BRISA:
Combining Efficiency and Reliability in Epidemic Data Dissemination

Miguel Matos[*], Valerio Schiavoni[+], Pascal Felber[+]
Rui Oliveira[*] and Etienne Rivière[+]

[*]INESC TEC & University of Minho, Portugal
[+]Université de Neuchâtel, Switzerland

miguelmatos@di.uminho.pt

International Parallel & Distributed Processing Symposium
24 May, 2012

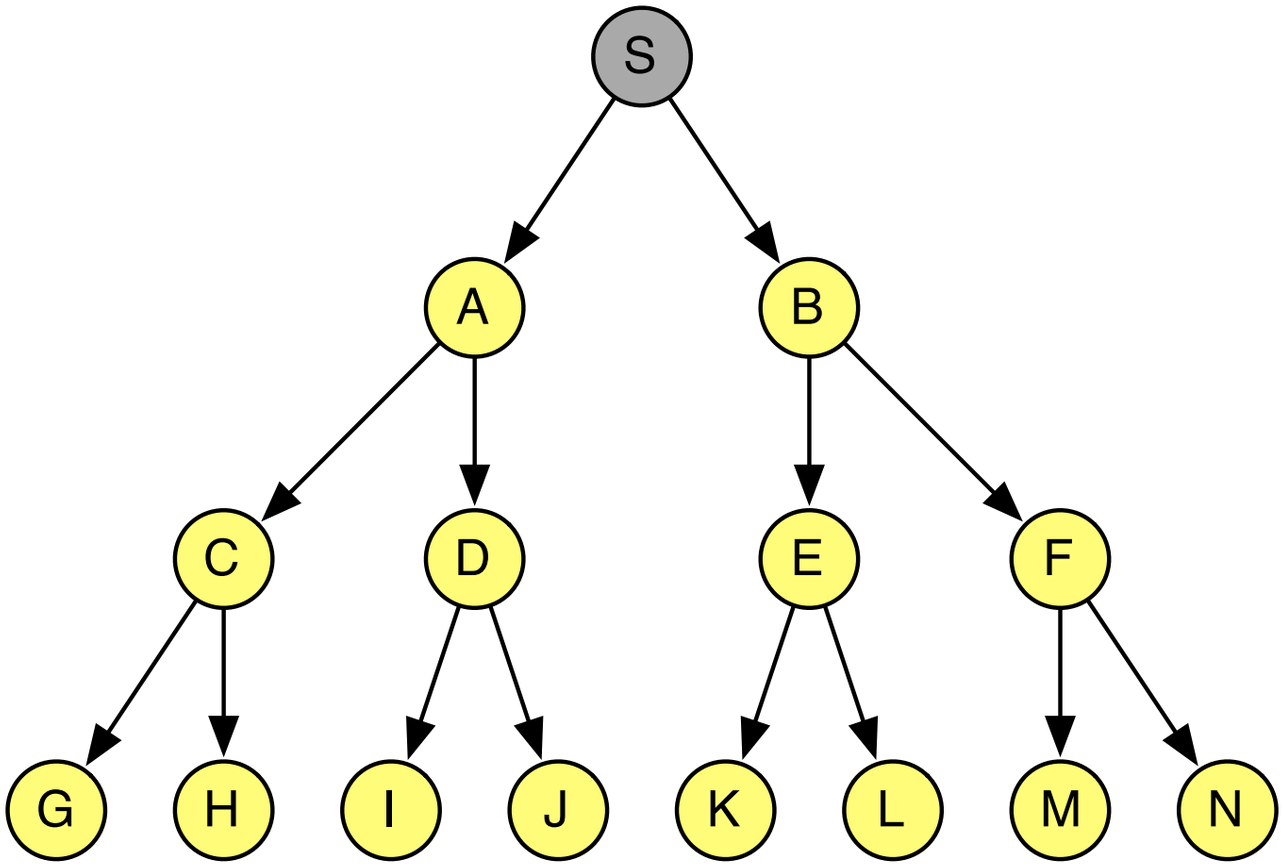
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

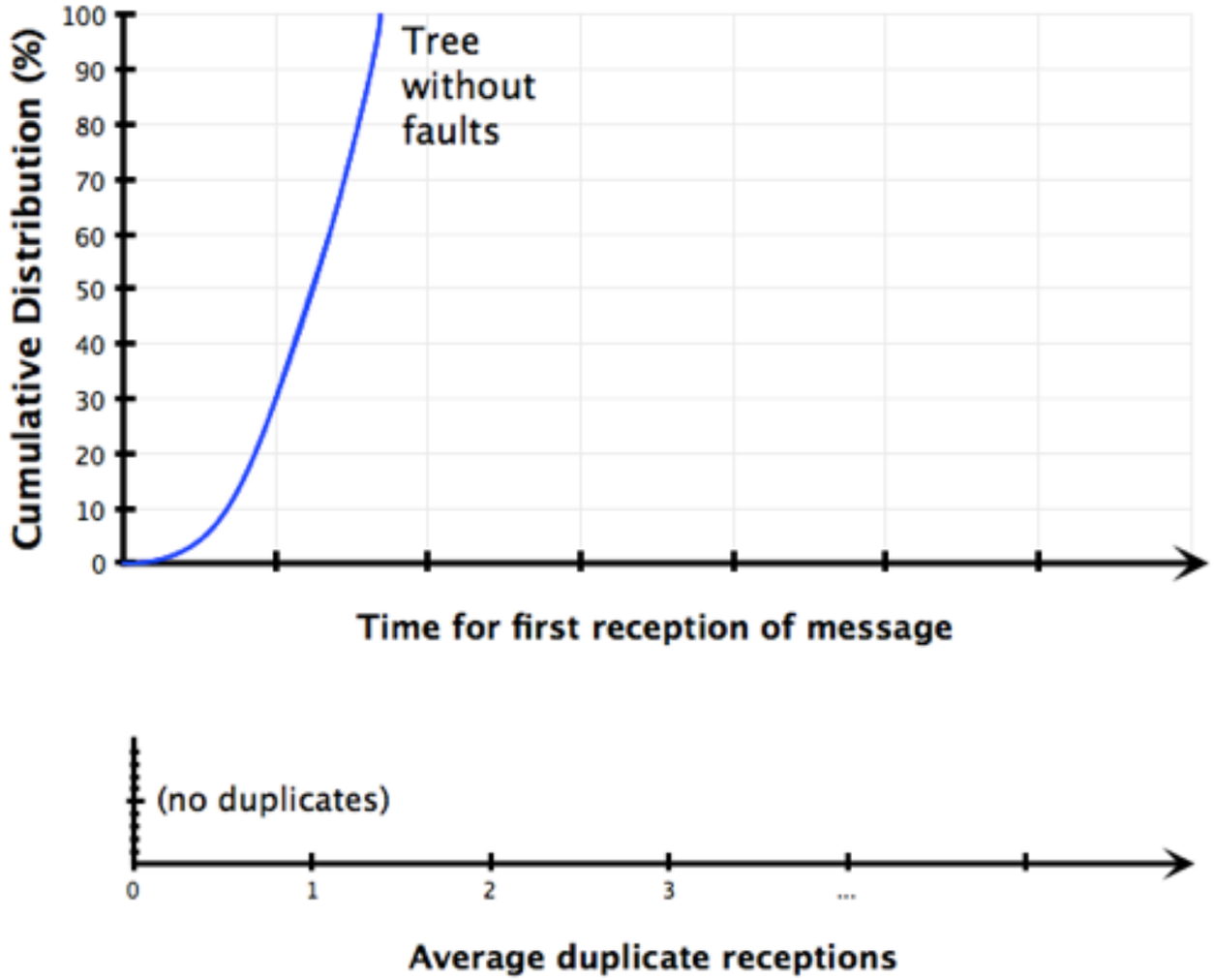
UNIVERSITÉ DE NEUCHÂTEL

Motivations

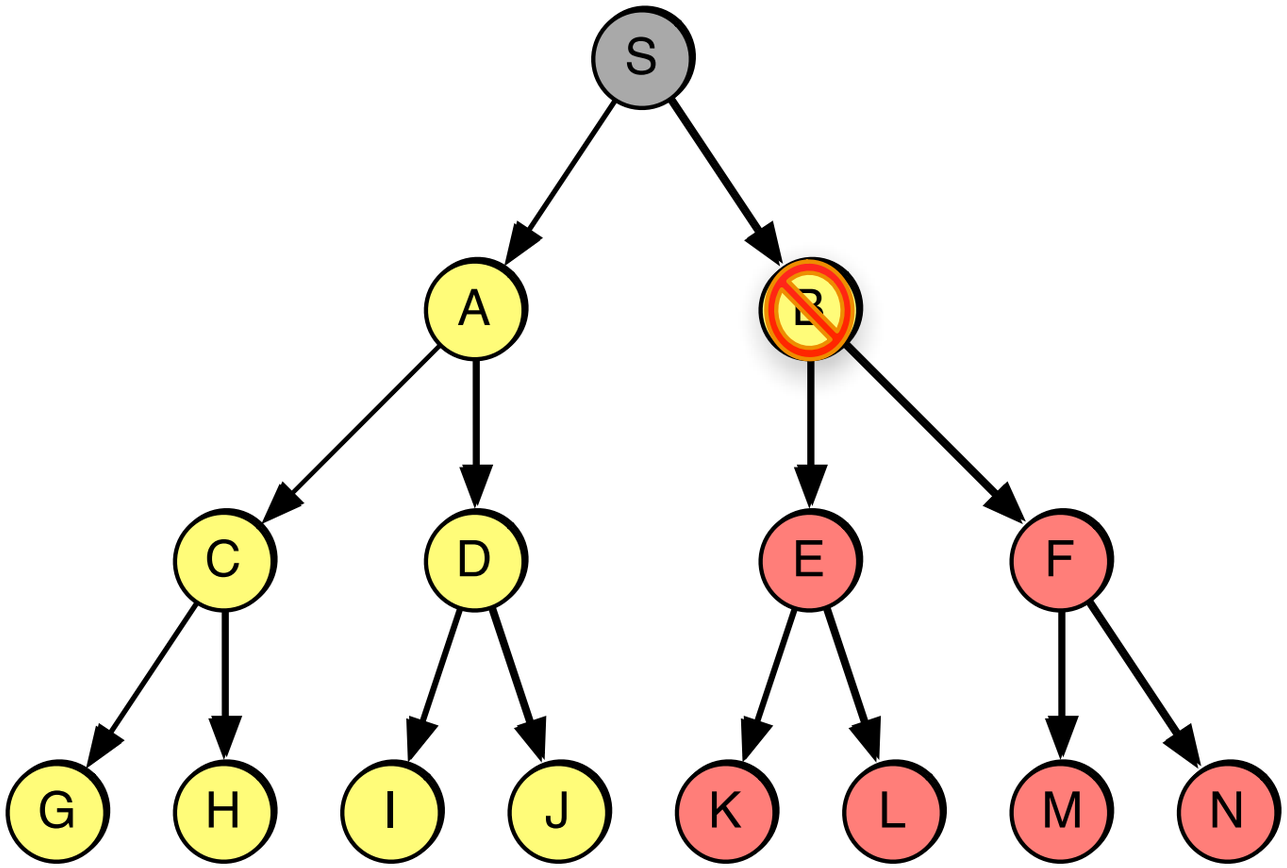
- Broadcast of small or medium sized messages
- Challenging environment
  - (very) Large scale
  - Highly dynamic
- Performance criteria
  - Low delays for all
    - Distribution of first reception time
  - Low overhead (overlay construction)
  - Low message count (for disseminations)
- + Simplicity!

