COSHER

Compact
Self-Healing Algorithms

Amitabh Trehan
Loughborough University

www.amitabhtrehan.net
www.huntforthetowel.wordpress.com

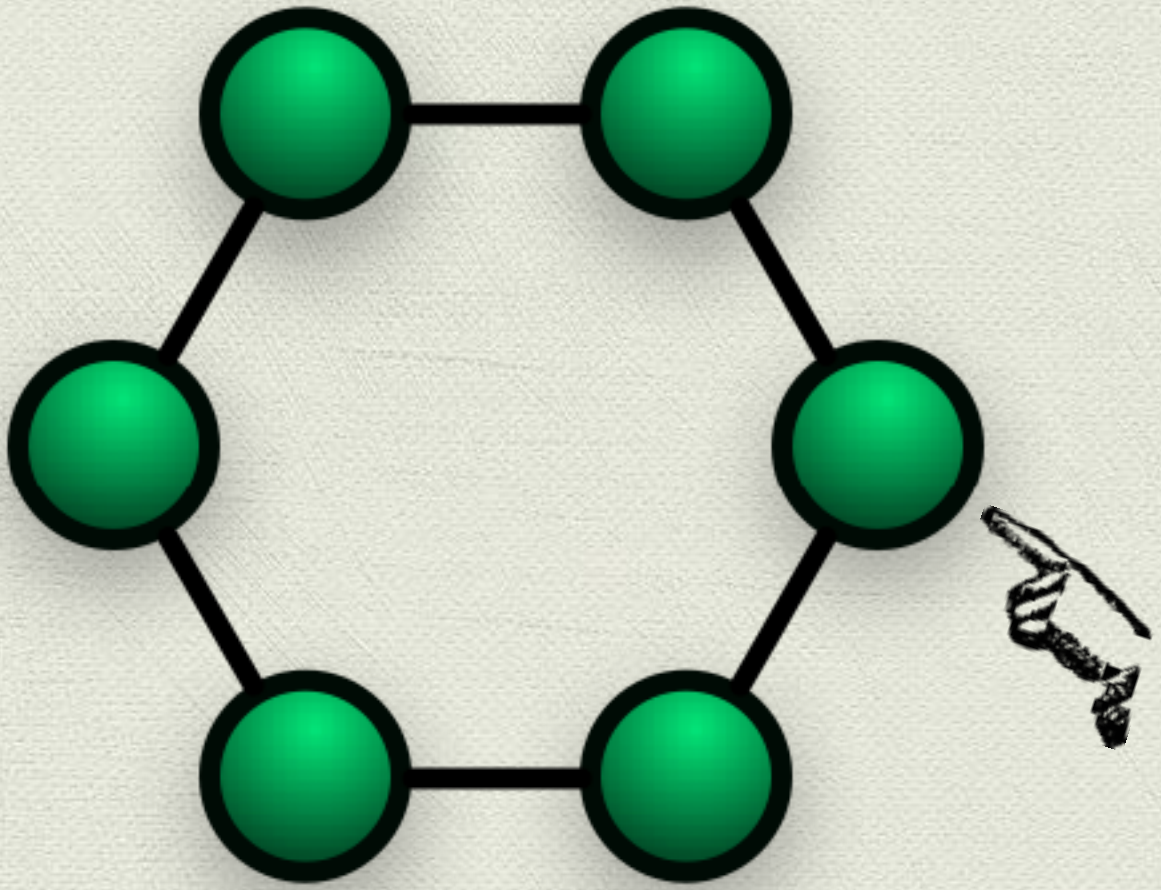
EPSRC research grant EP/P021247/1

The Internet has (almost instantaneously) taken over global communications: from 1% in the year 1993, to 51% by 2000, and more than 97% in 2007

[Hilbert et al, The worlds technological capacity to store, communicate, and compute information. Science, 2011]

Centralised:
Who gets to print?

A Network



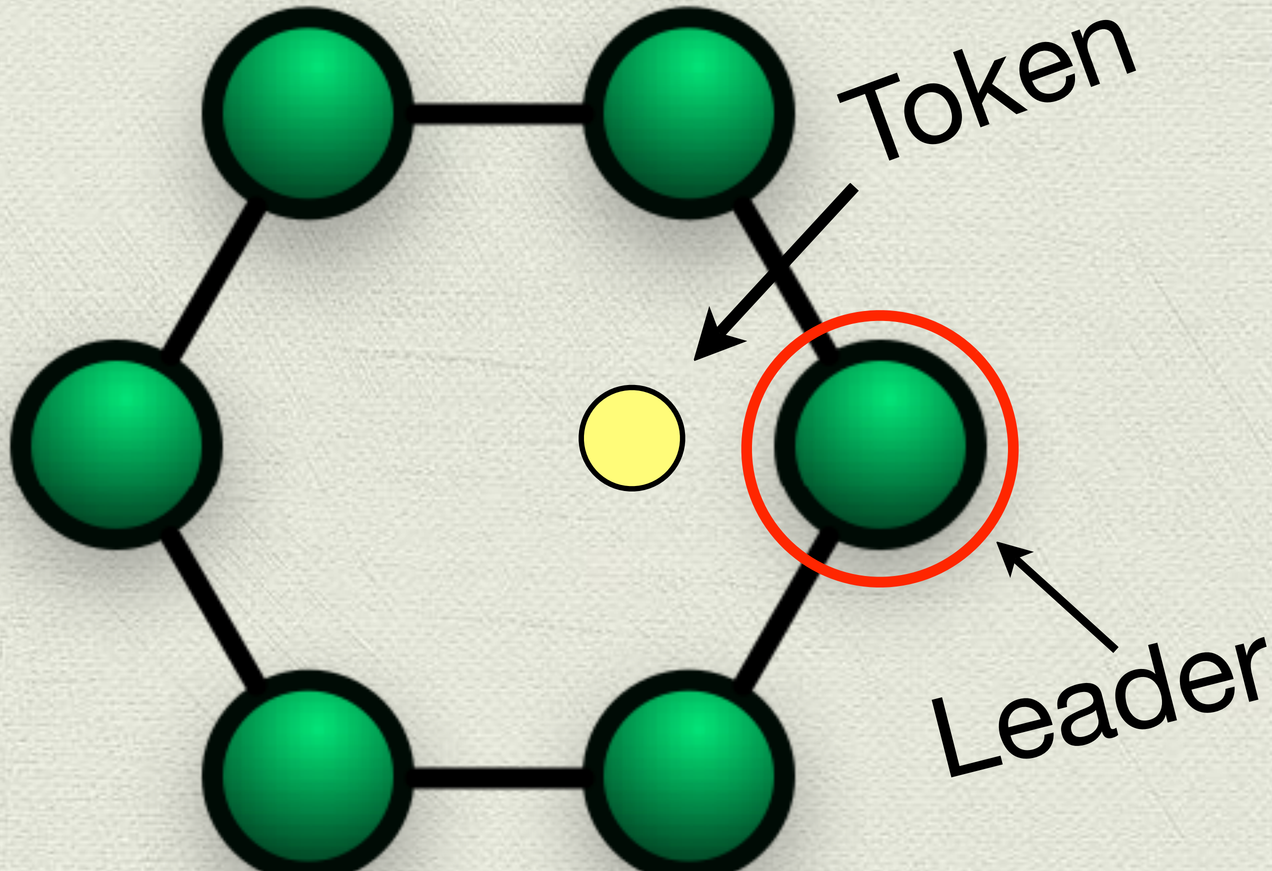
A Printer

<u>Centralised Algorithms:</u>
Single computer with the whole problem instance/data available.

Q:
which one of <u>them</u> will get the printer?

Distributed:
Who gets to print?

