Listing Regional Failures

Balázs Vass[*][†],
[*]Department of Operations Research, Eötvös University, Budapest, Hungary, vassbalazs@gmail.com
[†]MTA-BME Future Internet Research Group, Budapest University of Technology and Economics

***Abstract*—Current backbone networks are designed to protect a certain pre-defined list of failures, called Shared Risk Link Groups (SRLG). During network design and operation protecting a failure not part of an SRLG is ignored as they assume to be extremely rare events. The list of SRLGs must be defined very carefully, because leaving out one likely failure event will significantly degrade the observed reliability of the network. The list of SRLGs is typically composed of every single link or node failure. It has been observed that some type of failure events manifested at multiple locations of the network, which are physically close to each other. Such failure events are called regional failures, and are often caused by a natural disaster. The number of possible regional failures can be very large, thus simply listing them as SRLGs is not a viable solution. In this study we focus on link failures only and assume nodes are never part of the failure. We provide a fast systematic approach for generating a list of SRLGs the protection of which is essential to increasing the observed reliability of the network. According to a practical assumption this list is very short with $O(|V|)$ SRLGs in total, and can be computed very fast, in $O(|V|^2)$ time.**

I. Introduction

Current backbone networks are built to protect a certain list of failures. Each of these failures (or termed failure states) are called *Shared Risk Link Groups* (SRLG), which is a set of links that is expected to fail simultaneously. The network is designed to be able to automatically reconfigure in case of a single SRLG failure, such that every connection further operates after a very short interruption. In practice it means the connections are reconfigured to by-pass the failed set of nodes and links. The list of SRLGs must be defined very carefully, because not getting prepared for one likely simultaneous failure event the observed reliability of the network significantly degrades.

On one extreme is listing every *single link or node failure* as an SRLG. Often there is a known risk of a simultaneous multiple failure that can be added as an SRLG, for example if two links between different pair of nodes traverse the same bridge, etc. On the other hand, we have witnessed serious network outages because of a failure event that takes down almost every equipment in a physical region as a result of a disaster, such as weapons of mass destruction attacks, earthquakes, hurricanes, tsunamis, tornadoes, etc. For example the 7.1-magnitude earthquake in Taiwan in Dec. 2006 caused simultaneous failures of six submarine links between Asia and North America and hurricane Sandy in 2012 cased a power outage that silenced 46% of the network in the New York area. These type of failures are called **regional failures**. It is still a challenging open problem how to prepare a network to protect such failure events, as their location and size is not known at planning stage. In the study we propose a solution to this problem with a technique that can significantly reduce the number of possible failure states that should be added as an SRLG to cover all regional failures that does not affect any node.

In practice, regional failures can have any location, size and shape. It is a common best practice to fix the size or shape of regional failures, for example as cycles with given size (also called disk) [1]. Another approach to analyse the network vulnerability against regional failures is using probabilistic failure models, where each link in the SRLG has some probability to fail [2].

In this study we show an efficient way to generate the SRLGs of **regional link failures** which erase the network elements in a circular area and do not affect nodes. Based on the model and assumptions described in Section II, we have shown that with these assumption the number of SRLGs is small, $O(|V|)$, in typical backbone network topology, and can be at most $O(|E||V|)$ in an artificial worst case scenario. We propose a systematic approach based on computational geometric tools that can generate the list of SRLGs in $O(|V|^2)$ steps on typical networks.

We believe this result is a step towards filling the gap between the conventional SRLG based pre-planned protection and regional failures.

The paper is organised as follows. In Section II we present the core mathematical model with several observations and in Section III we show the main result the $O(|V^2)$ algorithm for generating the list of SRLG covering every regional link failures with a shape of disk that do not contain network nodes.

II. Model and Assumptions

We model the network as an undirected geometric graph $G(V,E)$ with $n = |V|$ nodes and $m = |E|$ edges, we assume $n \geq 3$. The nodes of the graph are embedded as points in the Euclidean plane, and the edges are embedded as line segments. The position of node v is denoted by (v_x, v_z). A *disk* $c(x,y,r)$ is a circle with a centre point (x,y) and radius r. The failure caused by a disk is modelled as every interior node and edge with interior part is erased from the graph.

For every disk c let E_c denote the set of edges and nodes erased by c.

