Paper

An Algorithm for Enumerating SRLG Diverse Path Pairs

Teresa Gomes and José Craveirinha

Abstract—Telecommunication networks are intrinsically multi-layered, a single failure at a lower level usually corresponds to a multi-failure scenario at an upper layer. In this context, the concept of shared risk link group (SRLG) allows an upper layer to select, for a given active path (AP), a backup path (BP), which avoids every SRLG that may involve the selected AP, in the event of a failure. That is a SRLG diverse path set maybe defined as a set of paths, between an origin and a destination, such that no pair of paths can be simultaneously affected by any given failure (or risk) in a single failure scenario. Firstly we present the formulation of the SRLG diverse path pair calculation problem in a directed network. An algorithm for enumerating SRLG diverse paths, by non decreasing cost of their total (additive) cost will be presented, which is based on an algorithm proposed for generating minimal cost node disjoint path pairs. The SRLG diverse path pairs may be node or arc disjoint, with or without length constraints. Computational results will be presented to show the efficiency of the proposed algorithm for obtaining node or arc disjoint SRLG diverse path pairs in undirected networks.

Keywords—routing, SRLG disjoint shortest paths, telecommunication networks.

1. Introduction

Bandwidth usage optimization is one of the main issues when protection schemes are used in telecommunication networks. In global path protection, the path that carries the associated traffic flow under normal operating conditions is called the active path (AP), and the path that carries that traffic when some failure affects the AP is called the backup path (BP).

Many network providers consider sufficient to implement protection schemes which ensure their network (or certain connections in their network) is 100% reliable in single failure scenarios. Because telecommunication networks are intrinsically multi-layered, a single failure at a lower level usually corresponds to a multi-failure scenario at an upper layer.

A failure risk may represent a fibre cut, a card failure at a node, a software failure, or any combination of these factors [1], which may affect one or more links at a given network layer. In this context, the concept of shared risk link group (SRLG) is very important in teletraffic engineering since allows an upper layer to select, for a given AP, a BP, which avoids every SRLG that may involve the selected AP, in the event of a failure. Note that this may not be feasible for all possible APs. That is a SRLG diverse path set maybe defined as a set o paths, between an origin and a destination, such that no pair of paths can be simultaneously affected by any given failure (or risk) in a single failure scenario. Therefore, to ensure global path protection against a single failure affecting a single SRLG, a SRLG diverse path pair must be calculated.

The problem of finding a SRLG diverse path pair has been shown to be NP-complete [1]. The minimum-cost diverse routing problem, in which the objective is finding two paths, SRLG diverse, with minimal total arc cost (also designated as the min-sum problem), is also NP-complete [1]. Hu [1] proposed an integer linear programming (ILP) formulation for the min-sum problem, and provide numerical results showing that the ILP formulation quite effective in networks with a few hundreds of nodes.

The necessity of calculating SRLG diverse path pairs arises in optical and multiprotocol label switching (MPLS) networks, where certain connections require two paths, the AP and the BP, in order to satisfy service level agreements (SLA) regarding reliability. The possibility of enumerating, by non-decreasing cost, SRLG diverse path pairs, may allow more elaborate, and possibly more efficient, forms of SRLG diverse routing. Furthermore the ordered enumeration of diverse SRLG paths will make it possible a multi-objective routing approach, with survivability requirements.

Rostami *et al*. [2] proposed an algorithm, named CoSE (conflicting SRLG exclusion), which is an extension to SRLG-disjoint routing of a link-disjoint routing algorithm called CoLE (conflicting link exclusion), proposed in [3], which can quickly find an optimal solution path pair. The CoSE algorithm iteratively separates the network SRLGs into two sets and then computes the working and backup paths. Furthermore, in [2] the authors also propose a way of calculating two maximally SRLG diverse paths in a network where no two completely-disjoint paths exist. The CoSE algorithm can be used for solving the min-min problem, by selecting the appropriate solution from the set of generated solutions (although the optimality of the solution is not guaranteed).

Todimala and Ramamurthy [4] proposed an iterative heuristic, based on a modification of Suurballe's algorithm [5], [6], for diverse routing under SRLG constraints that computes the least cost SRLG diverse paths pair. In [7] the same authors propose a heuristic for solving the problem of computing optimal SRLG/link diverse paths under shared protection (considering the definition of an optimal SRLG diverse path pair under shared protection as asymmetrically-weighted [8]).

In [9], [10] the authors consider the problem of path protection in wavelength-routed networks with SRLG and propose a heuristic method, which they compare with the trap avoidance (TA) algorithm [11]. They conclude the new algorithm, minimum total weight (MTW) algorithm, outperforms TA algorithm within the first few iterations. If more iterations are considered there is no clear advantage of one algorithm, over the other.

This work presents an exact algorithm for enumerating SRLG diverse path pairs in a multi-layered network by decreasing order of the total cost. Firstly we present the formulation of the SRLG diverse path pair calculation problem in a directed (implicitly multi-layered) network. Secondly we formalize the proposed algorithm, that is based on Algorithm 1 proposed in [12] for generating minimal cost node disjoint path pairs. The SRLG diverse path pairs may be node or arc disjoint, and with or without length constraints, as will be explained. Finally, computational results will be presented to show the efficiency of the proposed algorithm for obtaining node disjoint SRLG diverse path pairs.

The paper is organized as follows. In Section 2 the notation and the problem formulation are given. An algorithm for enumerating SRLG diverse paths, by non decreasing cost of their total (additive) cost is presented in Section 3. The application to path pairs node or arc disjoint and with length constraints, is briefly explained in Subsections 3.3 and 3.4. In Section 4 results which illustrate the algorithm efficiency in obtaining SRLG node disjoint path pairs, are presented. Finally, some conclusions are presented in Section 5.

