Instituto de Engenharia de Sistemas e Computadores de Coimbra
Institute of Systems Engineering and Computers INESC - Coimbra

Teresa Gomes
José Craveirinha

An algorithm for enumerating SLRG diverse path pairs

No. 9 2009

ISSN: 1645-2631

Instituto de Engenharia de Sistemas e Computadores de Coimbra
INESC - Coimbra
Rua Antero de Quental, 199; 3000-033 Coimbra; Portugal
www.inescc.pt

An algorithm for enumerating SLRG diverse path pairs

Teresa Gomes[(1,2)] and José Craveirinha[(1,2)]

[(2)]Departamento de Engenharia Electrotécnica e de Computadores da FCTUC,
Pólo 2 da Univ. Coimbra, 3030-290 Coimbra, Portugal

[(3)]INESC-Coimbra, Rua Antero de Quental 199,
3000-033 Coimbra, Portugal.

e-mail: teresa@deec.uc.pt, jcrav@deec.uc.pt

June 18, 2009

Abstract

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. In this context, a working path and a protection path, can be used to set up a connection, such that only one of them will be affected by the failure of a single network element

A network element may affect several arcs in an optical or MPLS network, due to the multi-layer nature of telecommunication networks. This lead to the concept of shared risk link group (SRLG). To ensure global path protection against a failure affecting a single SRLG, a SRLG diverse path pair must be calculated. The problem of finding a shared risk link group (SRLG) diverse path pair was shown to be NP-Complete.

An algorithm for enumerating SRLG diverse paths, by non decreasing total cost will be presented. The paths may be node or arc disjoint, and may satisfy a length constraint. Furthermore, the possibility of ordered enumeration of diverse SRLG paths will enable a multi-objective routing approach, with survivability requirements.

1 Introduction

Bandwidth usage optimization is one of the main issues when protection schemes are used in telecommunication networks. In global path protection, the 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 fiber cut, a card failure at a node, a software failure, or any combination of these factors [5], which may affect one or more links at a given network layer. In this context, the concept of shared risk link group (SRLG) allows an upper layer to select, for a given AP, a BP, which avoids every SRLG that may affect the selected AP. 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 shared risk link group (SRLG) diverse path pair has been shown to be NP-Complete [5]. The minimum-cost diverse routing, in which the objective is finding two paths, SRLG diverse, with minimal total arc cost (also designated the min-sum problem), is also NP-Complete [5]. Hu *et al.* [5] 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.

A similar problem is the min-min problem which seeks to minimise the cost of the shorter path (the AP). In [6] an algorithm (CoSE) is proposed for finding a min-min SRLG disjoint path pair, which is based in the CoLE algorithm [9] proposed for solving the min-min problem for link disjoint routing.

The necessity of calculating SRLG diverse path pairs arises in optical and 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. For example, the ordered enumeration of diverse SRLG paths will make possible a multi-objective routing approach, with survivability requirements, similar to the one considered in [2].

The report is organised as follows. In the next section 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 path pairs may be node or arc disjoint, and with length constraints, as briefly explained in sections 4 and 5. This is a work in progress, therefore no results regarding the efficiency of the proposed algorithm will be included at this time.

2 Notation and problem definition

The algorithm in section 3 is based on Algorithm 1 in [2]. Therefore we will use a notation similar to the one in [2].

Let $G = (N, A)$ be a directed network with node set $N = \{v_1, v_2, \ldots, v_n\}$ and arc set $A = \{a_1, a_2, \ldots, a_m\}$ (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 2.1 *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 will only use the expression "simple path" when required, namely in the algorithm.

Let path $p = \langle v_1, a_1, v_2, \ldots, v_{i-1}, a_{i-1}, v_i \rangle$, be given as an alternate sequence of nodes and arcs from G, such that the tail of a_k is v_k and the head of a_k is v_{k+1}, for $k = 1, 2, \ldots, i-1$ (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 $p = \langle v_1, a_1, v_2, \ldots, v_{i-1}, a_{i-1}, v_i \rangle$ 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 [1] if $\{v_2, \ldots, v_{i-1}\} \cap V^*(q) = \emptyset$. 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 fiber cut, a card failure at a node, a software failure, or any combination of these factors. Let A_r represent the sub-set 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 (Shared Risk Link Group associated with r). Let, as in [5],

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

The SRLG problem can be defined as follows [5].