Proposition 1. *For any $c_1, c_2 \in C$, $c_1 \subseteq c_2$ it holds that $E_{c_1} \subseteq E_{c_2}$.* $\square$

Let C_0 denote the set of disks that do not have any node of V in their interior. Clearly, $|C_0|$ is infinite.

Claim 2. *For any $c_1(x,y,r) \in C_0$ there exists a $c_2 \in C_0$ such that $E_{c_1} \subseteq E_{c_2}$ and c_2 has at least 2 nodes of V on its boundary.*

Proof. If $E_{c_1} = \emptyset$, than we can choose c_2 arbitrarily from among the non-empty set of disks with at least 2 points of V on their boundary.

If there exists an edge $e = \{a,b\}$ in E_{c_1}, than we generate c_2 as follows. We start with disk $c_1(x,y,r)$ and start to increase its radius. We do it until we reach a node $u \in V$. We can further blow the circle larger without loosing any covered area by moving the central point along the line $(x,y)-(u_x,u_y)$ while keeping u on the boundary.

Assume indirectly that it never reaches a second node. We get a contradiction because c_1 intersects line ab and $a,b \in V$.

Thus the circle will reach a second node $v \in V$. Let $c_2 \supseteq c_1$ be this circle having $u,v \in V$ on its boundary. Clearly, $E_{c_2} \supseteq E_{c_1}$. □

For nodes u and v, let $C_0^{u,v}$ be the set of disks from C_0 which have both u and v on the boundary.

Firs let us ignore the edges of the network and focus only on the nodes. We are searching for disks of maximum size that do not have any nodes in interior. Clearly, each disk of maximum size passes through at least three nodes, otherwise its size could be further increased. By simplicity we assume that the nodes are in general position i.e. no four nodes are on the same cycle and no three nodes are on the same line. In this case connecting the three nodes we get triangles. The problem was deeply investigated in the past and it was shown the union of these triangles results a triangulation of the graph, called Delaunay triangulation [3]. Let $D_\bigtriangledown = (E_\bigtriangledown, V)$ denote the Delaunay triangulation on the set of nodes, where $E_\bigtriangledown$ denotes the edges of the triangulation, which can be very different form the edges of the network. In Delaunay triangulation no circumcircle of any triangle contains node in interior. Another interesting property that the dual graph of the Delaunay triangulation is called Voronoi diagram [4]. An important observation is the following.

Proposition 3. *$C_0^{u,v}$ is non-empty iff $\{u,v\}$ is an edge of the $D_\bigtriangledown = (V, E_\bigtriangledown)$ Delaunay triangulation.*□

Let $\mathscr{F}_0$ and $\mathscr{F}_0^{u,v}$ be the set of failures caused by elements of C_0 and $C_0^{u,v}$, respectively. Formally, $\mathscr{F}_0 = \{E_c | k \in C_0\}$ and $\mathscr{F}_0^{u,v} = \{E_c | k \in C_0^{u,v}\}$. We call the elements of $\mathscr{F}_0$ *regional link failures*, or simply *link failures*.

Let denote $\mathscr{M}_0$ and $\mathscr{M}_0^{u,v}$ the exclusion-wise maximal elements of $\mathscr{F}_0$ and $\mathscr{F}_0^{u,v}$, respectively. Our goal is to determine $\mathscr{M}_0$.

Claim 4. $\mathscr{M}_0 \subseteq \bigcup_{\{u,v\} \in E_\bigtriangledown} \mathscr{M}_0^{u,v}$.

Proof. Clearly, for all $f \in \mathscr{M}_0$ there exists a $c_1 \in C_0$ such that $f = E_{c_1}$. According to Claim 2 and Prop. 3, there exists a $c_2 \in \cup_{\{u,v\} \in E_\bigtriangledown} C_0^{u,v}$ for which $E_{c_2} \supseteq E_{c_1}$. This implies $f \subseteq E_{c_2}$. Since f is an exclusion-wise maximal element of $\mathscr{F}_0$ by definition of $\mathscr{M}_0$, this is possible only if $f = E_{c_2}$.