A Network



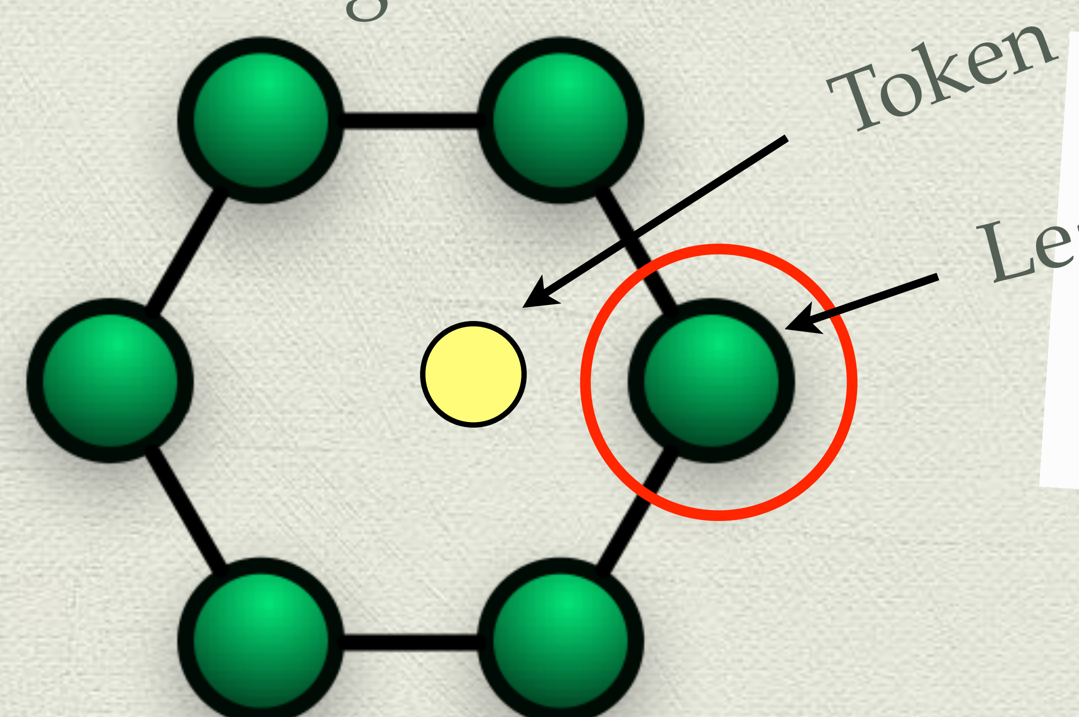
A Printer

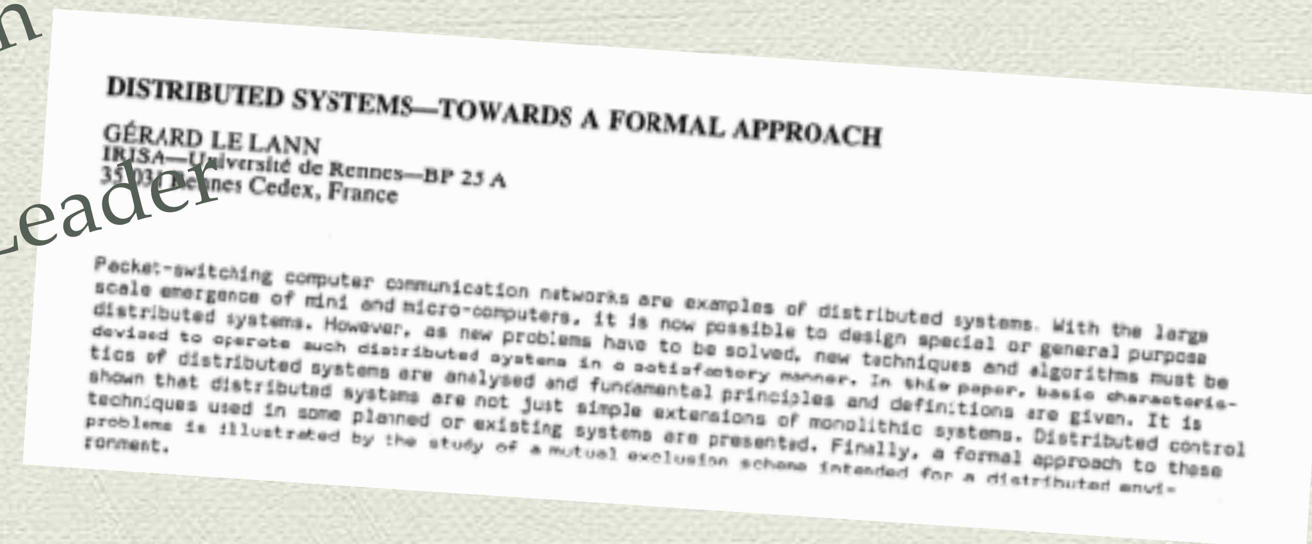
<u>DeCentralised/</u>
<u>Distributed Algorithm:</u>
Multiple `computers' each with it's own local view/data.

Q:
which one of <u>us</u> will get the printer?

A solution: 'Leader Election'!

Token Ring Networks





DISTRIBUTED SYSTEMS—TOWARDS A FORMAL APPROACH

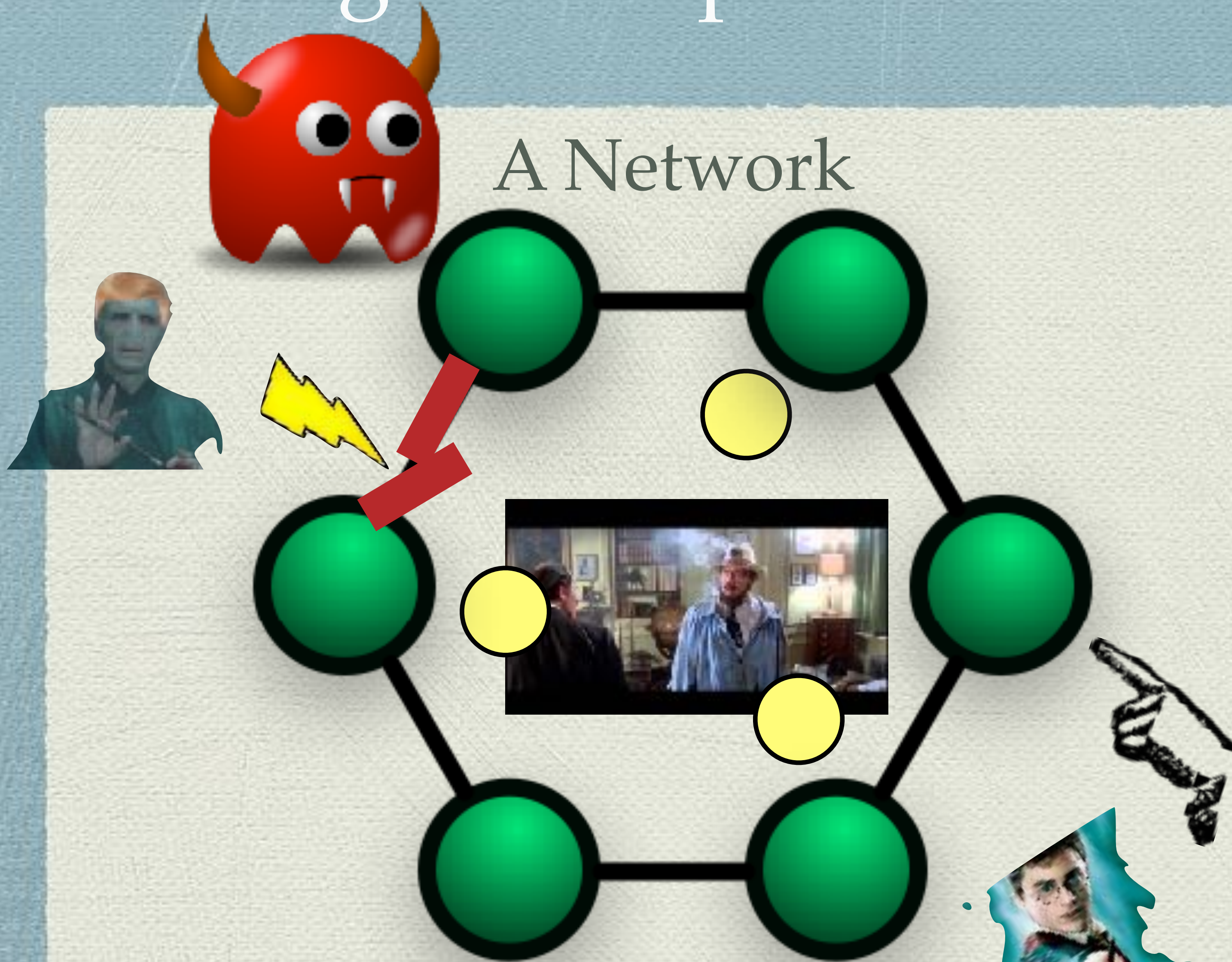
GÉRARD LE LANN
IRISA—Université de Rennes—BP 25 A
35031 Rennes Cedex, France

Packet-switching computer communication networks are examples of distributed systems. With the large scale emergence of mini and micro-computers, it is now possible to design special or general purpose distributed systems. However, as new problems have to be solved, new techniques and algorithms must be devised to operate such distributed systems in a satisfactory manner. In this paper, basic characteristics of distributed systems are analysed and fundamental principles and definitions are given. It is shown that distributed systems are not just simple extensions of monolithic systems. Distributed control techniques used in some planned or existing systems are presented. Finally, a formal approach to these problems is illustrated by the study of a mutual exclusion scheme intended for a distributed environment.

- Fundamental problem (Le Lann, 1977)
- Massively useful e.g. in large scale networks (P2P(Akamai), ad-hoc, sensor etc.)
- Symmetry breaking

Distributed in a `faulty/dynamic' environment: Who gets the printer?