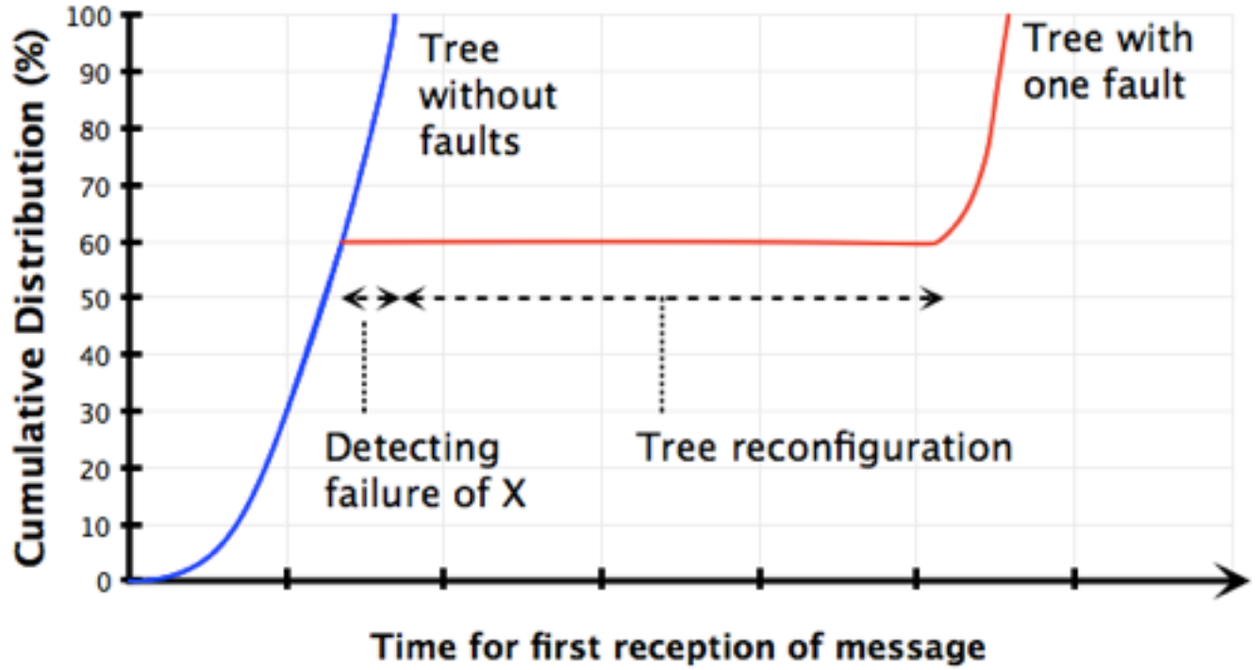
Using a tree?

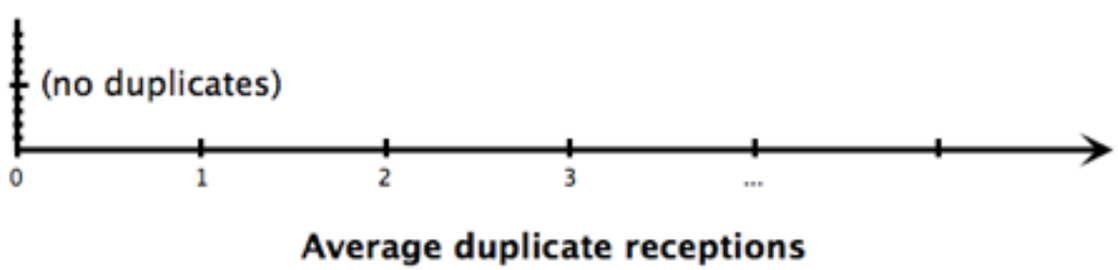

S
A
B
C
D
E
F
G
H
I
J
K
L
M
N


Cumulative Distribution (%)
100
90
80
70
60
50
40
30
20
10
0
Tree without faults
Time for first reception of message
(no duplicates)
0
1
2
3
...
Average duplicate receptions