We get that for every $f \in \mathscr{M}_0$ there exists a $c_2 \in \cup_{\{u,v\} \in E_\bigtriangledown} C_0^{u,v}$ such that $f = E_{c_2}$. This implies $\mathscr{M}_0 \subseteq \cup_{\{u,v\} \in E_\bigtriangledown} \mathscr{M}_0^{u,v}$. □

Before presenting our algorithm for determining $\mathscr{M}_0$, we should take a look on its size. It turns out that $|\mathscr{M}_0|$ is $O(nm)$.

We use parameter θ_0 for the maximum number of edges crossing the circumcircle of a Delaunay triangle.

Since $|\mathscr{M}_0|$ can be asymptotically large, it is not possible to give an algorithm which is "really fast" on all graphs. On the other hand, our algorithm computes $\mathscr{M}_0$ in $O(n^2\theta_0^3)$ time (Thm. 5), what gives $O(n^2)$ if θ_0 is constant, which is a natural assumption for many types of networks.

Our algorithm computes $\mathscr{M}_0$ in the following way. First it generates the Delaunay triangulation $D_\bigtriangledown = (E_\bigtriangledown, V)$. After that for every $\{u,v\} \in E_\bigtriangledown$ it generates sets $\mathscr{M}_0^{u,v}$. Finally it computes $\mathscr{M}_0$ by gathering the globally maximal elements of sets $\mathscr{M}_0^{u,v}$. We will realise this plan in Section III.

III. The Algorithm

Consider the Delaunay triangulation $D_\bigtriangledown = (V, E_\bigtriangledown)$ and the set T_0 of Delaunay triangles given by their vertices. Since $D_\bigtriangledown$ is a planar graph, for an edge (u,v) there exists one or two nodes in V that are neighbours of both u and v.

If there exist two points of this kind, let us call them w_1 and w_2. In this case both $\{u,v,w_1\}$ and $\{u,v,w_2\}$ are elements of T_0. Let $C_{u,v}^1$ and $C_{u,v}^2$ be the disks with u,v,w_1 and u,v,w_2 on the boundary.

If there exists only one common neighbour w_1 of u and v, let $C_{u,v}^1$ be the same as before, and let $C_{u,v}^2$ be an infinitely large disk with boundary going through u and v not containing w_1. Thus $\{u,v,w_1\} \in T_0$ and $\{u,v,w_2\} \notin T_0$.

Let T_0^t denote the set of Delaunay-triangles which have common edge with t, for all $t \in T_0$.

It is easy to see that all disks in $C_0^{u,v}$ are covered by $C_{uv}^1 \cup C_{u,v}^2$.

Let $e \in E_{u,v}^3$ iff $e \in E$ intersects $C_{u,v}^1 \cap C_{u,v}^2$, $e \in E_{u,v}^1$ iff $e \in E \setminus E_{u,v}^3$ intersects $C_{u,v}^1 \setminus C_{u,v}^2$ and $e \in E_{u,v}^2$ iff $e \in E \setminus E_{u,v}^3$ intersects $C_{u,v}^2 \setminus C_{u,v}^1$ (see Figure 1).

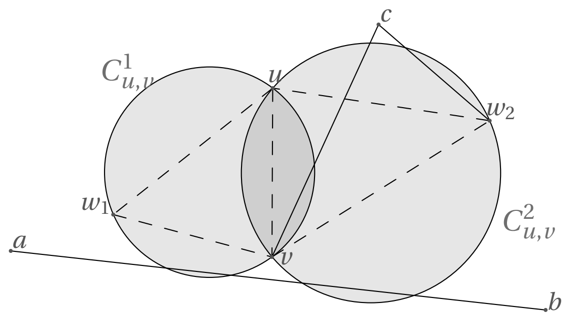

Fig. 1: Example on a Delaunay-edge $\{u,v\}$ with w_1, w_2, $C_{u,v}^1$ and $C_{u,v}^2$. Here $E_{u,v}^1 = \{\{a,b\}\}$, $E_{u,v}^2 = \{\{a,b\},\{c,v\},\{c,w_2\}\}$, $E_{u,v}^3 = \{\{c,v\}\}$.

Using the previous plan and the specific properties of the Delaunay triangulation we proved the next theorem.

