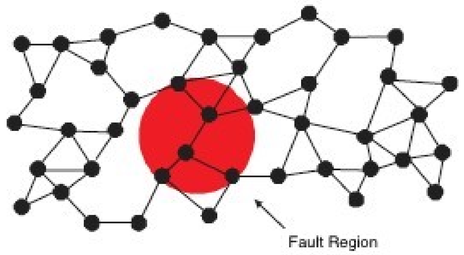
Optimised algorithms identifying all possible regional cuts in a graph

Ziheng Cao, supervised by Prof. Charl Ras

2023/2024 Mathematics and Statistics Vacation Scholarships Program. The University of Melbourne

Introduction

There are many large-scale networks everywhere around the world, playing significant parts in human society, such as NBN, water and gas pipelines, transportation networks and so forth. However, real networks are potentially vulnerable to disasters or terrorists. Such interruptions can sometimes cause serious consequences.



This project is to contribute to such an algorithm that helps design robustly connected networks against any possible events that may cause partial disconnections. Intuitively, when a disaster is modeled by some geometric region, the algorithm aims to identify all possible interruptions and enhance the network without blowing the budget.

Preliminaries[1]

Due to space constraints and for simplicity, in this project, we only consider rectilinear cases. And to help understand, here are some preliminaries explained.

- A **cut** C is defined as a subset of edges in a connected graph G whose removal disconnects G.
- An **inducing region** to C is a connected region in the plane which intersects all the edges in C. And it is considered as **minimum inducing region** $R(C)$ (see Fig. 1) when it can not be further diminished by any arbitrarily small value, ensuring continued intersection with every edge in C.
- C is called **1-D cut** if $R(C)$ is a segment. If $R(C)$ is an rectangle, then C is called a **2-D cut**.
- All 1-D cut can be found in $O(n \times MaxC \times MaxQU)$ time and all 2-D cut can be found in $O(n^2 \times MaxC \times MaxQU)$ time, where n is the number of edges, $MaxC$ is the maximum size of cut, $MaxQU$ is the maximum query time.[1]

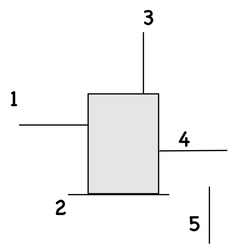


Figure 1. $R(C)$, where cut $C = \{1, 2, 3, 4\}$

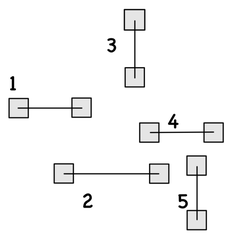


Figure 2. Protection cases

Assumption

To be a milestone towards the algorithm, this project aims to identify all possible cuts when given the shape of fault regions without specified positions. To focus on scenarios where the region intersects edges, we do not consider cases where the region covers a vertex and naturally cuts all edges connected with it. We avoid this by putting protection cases (squares with side length equal to the length of the fault region) on every vertex (see Fig. 2)

Optimised Algorithm for 1-D Cuts

The algorithms[1] initially concentrated on the scenarios where there is no protection case. To align with the objectives of this project, they have been modified and enhanced. By integrating the plane sweep algorithms[2], we refine the approach by considering more suitable event points and corresponding methods for each event.

Identify all 1-D cuts

To identify all 1-D cuts, the algorithm runs both vertical and horizontal sweeps in the plane. The vertical sweep is illustrated here and the horizontal follows similarly.

In this line sweeping algorithm, a vertical sweeper L sweeps the plane progressing from left to right. It stops each time newly encountering unprotected parts of an edge (see Fig. 3). At each of this points, we need only to identify all 1-D cuts containing the newly encountered edge, which means that the algorithms need only to consider the nearby parts of the edge. As shown in Fig. 3, the algorithm only considers a part with length of $2l$ near edge 1, where l is the fault region's length, hence it checks whether $\{1\}$ forms a cut. And the second time L stops (see Fig. 4), it checks $\{1\}$ and $\{1, 2\}$

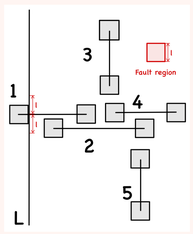


Figure 3.