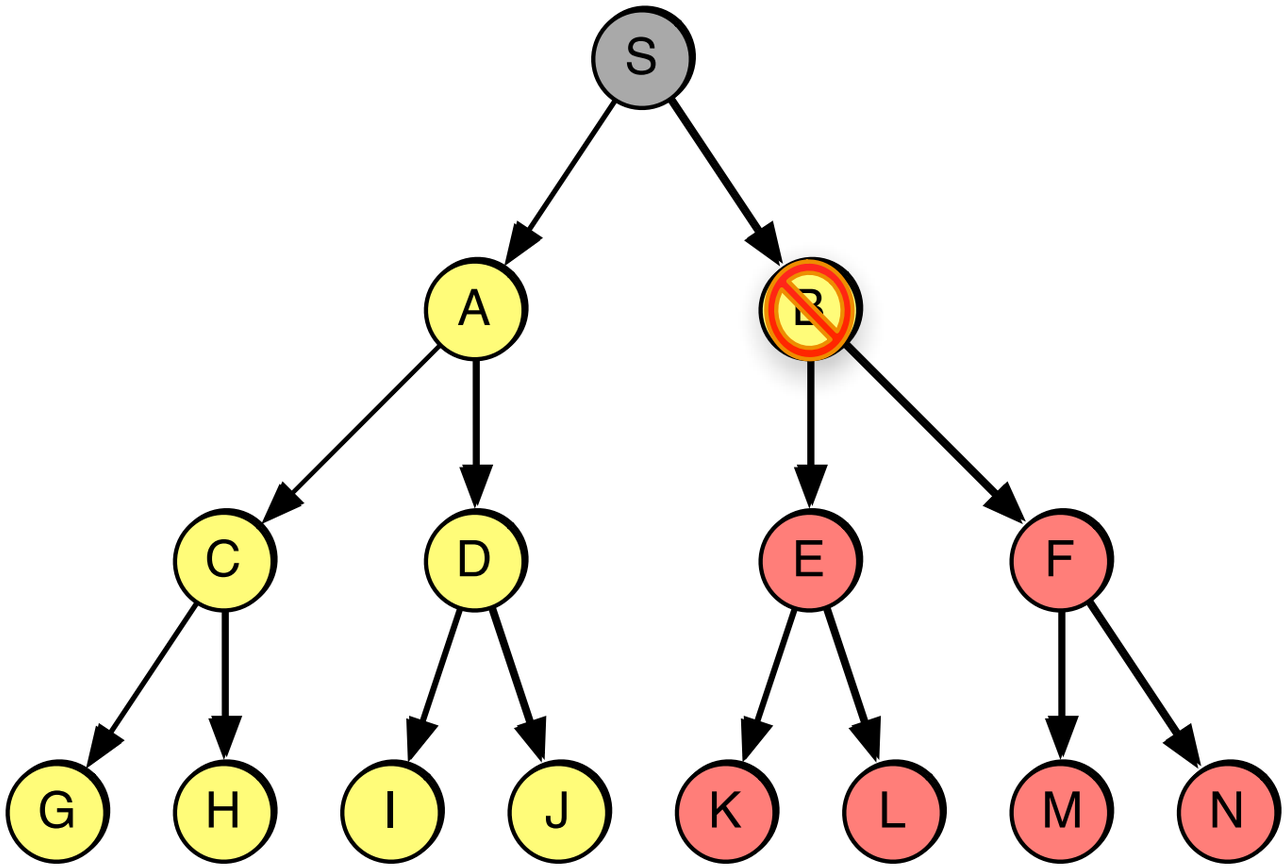
Faults happen

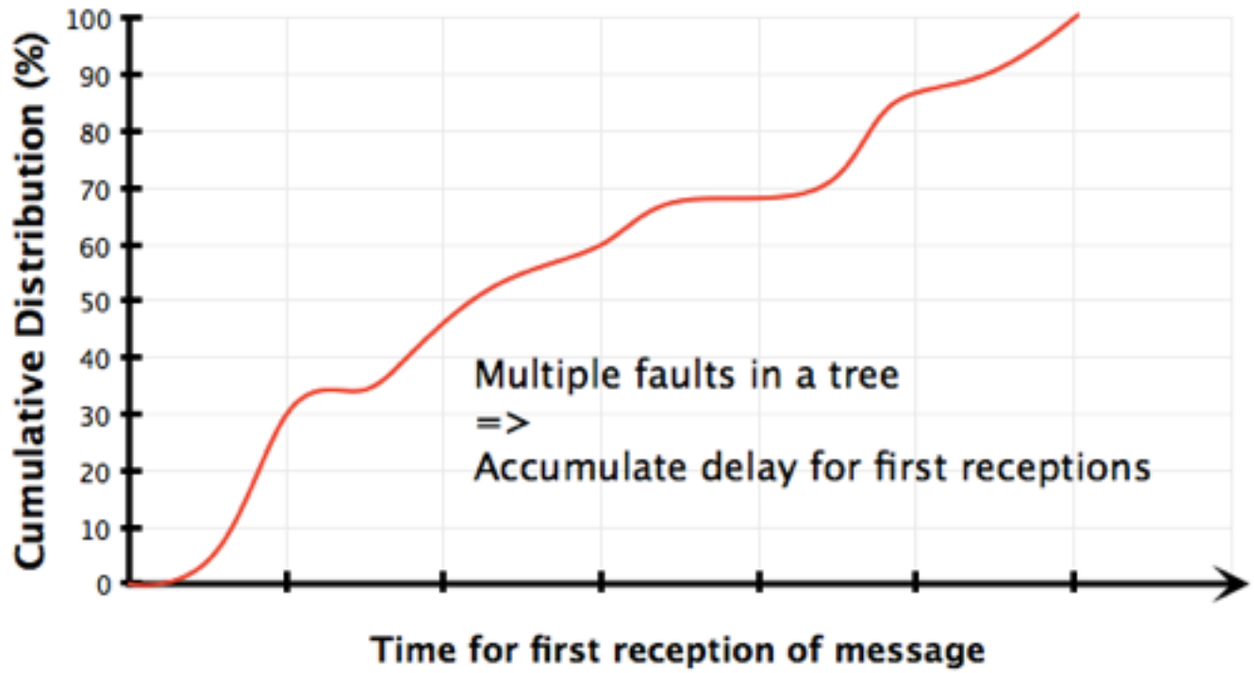


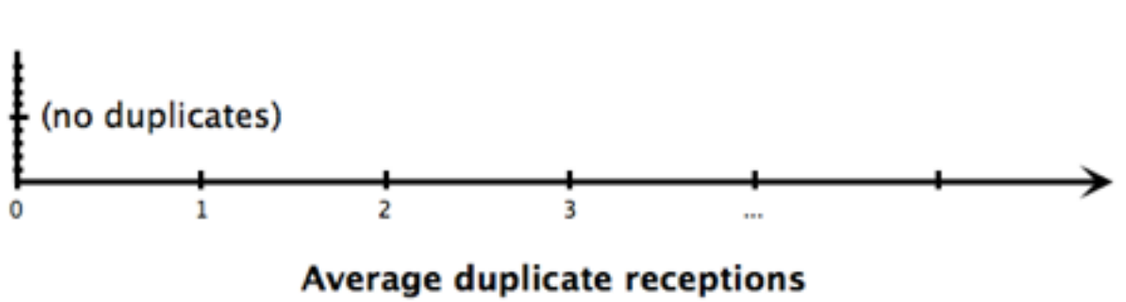




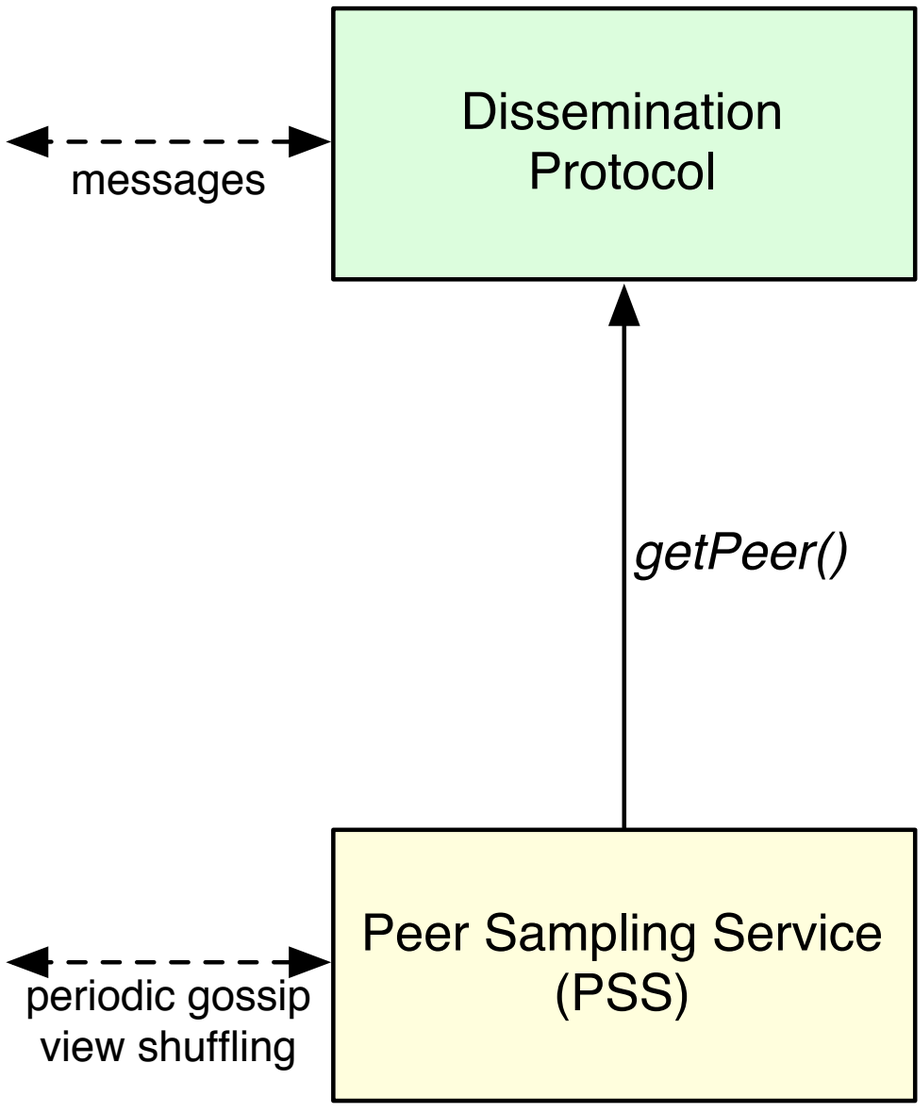
Faults cumulate







Classic epidemic-based dissemination



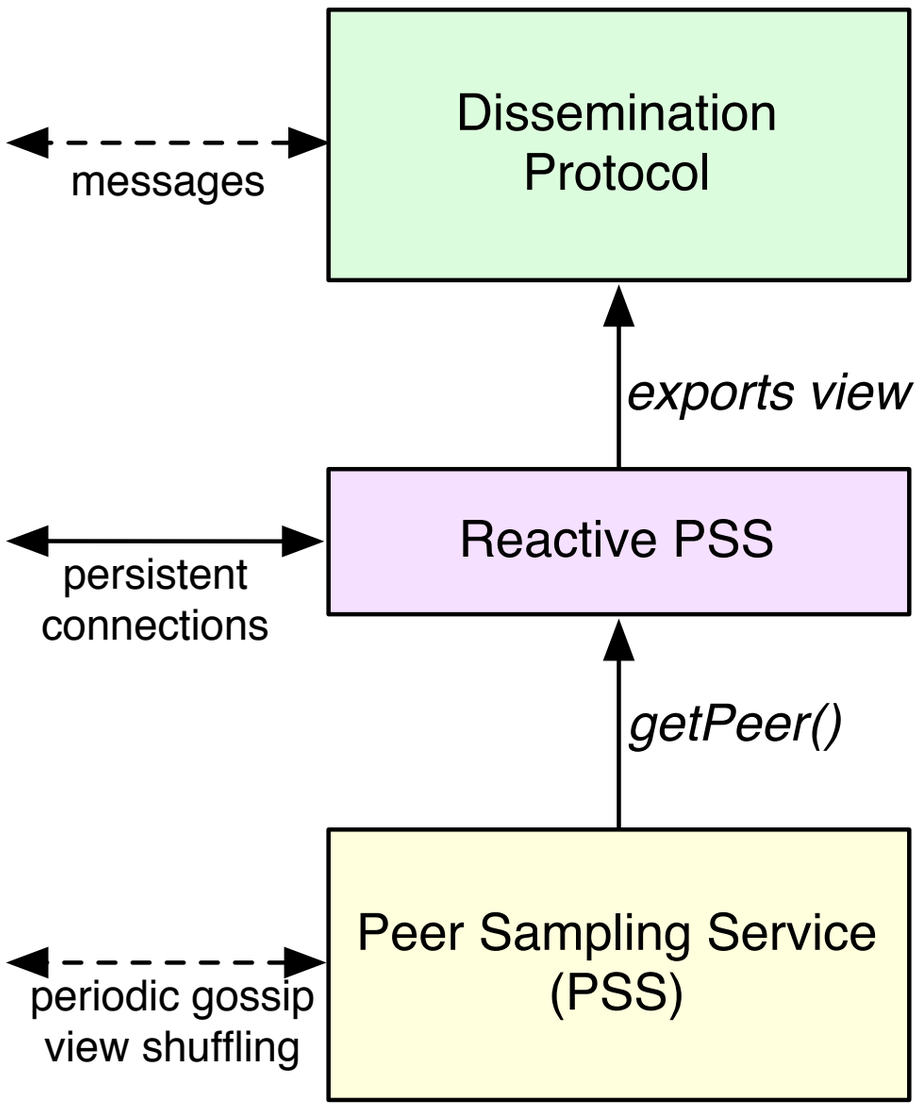
Flooding: send message to all nodes upon first reception

=> Probabilistic Broadcast

Provides constant *stream* of random peers

=> Membership Management
=> Avoids Partition
=> Insert new peers, forget old ones