A Network



A Printer

<u>Fault-Tolerant/Dynamic Algorithms:</u>

In faulty/dynamic environments

Q:

which one of <u>us</u> will get the printer despite failures or changes?

The Adversary!

- Theoretical abstraction for opponent(s), environment, internal faults and limitations
- Success against stronger adversaries => Stronger guarantees
- Allows us to reason about faulty and/or dynamic systems

Example publications vs. Adversary!

None	Edge Dynamic	Node Dynamic (Self-healing)	Byzantine	Game Theoretic	Memory (Self-Stablisation)
ICDCN '13, PODC '13, J. ACM '15 J. TCS '15 (Leader Election)	DISC '12, PODC '12,	IPDPS '08, '14, PODC '08, '09, '11, INFOCOM '12, ICDCN '16, J. Dist Computing '12, '14, '16 J. TCS '16	ICDCN'11	DISC '11, INFOCOM '12 AAAI '13	-

Main self-healing themes

- Resilient Distributed Architectures by Self-healing (~P2P topology maintenance)

COSHER

- Low Memory devices: *Compact Routing over Compact Self-healing Networks* ~ Internet of Things

- Under Investigation: Self-Healing Software Defined Networks, Social resilience (e.g. Terror networks)

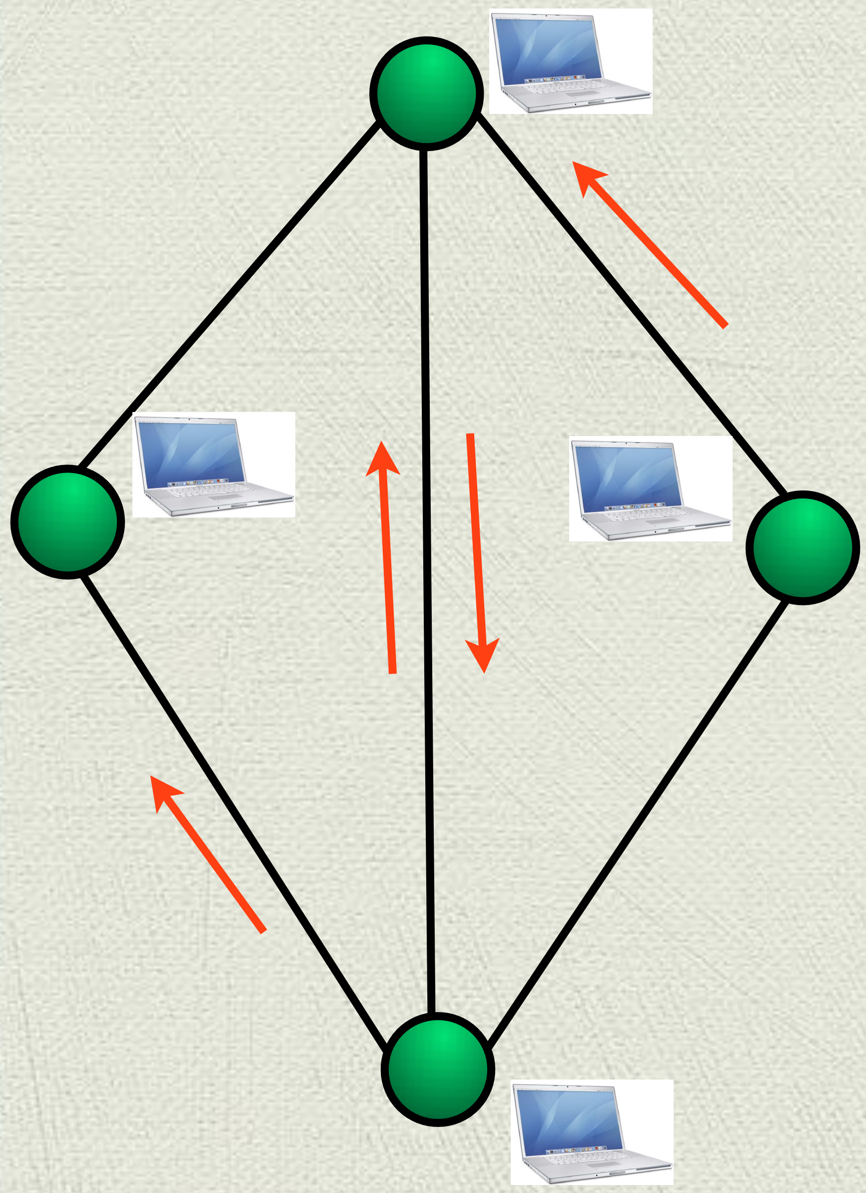
Self-healing

- A self-healing system, starting from a correct state, under attack from an adversary, goes only temporarily out of a correct state.
- Under attack, maintain certain properties within acceptable bounds.