2. Notation and Problem Definition

The algorithm in Section 3 is based on Algorithm 1 in [12]. Therefore, we will use a notation similar to the one in [12]. Let $G = (N,A)$ be a directed network with node set $N = \{v_1, v_2, \dots, v_n\}$ and arc set $A = \{a_1, a_2, \dots, a_m\}$ (where n and m designate the number of nodes and arcs in G, respectively). Let a non-negative cost function (or metric) in the arcs, be defined:

$$c_{v_a v_b} \geq 0, \quad (v_a, v_b) \in A\,, \tag{1}$$

where $c_{v_a v_b}$ represents the cost of using arc (v_a, v_b).

The cost $c(p)$ of a path p in G with respect to metric c is:

$$c(p) = \sum_{(v_a, v_b) \in p} c_{v_a v_b}\,. \tag{2}$$

Definition 1: A path p is said to be simple (or loopless) if all its nodes are different.

We will use the word path to refer to simple paths, and we will only use the expression "simple path" when required, namely in the algorithm.

Let path $p = \langle v_1, a_1, v_2, \dots, v_{i-1}, a_{i-1}, v_i \rangle$, be given as an alternate sequence of nodes and arcs of G, such that the tail of a_k is v_k and the head of a_k is v_{k+1}, for $k = 1, 2, \dots, i-1$ (all the v_i in p are different). Let the set of nodes in p be $V^*(p)$ and the set of arcs in p be $A^*(p)$. Two paths $p = \langle v_1, a_1, v_2, \dots, v_{i-1}, a_{i-1}, v_i \rangle$ and q are arc-disjoint if $A^*(p) \cap A^*(q) = \emptyset$. Two paths p and q are disjoint if $V^*(p) \cap V^*(q) = \emptyset$, and are internally disjoint [13] if $\{v_2, \dots, v_{i-1}\} \cap V^*(q) = \emptyset$. We will say that two paths are node disjoint if they are internally disjoint.

Let R be a set representing the risks (failures) in the functional network. Each risk may correspond to a fibre cut, a card failure at a node, a software failure, or any combination of these factors. Let A_r represent the subset of network arcs (or links) in the network logical representation (corresponding to a capacitated graph) that can be affected by risk $r \in R$. Thence A_r is a SRLG (associated with r).

Let

$$r_p = \{r \in R:\ \text{path } p \text{ contains elements of } A_r\}\,. \tag{3}$$

The SRLG problem can be defined as follows [1].

Definition 2: Find two paths p and q, between a pair of nodes, such that $r_p \cap r_q = \emptyset$. We also say that p and q are two SRLG diverse paths (with respect to R).

The first addressed problem is to enumerate node disjoint simple path pairs (p_i, q_i) $(i = 1, 2, \dots)$, in G, from a source s to a destination node t $(s \neq t)$, which are SRLG diverse, by non-decreasing total cost of the pair, defined by

$$c[(p_i, q_i)] = c(p_i) + c(q_i), \quad i = 1, 2, \dots\,, \tag{4}$$

where p_i and q_i have the same source and sink node.

Let R_a be the set of risks that can affect arc $a \in A$:

$$R_a = \{r : a \in A_r\}, \quad \forall a \in A\,, \tag{5}$$

R_a can be obtained from A_r $(r = 1, \dots, |R|)$ and

$$r_p = \cup_{a \in p} R_a\,, \tag{6}$$

which is much more adequate for generating SRLG diverse paths in the proposed algorithm.

If a path pair (p, q) is SRLG diverse then it is arc disjoint (regardless of whether the the network is directed or not).

Definition 3: Two arcs, $a_i, a_j \in A$ are SRLG diverse if $R_{a_i} \cap R_{a_j} = \emptyset$.

Definition 4: An arc $a \in A$ is SRLG diverse with a path p if $R_a \cap r_p = \emptyset$.

The algorithm proposed in Section 3 is based on Algorithm 1 in [12], which uses the MPS algorithm [14] in its loopless version [15]. The algorithm MPS is a *deviation* algorithm. Each time a path p is chosen from a set of candidate paths, X, new paths may be added to X. In the context of the algorithm the node v_k of path p, from which a new candidate path is generated, is the *deviation node* of that new path (which coincides with p up to v_k). In a path the link the tail of which is the deviation node, is called the *deviation arc* of that path [14]. By definition s is the deviation node of p_1 (the shortest path from s to t). The *concatenation* of path p, from v_i to v_j, with path q, from v_j to v_l, is the path $p \diamond q$, from v_i to v_l, which coincides with p from v_i to v_j and with q from v_j to v_l.

Let $\mathcal{T}_t$ designate a tree where there is a unique path from any node v_i to t (tree rooted at t as defined in [14]) and

π_{v_i} denote the cost of the path p, from v_i to t, in $\mathcal{T}_t$; the *reduced cost* $\bar{c}_{v_i v_j}$ of arc $(v_i, v_j) \in A$ associated with $\mathcal{T}_t$ is $\bar{c}_{v_i v_j} = \pi_{v_j} - \pi_{v_i} + c_{v_i v_j}$. So all arcs in $\mathcal{T}_t$ have a null reduced cost. The reduced cost of path p is given by $\sum_{(v_i,v_j)\in p} \bar{c}_{v_i v_j}$ and it can be proved that $c(p) = \bar{c}(p) + \pi_s$. The advantage of using reduced costs was first noted by Eppstein [16] and they are shown by Theorems 8 and 9 in [14] and by Theorem 2.1 in [15] (in the context of the MPS algorithm) to lead to less arithmetic operations and to sub-path generation simplification.
Let $\mathcal{T}_t^*$ be the tree of the shortest paths from all nodes to t and $\mathcal{T}_t^*(v_j)$ the shortest path from v_j to t in $\mathcal{T}_t^*$ (hence $\pi_{v_i} = c[\mathcal{T}_t^*(v_j)]$). The sub-path from v_k to v_j in p is represented by $\text{sub}_p(v_k, v_j)$. The set of arcs of A of $G = (N, A)$ is arranged in the *sorted forward star form* – for details, see [17]. That is, the set A is sorted in such a way that, for any two arcs $(v_i, v_j), (v_k, v_l) \in A$, $(v_i, v_j) < (v_k, v_l)$ if $v_i < v_k$ or ($v_i = v_k$ and $\bar{c}_{v_i v_j} \le \bar{c}_{v_k v_l}$).