Epidemic-based dissemination on reactive PSS



Deterministic flooding: send message to all nodes on first reception

=> Total Broadcast

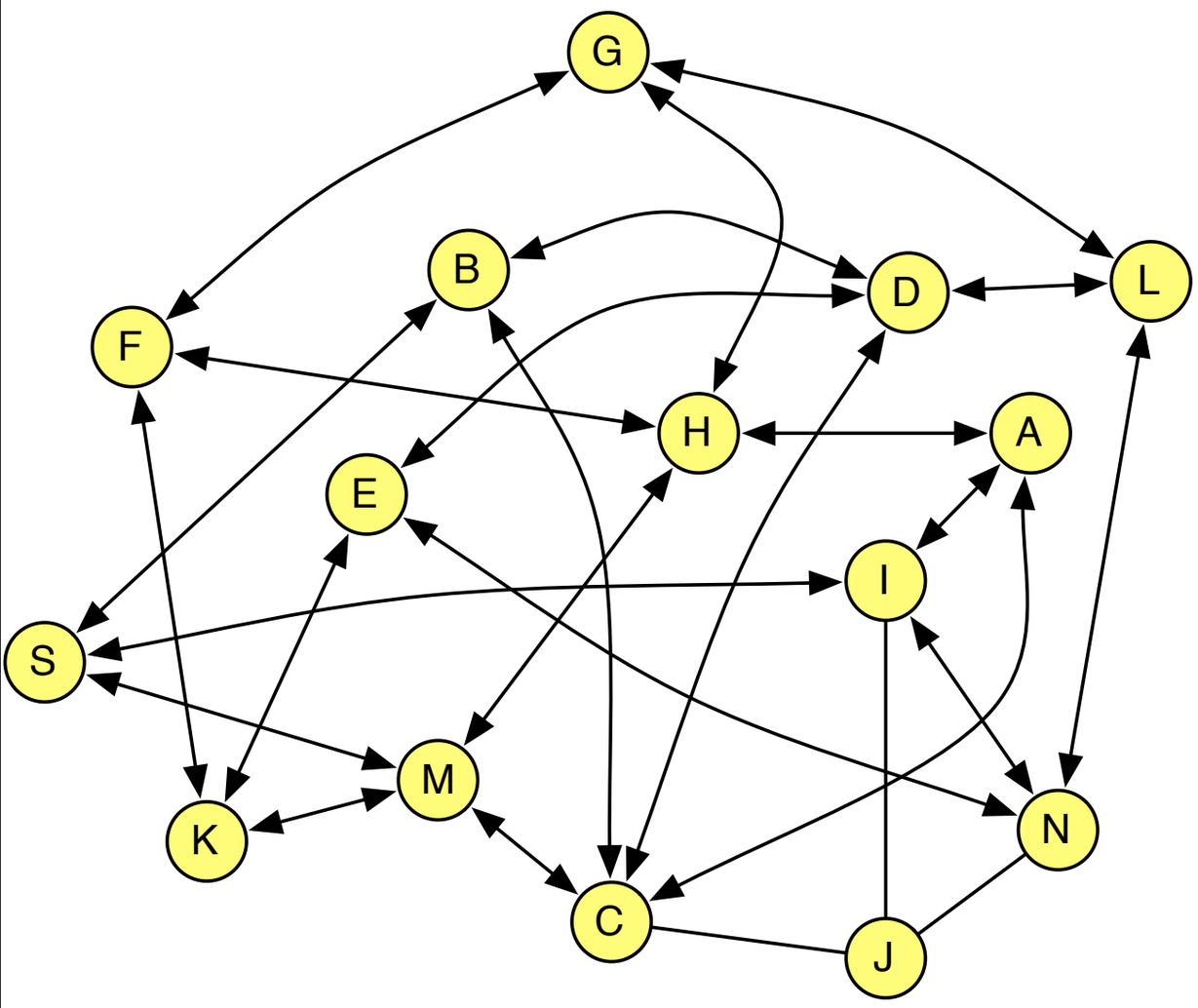
Reactive view management

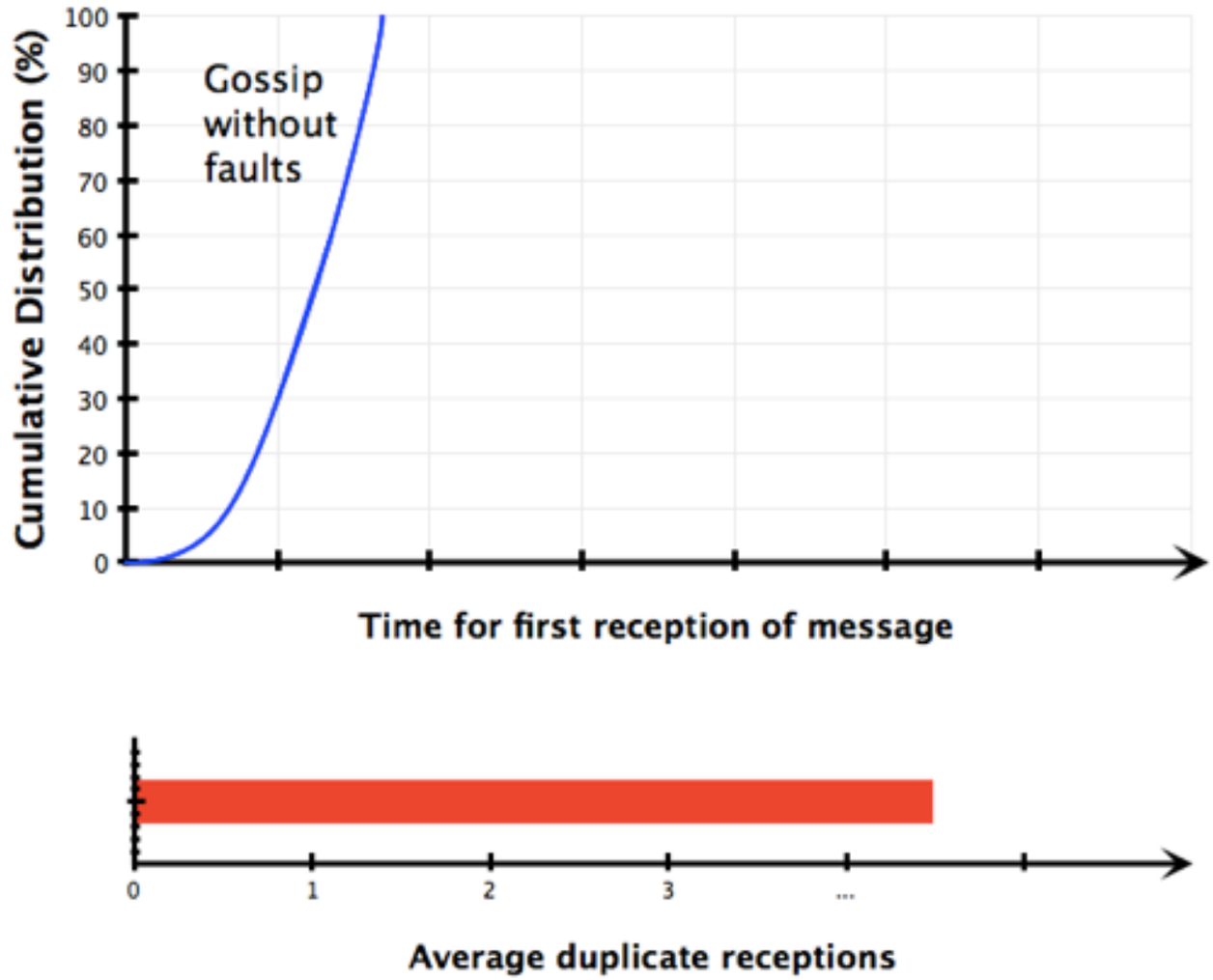
Maintains **persistent** connections to peers
Bi-directional links
Fault Detection
Replaces failed peers from PSS getPeer()

Provides constant *stream* of random peers

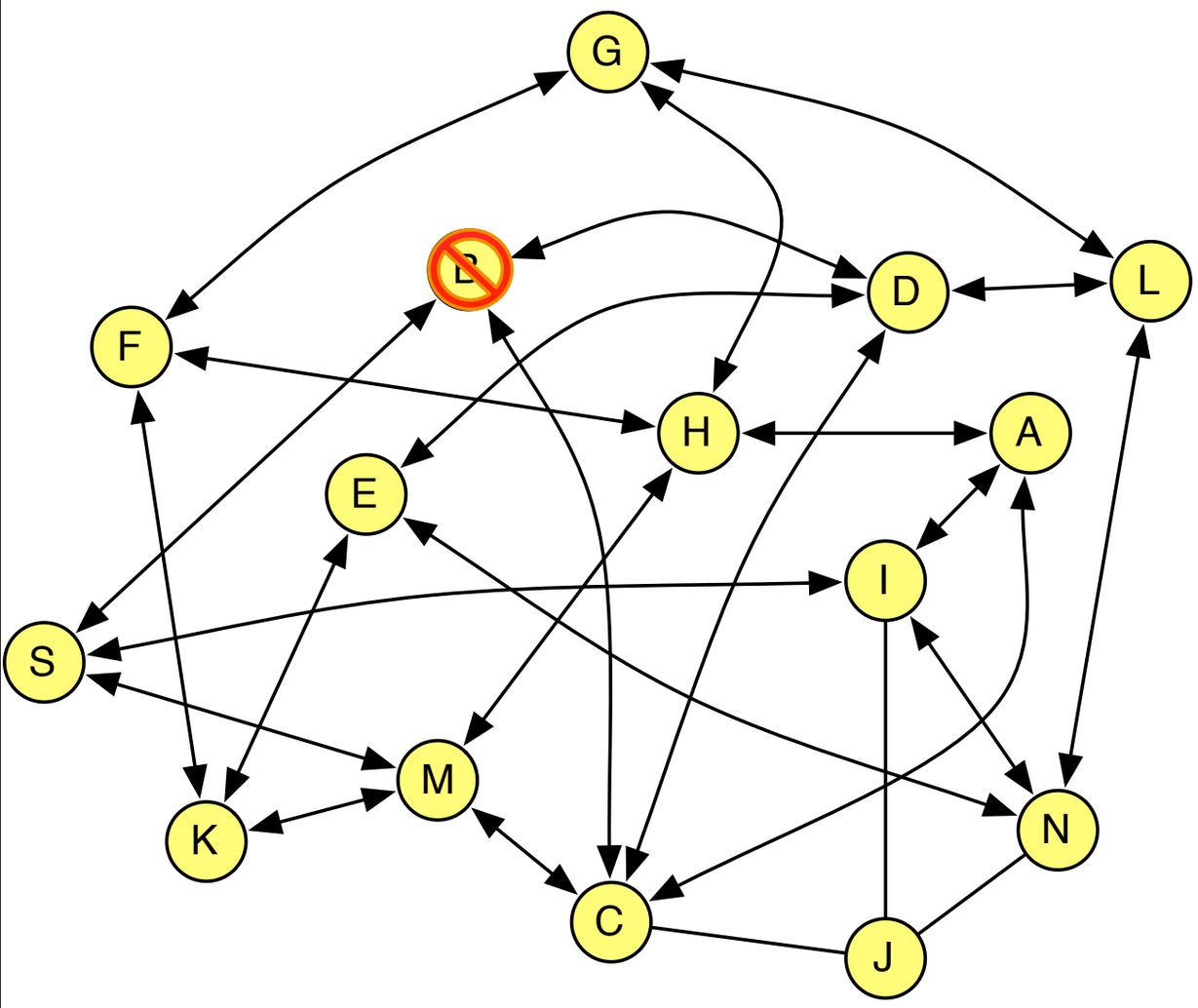
=> Membership Management
=> Avoids Partition
=> Insert new peers, forget old ones