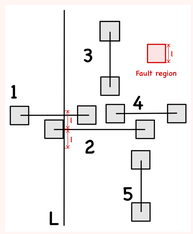


Figure 4.

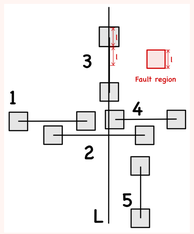


Figure 5.

In the scenarios where L encounters vertical edges (e.g. Fig. 5), it is clear that the only possible cut containing the encountered edge is the edge itself, whose identification, however, will also be completed in horizontal sweep, which causes redundancy. Hence, in the case of vertical sweep, the algorithm does not stop at vertical edges.

Time Complexity

This integrated algorithm can identify all 1-D cuts in $O(n \times l \times MaxQU)$ time. In our scenario, it is more efficient, compared with the initial algorithm, since we need only to take the cuts, whose minimum inducing region can be encompassed by the fault region, into consideration.

Optimised Algorithm for 2-D Cuts

Similar to 1-D algorithm, the initial 2-D algorithm has been integrated into our specific scenario.

Identify all 2-D cuts

To identify all 2-D cuts, the algorithm also executes both vertical and horizontal sweeps but with double sweep lines. Now the primary sweeper L stops at not only the point when it encounters a new edge but also when it is about to leave an edge (see Fig. 6 and 7).

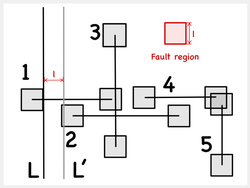


Figure 6.

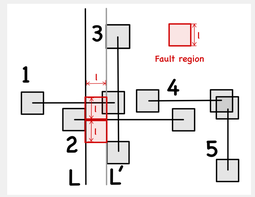


Figure 7.

At each time L stops, the secondary sweeper L' progresses from L's current location to the right until the distance between them reaches l, stopping at each time it encounters a new edge (see Fig. 8) and the moment the distance reaches l. Then, at each time L' stops, the algorithm checks the area bounded by L from left and L' from right, spanning a length of $2l$ (see Fig. 7).

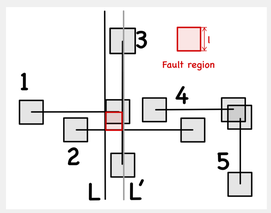


Figure 8.

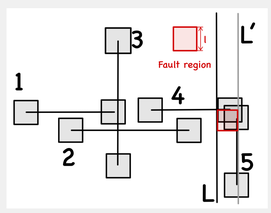


Figure 9.

For instance, as illustrated in Fig. 8, the red region can be encompassed by the fault region, forming a cut $\{1, 2, 3\}$. Similarly, in Fig. 9, the resulting cut is $\{4, 5\}$.

Time Complexity

Similar to 1-D cuts, the 2-D cuts identification algorithm is optimised based on our scenario and performs higher efficiency, $O(n \times l^2 \times MaxQU)$.

Conclusion

Integrating the initial algorithms[1], optimised algorithms identify all cuts in $O(nl^2 \log n(\log\log n)^3)$ time, given that $MaxQU = \log n(\log\log n)^3$ [3]. This potentially helps research towards algorithms of designing robustly connected networks.

References

[1] E. P. Jinhui Xu, Lei Xu. Computing the map of geometric minimal cuts. Algorithmica, 2012.
[2] S. Fortune. A sweepline algorithm for voronoi diagrams, 153–174. Algorithmica. 1987.
[3] M. Thorup. Near-optimal fully-dynamic graph connectivity, 343–350. STOC'00,pp. 2000.

zihengc1@student.unimelb.edu.au