3. Node Disjoint and SRLG Diverse Path Pairs

The algorithm is based on the Algorithm 1 in [12] for enumerating node disjoint path pairs, by non-increasing total additive cost which requires a network topology transformation as described in the next subsection.

3.1. Network Topology Modification

Let s, t be a source and destination in G. Let P_{xy} be the set paths (loopless or not) from node x to node y in G. Let $G' = (N', A')$ be a transformed network where, such that [12]:

- the former nodes are duplicated: $N' = N \cup \{v_i' : v_i \in N\}$;
- the former arcs are duplicated, and a new one, linking t and the new node s', is added: $A' = A \cup \{a' = (v_a', v_b') : a = (v_a, v_b) \in A\} \cup \{(t, s')\}$;
- $c(v_a', v_b') = c(v_a, v_b), \quad \forall (v_a, v_b) \in A$;
- $c(t, s') = 0$;
- $R_{a'} = R_a, \quad \forall a, a' \in A'$.

In this new network the source node is s and the destination node is t'. Each path from s to t' in G' is such that:

$$p = q \diamond (t, s') \diamond q', \tag{7}$$

where $q \in P_{st}$ and $q' \in P_{s't'}$. If q and q' are simple and do not share corresponding nodes in N and N' (except s, s' and t, t') then they are disjoint simple paths. If, additionally, $R_q \cap R_{q'} = \emptyset$, then q and q' are SRLG diverse.
Let $\mathcal{T}_{t'}^*$ be the tree of the shortest paths from all nodes to t', in G' (the modified graph). If $\mathcal{T}_t^*$ is calculated before transforming the network, then $\mathcal{T}_{t'}^*$ can easily be obtained. This process of building $\mathcal{T}_{t'}^*$ ensures that $\mathcal{T}_{t'}^*(s) = p \diamond (t, s') \diamond p'$, where p and p' correspond to the same path. In the transformed network, $\pi_{v_i'} = c(\mathcal{T}_t^*(v_i)), \forall v_i' \in N' \backslash N$ and $\pi_{v_i} = \pi_{v_i'} + \pi_{s'}$, for any $v_i \in N$ [12][1].
In Remark 1 of [12] it is suggested that there is no need to explicitly represent the new arcs in G' except the new arc (t, s'), because every new arc is a copy of another existing arc, and $\bar{c}_{v_i' v_j'} = \bar{c}_{v_i v_j}$. However, implementing Remark 1 is only feasible if $\mathcal{T}_{t'}^*$ is built as described in the previous paragraph – a fact which is not pointed out in [12].
If at least two different paths, p and q, with the same minimal cost exist from v_i to t, (with the successor of v_i in p different from the successor of v_i in q), then, using Dijkstra's algorithm in G' for calculating $\mathcal{T}_{t'}^*$, we may obtain $\mathcal{T}_{t'}^*(v_i) = (v_i, v_j) \diamond \mathcal{T}_{t'}^*(v_j)$ and $\mathcal{T}_{t'}^*(v_i') = (v_i', v_k') \diamond \mathcal{T}_{t'}^*(v_k')$, with $v_j \ne v_k$ (and $v_j' \ne v_k'$). When this happens, two different arcs with the "same tail", v_i and v_i', will belong to $\mathcal{T}_{t'}^*$, and when building the sorted forward star form of the arcs $A \cap (t, s')$, both arcs must be the first arc with tail v_i, which is not possible! This detail is very important because the MPS algorithm [14], which is the base of Algorithm 1 in [12], requires the ordering of the arcs in the ordered forward star form, such that the first arc with tail v_i (equivalent to v_i') $\forall v_i \in A$, belongs to $\mathcal{T}_{t'}^*$, in order to be able to generate every path by non-decreasing order of its cost.

3.2. The Algorithm

A infeasibility test can be made at the very beginning of the algorithm:

- if we can not find at least two arcs with tail node s, which are SRLG diverse, then there is no solution;
- if we can not find at least two arcs with head node t, which are SRLG diverse, then there is no solution.

If this infeasibility test fails, then we can proceed to try and find SRLG diverse path pairs.
In order to speed up path generation, the network should be pruned of the arcs with tail s and head t such that no SRLG diverse paths can be obtained if they belong to any of the paths. We will say that the remaining arcs of tail s and head t can be SRLG protected (by at least another arc of tail s or head t, respectively). These arcs can be identified during the infeasibility test and removed from the network[2] before running the Dijkstra algorithm for obtaining $T_{t'}$.
A path, p, obtained in the augmented network (see Subsection 3.1 or [12; Subsection 3.2]), is made of $q \diamond (t, s') \diamond q'$ and we assume it has deviation node d_p, deviation arc a_h,

[1]In [12], where is $\pi_i = \pi_{i'} + \pi_s$ should be $\pi_i = \pi_{i'} + \pi_{s'}$.
[2]In order to reduce the need for graph transformation, these arcs can be simply marked as useless, as long as an adequate Dijkstra's algorithm is implemented.

