Improving the resilience of networks against geographically correlated failure

Weilun Xu
School of Computing and Information Systems
The University of Melbourne, VIC3010, Australia
weilunx1@student.unimelb.edu.au

Introduction

Much of society's infrastructure can be modelled by networks, including Wide Area Networks, the NBN and major road networks. It is crucial to ensure that these networks do not become disconnected as a result of natural disasters (eg., earthquakes, tsunamis, and bushfires) or malicious attacks. The disconnection of these networks can have significant negative impacts on the basic functioning of society. Thus, increasing a networks' resilience against these types of failures is of great importance.

Recent work by Andres-Thio et al.[1] proposed an augmentation method that improves the resilience of a network against natural disasters, where the disaster is defined as a translation of an open horizontal line segment that intersects the network. The corresponding intersected edges of the graph are considered to be disrupted or destroyed. The augmentation algorithm by Andres-Thio et al. significantly improves the resilience of networks, as demonstrated through extensive computational experiments. However, the computational cost is relatively high and the solutions are not guaranteed to be optimal. The purpose of this project is to improve the algorithm in [1] by finding an exact polynomial-time algorithm for the problem based on dynamic programming.

Research Problem

A network is defined as *G = (V, E)*, which is an arbitrary connected planar graph with a straight-line plane embedding. A disaster is defined as a translation of an open horizontal line-segment *D* of length l in the plane. If *p* is the midpoint of a given disaster *D*, then we say that *D* is centred at *p*. If an edge *e* of *G* intersects *D*, *D* is considered to disrupt *e*. We think of every vertex of *G* as being protected, that is if *D* only intersects an edge *e* at one (or both) of its endpoints, then *D* does not disrupt e. A disaster *D* is said to disrupt a given edge set if it disrupts every element of that edge set. The aim then is, given a graph *G*, find a set of edges of minimum total length that we can add to *G* so that no set of edges that are disrupted by a single disaster will disconnect *G*. This is called the augmentation problem on *G*. We solved this problem by designing an algorithm based on dynamic programming which employs various structural properties of planar graphs.

Methods

Algorithm 5: Augment

```
1  Input  : A graph G = (V, E), a disaster length l and a scheme S;
2  Output: A graph G* = (V, E*) which is l-resilient, with E ⊆ E*;
3  G* := G;
4  while G* is not l-resilient do
5      ℰ := Find-l-cuts(G*, l);
6      Find the l-blocks and l-leaves of G* corresponding to the l-cuts in ℰ;
7      𝒞 := {}        \\ set of candidate edges;
8      for every tuple (ℰ0, V1, V2) satisfying scheme S do
9          e := FindShortestEdge(G, ℰ0, V1, V2);
10         𝒞 := 𝒞 ∪ {e};
       end
11     Choose best edge e ∈ 𝒞 according to scheme S;
12     E* := E* ∪ {e};
13     G* := (V, E*);
   end
14 return G*
```

- The augmentation method proposed by [1] is presented as pseudo-code here. The first is to find the remaining l-cuts in the graph. The l-cuts represent the vulnerable places in the graph. The details of the algorithm to find l-cuts can be found in [1]. Sequentially, the l-blocks and l-leaves will be correspondingly found. These can be seen as the shores of the disconnected edges, where we need to consider how to increase the resilience by augmenting more edges that connect these shores. Accordingly, lines 8-13 is to find the edge with minimal cost to "double ensure" the disaster will not isolate a node if it only disconnects a single edge.

- The scheme S is composed of up to four parameters of different types. It is used to adjust the functionality of the functions of the algorithm, namely a parameter specifies how the end-node subsets for e (V1 and V2) are selected. a parameter specifies the set of l-cuts that passed to subroutine FindShortestEdge, a parameter modifies the way that the cost of an edge is defined, and two miscellaneous scheme options.