Autonomic Computing

- IBM's autonomic computing initiative
- Self-CHOP

Distributed Computing: Message Passing model

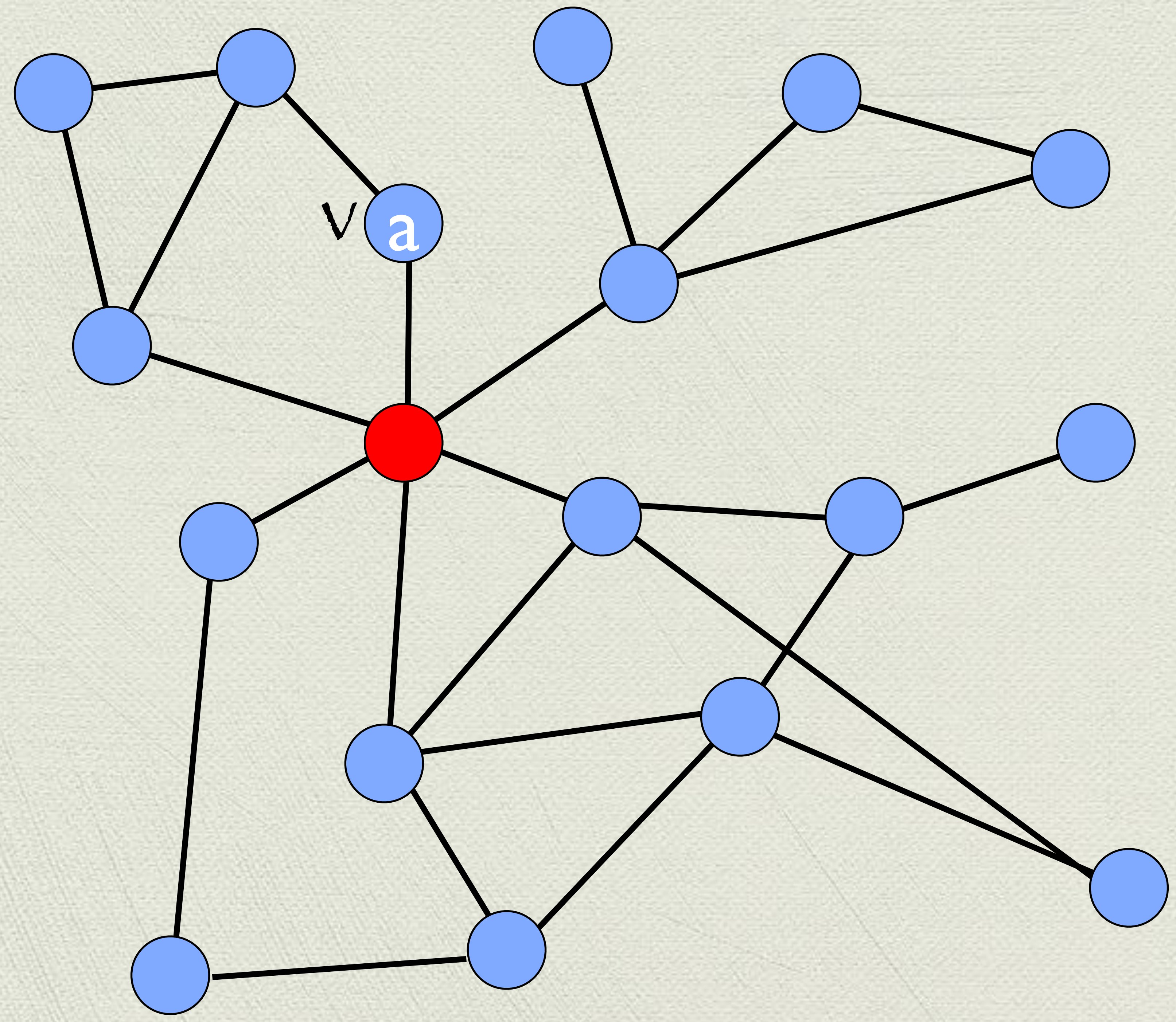


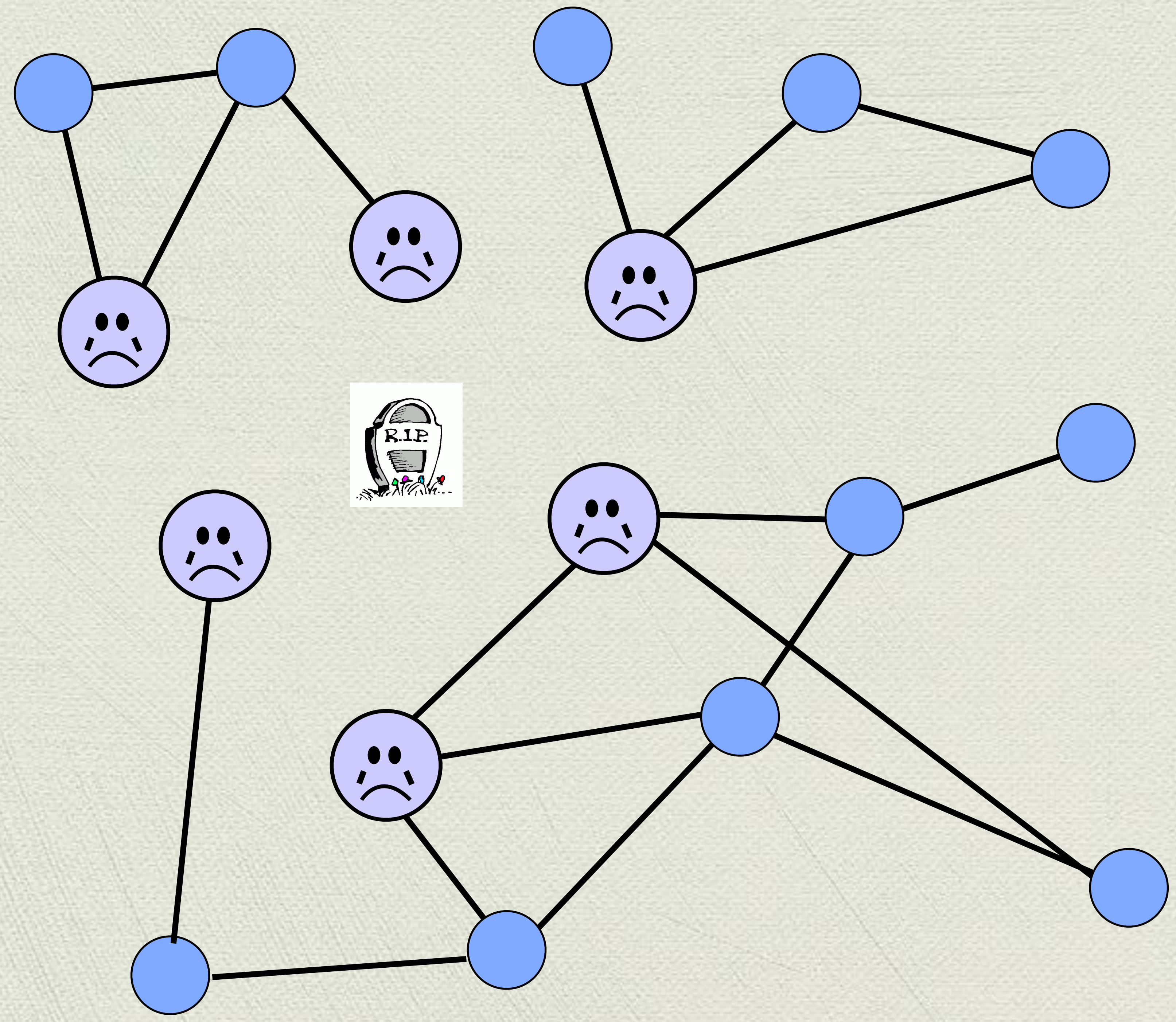
- Suitable for Networks!
- Each node is a processor
- Communication is by sending messages along edges
- Only local knowledge initially
- Metrics:
    - time (# parallel rounds)
    - # messages exchanged etc.