Algorithm 1: **Determination of the *K* shortest SRLG diverse simple path pairs**

```
**Data**: Network directed graph $G = (N,A)$ and a source
        destination node pair $(s,t)$, and $c$ cost of the links
**Result**: $S$, the set of the $K$ shortest SRLG diverse simple
        path pairs from $s$ to $t$
1  if *the infeasibility test is successfully* then Stop end
2  Remove from $A$ arcs emerging form $s$ or incident in $t$,
   which can not be SRLG protected. Remove from $A$ all
   arcs with tail node $t$
3  $\mathcal{T}^*_{t'} \leftarrow$ tree of the shortest paths from $i \in N'$ to $t'$ using $c$
4  $p \leftarrow \mathcal{T}^*_{t'}(s)$
5  if *$p$ is not defined* then Stop end
6  $\bar{c}_{v_i v j} \leftarrow \pi_{v_j} - \pi_{v_i} + c_{v_i v_j}, \quad \forall (v_i, v_j) \in A'$
7  Represent $A'$ in the sorted forward star form concerning $\bar{c}$
   // Consider: $p = (s \equiv v_1, v_2, ..., v_{y-1}, v_y \equiv t)$, $(s, v_2)$
   can be SRLG protected
8  $d_p \leftarrow s$ // Deviation node of $p$
9  $X \leftarrow \{p\}$
10 $S = \emptyset$
11 while $X \neq \emptyset \wedge |S| < K$ do
12     $p \leftarrow$ path in $X$ such that $\bar{c}(p)$ is minimum
13     if *($p$ is simple)* $\wedge$ Disjoint($p$) $\wedge$ SRLGDiverse($p$)
       then
14         $S \leftarrow S \cup \{p\}$
15     end
16     $X \leftarrow X \backslash \{p\}$
17     $i \leftarrow$ min index such that $v_i = d_p$
18     break $\leftarrow$ false // Candidate paths might be
           derived from $p$
19     repeat
20         $l \leftarrow$ index such that $a_l = (v_i, v_{i+1})$
21         repeat
22             $l \leftarrow l + 1$
23             $v_j \leftarrow$ head node of $a_l$ // if $l > m+1$ then
                   $v_j \leftarrow 0$
24             if *($v_i$ is the tail node of $a_l$)*$\wedge$
               EquivalentPair(sub$_p(s, v_i) \diamond a_l \diamond \mathcal{T}^*_{t'}(v_j)$)
               then
25                 break $\leftarrow$ true // No candidate paths
                       will derive from $p$ at $v_i$
26             end
27         until *break* $\vee$(*$v_i$ is not the tail node of $a_l$*)$\vee$ [(*$a_l$
           does not form a loop with* sub$_p(s, v_i)$)$\wedge$
           SRLGDiverse(sub$_p(s, v_i) \diamond a_l$) $\wedge$
           Disjoint(sub$_p(s, v_i) \diamond a_l$)]
28         if ($\neg$ *break* )$\wedge$(*$v_i$ is the tail node of $a_l$*) then
29             $q \leftarrow$ sub$_p(s, v_i) \diamond a_l \diamond \mathcal{T}^*_{t'}(v_j)$; $d_q \leftarrow v_i$
30             $X \leftarrow X \cup \{q\}$
31         end
32         $v_i \leftarrow v_{i+1}$ // Next node of $p$
33     until ($v_i = t'$) $\vee$ $\neg$(sub$_p(s, v_i)$ *is simple*) $\vee$
       $\neg$ Disjoint(sub$_p(s, v_i)$) $\vee$
       $\neg$ SRLGDiverse(sub$_p(s, v_i)$)
34 end
```

and that the first arc in q is a_f (where $a_f = a_h$ if the deviation node is s). Paths will only be placed in the set of candidate paths if:

- the deviation node, d_p, belongs to N and the path sub$_p(s, d_p) \diamond a_h$ is simple;
- the deviation node, d_p, belongs to $N' \backslash N$:
  - the path sub$_p(s', d_p) \diamond a_h$ is simple;
  - $c(\text{sub}_p(s,t)) \geq c(\text{sub}_p(s',t'))$ (or $c(q) \geq c(q')$); note that $c(q') = c(\text{sub}_p(s', d_p) \diamond a_h)$, and that $c(q') = c(p) - c(q)$;
  - the paths sub$_p(s,t)$ and sub$_p(s', d_p) \diamond (a_h)$ are node-disjoint;
  - a_h is SRLG diverse with sub$_p(s,t)$.

In Algorithm 1 we chose to remove from the network graph arcs which are not useful for obtaining SRLG diverse path pairs. This is not strictly necessary, but improves the algorithm efficiency.

Note that in set X all paths are simple, disjoint and SRLG diverse up to and including the deviation arc. Due to this fact we have replaced all the interior **while** cycles of Algorithm 1 [12] with **repeat until** cycles.

Function **Disjoint**(p), $p = q \diamond (t, s') \diamond q'$, returns true if q and q' are node disjoint. Function **SRLGDiverse**(p) returns true if q and q' are SRLG diverse. At Steps 27 and 33 the value of functions **Disjoint()** and **SRLGDiverse()** is true whenever v_i belongs to N. This implies that the evaluation of disjointness or SRLG diverseness is only effectively required at Steps 27 and 33 of the algorithm when the deviation node belongs to $N' \backslash N$. Also note that for the calculation of **SRLGDiverse**(sub$_p(s, v_i) \diamond a_l$), in Step 27, it is sufficient to evaluate if sub$_p(s, v_i)$ is SRLG diverse with arc a_l.