- Figure 1 demonstrated the process of the augmentation process of a connected graph. On the left of the figure, it represents an input graph G and a representative disaster disrupting a cut-set incident to node x. The diagram on the right shows an augmentation of G where an edge has been added joining nodes x and y. Here e avoids certain regions associated with cut-sets of G.

- There are five l-cuts in G and five corresponding regions. If e were to properly intersect one of these regions, for example, the top region D, then some disaster would be able to simultaneously disrupt both e and the cut-set incident to x, meaning that the graph generated by adding e to G would not be l-resilient.

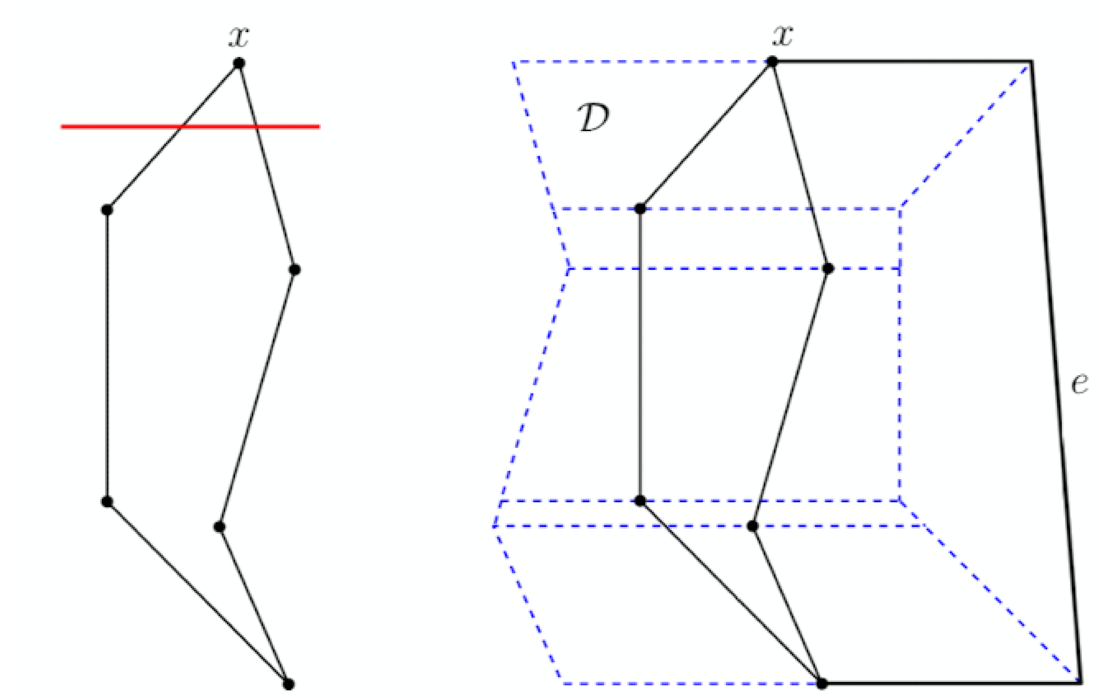

Figure 1 Example of an l-augmentation of an embedded graph.