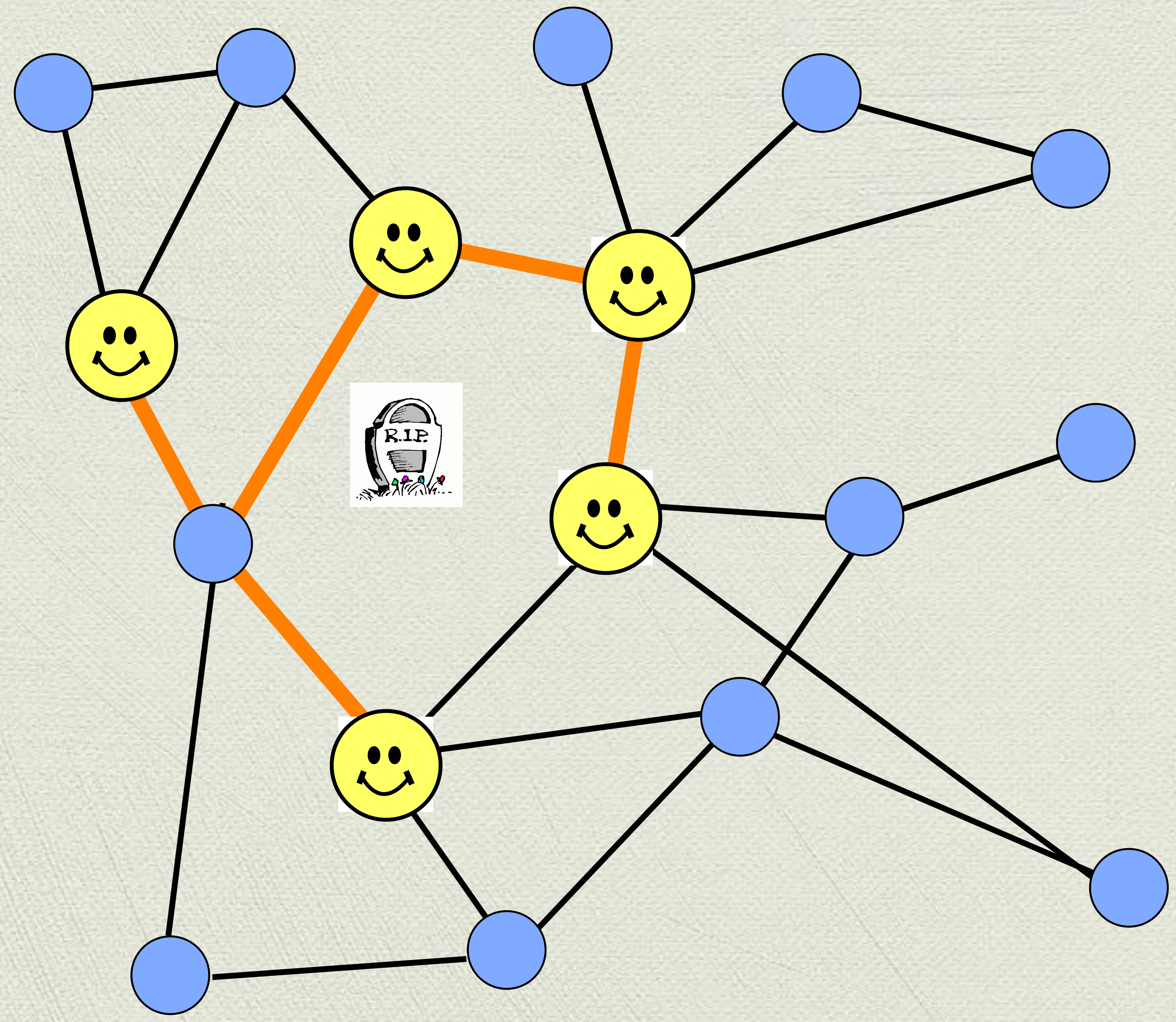
Self-Healing Model

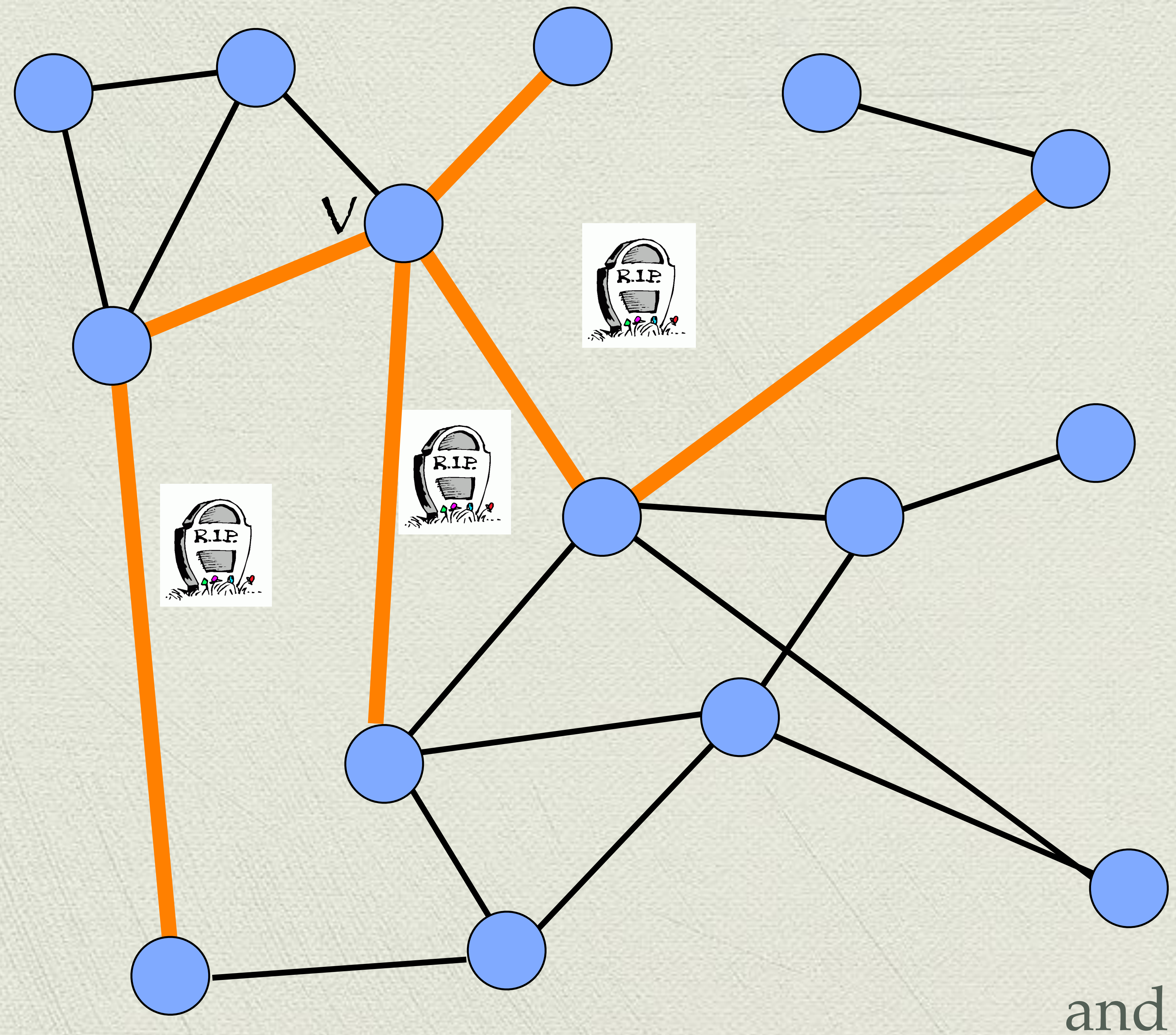
- Start: a distributed network G
- Attack: An adversary inserts or deletes one node per round
- Healing: After each adversary action, we add and/or drop some edges between pairs of nearby nodes, to "heal" the network

Self-Healing Illustration




R.I.P.

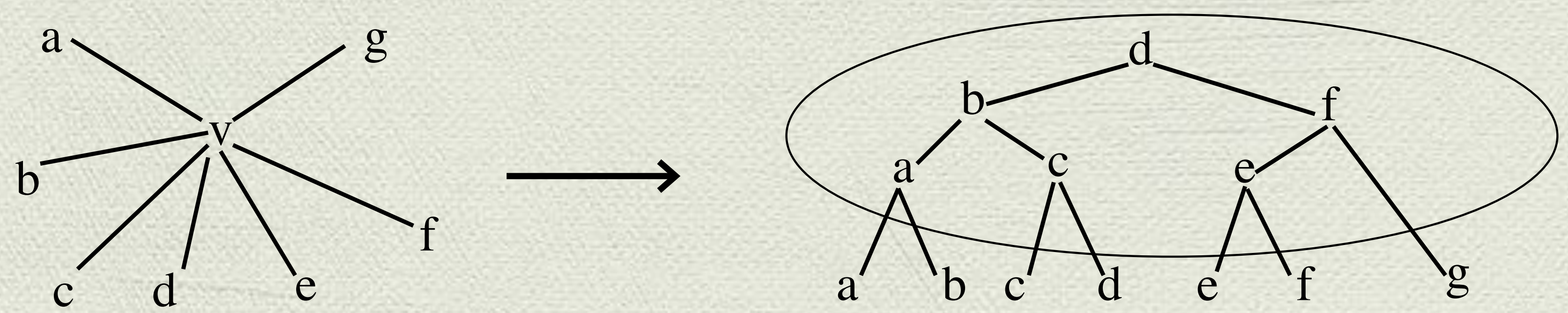

R.I.P.



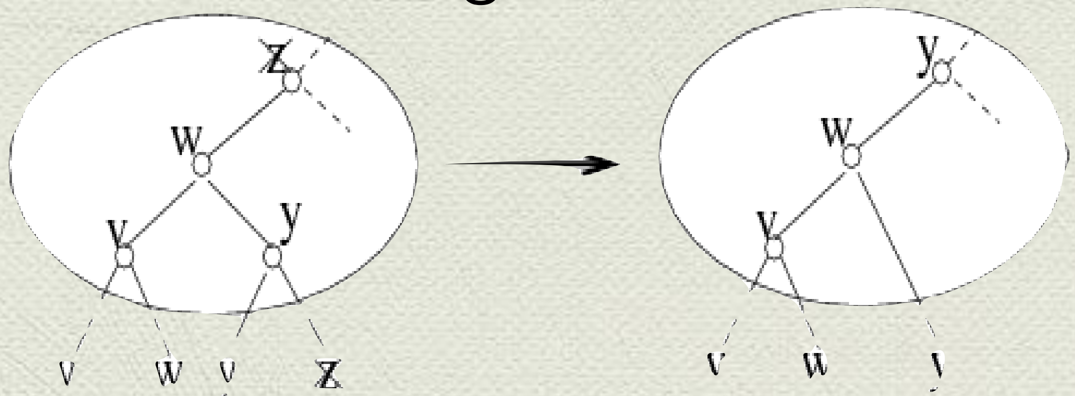
and so on ....

*Forgiving Tree**: Basic Technique

On a rooted spanning tree of G:



- Replace (**non-leaf**) v by a **Reconstruction Tree** (**RT**) of virtual nodes (in oval). The 'real' neighbors are the leaves of the tree.



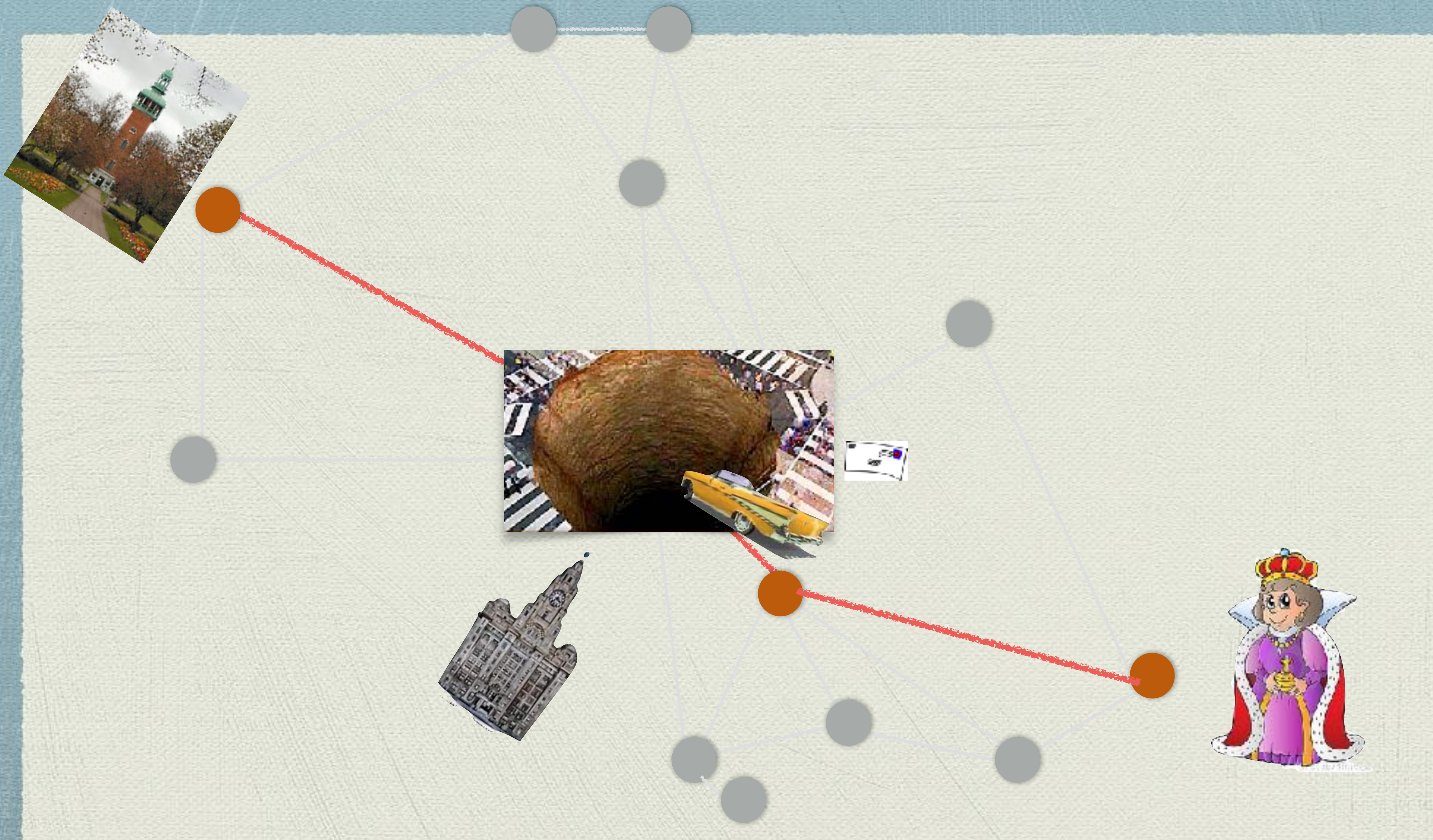
- (**leaf** deletions) *Short-circuiting* a redundant virtual node