Theorem 5. *$\mathcal{M}_0$ can be computed in $O(n^2\theta_0^3))$ using Algorithm 1, and has $O(n\theta_0)$ elements, each of them consisting of $O(\theta_0)$ edges.*

Proof. At line 1 the Delaunay triangulation can be computed in $O(n \log n)$ time [3]. Sets T_0 and T_0^t for all $t \in T_0$ also can be computed in $O(n \log n)$.

In line 2 we do some preparation in constant time for every Delaunay edge $\{u,v\}$, then we calculate sets $E_{u,v}^i$ for all $\{u,v\} \in E_\triangledown$ in $O(n^2\theta_0)$ time according to Lemma 8.

In line 3 we generate sets $\mathcal{M}_0^{u,v}$ in $O(n^2\theta_0^2)$ time (Lemma 9).

Finally, in line 4 $\mathcal{M}_0$ is calculated from lists $\mathcal{M}_0^{u,v}$ in $O(n^2\theta_0^3)$ time (Lemma 12).

According to these results we can derive Theorem 5.

□

Corollary 6. *Assuming θ_0 is upper bounded by a constant, $\mathcal{M}_0$ can be computed in $O(n^2)$ time, and the total length of it is $O(n)$.*

Algorithm 1: Generating the maximal regional link failures

```
Input: $G = (V,E)$
Output: The set $\mathcal{M}_0$ of maximal single regional failures.
begin
1   $E_\triangledown, T_0, T_0^t (t \in T_0) \leftarrow$DELAUNAY($V$);
2   $E_{u,v}^i (i \in \{1,2,3\}, \{u,v\} \in E_\triangledown) \leftarrow$
    GETEDGESETS($V, E, E_\triangledown, T_0$);
3   $\mathcal{M}_0^{u,v} (\{u,v\} \in E_\triangledown) \leftarrow$ GENERATE
    ($V, E, E_\triangledown, E_{u,v}^i (i \in \{1,2,3\}, \{u,v\} \in E_\triangledown)$);
4   $\mathcal{M}_0 \leftarrow$
    ELIMINATEREDUNTANTS($\mathcal{M}_0^{u,v}, \forall \{u,v\} \in E_\triangledown$);
    return $\mathcal{M}_0$
```

A. On the number of edges and size of θ_0

Lemma 7. *The number of edges is $O(n\theta_0)$, more precisely $m \le (2n-5)\theta_0$.*

Proof. The Delaunay triangulation is a planar graph and thus $|E_\triangledown| \le 3n-6$. Since every Delunay triangle has 3 Delaunay edges, a Delaunay edge is edge of at most 2 Delaunay triangles, and there are at least 3 Delaunay edges on the convex hull of V, the number of Delaunay triangles is at most

$$\frac{2|E_\triangledown|-3}{3} \le \frac{2}{3}(3n-6)-1 = 2n-5.$$

Since every $c \in C_0$ intersects at most θ_0 edges of the network, and contains a Delaunay triangle and every edge intersects at least one triangle, the m number of the links cannot be larger than θ_0 times the number of the Delaunay triangles. We get $m \le (2n-5)\theta_0$. □

A graph family may have $O(n^3)$ single regional failures. However, we are convinced that θ_0 is small in case of typical backbone networks and there exists a small constant c that it never exceeds and thus $|\mathcal{M}_0| \le cn$.

B. Algorithm 2 (Method Getedgesets)

Lemma 8. *Algorithm 2 computes sets $E_{u,v}^i$ for all $\{u,v\} \in E_\triangledown$ in $O(n^2\theta_0)$. If θ_0 is constant, this gives $O(n^2)$ time.*

Proof. First we have to show the correctness of the algorithm.

Circles $C_{u,v}^i$, $i \in \{1,2\}$ are the circumcircles of the Delaunay triangles, unless $\{u,v\}$ is on the convex hull of V. By definition, if $\{u,v\}$ is on the convex hull of V, $E_{u,v}^2$ is empty set. Therefore, it is easy to see that since in lines 5 - 8 we compute every edge set E_t covered by circumcirlce of Delaunay tirangle t, in lines 9 - 11 we also get set $E_{u,v}^i$. Thus Algorithm 2 is correct.

We make an estimation of the complexity of Algorithm 2 as follows.

Clearly, calculations in lines 1-4 can be done overall in $O(n)$ time.