Results

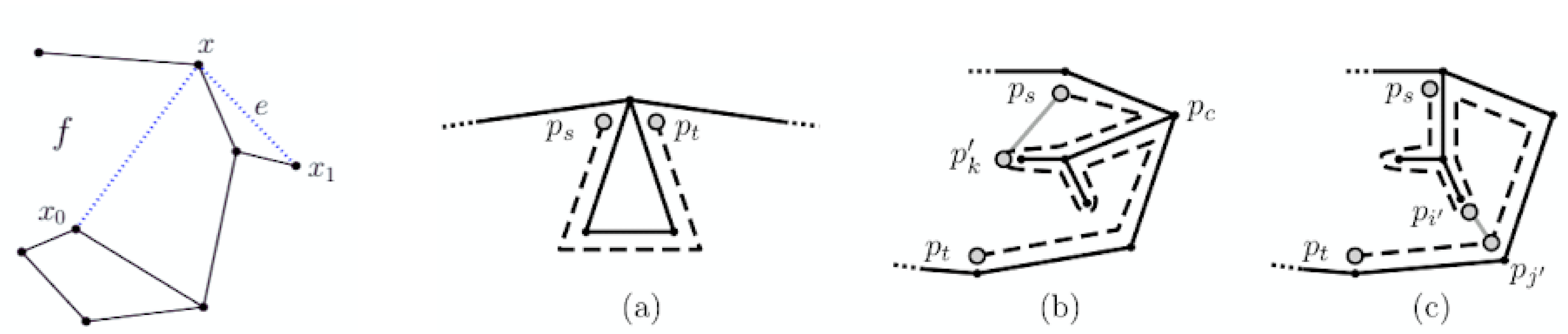

Figure 2 Augmentation and examples of cases (ii) (iii) and (ib) of the DP Approach

(i) If $W_{s,t}$ does not contain any cut vertex relative to $W_{s,t}$, then $C[s,t] = 0$.

(ii) If $p_s = p_t$ and $s \neq t$, then $C[s,t] = \infty$.

(iii) If p_s is not a cut vertex relative to $W_{s,t}$, then $C[s,t] = \min\{C[s+1,t]$, and $\min_{k\in\{s+2,\dots,t-1\}}\{C[s,k]+C[k,t]+f(s,k)\}\}$.

(iv) If p_s is a cut vertex relative to $W_{s,t}$, then let $X = \{\text{descendants of } p_s \text{ in } W_{s,t}\} \times \{\text{non-descendants of } p_s \text{ in } W_{s,t}\}$. Set $C[s,t] = \min_{(p_i,p_j)\in X}\{C[s,i]+C[i,j]+C[j,t]+f(i,j)\}$.

The algorithm of the DP approach to the 2-connected augmentation problem

We applied the idea of the DP algorithm proposed in [2] as shown left to the augmentation algorithm.

- In case (ii), we find the vertex that is a cut vertex, and all other vertices in W[s, t] are descendants. Therefore, there is no edge incident to a non-descendant of Ps in W[s, t], and Ps cannot be satisfied in W[s, t].
- In case (iii), since s does not have descendants, the edge set corresponding to C[s, t] either has no edge incident to Ps or it has an edge between Ps and some descendant Pk of a cut vertex pc in W[s, t]. In the former case, we have C[s, t] = C[s+1, t] since the edge set satisfies all cut vertices relative to W[s+1, t]. In the latter, the edge {Ps, Pk} creates a cycle satisfying the group of descendants containing the vertex for every cut vertex in W[c, k]. It divides F into two faces, whose facial walk contains Ws,k (resp., W[k, t]). Every group still needs to be satisfied in either of the two faces.
- In case (iv), all non-descendants of ps in Ws,t are collected in the set. In particular, there exists no edge in the edge set between a vertex in Ws, I and Wj,t. We can partition the edge set into small edge sets such that they each contain only edges between vertices in Ws, i, Wi,j, and Wj,t respectively, each of them being the minimum-weight set that satisfies the groups of descendants in their respective subproblems. The edges in E1, E2, E3 cannot cross since they correspond to edge-disjoint walks in the walking face. Thus, we have C [s, t] = C [s, i] + C [i, j] + C [j, t] + f (i, j).

Related Literature

[1] "Network augmentation for disaster-resilience against geographically correlated failure", Nicolau Andres-Thio, Marcus Brazil, Charl Ras, and Doreen Thomas, preprint
[2] "Minimum Weight Connectivity Augmentation for Planar Straight-Line Graphs", Hugo A. Akitaya, R. Inkulu, Torrie L. Nichols, Diane L. Souvaine,Csaba D. T´oth, Charles R. Winston preprint

Acknowledgement

Many thanks to the School of Maths and Stats at the University of Melbourne for this opportunity. I'd also like to thank my supervisor, Dr. Charl Ras, for his guidance and many helpful conversations during the program.