Function **EquivalentPair()** was first introduced in [12], for including Remark 2 in Algorithm 1. Due to Remark 2 in [12] we may choose to store paths pairs that $\bar{c}[\text{sub}_p(s,t)] \leq \bar{c}[\text{sub}_p(s',t')]$ or $\bar{c}[\text{sub}_p(s,t)] \geq \bar{c}[\text{sub}_p(s',t')]$. If we choose to store in X paths q such that $\bar{c}[\text{sub}_q(s,t)] \geq \bar{c}[\text{sub}_q(s',t')]$, then function **EquivalentPair()** will only be required when v_i belongs to $N' \backslash N$ – that is Step 24 could be rewritten:

$(v_i \in N' \backslash N) \wedge (v_i$ is the tail node of $a_l) \wedge$
EquivalentPair(sub$_p(s, v_i) \diamond a_l \diamond \mathcal{T}^*_{t'}$(head node of a_l)).

Function **EquivalentPair**(p) returns true whenever $\bar{c}[\text{sub}_p(s,t)] < \bar{c}[\text{sub}_p(s',t')]$. Consider that v_i belongs to $N' \backslash N$ and let $q = \text{sub}_p(s, v_i) \diamond a_l \diamond \mathcal{T}^*_{t'}$(head node of a_l), in Step 24. In this case sub$_p(s,t) = $ sub$_q(s,t)$, therefore the execution of **EquivalentPair()** can be simply the evaluation of $\underbrace{\bar{c}[\text{sub}_p(s,t)]}_{\bar{c}[\text{sub}_q(s,t)]} < \underbrace{\bar{c}(q) - \bar{c}[\text{sub}_p(s,t)]}_{\bar{c}[\text{sub}_q(s',t')]}$.

The proposed algorithm requires a directed network graph. For obtaining SRLG diverse path pairs in an undirected network, we must build the equivalent directed graph: each undirected link is represented by two directed arcs, in opposite directions,with the same costs, belonging to the same SRLGs as the corresponding undirected link.

3.3. Link Disjoint SRLG Diverse Path Pairs

If the path pair does not need to be node disjoint, then the only modification required in Algorithm 1 is the suppression of the function **Disjoint()**, assuming each undirected link belongs to at least one SRLG.

3.4. SRLG Diverse Path Pairs With Length Constraints

Let $p = q \diamond (t,s') \diamond q'$, represent a path pair (q,q'). If the path pairs have length restrictions (maximum number of allowed arcs), then two new conditions must be evaluated: the depth of the deviation node $i \in q$ and $j' \in q'$ must be less than the length constraint (assuming node s has depth 0).

4. Computational Results

Two sets of experiments were made. The first set used:

- Randomly generated undirected networks with $n = 25, 50, 100, 200, 400$ and $m = 3n, 4n$ (where n is the number of nodes and m is the number of undirected arcs).
- The cost of each link was randomly generated in [1..65535].
- Each undirected arc was associated with a single SRLG.
- For each value of n and m ten randomly generated networks were considered.
- For each network 50 end-to-end node pairs, where selected and $K = 1000$ diverse path pairs were sought.

The second set of experiments considered:

- Randomly generated undirected networks with $n = 100, 1000$ and $m = 3n, 4n$ (where n is the number of nodes and m is the number of undirected arcs).
- The cost of each link was randomly generated in [1..65535].
- Each undirected arc was associated with a single SRLG.
- For each value of n and m ten randomly generated networks were considered.
- For each network 50 end-to-end node pairs, where selected and $K = 5000$ diverse path pairs were sought.

The computer used was a PC, Intel(R) Core(TM) 2 Duo Processor, 1.82 GHz, RAM 1 GB, under Kubuntu. The maximum number of allowed paths was 10^7.

Observing the average central processing unit (CPU) time per node pair, for obtaining $K = 1000$ path pairs, presented in Figs. 1 and 2, it may be concluded that it is more efficient for obtaining node disjoint than arc disjoint path pairs, as expected.

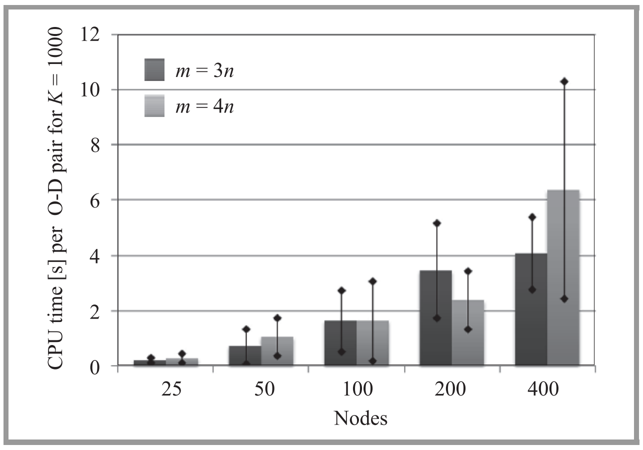

Fig. 1. CPU times for obtaining $K = 1000$ arc disjoint and SRLG diverse path pairs.

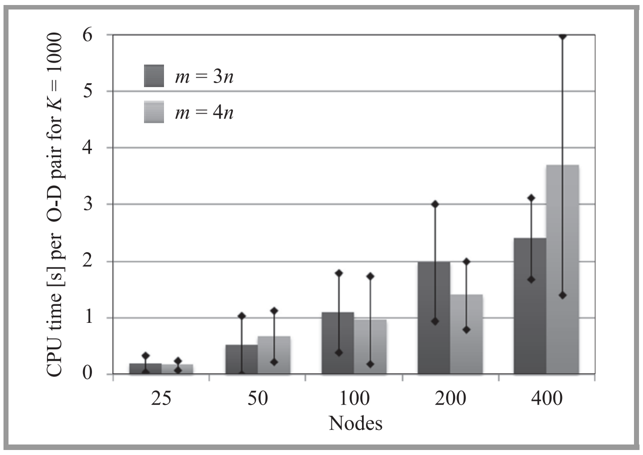

Fig. 2. CPU times for obtaining $K = 1000$ node disjoint and SRLG diverse path pairs.