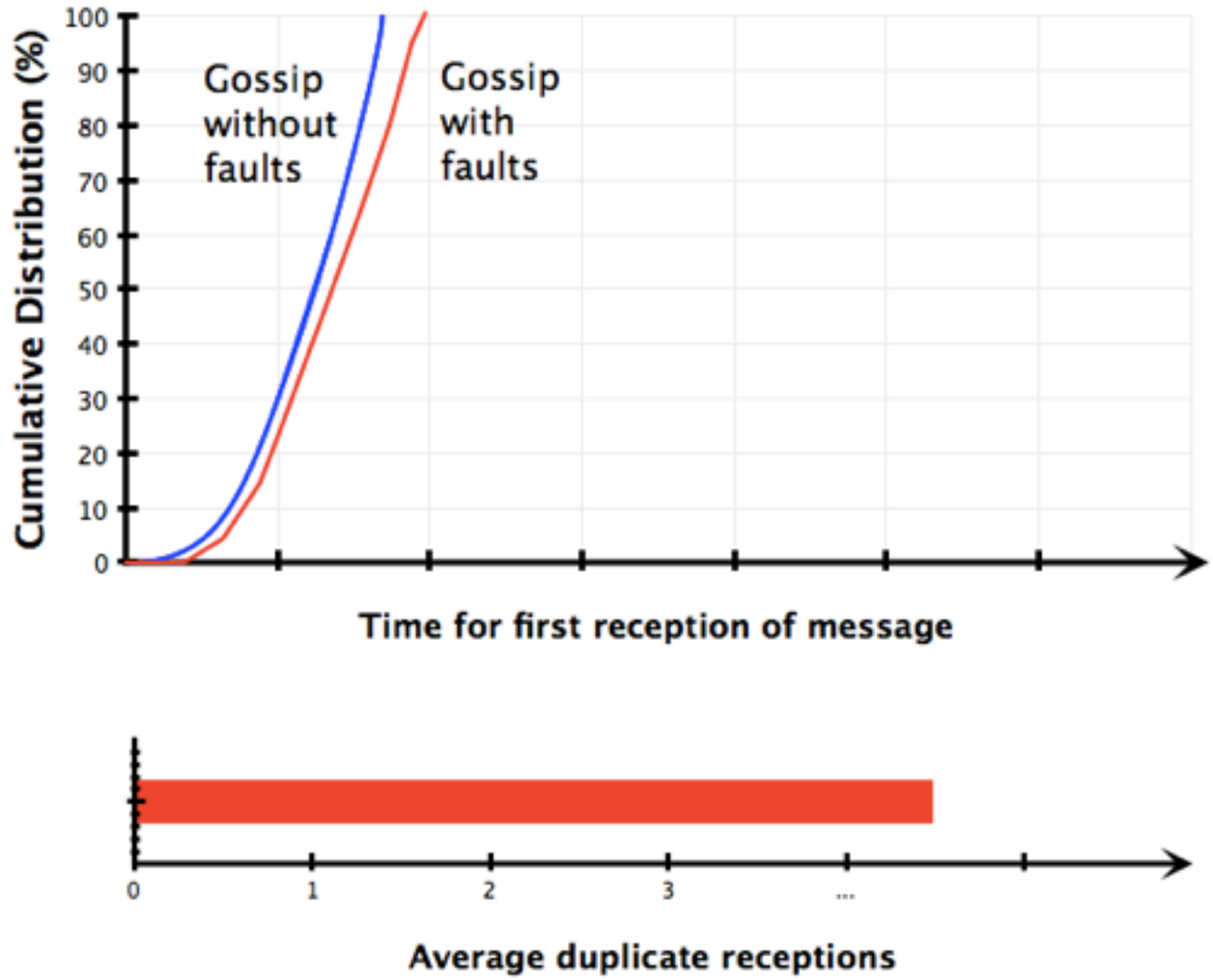
Epidemic-based dissemination



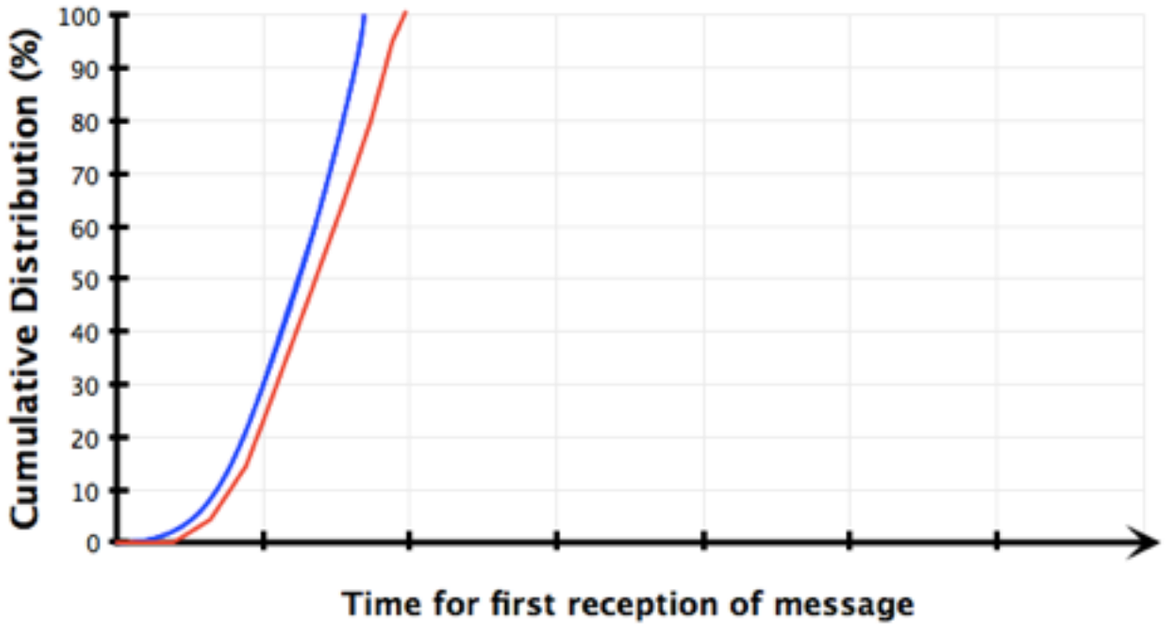


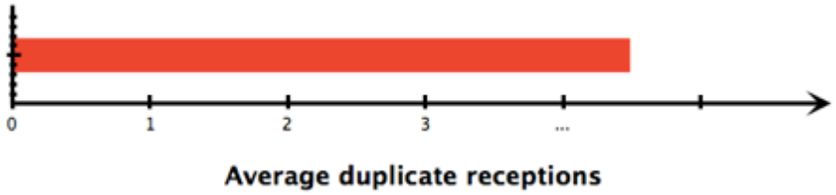
Epidemic-based dissemination



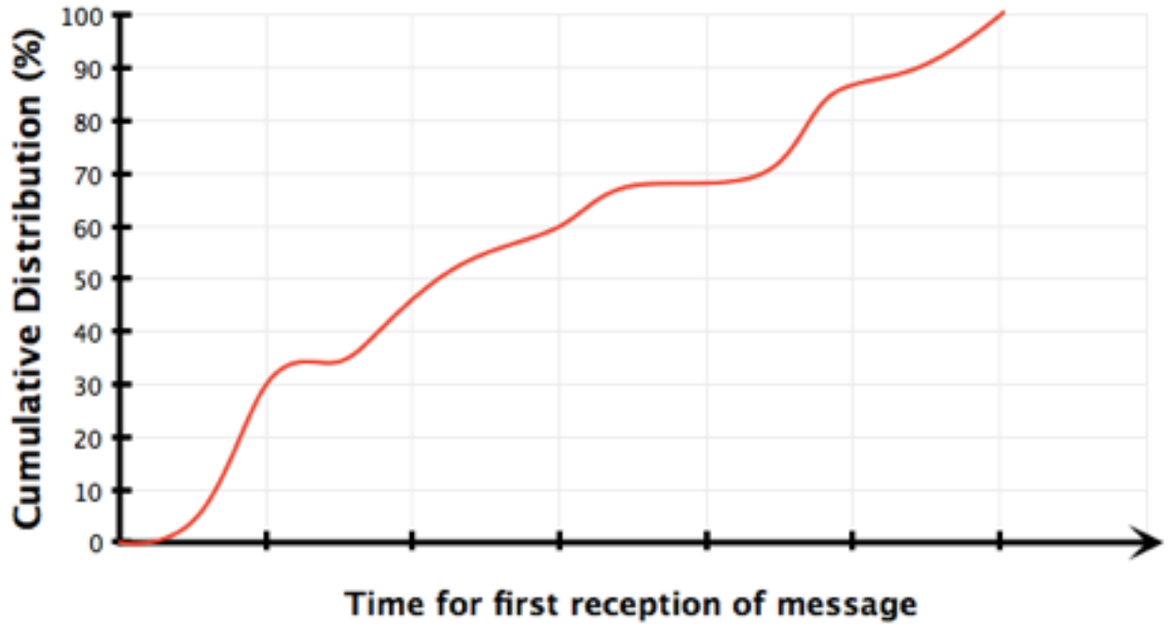


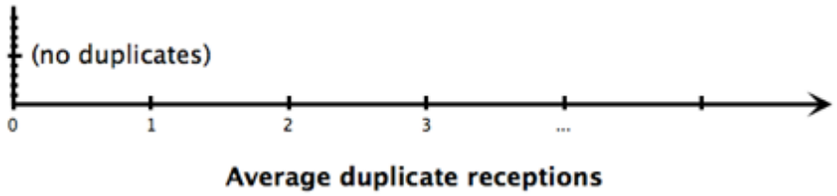
Gossip



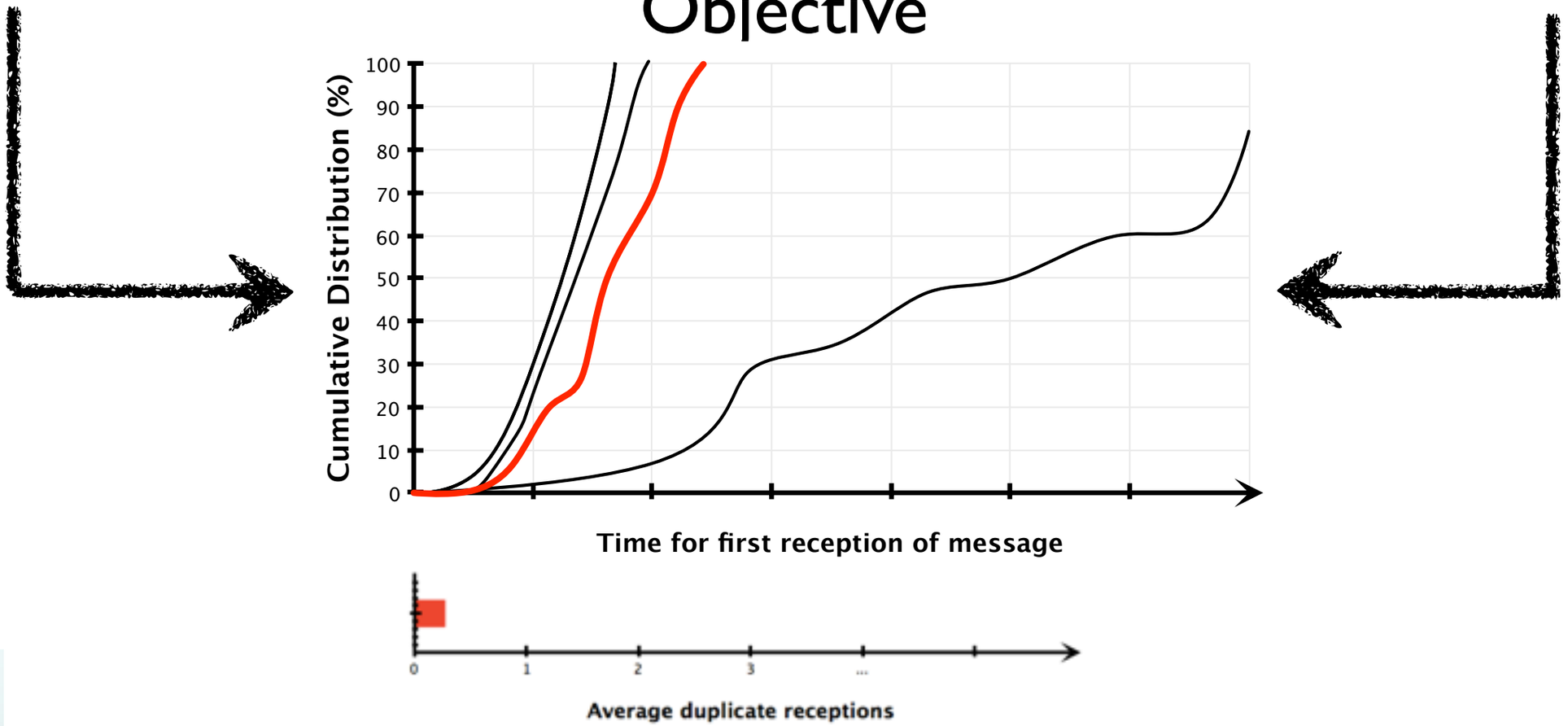


Tree





Objective



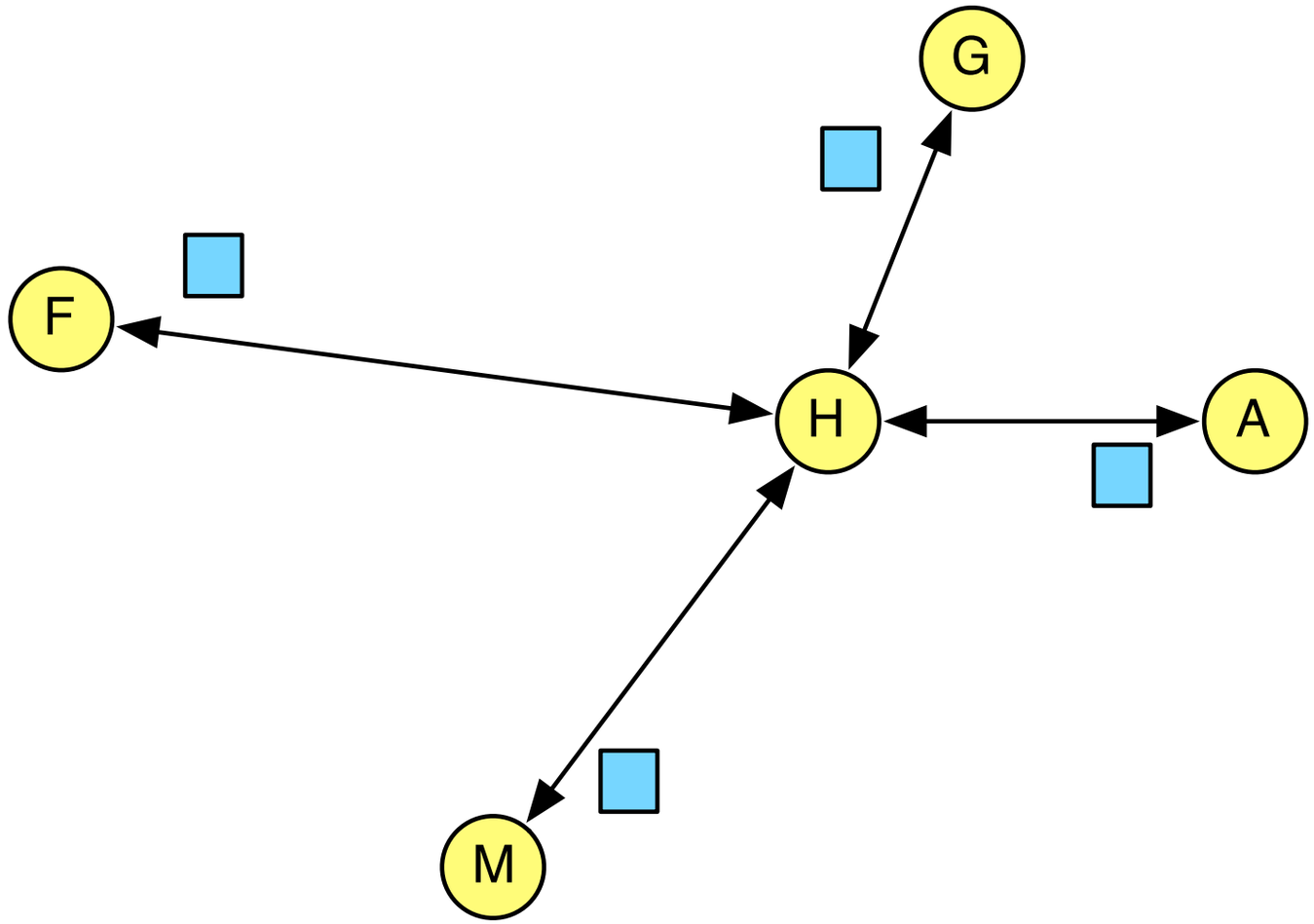
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