Definition 2.2 *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) $(i = 1, 2, \ldots)$, 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:

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

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 $a \in A$:

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

R_a can be obtained from A_r $(r = 1, \ldots, |R|)$ 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 2.3 *Two arcs, $a_i, a_j \in A$ are SRLG diverse if $R_{a_i} \cap R_{a_j} = \emptyset$.*

Definition 2.4 *An arc $a \in A$ is SRLG diverse with a path p if $R_a \cap r_p = \emptyset$.*

The algoritm proposed in section 3 is based on algorithm 1 in [2], wich uses the MPS algorithm [8], in its loopless version [7]. 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 [8]. 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 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 [8]) 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 [4] and they are shown by theorems 8 and 9 in [8] and by theorem 2.1 in [7] (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 [3]. 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 $\bar{c}_{v_i v_j} \leq \bar{c}_{v_k v_l}$).

3 Node disjoint and SRLG diverse path pairs

The algorithm is based on the Algorithm 1 in [2] for enumerating node disjoint path pairs, by non-increasing total additive cost. That algorithm requires a network topology transformation as described in the next subsection.

3.1 Network topology modification

Let s, t be a source and destination network 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 [2]:

- 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 form 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.

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 $v_i \in N$ [2][1].

In remark 1 of [2] 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 [2].

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 $v_j \neq v_k$ (and $v'_j \neq v'_k$). 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 extremely important because the MPS algorithm [8], which is the base of Algorithm 1 in [2], 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

In the algorithm we will use a notation similar to the one in [2]. 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.

[1] In [2], where is $\pi_i = \pi_{i'} + \pi_s$ should be $\pi_i = \pi_{i'} + \pi_{s'}$.

If this infeasibility test fails, then we can proceed to try and find SRLG diverse path pairs.

Algorithm 1: Determination of the k shortest SRLG diverse simple path pairs.

Data: Network graph $G = (N, A)$ and a source destination node pair (s, t)
Result: S, the set of the k shortest SRLG diverse simple path pairs from s to t

if *the infeasibility test is successfully* **then** Stop **end**
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 `// Network pruning`
$\mathcal{T}_{t'} \leftarrow$ tree of the shortest paths from $i \in N'$ to t' concerning c `//` $\mathcal{T}'_t(i') = \mathcal{T}_t(i)$
$p \leftarrow \mathcal{T}_{t'}(s)$
if p *is not defined* **then** Stop **end**
$\bar{c}_{v_i v j} \leftarrow \pi_{v_j} - \pi_{v_i} + c_{v_i v_j}, \quad \forall (v_i, v_j) \in A'$`// Calculates reduced costs`
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`
$d_p \leftarrow s$ `// Deviation node of` p, `the first candidate path`
$X \leftarrow \{p\}$
$S = \emptyset$ `// Set of identified SRLG diverse simple path pairs`
while $X \neq \emptyset \wedge |S| < k$ **do**
  $p \leftarrow$ path in X such that $\bar{c}(p))$ is minimum
  if *(p is simple)* $\wedge$ `Disjoint(`p`)` $\wedge$ `SRLGDiverse(`p`)` **then**
    $S \leftarrow S \cup \{p\}$
  end
  $X \leftarrow X \backslash \{p\}$
  $i \leftarrow$ min index such that $v_i = d_p$
  break $\leftarrow$ false `// Candidate paths might be derived from` p
  repeat
    $l \leftarrow$ index such that $a_l = (v_i, v_{i+1})$
    repeat
      $l \leftarrow l + 1$
      $v_j \leftarrow$ head node of a_l `//` **if** $l > m + 1$ **then** $v_j \leftarrow 0$
      if (v_i *is the tail node of* a_l)$\wedge$ `EquivalentPair(`$\text{sub}_p(s, v_i) \diamond a_l \diamond \mathcal{T}_{t'}(v_j)$`))` **then**
        break $\leftarrow$ true `// No candidate paths will derive from` p `at` v_i
      end
    until *break* $\vee$(v_i *is not the tail node of* a_l)$\vee$ [(a_l *does not form a loop with* $\text{sub}_p(s, v_i)$)$\wedge$ `SRLGDiverse(`$\text{sub}_p(s, v_i) \diamond a_l$`)` $\wedge$ `Disjoint(`$\text{sub}_p(s, v_i) \diamond a_l$`)`]
    if ($\neg$ *break*) $\wedge$ (v_i *is the tail node of* a_l) **then**
      $q \leftarrow \text{sub}_p(s, v_i) \diamond a_l \diamond \mathcal{T}_{t'}(v_j)$; $d_q \leftarrow v_i$
      $X \leftarrow X \cup \{q\}$
    end
    $v_i \leftarrow v_{i+1}$ `// Next node of` p
  until ($v_i = t'$) $\vee$ $\neg$($\text{sub}_p(s, v_i)$ *is simple)* $\vee$ $\neg$ `Disjoint(`$\text{sub}_p(s, v_i)$`)` $\vee$ $\neg$ `SRLGDiverse(`$\text{sub}_p(s, v_i)$`)`
end

