}_{s,d}$, the "Loni" network has the greatest difference in the lengths of working and backup paths in both algorithms, with $\sum \Theta^{-}_{s,d} = 40\%$ and $\sum \Theta^{+}_{s,d} = 36\%$, percentages relative to $\sum C$. The smallest difference in $STLB$ and in $STMB$ was from the "USA" network, $\sum \Theta^{-}_{s,d} = 13\%$ and $\sum \Theta^{+}_{s,d} = 8\%$ relative to $\sum C$, meaning that this network has pairs with the least difference in length of working and backup paths.

The difference between the algorithms was obtained through the difference of working and backup paths of the $STMB$ with the differences of working and backup paths of the $STLB$. Then, the percentages shown in Fig. 7 are twice the difference between the two strategies, that is, 2Δ. The "Arnes" and "Cernet" networks obtained the greatest difference with $\sum \Theta^{-}_{s,d} - \sum \Theta^{+}_{s,d} = 18\%$, that is, $\Delta = 9\%$. On average for all networks the difference was 5%.

To verify the impact on the number of transponders when using these algorithms, we used Eqs. (1) and (2). Figure 8 shows the normalized difference between the number of transponders of the working paths computed by $STMB$ and $STLB$ for all networks. Among the 40 networks, the "Arnes" network uses up to 29% more transponders in $STMB$ than in $STLB$. This result confirms the expected difference of the algorithms presented above. On average for all networks, using $STMB$ would require 7% more transponders.

To identify the amount of additional capacity required to protect the network, the protection coefficient of each network was computed using Eq. (3). In Figure 9 one can see that the "Loni" network needs up to 235% extra capacity to ensure survivability against link failures when using $STLB$ and 214% using $STMB$. This case is the maximum amount of extra capacity required for the set of 40 networks excluding those that have the same lengths of working and backup paths in $STMB$ and $STLB$, that is, con-

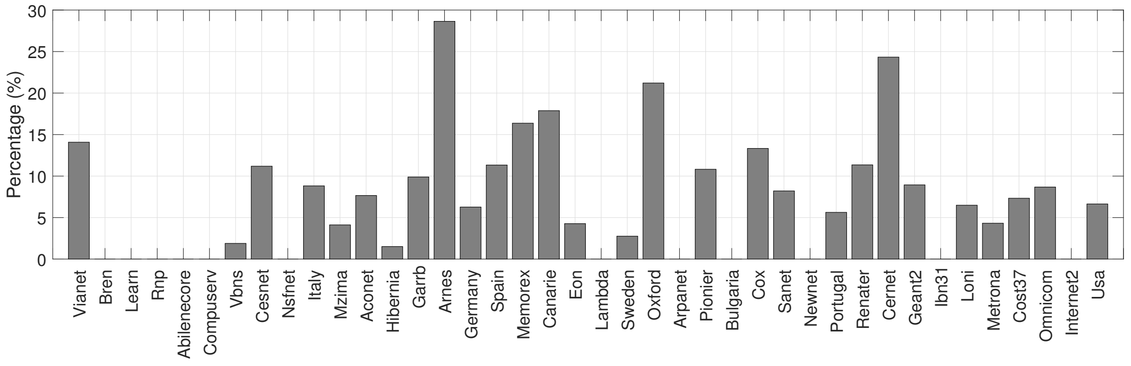


Figure 8. Normalized difference between the number of transponders used in working paths in $STMB$ and $STLB$ for all networks. The networks are organized by the number of nodes in ascending order.

sidering 28 networks. The minimum required extra capacity is 131% in $STLB$ and 116% in $STMB$, these results are both from the "USA" network. On average for all networks, 173% is needed in $STLB$ and 155% in $STMB$. The greatest difference between k_p of $STLB$ and of k_p of $STMB$ is in the "Arnes" network, for which the $STLB$ requires 71% extra capacity to protect the network. The k_p in $STLB$ presents results equal to or greater than the k_p in $STMB$, meaning that the $STLB$ will need more extra capacity in the network.

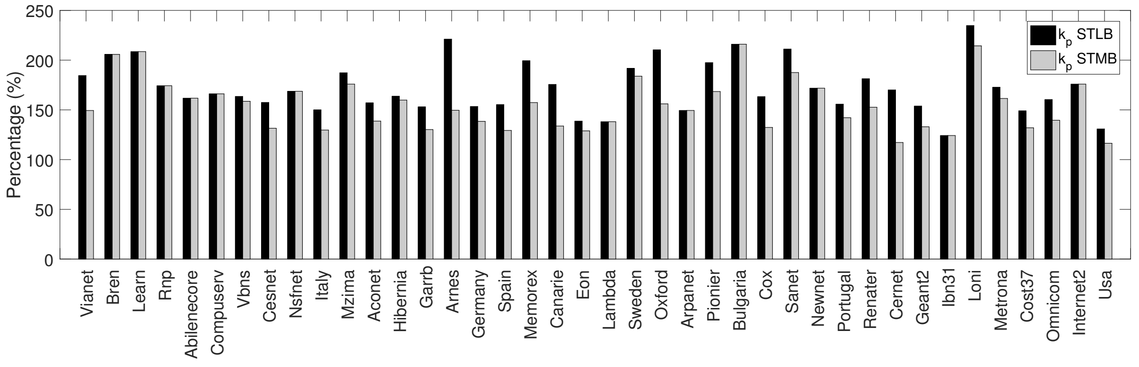


Figure 9. Protection coefficient (k_p) for each network using $STLB$ and $STMB$ algorithms.