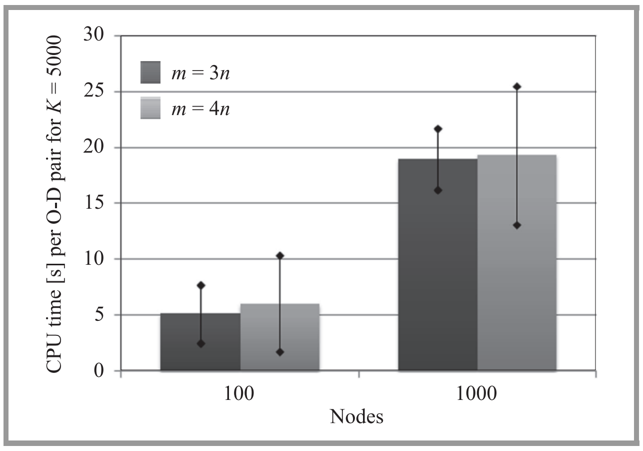

Fig. 3. CPU times for obtaining $K = 5000$ arc disjoint and SRLG diverse path pairs.

That statement is still true if $K = 500$ is used, as can be seen in Figs. 3 and 4. It should be noted that with $n = 100$ when K goes from 1000 to 5000, the CPU time grows proximately linearly with K. Also the CPU time with $n = 1000$, in Figs. 3 and 4, is less than 10 times the CPU time when $n = 100$.

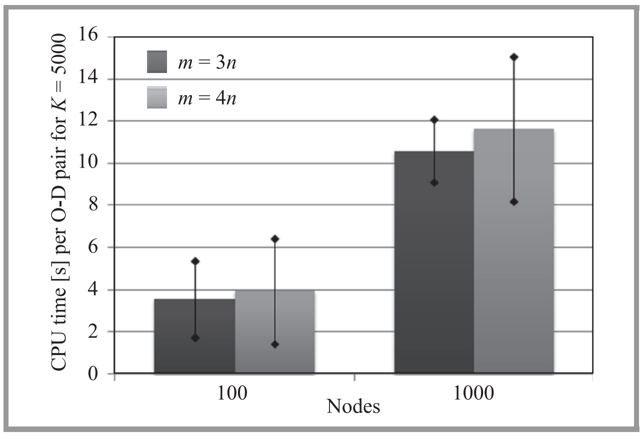

Fig. 4. CPU times for obtaining $K = 5000$ node disjoint and SRLG diverse path pairs.

The interval bars in the figures indicate the 95% confidence interval for the average CPU time.

The CPU time grows with the number of nodes, for the networks with the same average degree, and also tends to increase with the average node degree, for networks with the same number of nodes. In the cases where that does not happen, it was due to some(s) node pair(s) with a CPU quite above the average CPU time in one (or two) of the ten networks. The average CPU times presented in Figs. 1–4 were obtained for the node pairs for which the desired K (1000 or 5000) were obtained.

The CPU time is closely related to the total number of candidate paths that were generated and added to set X in the algorithm, as it can be seen in Figs. 5–8.

In some cases there is no solution (and in most cases this was discovered very fast due to the infeasibility test) or the maximum allowed number (10^7) of candidate path pairs was generated without obtaining the desired number K of SRLG disjoint path pairs – in this last case it is uncertain whether any more disjoint path pairs might have been obtained if the maximum allowed number of candidate paths was higher. In the experiments, we tried to obtain $K = 1000$ SRLG node disjoint path pairs for 5000 node pairs (5 different values of n, two different average node degrees, 10 different seeds for network generation and 50 node pairs per network) and failed to do it in 22 cases due to the fact that the maximum number of candidate paths was attained. This corresponds to a success rate of 99.56%. The results for arc and SRLG disjoint path pairs was similar: 99.54%. When $K = 1000$ the unsuccessful node pairs occurred only for $n = 200$ and $n = 400$ and the CPU times is approximately 2 minutes and 4 minutes, for the node and arc disjoint path pairs, respectively.

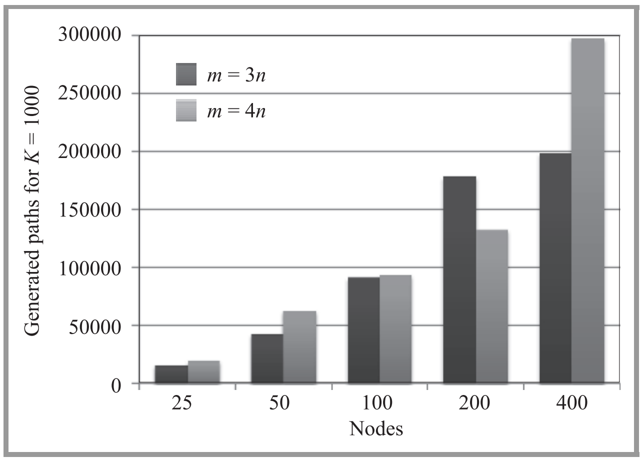

Fig. 5. Number of candidate path pairs added to X for obtaining $K = 1000$ arc disjoint and SRLG diverse path pairs.

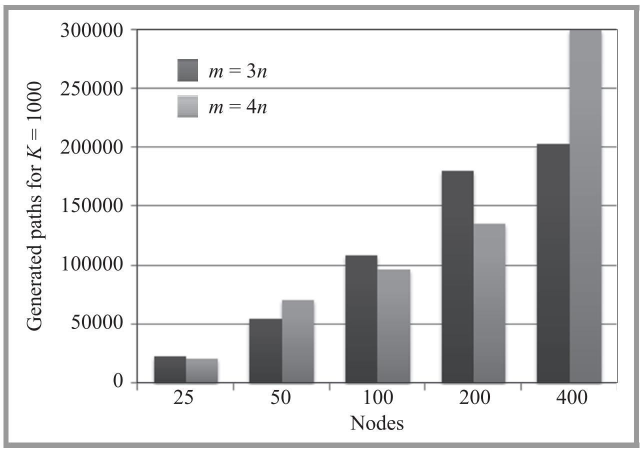

Fig. 6. Number of candidate path pairs added to X for obtaining $K = 1000$ node disjoint and SRLG diverse path pairs.