UNIVERSITÉ DE NEUCHÂTEL

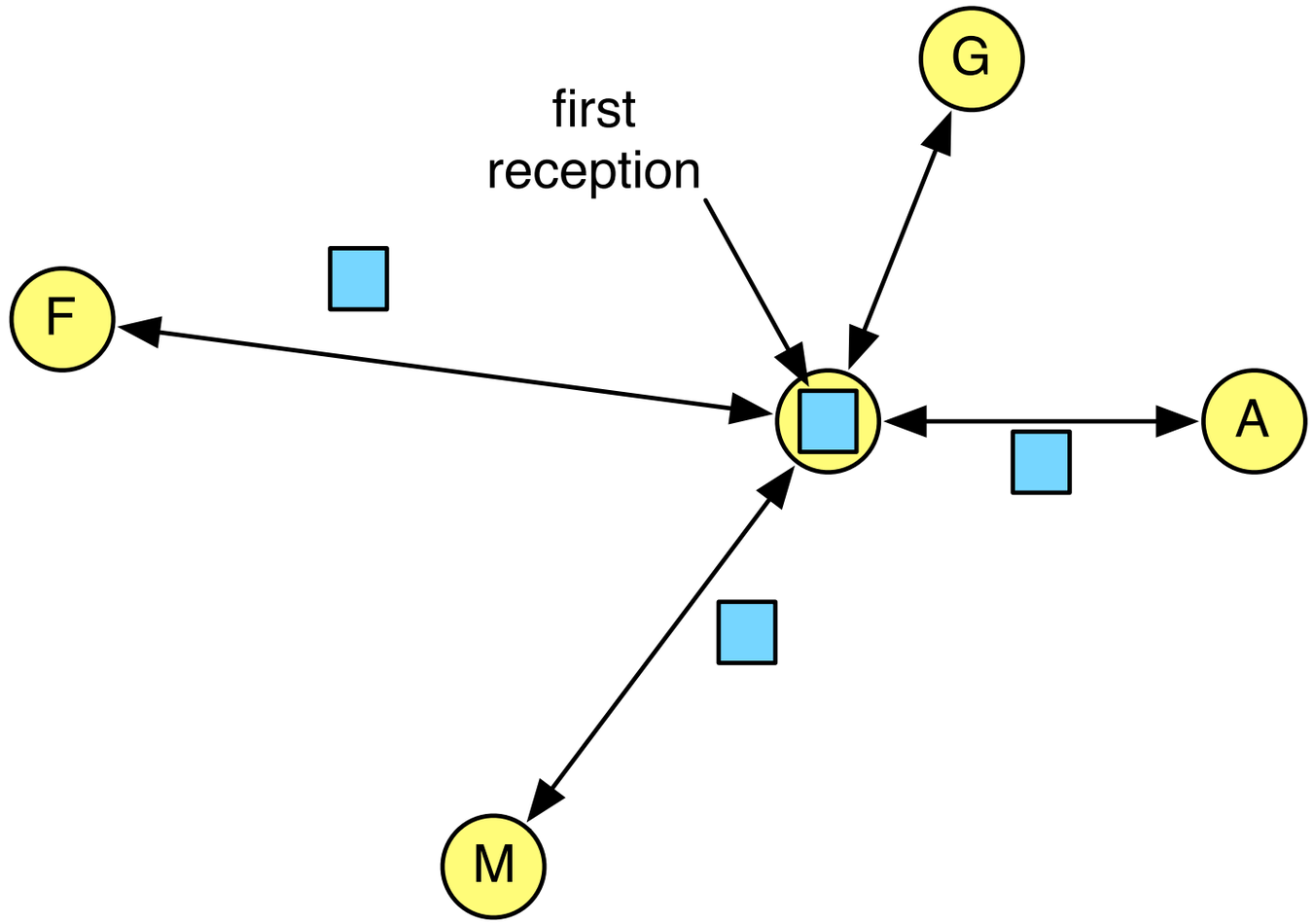
BRISA *in a nutshell*

- Start from existing *Reactive Peer Sampling Service* (RPSS)
  - First dissemination = flooding over RPSS
- Emerge an embedded tree
  - Select between the links maintained by the RPSS
    - One of the receptions = parent link
    - Others deactivated
- Allow only a small number of duplicates
  - Upon join
  - Upon failure
- Quick recovery based on RPSS persistent connections and quick fault detection