These results imply a significant impact on network costs. Because it is obtained a reduction of up to 22% in transponders costs when using $STLB$, and a reduction of up to 32% in extra protection capacity when using $STMB$.

6. Conclusion

We have identified that using the *Suurballe and Tarjan's* (ST) algorithm for a source and destination node pair, a diversity of pairs of paths of the same cost can be found and it is not possible to choose one of these pairs. Therefore, two versions of the ST algorithm were proposed and analyzed. The *Suurballe and Tarjan Most Balanced* ($STMB$) version finds the most balanced shortest pairs of working and backup paths whereas *Suurballe and Tarjan Least Balanced* ($STLB$) finds the least balanced combination. Using $STMB$

and $STLB$, it is possible to take advantage of that diversity since allowing the choice of working and backup paths length, making it possible to explore different protection strategies for applications in different scenarios.

The proposed algorithms were tested in a set of 40 real-world optical telecommunication networks. For each pair of source and destination nodes, the differences between working and backup paths of $STMB$ and $STLB$ were explored in each network. Subsequently, the sum of those differences for all node pairs was considered.

Our results show that, even when restricted to the use of shortest pairs of edge-disjoint paths, there is a considerable degree of freedom in choosing them. For instance, the differences between working and backup path lengths for a pair of source and destination nodes of “Arnes” network reach 63% in $STLB$ (percentage relative to the sum of working and backup path lengths).

Differences between $STMB$ and $STLB$ were observed for 70% of the analyzed networks. Considering the sum of those results for all node pairs, a difference of up to 9% is found for “Arnes” and “Cernet” networks (that percentage is relative to the sum of working and backup path lengths for all node pairs).

This observed difference impacts the protection coefficient and the number of transponders required by dedicated-path protection schemes. The “Arnes” network needs up to 29% more transponders when using $STMB$ and requires 71% more protection capacity when using $STLB$. The network that most needs extra capacity in both algorithms is the “Loni” network, which needs 235% extra capacity when using $STLB$ and 214% using $STMB$.

The proposed versions can be used on different types of service. If the need is for paths of the same length or of lengths as close as possible, the best strategy is to use the $STMB$. This strategy will achieve a reduction of up to 32% in extra protection capacity. However, it may require more transponders. If the need is for working paths as short as possible, the most appropriate strategy is the $STLB$. This strategy will achieve a reduction of up to 22% in transponders costs. Nevertheless, the extra capacity to protect the network in case of failure may be higher.

For future work, protection schemes based on $STMB$ and $STLB$ will be analyzed. Also, it will be investigated which topological characteristics lead to a more significant difference between the two versions.

Acknowledgment

This work was partially supported by CAPES, CNPq, and FAPES.

References

Berger, L., Bryskin, I., Farrel, A., and Papadimitriou, D. (2007). GMPLS segment recovery.

Grover, W. D. (2003). *Mesh-based survivable transport networks: options and strategies for optical, MPLS, SONET and ATM networking*. Prentice Hall PTR.

Korotky, S. K. (2004). Network global expectation model: A statistical formalism for quickly quantifying network needs and costs. *Journal of Lightwave Technology*, 22(3):703.

Labourdette, J.-F., Bouillet, E., Ramamurthy, R., and Akyamaç, A. A. (2005). Fast approximate dimensioning and performance analysis of mesh optical networks. *IEEE/ACM Transactions on Networking (TON)*, 13(4):906–917.

Lang, J., Rekhter, Y., and Papadimitriou, D. (2007). RSVP-TE extensions in support of end-to-end generalized multi-protocol label switching (GMPLS) recovery. Technical report.

Manchester, J., Bonenfant, P., and Newton, C. (1999). The evolution of transport network survivability. *IEEE Communications Magazine*, 37(8):44–51.

Nykänen, M. and Ukkonen, E. (2002). The exact path length problem. *Journal of Algorithms*, 42(1):41–53.

Oliveira, J. M. S. d. S. (2010). Protecção maxima de redes de telecomunicações. Mestrado, Universidade de Aveiro.

Pavan, C., de Lima, L., Paiva, M., and Segatto, M. (2015). How reliable are the real-world optical transport networks? *Journal of Optical Communications and Networking*, 7(6):578–585.

Pinto, A. N. (2014). Reference networks. `www.av.it.pt/anp/on/refnet2.html`.

Ramaswami, R., Sivarajan, K., and Sasaki, G. (2009). *Optical networks: a practical perspective*. Morgan Kaufmann.

Routray, S. K., Morais, R. M., da Rocha, J. R. F., and Pinto, A. N. (2013). Statistical model for link lengths in optical transport networks. *Journal of Optical Communications and Networking*, 5(7):762–773.

Schrijver, A. (2003). *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media.

Shen, G., Guo, H., and Bose, S. K. (2016). Survivable elastic optical networks: survey and perspective. *Photonic Network Communications*, 31(1):71–87.

Suurballe, J. and Tarjan, R. E. (1984). A quick method for finding shortest pairs of disjoint paths. *Networks*, 14(2):325–336.

Tessinari, R. S., Puype, B., Colle, D., and Garcia, A. S. (2016). ElasticO++: An elastic optical network simulation framework for OMNeT++. *Optical Switching and Networking*, 22:95–104.

Whitney, H. (1932). Congruent graphs and the connectivity of graphs. *American Journal of Mathematics*, 54(1):150–168.

Williams, R. (2009). Finding paths of length k in O*(2k) time. *Information Processing Letters*, 109(6):315–318.

Yen, J. Y. (1970). An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, 27(4):526–530.

Zang, H. (2012). WDM mesh networks: management and survivability. Springer Science & Business Media.