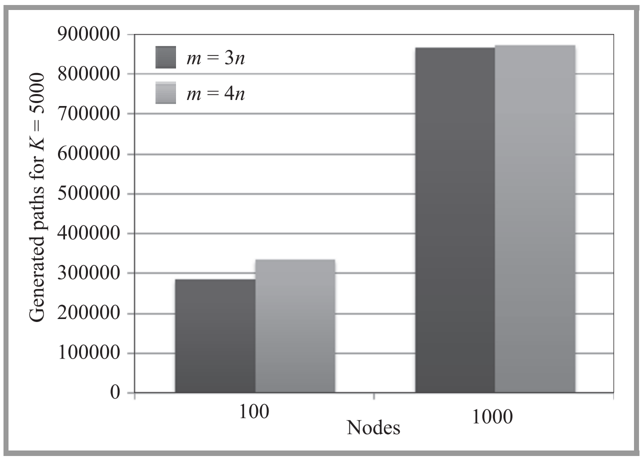

Fig. 7. Number of candidate path pairs added to X for obtaining $K = 5000$ arc disjoint and SRLG diverse path pairs.

When $K = 5000$ the rate of unsuccessful node pairs grows to 7.4% for $n = 1000$ but the CPU time remains similar to what was observed, for the unsuccessful cases when $K = 1000$.

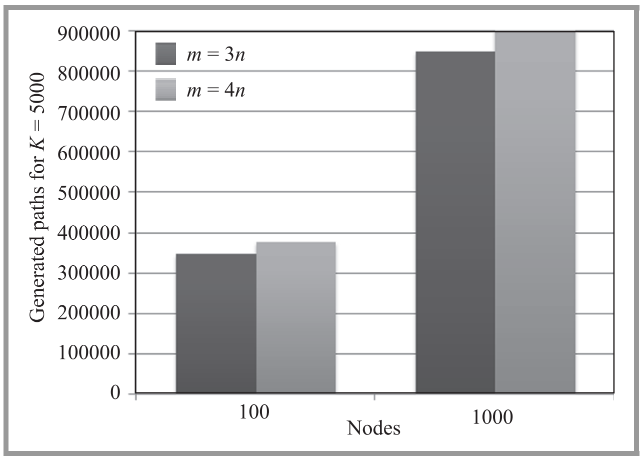

Fig. 8. Number of candidate path pairs added to X for obtaining $K = 5000$ node disjoint and SRLG diverse path pairs.

Note that in the case $n = 25, 50, 100, 200, 400$ the algorithm was able to detect that no solution could be found for thirteen node pairs in zero seconds. However, for $n = 1000$ there were two node pairs for which no solution was found even after generating 10^7 candidate path pairs, as it can be seen in Tables 1 and 2, in the line with "$k = 0$ (?)".

Table 1
Arc disjoint and SRLG diverse path pairs, for $m = 3n, 4n$: total number of node pairs and average CPU times when $K = 5000$ was not obtained

Arc disjoint	$m = 3n$		$m = 4n$	
n	100	1000	100	1000
$0 < k < 5000$	3	34	2	31
CPU(s)	212	244	222	232
$k = 0$ (?)	–	1	–	–
CPU(s)	–	277	–	–

Table 2
Node disjoint and SRLG diverse path pairs, for $m = 3n, 4n$: total number of node pairs and average CPU times when $K = 5000$ was not obtained

Node disjoint	$m = 3n$		$m = 4n$	
n	100	1000	100	1000
$0 < k < 5000$	4	36	1	29
CPU(s)	125	139	125	132
$k = 0$ (?)	–	1	–	–
CPU(s)	–	155	–	–

The results show that the algorithm solves exactly the problem of obtaining K path pairs, node disjoint (arc disjoint) and SRLG diverse in most cases. When the algorithm generates, k, $0 \leq k < K$ paths (due to the allowed maximum number of generated paths), this can be CPU time consuming. Therefore, in order to avoid the high CPU time, sometimes required by the algorithm, a CPU time limit should be imposed for obtaining the desired number of solutions (K), so that it can be used as a subroutine in a multicriteria approach to reliable routing taking into account SRLGs.

5. Conclusion

The multi-layer nature of telecommunication networks, makes it more difficult to implement recovery mechanisms to ensure routing resiliency. The introduction of the concept of SRLGs allows an upper layer to select, for a given active path, a backup path, which avoids every SRLG that may involve the selected AP in the event of afailure. A formulation of the SRLG diverse path pair calculation problem, in a directed network was put forward. An exact algorithm for enumerating SRLG diverse paths in (un)directed networks, by non-decreasing cost of their total (additive) cost was proposed. The considered SRLG diverse path pairs may be node or arc disjoint, with or without length constraints.
Computational results displayed the efficiency of the proposed algorithm for obtaining node or arc disjoint SRLG diverse path pairs in undirected networks. The experimental results show the algorithm solves the problem of enumerating $K = 1000$ disjoint paths pairs in most cases, using less than one second for the smaller networks. However, when the desired value for K can not be attained, the CPU time can grow significantly. Considering that this algorithm can be used as a subroutine in a multicriteria approach to resilient routing, taking into account SRLGs, we would advise a lower number of allowed maximum candidate paths or a limit of CPU time per node pair.