In lines 5 - 8 we get sets E_t in $O(n^2\theta_0)$ time by Lemma 7. If θ_0 is constant, this gives an $O(n^2)$ complexity.

It is easy to see that lines 9 - 11 we find the desired $E_{u,v}^i$ sets in $O(n^2\theta_0)$, or if θ_0 is constant, in $O(n)$ complexity.

The overall complexity of Algorithm 2 is $O(n^2\theta_0)$ time. If θ_0 is constant, this means $O(n^2)$.

□

C. Algorithm 3 (Method Generate $\mathcal{M}_0^{u,v}$)

Lemma 9. *Algorithm 3 generates sets $\mathcal{M}_0^{u,v}$ in $O(n\theta_0^2)$ time.*

Proof. Proposition 10 shows the correctness of Algorithm 3.

We assume line 2 runs in constant time. This means that for a given $\{u, v \in E_\triangledown\}$ in lines 1 and 2 we get values $m_{a,b}^i$ in $O(\theta_0)$ time, in lines 3 and 4 we sort them in $O(\theta_0 \log \theta_0)$ time, and in lines 7-11 $M_0^{u,v}$ is calculated in $O(\theta_0^2)$ time. Since $|E_\triangledown| \le 3n-6$, this gives an overall complexity $O(n\theta_0^2)$.

□

Proposition 10. *It can be shown that if a $k \in C_0^{u,v}$ does not contain $L^1[i-1]$ but contains $L^1[i]$, then $L^1[j]$ is covered by k iff $j \ge i$. Similarly, if k contains $L^2[i-1]$ but does not contain $L^2[i]$, then $L^2[j]$ is covered by k iff $j \le i-1$. Trivially, E^3 is covered by k, and that is also clear that for any $e_1 \in E^1$ and $e_2 \in E^2$ the edges e_1, e_2 are covered by k iff $m_{e_1}^1 + m_{e_2}^2 \le \pi$.* □

Corollary 11. *For every $\{u,v\} \in E_\triangledown$, $|\mathcal{M}_0^{u,v}|$ is $O(\theta_0)$.*

Proof. It can be deducted from the description and correctness of Algortithm 3. □

D. Algorithm 4 (Method Eliminateredundants)

Lemma 12. *Algorithm 4 computes $\mathcal{M}_0$ in $O(n\theta_0^3)$ using sets $\mathcal{M}_0^{u,v}$.*

Proof. We can prove the correctness of Algorithm 4 by checking that it eliminates all globally non-maximal elements

Algorithm 2: GETEDGESETS $E^1_{u,v}, E^2_{u,v}, E^3_{u,v}$ for all $\{u,v\} \in E_\triangledown$

```
Input: $V, E, E_\triangledown, T_0$
Output: $E^i_{u,v}$ for all $i \in \{1,2,3\}, \{u,v\} \in E_\triangledown$
begin
1     for $\{u,v\} \in E_\triangledown$ do
2         Determine $w_1, w_2, C^1_{u,v}$ an $C^2_{u,v}$ ;
3         $P^1 \leftarrow$ the half plane having $uv$ line as boundary
          and containing $w_1$ ("the half plane on left hand
          side");
4         $P^2 \leftarrow$ the half plane having $uv$ line as boundary
          and containing $w_2$("the plane on right hand
          side");
5     for $t \in T_0$ do
          $E_t \leftarrow \emptyset$
6     for $\{a,b\} \in E$ do
7         for $t \in T_0$ do
              if $[a,b] \cap C_t \neq \emptyset$ then
8                 $E_t \leftarrow E_t \cup \{\{a,b\}\}$
9     for $\{u,v\} \in E_\triangledown$ do
          for $i \in \{1,2\}$ do
              if $w_i \in V$ then
10                $E^i_{u,v} \leftarrow E_{w_i uv_\Delta} \setminus E_{w_{3-i} uv_\Delta}$ ;
              else if $\{u,v\} \in E$ then
                  $E^i_{u,v} \leftarrow \{\{u,v\}\}$
11        $E^3_{u,v} \leftarrow E_{w_1 uv_\Delta} \cap E_{w_2 uv_\Delta}$ ;
12    return $E^i_{u,v}$ for all $i \in \{1,2,3\}, \{u,v\} \in E_\triangledown$
```

