Self-Healing Networks

Amitabh Trehan[1] **Tom Hayes**[2] **Navin Rustagi**[1] **Jared Saia**[1]

1: Department of Computer Science, University of New Mexico, Albuquerque, NM 87131

2: Toyota Technological Institute, Chicago, IL 60637.

{amitabh, navin, saia}@cs.unm.edu, hayest@tti-c.org

Abstract

We present two distributed algorithms for self-healing in networks that are reconfigurable (such as peer-to-peer networks) in the sense that they can change their topology during an attack. Self-healing seeks to maintain connectivity and possibly other useful properties in the face of repeated attacks by an adversary, that in our model, is assumed to be omniscient.

The first algorithm DASH *is an algorithm that adds edges only among neighbors of deleted nodes (i.e. locality-aware) and provably maintains connectivity and limits the degree increase of any node . Our second algorithm* ForgivingTree *allows only $O(1)$ degree increase and also limits the diameter increase of the network. It, however, may need to use nodes other than the neighbors of the deleted node for reconstruction. Our approach is orthogonal and complementary to traditional topology-based approaches to defending against attack.*

1. Problem

Many modern networks are *reconfigurable*, in the sense that the topology of the network can be changed by the nodes in the network. For example, peer-to-peer, wireless and mobile networks, many social and biological networks. Algorithmic tools have not yet developed to the point that we are able to fully understand and exploit the flexibility of reconfigurable networks. Here, we present a new, *responsive* approach for maintaining robust reconfigurable networks.

Our model assumes that the network is under repeated attack by an omniscient adversary and the following process continues for up to n rounds where n is the total number of nodes initially in the network: the adversary deletes an arbitrary node from the network, then the network responds by quickly adding a small number of new edges. The goal of self-healing is to maintain connectivity and possibly other useful properties.

2. DASH

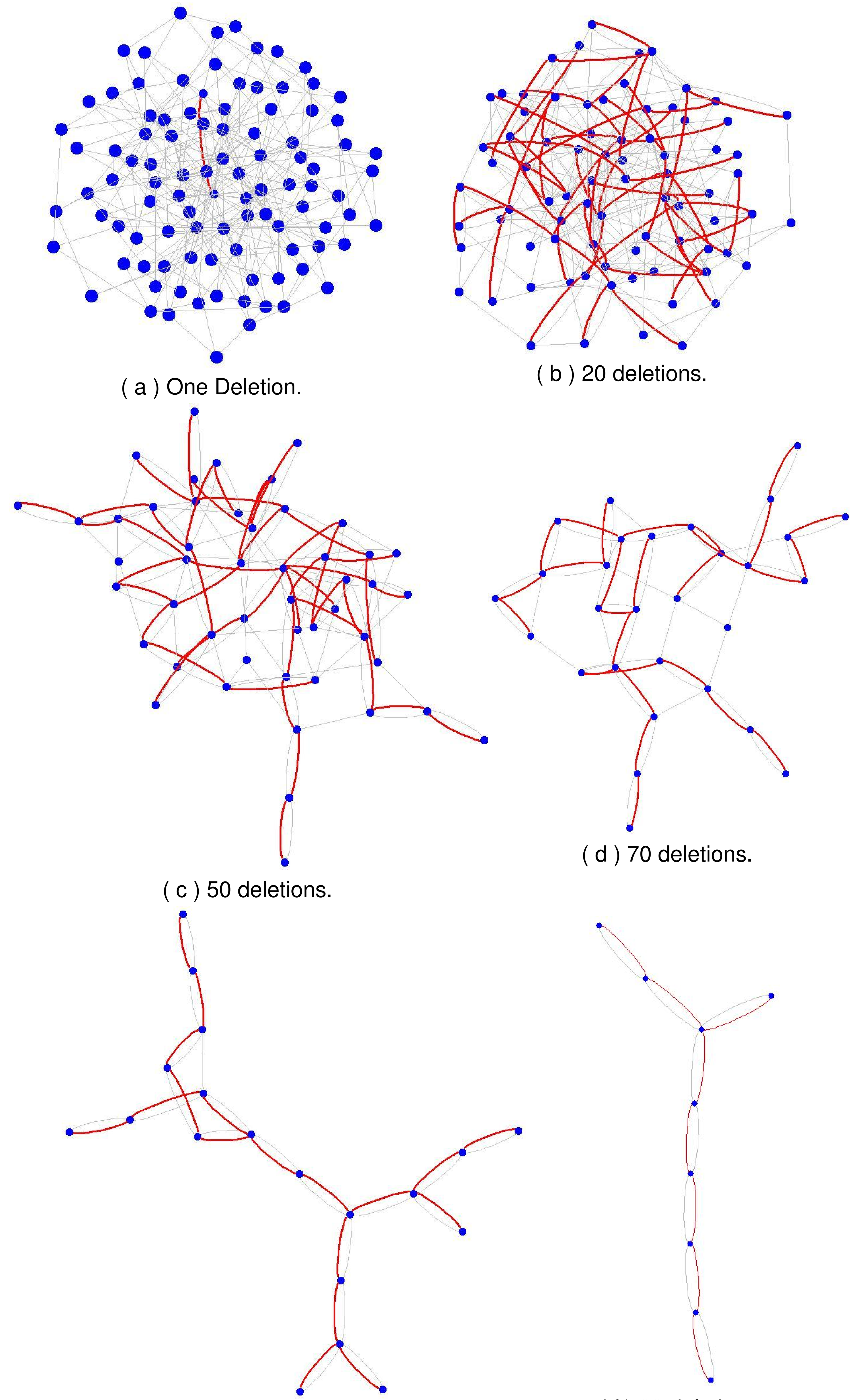
The abbreviation DASH stands for *Degree Based Self-Healing*. DASH is discussed in detail in [2]. In brief, when a node is deleted, a cleverly chosen subset of its neighbors reconnect among themselves. They reconnect in the form a complete binary tree such that the nodes are organized in the ascending order of their previous degree increase from top to bottom, left to right.

