Scalable and Distributed Self-Healing Algorithms for Reconfigurable Networks

Amitabh Trehan  Jared Saia

Department of Computer Science  University of New Mexico

CS UNM Student Conference, 2007
Self-healing: A phrase applied to the process of recovery (generally from psychological disturbances, trauma, etc.), motivated by and directed by the patient, guided often only by instinct. [Wikipedia]

Our Goal?
Make this concept concrete.
**Self-healing.**

- *Self-healing:* A phrase applied to the process of recovery (generally from psychological disturbances, trauma, etc.), motivated by and directed by the patient, guided often only by instinct. [Wikipedia]
- Our Goal?
  Make this concept concrete.
Our Problem

- Given: a connected network.
- Goal: Keep the network connected and "small".
- Problem: An adversary deletes nodes in the network.
- Technique: Add edges.
1 Introduction
   - Self-healing in face of attacks
   - Previous Work

2 Our Work
   - Our Model
   - DaSH: Algorithm
   - Experiments
The network: a Graph G(V,E)
The attack: Deletion of nodes.
Self-healing goals:
- Maintain connectivity.
- Ensure degrees of all nodes stay small.
- The algorithm must be efficient.
Outline

1. Introduction
   - Self-healing in face of attacks
   - Previous Work

2. Our Work
   - Our Model
   - DaSH: Algorithm
   - Experiments
Reconfigurable Networks.

- Networks in which we can add new connections between nodes.
- Examples:
  - Peer-to-Peer (P2P) networks.
  - Cellular networks.
  - Ad-hoc networks.
  - Social Networks.
Reconfigurable Networks.

- Networks in which we can add new connections between nodes.
- Examples:
  - Peer-to-Peer (P2P) networks.
  - Cellular networks.
  - Ad-hoc networks.
  - Social Networks.
Applications

- **Sensor Networks**
  - Node: Sensor.
  - Edge: Communication link.

- **P2P Networks**
  - Node: Peer.
  - Edge: Communication link.

- **Social Networks**
  - Node: Person.
  - Edge: Social connection.
Applications

- **Sensor Networks**
  - Node: Sensor.
  - Edge: Communication link.

- **P2P Networks**
  - Node: Peer.
  - Edge: Communication link.

- **Social Networks**
  - Node: Person.
  - Edge: Social connection.
Applications

- **Sensor Networks**
  - Node: Sensor.
  - Edge: Communication link.

- **P2P Networks**
  - Node: Peer.
  - Edge: Communication link.

- **Social Networks**
  - Node: Person.
  - Edge: Social connection.
Applications

- **Sensor Networks**
  - Node: Sensor.
  - Edge: Communication link.

- **P2P Networks**
  - Node: Peer.
  - Edge: Communication link.

- **Social Networks**
  - Node: Person.
  - Edge: Social connection.
Outline

1. Introduction
   - Self-healing in face of attacks
   - Previous Work

2. Our Work
   - Our Model
   - DaSH: Algorithm
   - Experiments
Non-adaptable networks.

- Spare capacity and rerouting. [XM 1999]
- Redundant trees. [MFB 1999]
- Resilient Overlay networks. [ABKM ’01]
- Independent redundant network components. [GBI ’04]
Reconnecting neighbours of deleted nodes in a line. [BASS ’06].
Pluses

- Keeps degrees small.
- Ensures connectivity.
- Simple algorithm.
Problems

- Not scalable.
- Too many messages exchanged $O(n)$.
- Too slow $O(n)$.
- Diameter can increase.
Outline

1. Introduction
   - Self-healing in face of attacks
   - Previous Work

2. Our Work
   - Our Model
   - DaSH: Algorithm
   - Experiments
Our Model

- The Adversary:
  - Eats Nodes.
  - Omniscient: has knowledge of our network and algorithms.
  - Eats one node at a time.

- The Home team (Nodes):
  - Have a small time to recover after each attack.
  - Can set up new links (reconfigure).
  - Maintain Neighbour-of-Neighbour information.
Outline

1 Introduction
   - Self-healing in face of attacks
   - Previous Work

2 Our Work
   - Our Model
   - DaSH: Algorithm
   - Experiments
Some definitions

For a fixed time $t$:

- $G(V, E)$: The actual network.
- $E'$: The edges added by algorithm. ($E' \subseteq E$).
- $G' = (V, E')$: $G'$ will be a forest.
- $N(v, G')$: neighbors of $v$ in $G'$.
- $UN(v, G)$ (Unique Neighbours): Set of neighbours of $v$ in $G$ such that no subtree in $G'$ has more than one representative.
**init:** for given network $G(V, E)$, Initialise each vertex with a random number $ID$ between $[0,1]$ selected uniformly at random.

**while** true do

**if a vertex $v$ is deleted,** do

Nodes in $UN(v, G) \cup N(v, G')$ are reconnected into a complete binary tree. To connect the tree, go right to left, bottom up, mapping nodes to the complete binary tree in decreasing order of degree value.

Let $MINID$ be the minimum $ID$ of any node in $UN(v, G) \cup N(v, G')$. Propagate $MINID$ to all the nodes in the tree of $UN(v, G) \cup N(v, G')$ in $G'$.

**end while**
Figure: Reconfiguration on deletion of node $V$. 
DaSH Properties.

**Theorem**

*DaSH has the following properties:*

- The degree of any vertex is increased by at most $2\log n + 1$.
- The latency to do healing after a deletion is constant.
- The number of messages any node sends out and receives is $O(\log n)$ with high probability.
- The algorithm is completely distributed.
Outline

1. Introduction
   - Self-healing in face of attacks
   - Previous Work

2. Our Work
   - Our Model
   - DaSH: Algorithm
   - Experiments
Attack strategies:

- Max degree: Delete node of maximum degree.
- Max Degree Neighbour: Keep deleting neighbours of maximum degree node.

Healing strategies:

- Binary Graph: reconnect all neighbours; naive.
- Binary Tree: reconnect neighbours keeping $G'$ as forest.
- Degree based Binary Tree (DaSH)
Figure: Self-healing demonstrated by DaSH and related Algorithms.
Concrete definition of self-healing: maintaining an invariant over multiple attacks.

Provably efficient algorithm for maintaining networks.
Additionally, keep Stretch\(^1\) of the network low.

\[^1\text{maximum } \frac{\delta'(u,v)}{\delta(u,v)}\text{ for all nodes } u, v, \text{ where } \delta' \text{ is distance in new graph, } \delta \text{ distance in original graph.}\]
Question Time