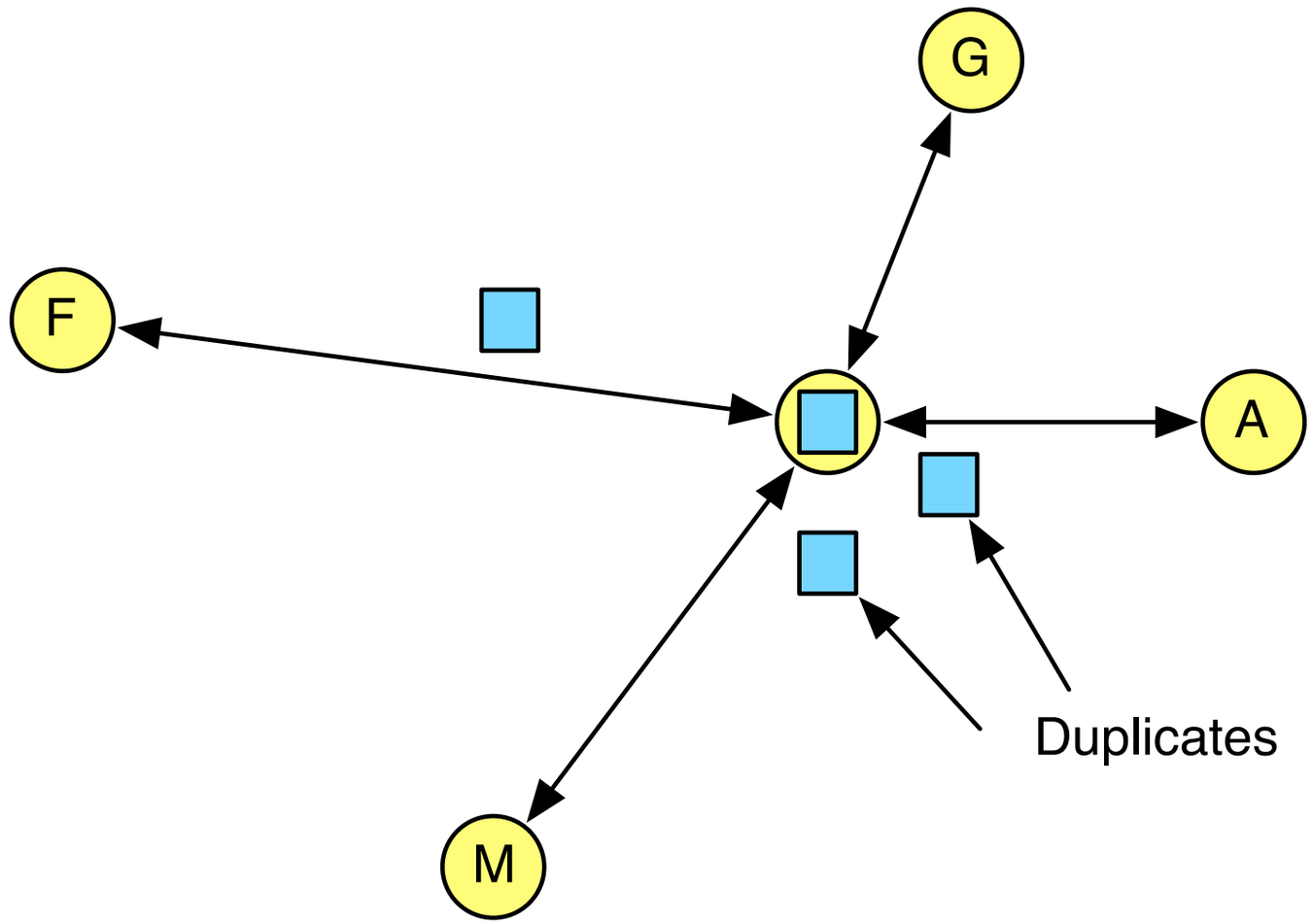
Emerging a tree through deactivations



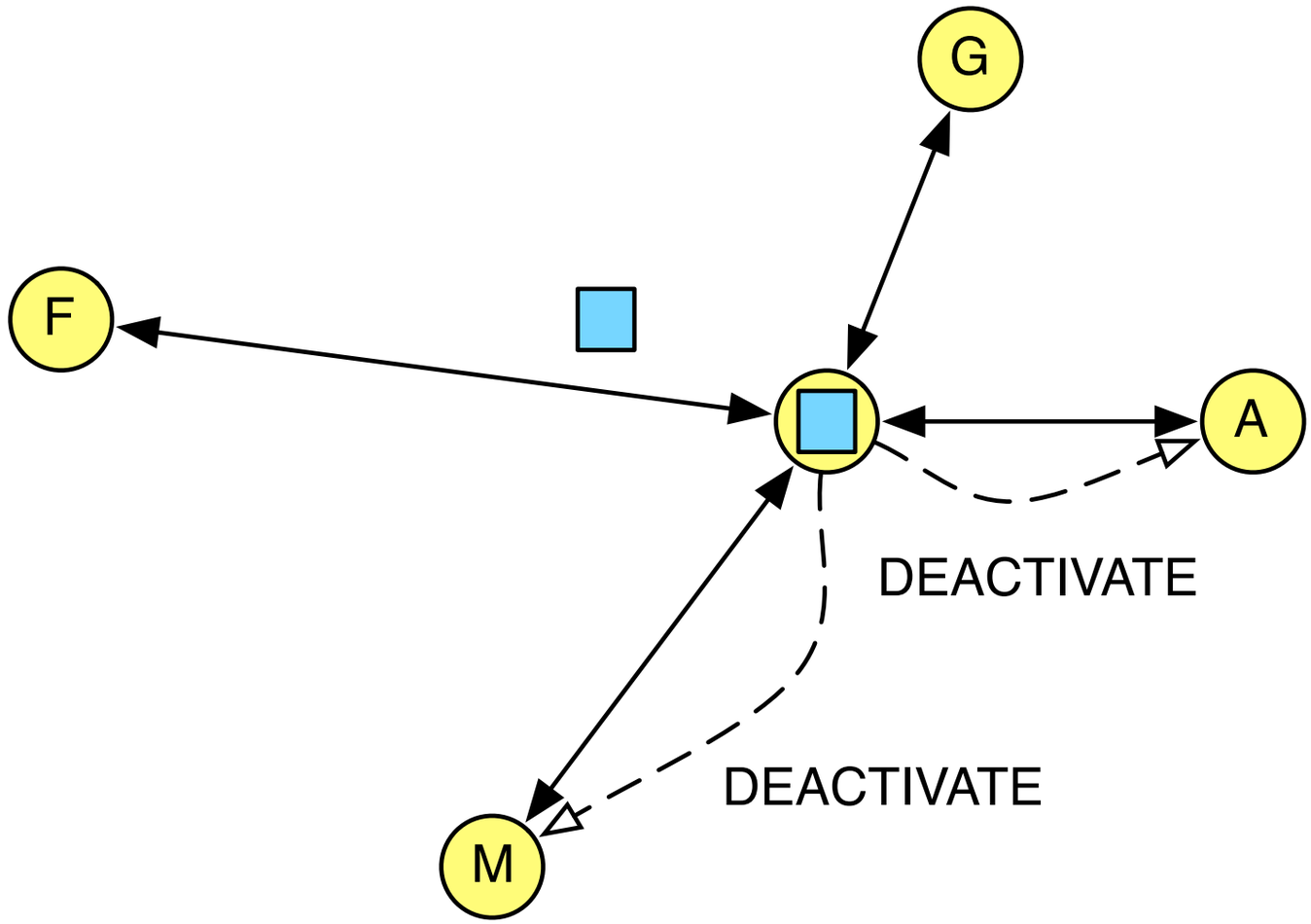
Emerging a tree through deactivations



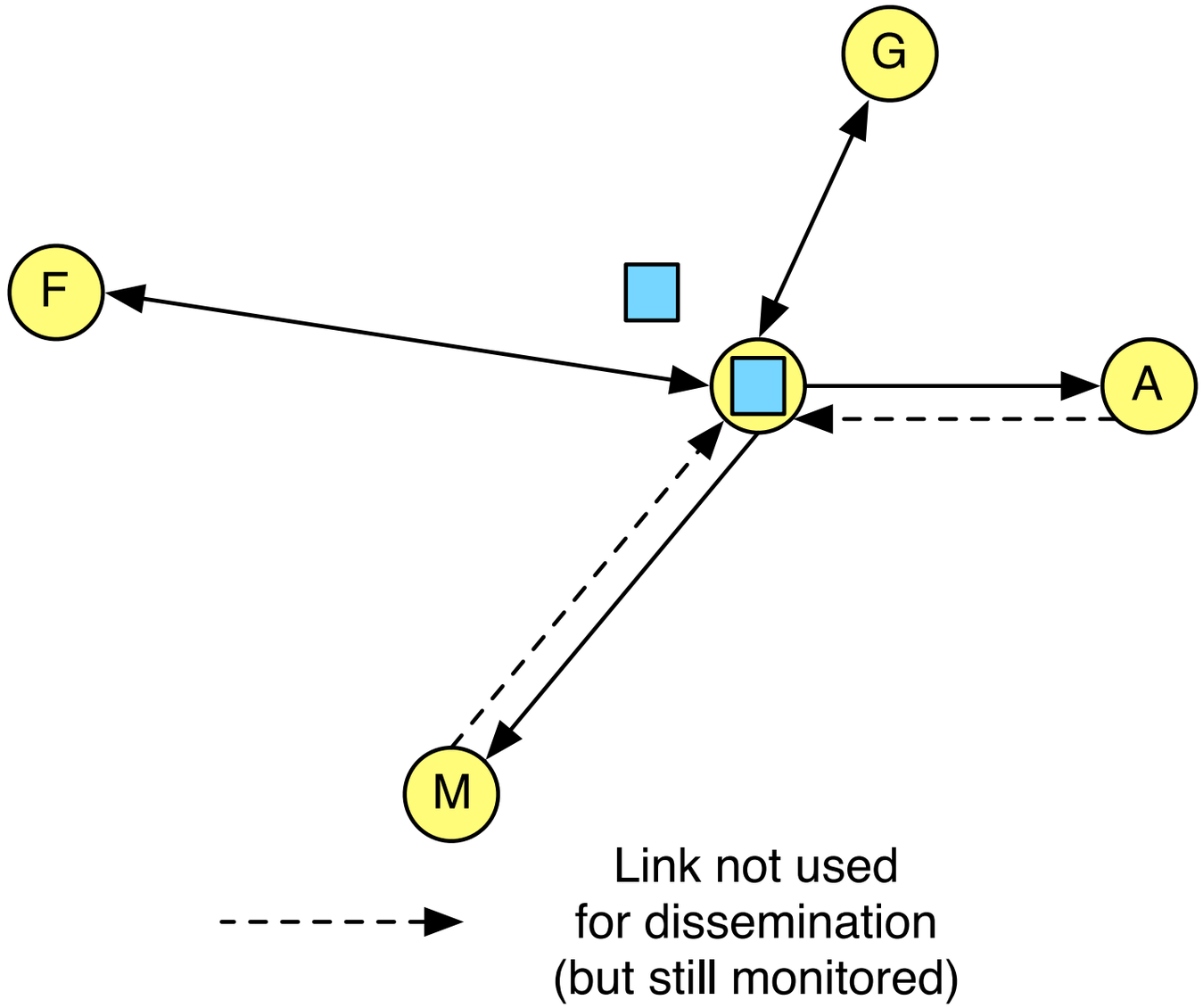
Emerging a tree through deactivations



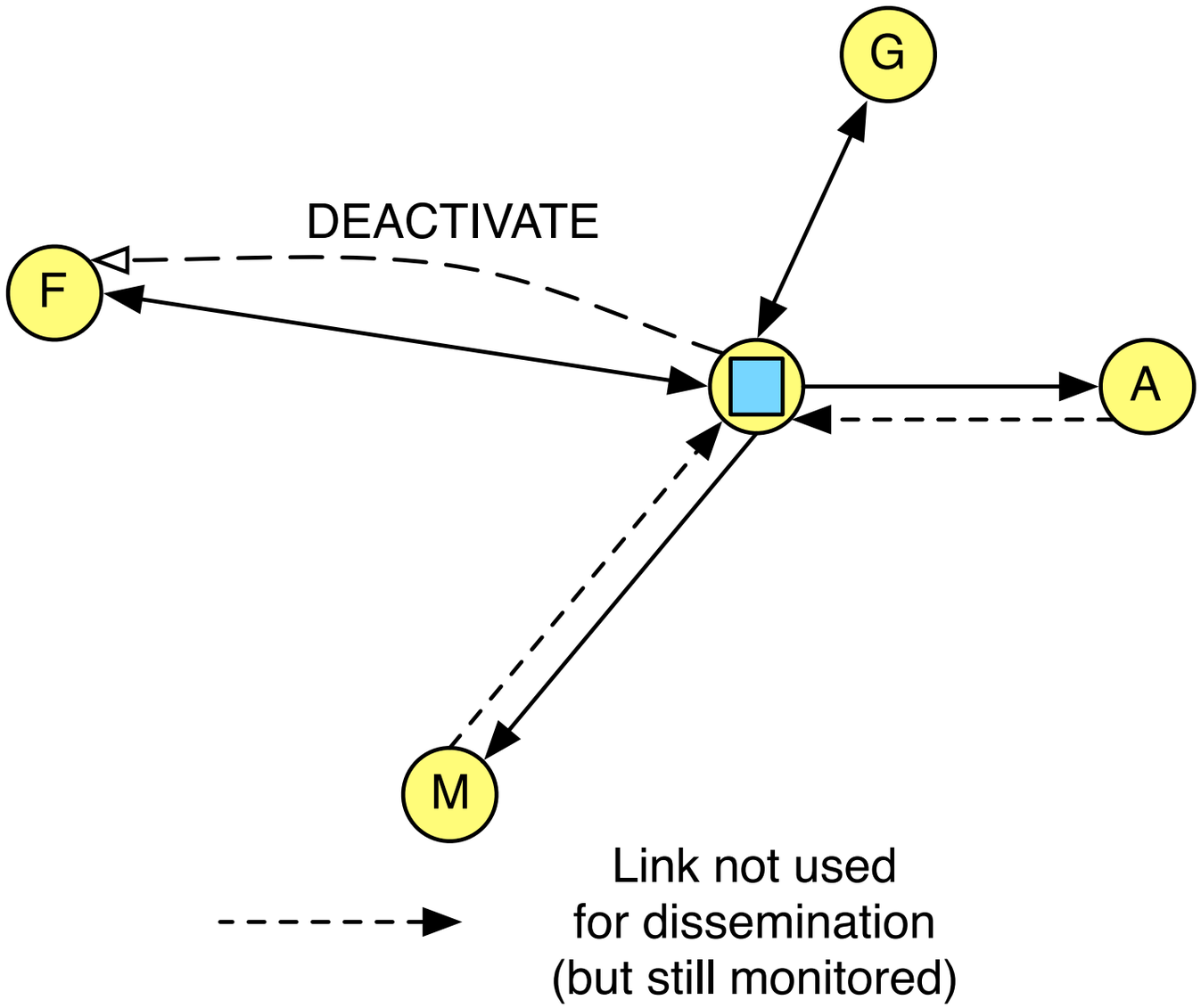
Emerging a tree through deactivations