Algorithm 3: Generaring sets $\mathcal{M}^{u,v}_0$ for all $\{u,v\} \in E_\triangledown$

```
Input: $V, E, E_\triangledown, E^i_{u,v} (i \in \{1,2,3\}, \{u,v\} \in E_\triangledown)$
Output: Sets $\mathcal{M}^{u,v}_0$ for all $\{u,v\} \in E_\triangledown$ .
begin
      for $\{u,v\} \in E_\triangledown$ do
1         for $i \in \{1,2\}$ and $\{a,b\} \in E^i_{u,v}$ do
2             $m^i_{a,b} \leftarrow \max\{m(\widehat{ucv}) | c \in [a,b] \cap P^i\}$
3         $L^1 \leftarrow$SORT($E^1$ by $m^1_{a,b}$ values increasingly);
4         $L^2 \leftarrow$SORT($E^2$ by $m^2_{a,b}$ values decreasingly);
5         $i \leftarrow 1, j \leftarrow 1$;
6         $\mathcal{M}^{u,v}_0 \leftarrow \emptyset$ ;
7         repeat
8             while $m^1_{L^1(i)} + m^2_{L^2(j)} \leq \pi$ and $j \leq length(L^2)$
              do
                  $j \leftarrow j+1$
9             $\mathcal{M}^{u,v}_0 \leftarrow \mathcal{M}^{u,v}_0 \cup$
              $\{\{L^1(i), \dots, L^1(n), L^2(1), \dots, L^2(j-1)\}\} \cup E^3$;
10            while $m^1_{L^1(i)} + m^2_{L^2(j)} > \pi$ and $i \leq length(L^2)$ do
11                $i \leftarrow i+1$
          until $i > length(L^1)$ or $j > length(L^2)$;
      return $\mathcal{M}^{u,v}_0$
```

Algorithm 4: ELIMINATEREDUNTANTS

```
Input: $\mathcal{M}^{u,v}_0$ for all $\{u,v\} \in E_\triangledown$
Output: $\mathcal{M}_0$
begin
      for $\{u,v\} \in E_\triangledown$ do
          for $\{w,z\} \in E_\triangledown, \{w,z\} \neq \{u,v\}$ do
              for $f_{u,v} \in \mathcal{M}^{u,v}_0$ do
                  for $f_{u,v} \in \mathcal{M}^{u,v}_0$ do
                      for $f_{w,z} \in \mathcal{M}^{w,z}_0$ do
                          if $f_{u,v} \supseteq f_{w,z}$ then
                              $\mathcal{M}^{w,z}_0 \leftarrow \mathcal{M}^{w,z}_0 \setminus f_{w,z}$
                          else
                              if $f_{u,v} \subset f_{w,z}$ then
1                                 $\mathcal{M}^{u,v}_0 \leftarrow \mathcal{M}^{u,v}_0 \setminus f_{u,v}$
2     $\mathcal{M}_0 \leftarrow \bigcup_{\{u,v\} \in E_\triangledown} \mathcal{M}^{u,v}_0$;
3     return $\mathcal{M}_0$
```

of $\mathcal{M}^{u,v}_0$ and leaves exactly one copy of each element of $\mathcal{M}^{u,v}_0$ in the end.

The proof of complexity is as follows.

We have to compare $O(n^2\theta_0^2)$ times two sets of size $O(\theta_0)$.

Thus the overall complexity of Algorithm 4 is $O(n^2\theta_0^3)$. If θ_0 is constant, this means $O(n^2)$.

□

Acknowldedgements

I would like to express my deep gratitude to Erika Bérczi-Kovács, Attila Kőrösi and János Tapolcai, my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work.

References

[1] S. Neumayer, A. Efrat, and E. Modiano, "Geographic max-flow and min-cut under a circular disk failure model," *Computer Networks*, vol. 77, pp. 117–127, 2015.

[2] S. Neumayer, G. Zussman, R. Cohen, and E. Modiano, "Assessing the vulnerability of the fiber infrastructure to disasters," *Networking, IEEE/ACM Transactions on*, vol. 19, no. 6, pp. 1610–1623, 2011.

[3] F. Aurenhammer, "Voronoi diagramsÑa survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 345–405, 1991.

[4] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf, *Computational geometry*. Springer, 2000.