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, z, obtained in the augmented network (see sub-section 3.1 or [2, section3.2]), is made of $p \diamond (t, s') \diamond q$ and we assume it has deviation node d_z, deviation arc a_h, and that the first arc in p 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_z, belongs to N and the path $\text{sub}_z(s, d_z) \diamond a_h$ is simple;
- the deviation node, d_z, belongs to $N' \backslash N$:
  - the path $\text{sub}_z(s', d_z) \diamond a_h$ is simple;
  - $c(\text{sub}_z(s, t)) \geq c(\text{sub}_z(s', t'))$ (or $c(p) \geq c(q)$). Note that $c(q) = c(\text{sub}_z(s', d_z) \diamond a_h)$, and that $c(q) = c(z) - c(p)$.
  - the paths $\text{sub}_z(s, t)$ and $\text{sub}_z(s', d_z) \diamond (a_h)$ are node-disjoint;
  - a_h is SRLG diverse with $\text{sub}_z(s, t)$.

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 [2] with **repeat until** cycles.

Function **Disjoint**(z), $z = p \diamond (t, s') \diamond q$, returns true if p and q are node disjoint. Function **SRLGDiverse**(z) returns true if p 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**($\text{sub}_p(s, v_i) \diamond a_l$), in step 27, it is sufficient to evaluate if $\text{sub}_p(s, v_i)$ is SRLG diverse with arc a_l.

Function **EquivalentPair()** was first introduced in [2], for including remark 2 in Algorithm 1. Due to remark 2 in [2] 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:

[2] In order to reduce the need for graph transformation, these arcs can be simply be marked as useless, as long as an adequate Dijkstra's algorithm is implemented.

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

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

4 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 arc belongs to at least one SRLG.

5 SRLG diverse path pairs with length constraints

Let $z = p \diamond (t, s') \diamond q$, represent a path pair (p, 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 $i \in p$ and $j' \in q$ must less then the length restriction (assuming node s has depth 0).

References

[1] J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer Monographs in Mathematics. Springer-Verlag, May 2002.

[2] João C.N. Clímaco and Marta M.B. Pascoal. Finding non-dominated bicriteria shortest pairs of disjoint simple paths. *Computers & Operations Research*, 36(11):2892–2898, 2009.

[3] R. Dial, F. Glover, D. Karney, and D. Klingman. A computational analysis of alternative algorithms and labeling techniques for finding shortest path trees. *Networks*, 9:215–348, 1979.

[4] D. Eppstein. Finding the k shortest paths. *SIAM Journal on Computing*, 28:652–673, 1998.

[5] J. Q. Hu. Diverse routing in optical mesh networks. *IEEE Transactions on Communications*, 51(3):489–494, March 2003.

[6] M. J. Rostami, S. Khorsandi, and A. A. Khodaparast. CoSE: ASRLG-disjoint routing algorithm. In *Proceedings of the Fourth European Conference on Universal Multiservice Networks (ECUMN'07)*, 2007.

[7] E. Martins, M. Pascoal, and J. Santos. An algorithm for ranking loopless paths. Technical Report 99/007, CISUC, 1999. http://www.mat.uc.pt/~marta/Publicacoes/mps2.ps.

[8] E. Martins, M. Pascoal, and J. Santos. Deviation algorithms for ranking shortest paths. *International Journal of Foundations of Computer Science*, 10(3):247–263, 1999.

[9] 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*. IEEE, 2004.