Acknowledgements

We thank Marco António do Carmo Silva for collecting the data and producing the figures used in this work.
This work was partially supported by programme POSC of the EC programme cosponsored by national funds.

References

[1] J. Q. Hu, "Diverse routing in optical mesh networks", *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 489–494, 2003.

[2] M. J. Rostami, S. Khorsandi, and A. A. Khodaparast, "CoSE: ASRLG-disjoint routing algorithm", in *Proc. Fourth Eur. Conf. Univ. Multiserv. Netw. ECUMN'07*, Toulouse, France, 2007.

[3] D. Xu, Y. Chen, Y. Xiong, C. Qiao, and X. He, "On finding disjoint paths in single and dual link cost networks", in *IEEE INFOCOM 2004 Conf.*, Hong Kong, 2004.

[4] A. K. Todimala and B. Ramamurthy, "IMSA: an algorithm for SRLG diverse routing in WDM mesh networks", in *Proc. Int. Conf. Comput. Commun. Netw. ICCCN 2004*, Chicago, USA, 2004, pp. 199–204.

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

[6] R. Bhandari, *Survivable Networks, Algorithms for Diverse Routing.* Norwell: Kluwer, 1999.

[7] A. K. Todimala and B. Ramamurthy, “A heuristic with bounded guarantee to compute diverse paths under shared protection in WDM mesh networks”, in *Proc. IEEE Globlecom 2005 Conf.*, St. Louis, USA, 2005, pp. 1915–1919.

[8] P. Laborczi, J. Tapolcai, P.-H. Ho, T. Cinkler, A. Recski, and H. T. Mouftah, “Algorithms for asymmetrically weighted pair of disjoint paths in survivable networks”, in *Proc. Des. Rel. Commun. Netw. DCRN 2001*, Budapest, Hungary, 2001, pp. 220–227.

[9] X. Pan and G. Xiao, “Algorithms for the diverse routing problem in WDM networks with shared risk link groups”, in *Int. Conf. Comput. Sci. ICCS 2004*, Krakow, Poland, 2004, pp. 381–385.

[10] X. Pan and G. Xiao, “Heuristics for diverse routing in wavelength-routed networks with shared risk link groups”, *Phot. Netw. Commun.*, vol. 11, no. 1, pp. 29–38, 2006.

[11] D. Xu, Y. Xiong, C. Qiao, and G. Li, “Trap avoidance and protection schemes in networks with shared risk link groups”, *J. Lightw. Technol.*, vol. 21, no. 11, pp. 2683–2693, 2003.

[12] J. C. N. Clímaco and M. M. B. Pascoal, “Finding non-dominated bicriteria shortest pairs of disjoint simple paths”, *Comput. Oper. Res.*, vol. 36, no. 11, pp. 2892–2898, 2009.

[13] J. Bang-Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer Monographs in Mathematics. Great Britain: Springer, 2002.

[14] E. Martins, M. Pascoal, and J. Santos, “Deviation algorithms for ranking shortest paths”, *Int. J. Found. Comput. Sci.*, vol. 10, no. 3, pp. 247–263, 1999.

[15] E. Martins, M. Pascoal, and J. Santos, “An algorithm for ranking loopless paths”, Tech. Rep. 99/007, CISUC, 1999 [Online]. Available: http://www.mat.uc.pt/∼marta/Publicacoes/mps2.ps

[16] D. Eppstein, “Finding the *k* shortest paths”, *SIAM J. Comput.*, vol. 28, no. 2, pp. 652–673, 1999.

[17] R. Dial, F. Glover, D. Karney, and D. Klingman, “A computational analysis of alternative algorithms and labeling techniques for finding shortest path trees”, *Networks*, vol. 9, no. 3, pp. 215–348, 1979.

Teresa Gomes is Assistant Professor in telecommunications at the Department of Electrical Engineering and Computers of the Faculty of Sciences and Technology of the University of Coimbra, Portugal, since 1998, and a researcher at the INESC-Coimbra. She obtained the following degrees: undergraduate diploma in electrical engineering science (E.E.S.)-informatics at the Coimbra University (1984), M.Sc. in computer science (1989) and Ph.D. in E.E.S.-telecommunications and electronics (1998), both at the University of Coimbra. Her main present interests are routing, protection and reliability analysis models and algorithms for optical and MPLS networks.

e-mail: teresa@deec.uc.pt
Department of Electrical Engineering and Computers
University of Coimbra
Pinhal de Marrocos
3030-290 Coimbra, Portugal

José Craveirinha is Full Professor in telecommunications at the Department of Electrical Engineering and Computers of the Faculty of Sciences and Technology of the University of Coimbra, Portugal, since 1997. He obtained the following degrees: undergraduate diploma in electrical engineering science (E.E.S.)-telecommunications and electronics at IST, Lisbon Technical University (1975), M.Sc. (1981) and Ph.D. in E.E.S. at the University of Essex (UK) (1984) and Doct. of Science (Agregado) in E.E.S.-telecommunications at the University of Coimbra (1996). Previous positions were: Associate Professor and Assistant Professor at the FCUT, Coimbra University, telecommunication R&D engineer (at the CET-Portugal Telecom). He coordinated a research group in teletraffic engineering and network planning at the INESC-Coimbra R&D Institute since 1986 and was Director of this Institute during 1994–1999. He is author/co-author of more than 100 scientific and technical publications in teletraffic modeling, reliability analysis, planning and optimization of telecommunication networks. His main present interests are in multicriteria routing and reliability analysis models and algorithms for optical and multiservice-IP/MPLS networks.

e-mail: jcrav@deec.uc.pt
Department of Electrical Engineering and Computers
University of Coimbra
Pinhal de Marrocos
3030-290 Coimbra, Portugal