*Efficient implementation by '**wills**'

**Principles of Distributed Computing (PODC) 2008*

Routing under pressure….

Sometimes the messenger may have an accident …



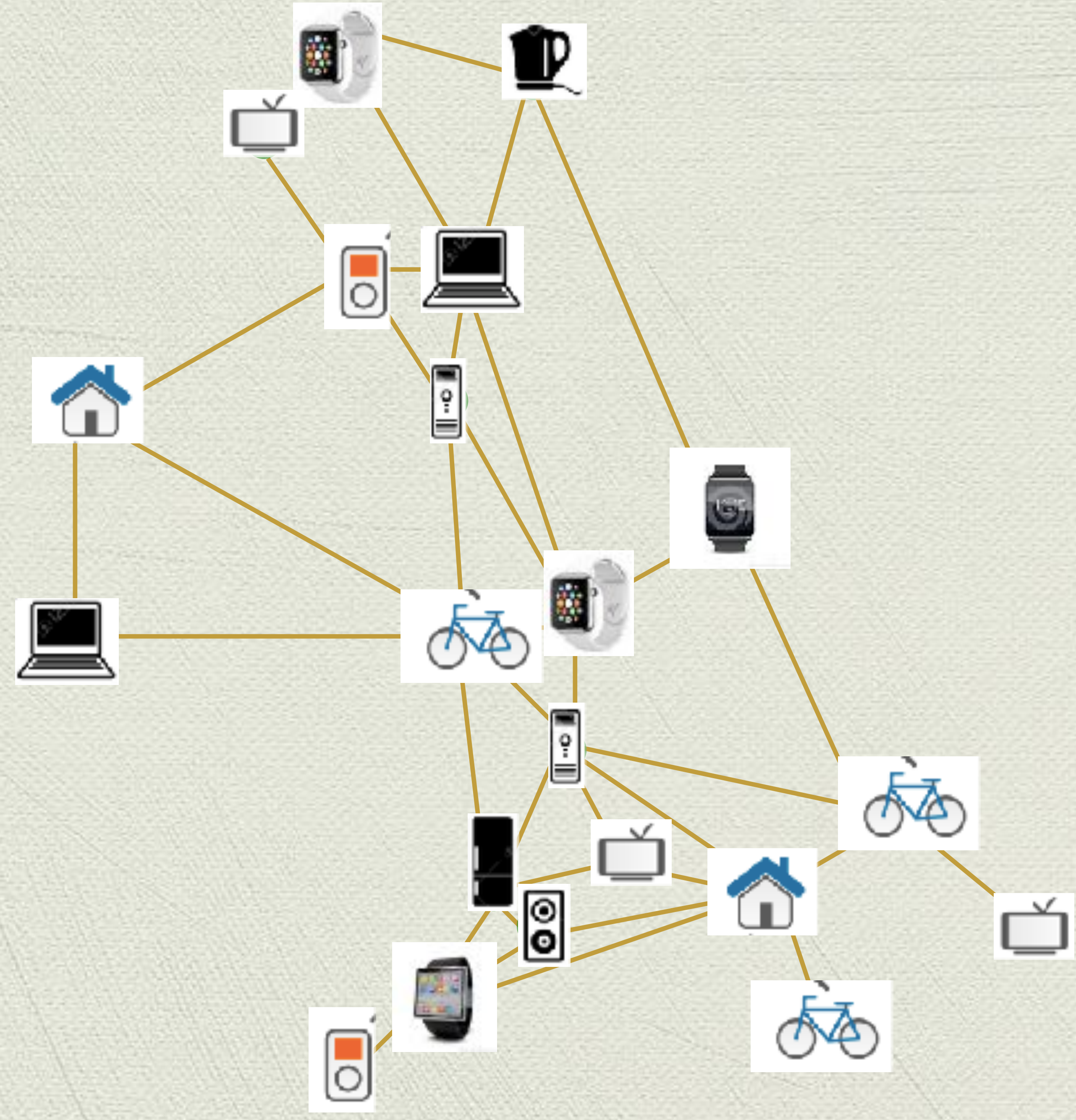
Problem 1a

- How to Route a message from a sender u to a receiver v despite failures in the network?

Problem 1b

- How to Route a message from a sender u to a receiver v in a network of low memory nodes?

Imagine IOT / Overlay / large networks of small devices



History of ‘provably good’ compact routing

1985 Santoro and khatib	*The Computer journal*
:	
1988 Peleg and Upfal	*STOC*
:	
1999 Cowen	*SODA*
2001 Thorup andZwick	*SPAA*
:	
2002 Korman et al	*STACS*
2013 Chechik	*PODC*

Problem 2

- How to Route a message from a sender u to a receiver v despite node failures in a network of nodes with low memory?

Problem 2a

- How to Route a message from a sender u to a receiver v despite node failures in a reconfigurable/overlay network of nodes with low memory?

Compact/ Bounded memory Self-healing Routing

- ***Compact Routing Messages in Self-Healing Trees*** [£]
- Adversary is Deletion only
- Each node uses less than linear (i.e. o(n)) memory (=**Compact**)[*, !]

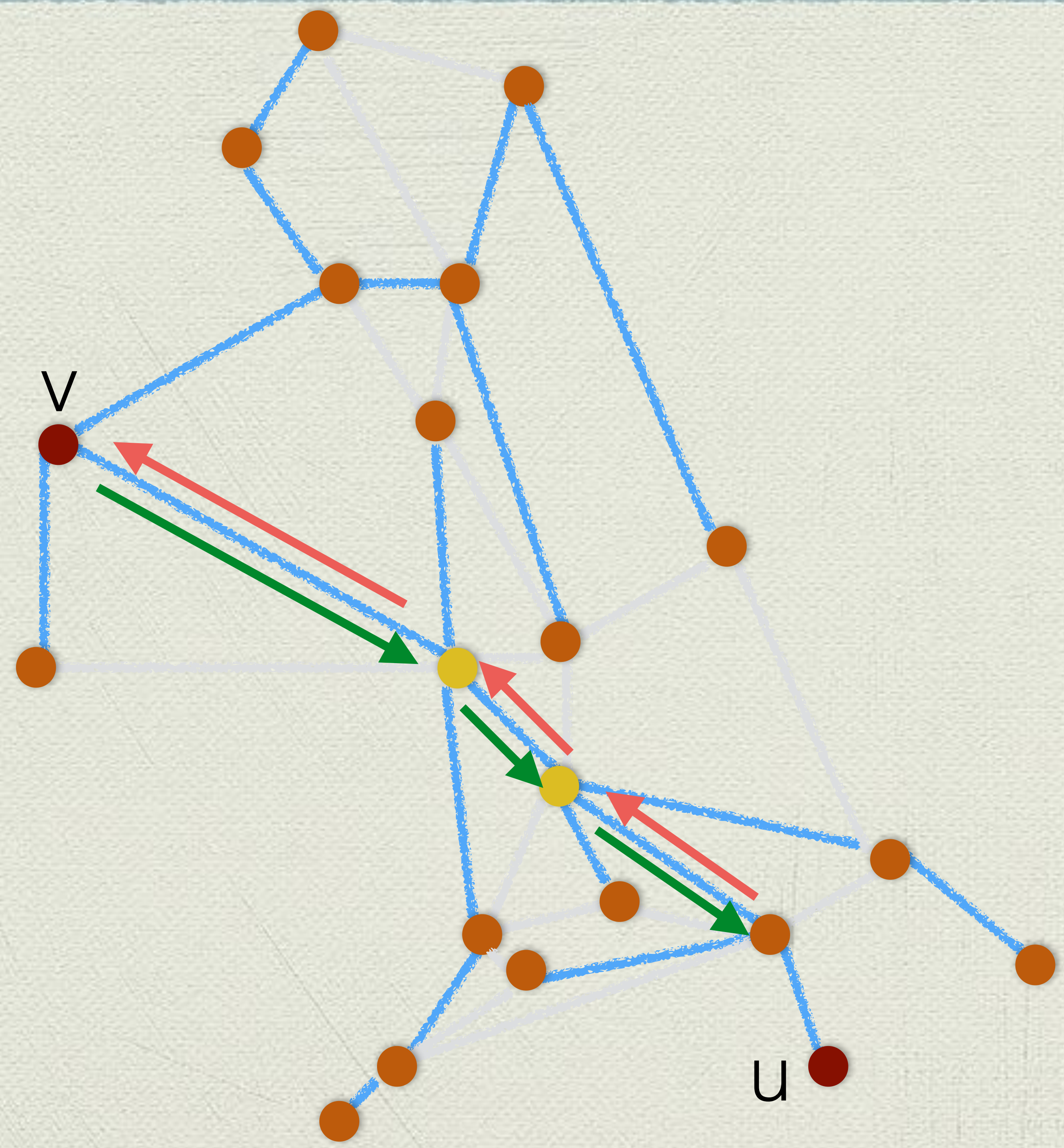
* Implies finite computation at individual nodes!