The main characteristics of DASH are summarized as follows:

Theorem 1 DASH *guarantees the following properties even if up to all the nodes in the network are deleted:*

1. *The degree of any vertex is increased by at most $2\log n$.*
2. *The number of messages any node of initial degree d sends out and receives is no more than $2(d+2\log n)\ln n$ with high probability*[1] *over all node deletions.*
3. *The latency to reconnect is $O(1)$ after attack; and the amortized latency to update the state of the network over $\theta(n)$ deletions is $O(\log n)$ with high probability.*

Figure 1 shows the evolution of a network over a series of attacks and self-healing performed by DASH.



(a) One Deletion. (b) 20 deletions. (c) 50 deletions. (d) 70 deletions. (e) 80 deletions. (f) 90 deletions.

Figure 1: *Snapshots of a network, of 100 initial nodes, after deletions and healing by* DASH. *The red edges are the new edges added by* DASH.

[1] We use the phrase with high probability (w.h.p) to mean with probability at least $1-1/n^C$ for any fixed constant C.

3. ForgivingTree

The data structure ForgivingTree and the associated algorithms are described in [1]. The algorithm works on a spanning tree of the original graph. On the adversarial deletion of a node, it is replaced by a pre-computed Replacement Tree, as depicted in figure 2. Each node maintains the portions of replacement trees relevant to it and those required for its children through a system of wills, which are executed on deletion of that node.

The following theorem summarizes its performance (Here T is the original tree, D its diameter, and Δ its maximum degree):

Theorem 2 *The Forgiving Tree has the following properties:*

1. *The Forgiving Tree increases the degree of any vertex by at most 3.*
2. *The Forgiving Tree always always has diameter $O(D\log\Delta)$.*
3. *The latency per deletion and number of messages sent per node per deletion is $O(1)$; the size of each message is $O(\log n)$ bits.*

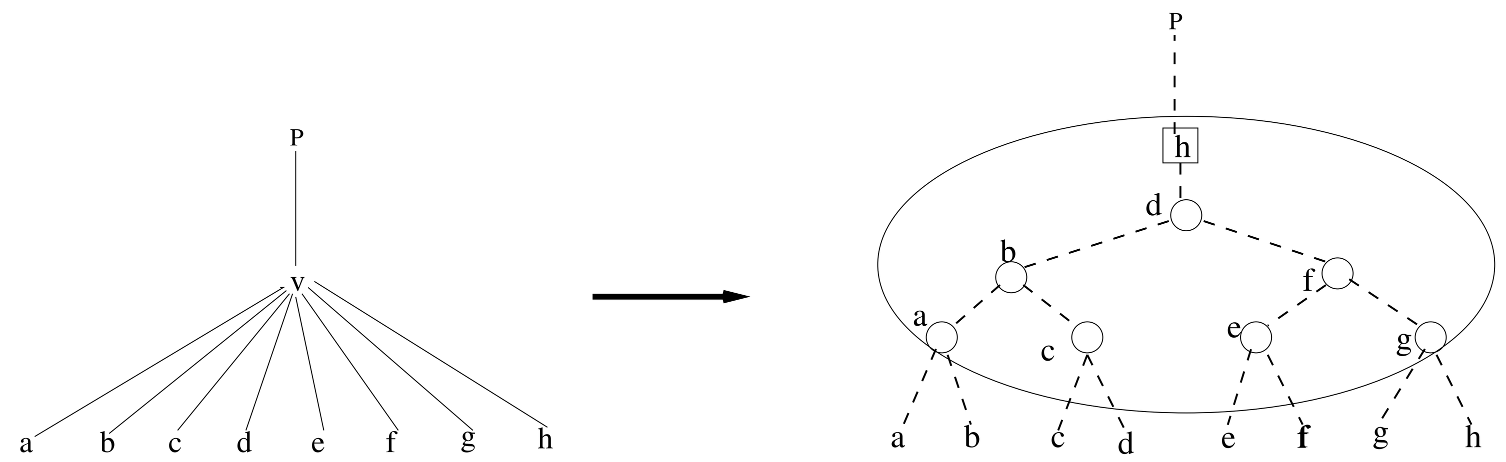


Figure 2: *Deleted node v replaced by its Reconstruction Tree. The nodes in the oval are helper nodes. Regular helper nodes are depicted by circles and the heir helper node by a rectangle.*

4. Future Work

This work can be extended in various directions. There are several open questions we can address: Can we design a distributed data structure to ensure that the stretch between any pair of nodes increases by no more than a certain amount? Can we design algorithms for less flexible networks such as sensor networks? What if the only edges we can add are those that span a small distance in the original network? Finally, can we extend the concept of self-healing to other objects besides graphs? For example, can we design algorithms to rewire a circuit so that it maintains its functionality even when multiple gates fail?

References

[1] T. Hayes, N. Rustagi, J. Saia, and A. Trehan. The forgiving tree: A self-healing distributed data structure. http://www.cs.unm.edu/~amitabh/pubs/ForgivingTreeNV19-525P.pdf.

[2] J. Saia and A. Trehan. Picking up the pieces: Self-healing in reconfigurable networks. In *IEEE International Parallel & Distributed Processing Symposium*, 2008.