! Nodes cannot even store list of their neighbours!

[£] ***17th International Conference on Distributed Computing and Networking (ICDCN 2016); Theoretical Computer Science (Journal version), 2017***

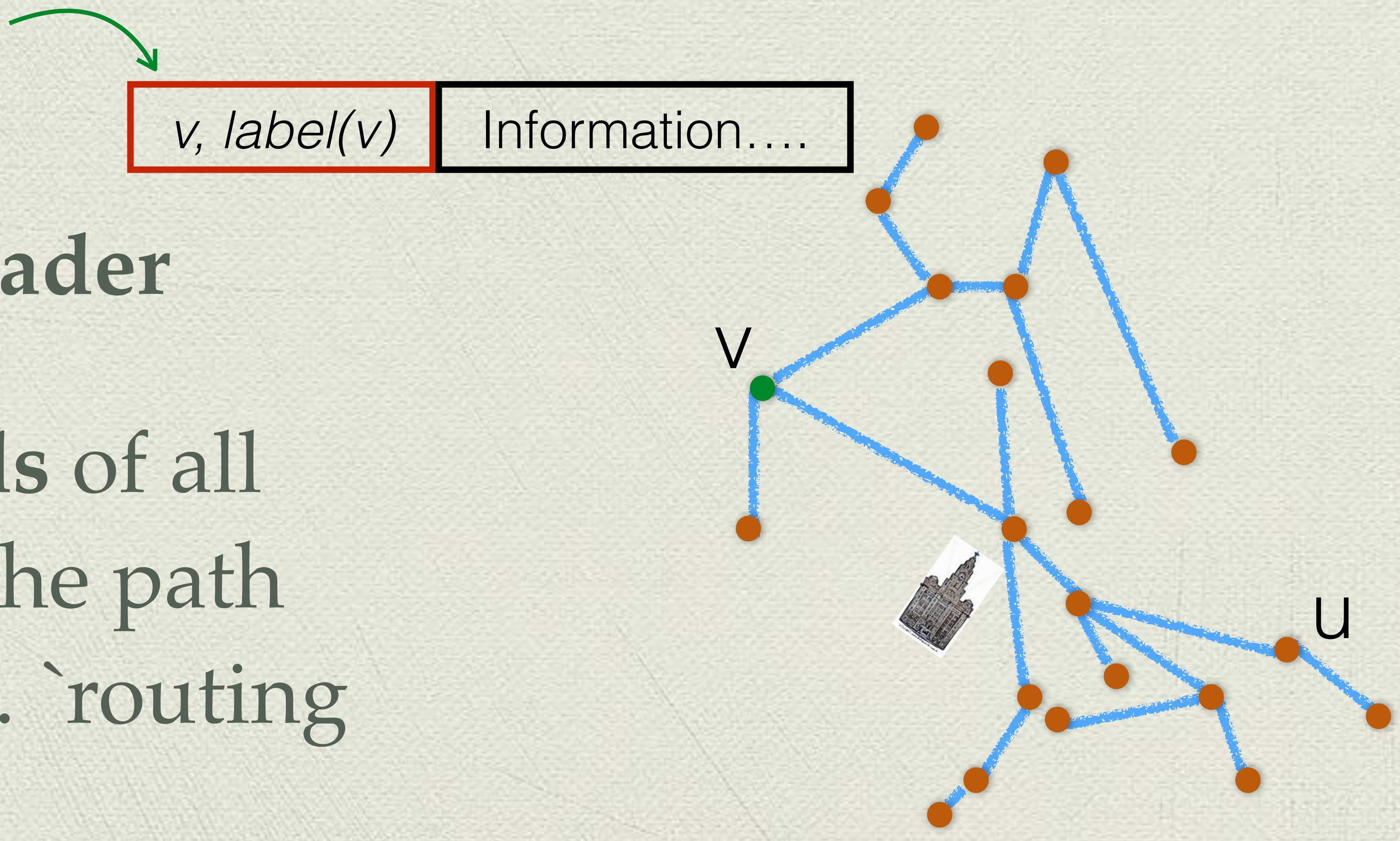
Solution: Intuition

1. A compact Routing Scheme 𝕾 (no failures)



𝔖: Static Routing

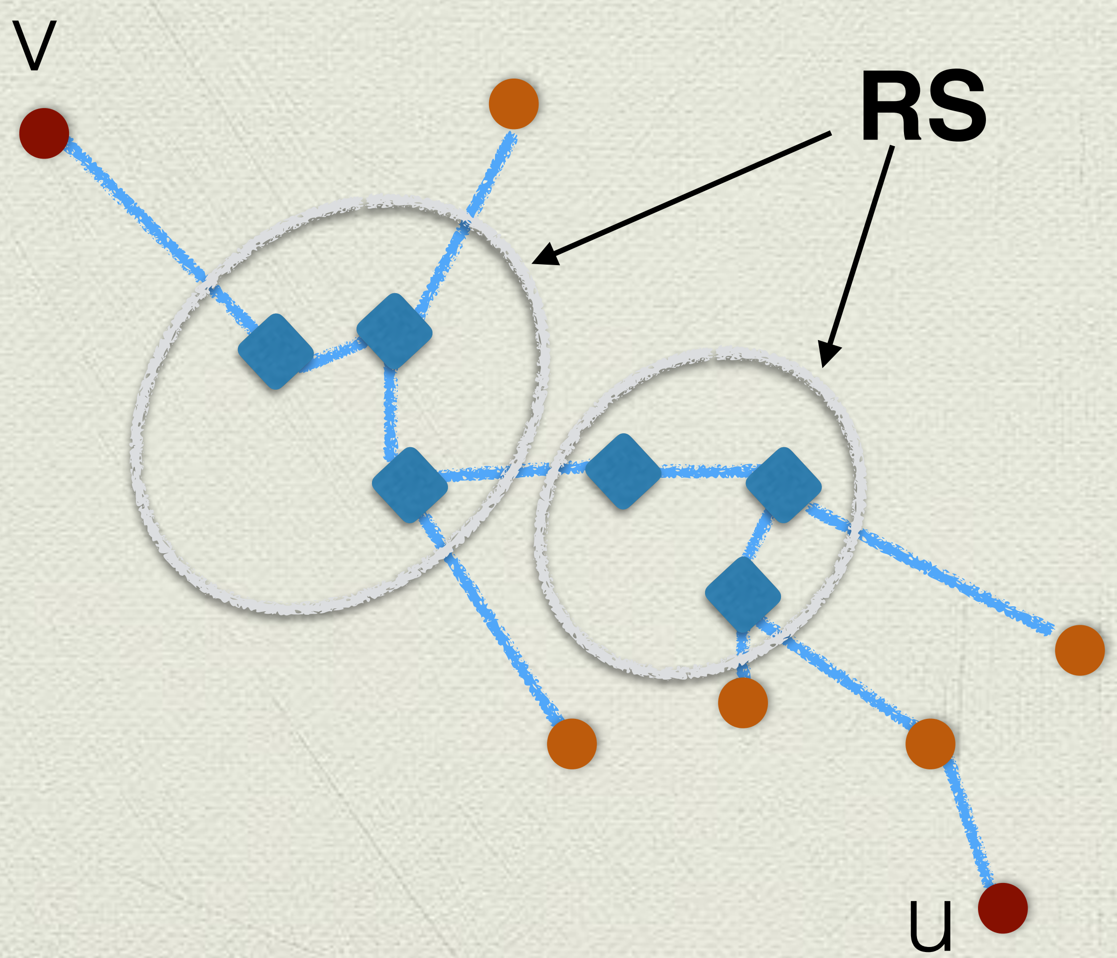
- u wants to send a packet to v. Two pieces of information:
  - **Packet Header**
  - **Local fields** of all nodes on the path p(u,v) (i.e. `routing tables')



Solution: Intuition

1. A compact Routing Scheme $\mathfrak{S}$ (no failures)

2. If node failure, compactly self-heal by replacing node with reconstruction structure (RS) following Routing Scheme $\mathfrak{R}$.

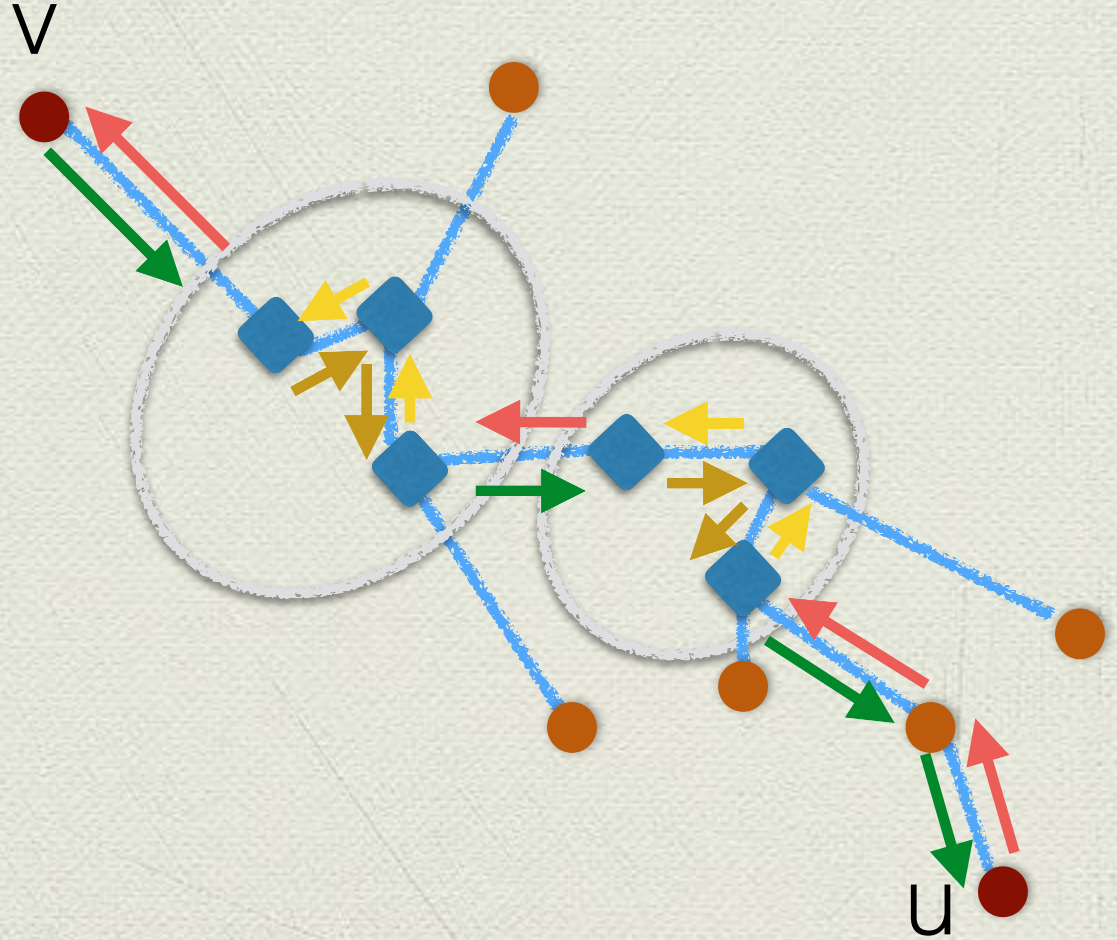


Solution: Intuition

1. A compact Routing Scheme $\mathfrak{S}$ (no failures)

2. If node failure, compactly self-heal by replacing node with reconstruction structure (RS) following Routing Scheme $\mathfrak{R}$.

3. $\mathfrak{S} + \mathfrak{R}$: Transparent integrated compact routing with small **stretch....**



Our Implementations / Algorithms

1. A compact Routing Scheme 𝔖 (no failures)

TZ: Modified Thorup-Zwick^ Routing over trees

2. If node failure, compactly self-heal by replacing node with reconstruction structure (RS) following Routing Scheme ℜ.

COMPACTFT: Compact version of ForgivingTree*

→ *Binary Search Tree Traversal*

3. 𝔖 + ℜ: Transparent integrated compact routing with small **stretch....**

COMPACTFTZ: SH Compact routing

^M. Thorup and U. Zwick. Compact routing schemes. In SPAA, pages 1–10, 2001.
*Tom Hayes, Navin Rustagi, Jared Saia and Amitabh Trehan, "The forgiving tree: a self-healing distributed data structure", in Principles of Distributed Computing (PODC), 2008.

Bringing it all together

1. Preprocessing (regular memory): BFS spanning tree of the network + DFS labelling and TZ setup + CompactFT data structure

Do forever
{

2. COMPACTFT: On node deletion, neighbours replace deleted node by its RT and maintain SH tree
3. $\mathfrak{H} + \mathfrak{R}$: TZ on real nodes + BST routing on RTs (virtual nodes)

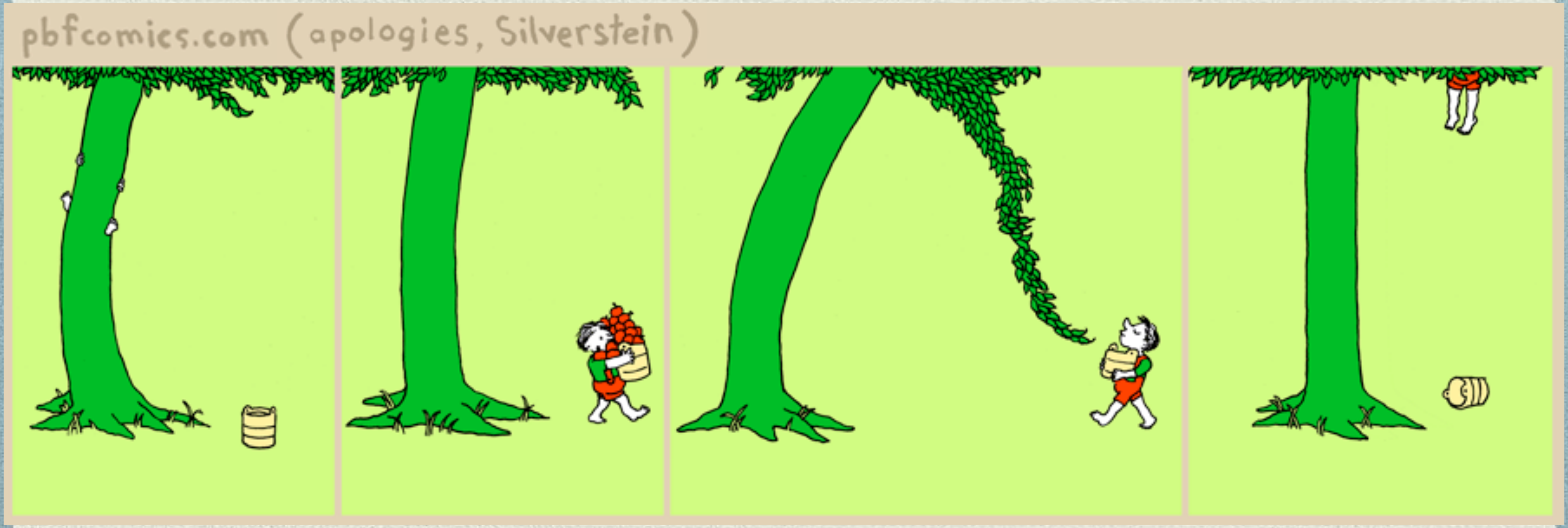
}

The dream:

- A unified theory of 'dynamicity' in networks
- Engineering solutions despite and along with the dynamicity…

Conclusion

- What can we do in practice?
    - Compact routing?
    - Resilience via self-healing? (e.g. Self-Healing SDN)
    - Compact Self-healing routing?



Thank You