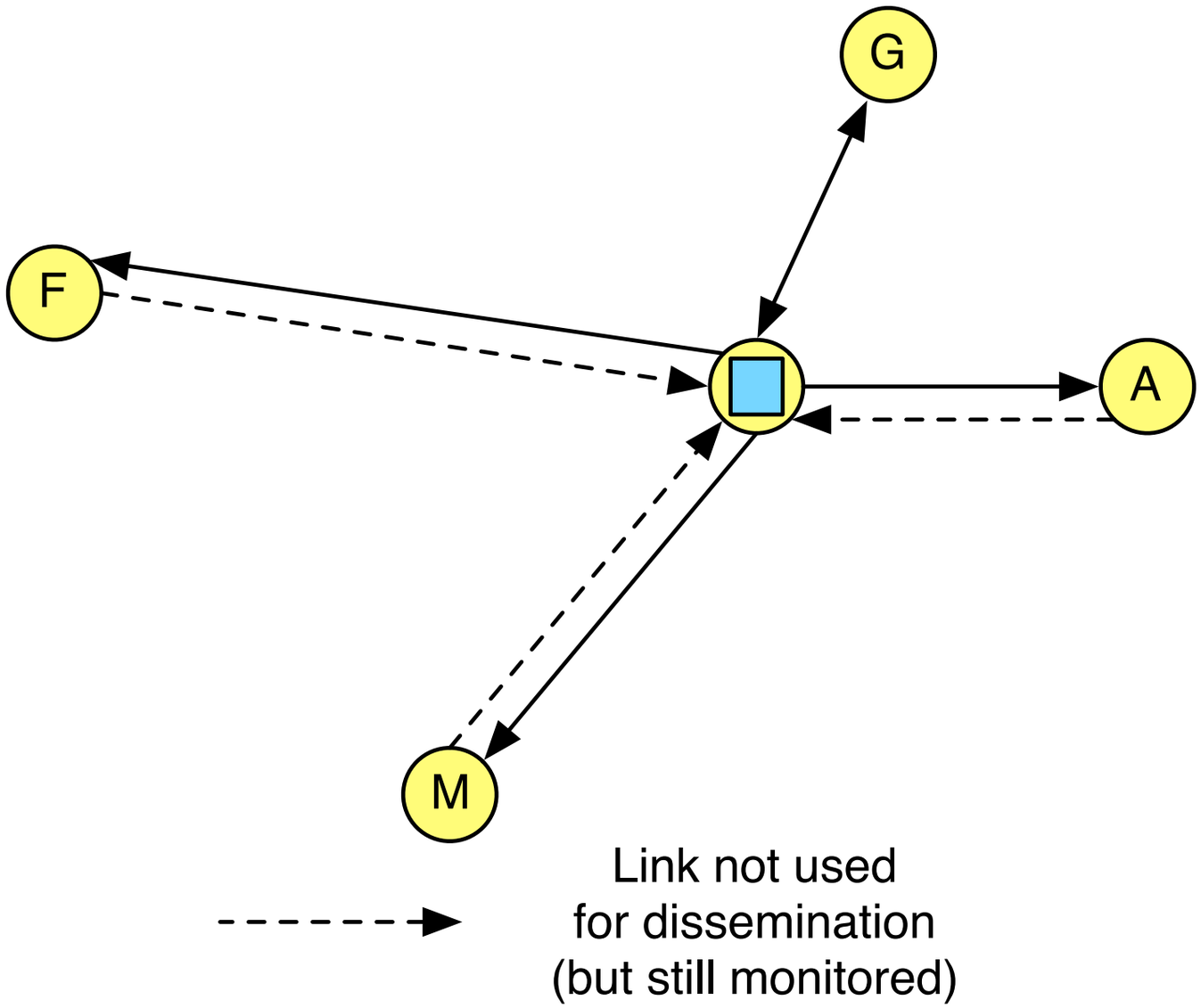
Emerging a tree through deactivations



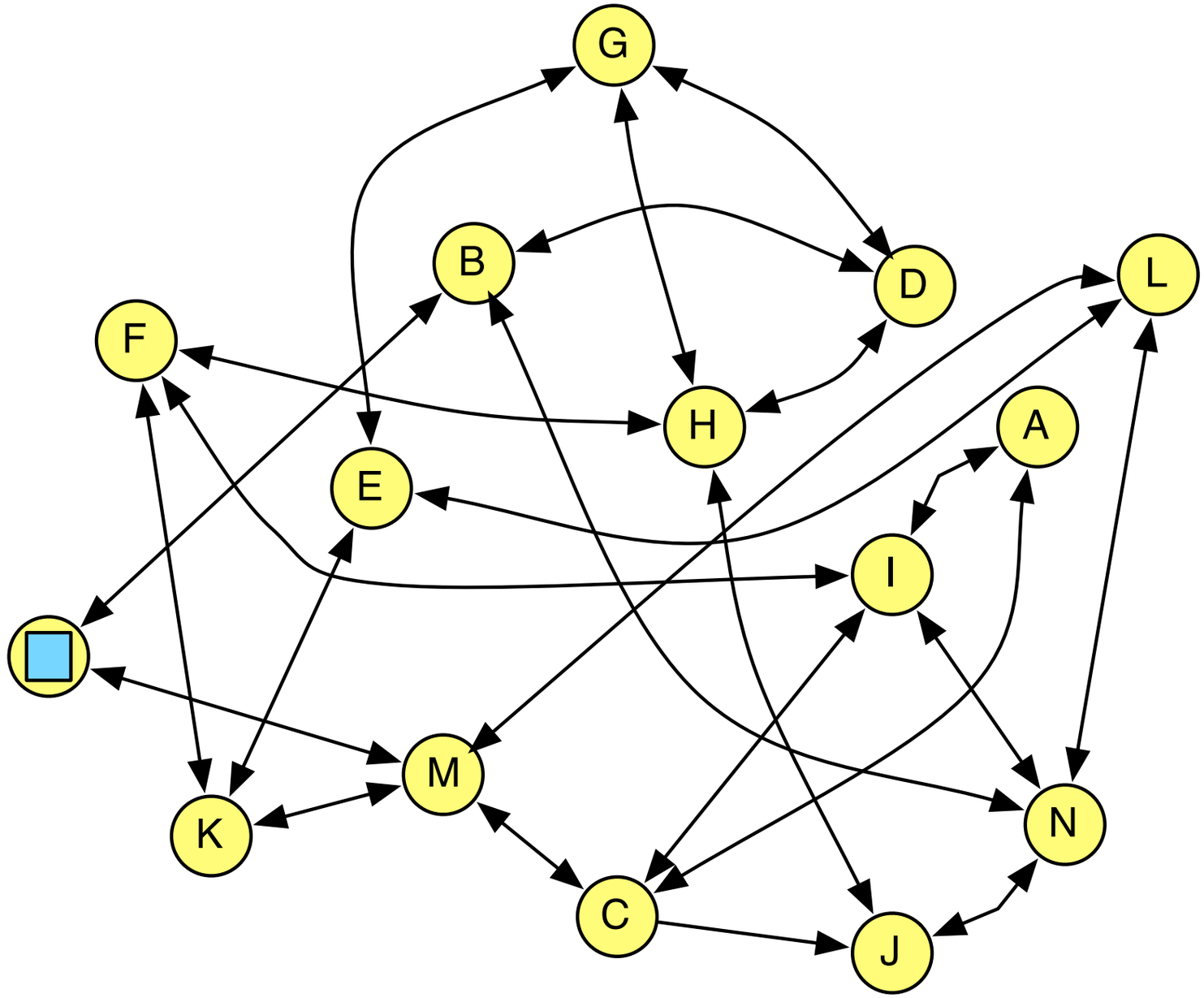
Emerging a tree through deactivations



Emerging a tree through deactivations

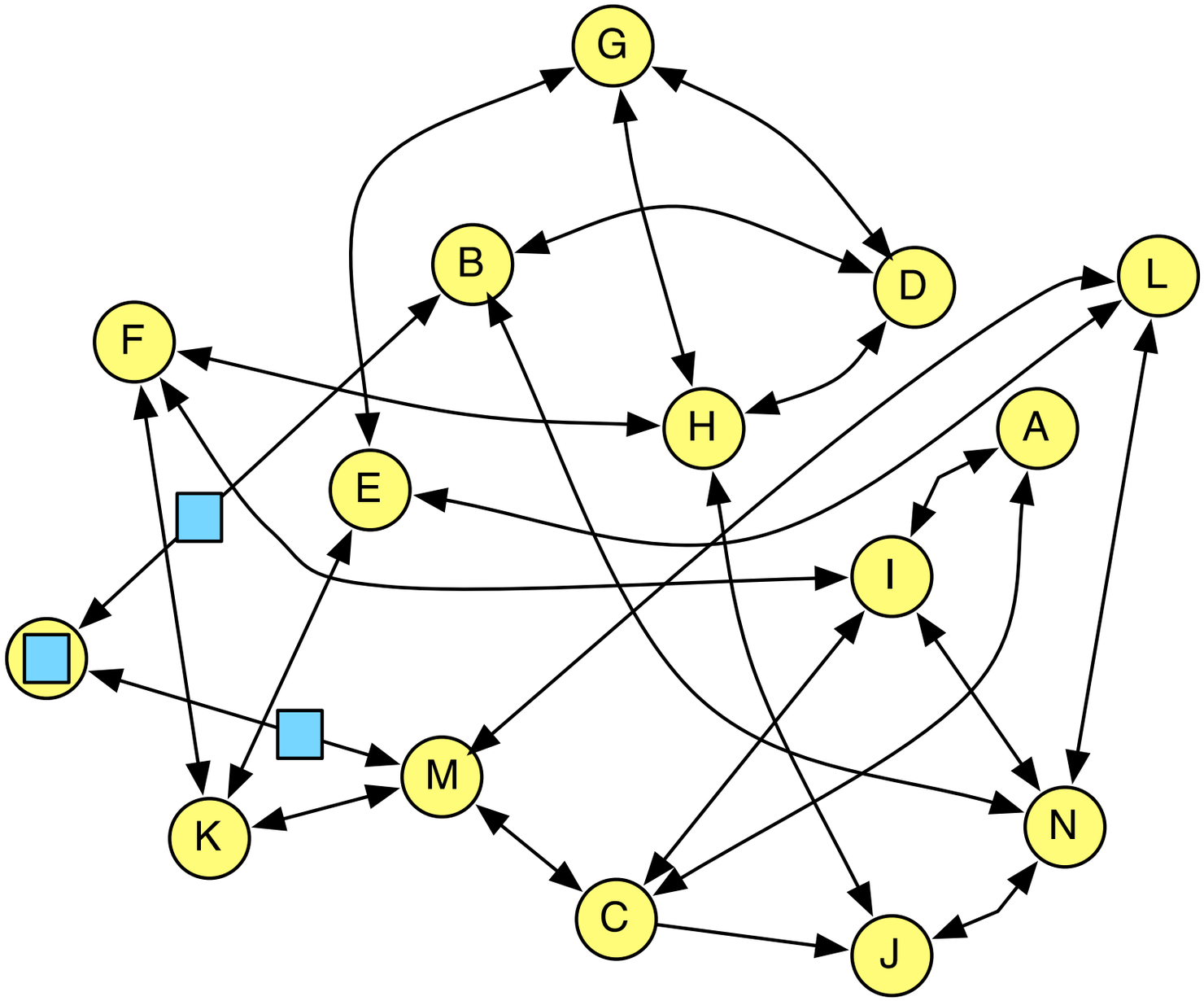


BRISA in action



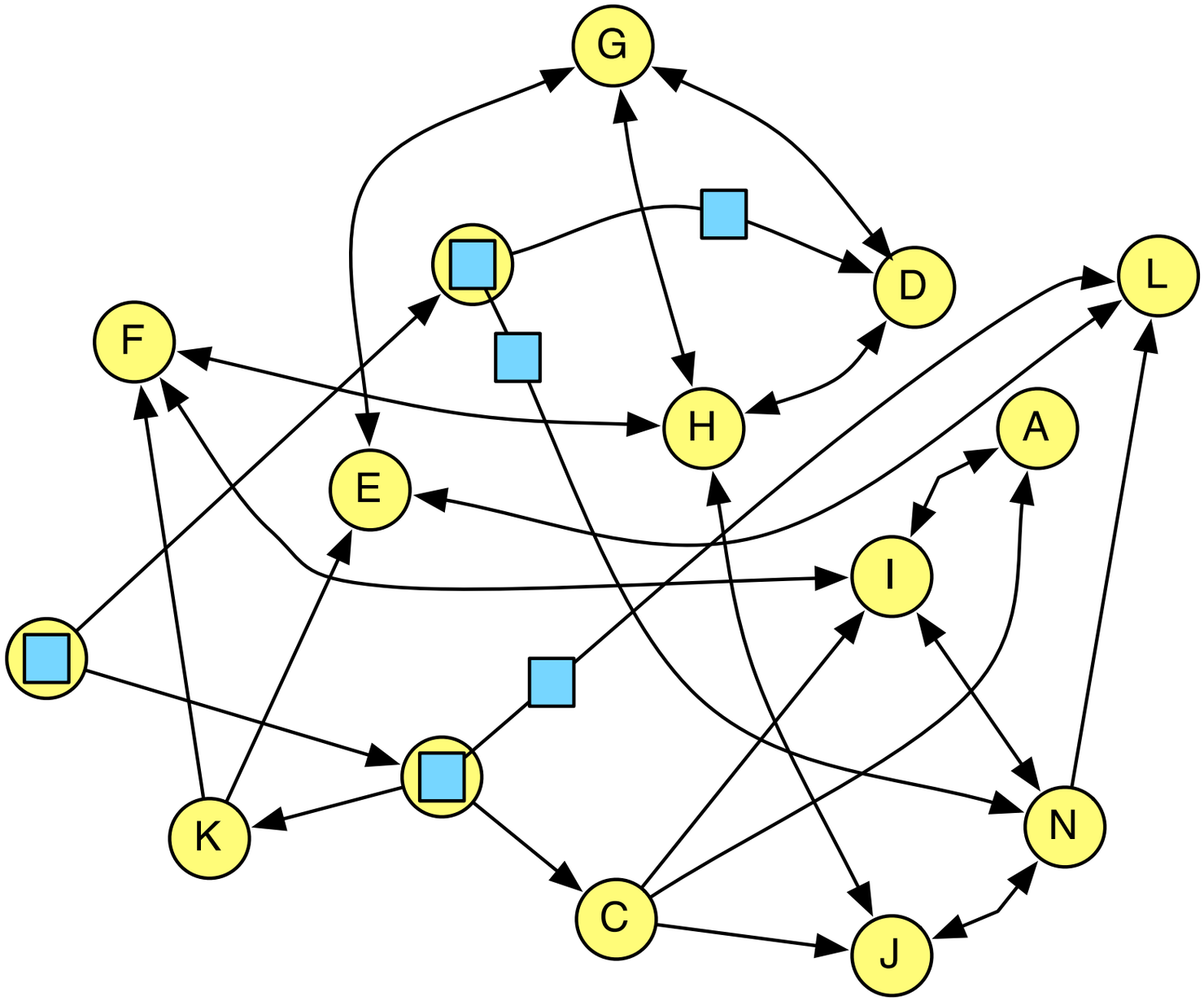
(Inactive Links not shown)

BRISA in action



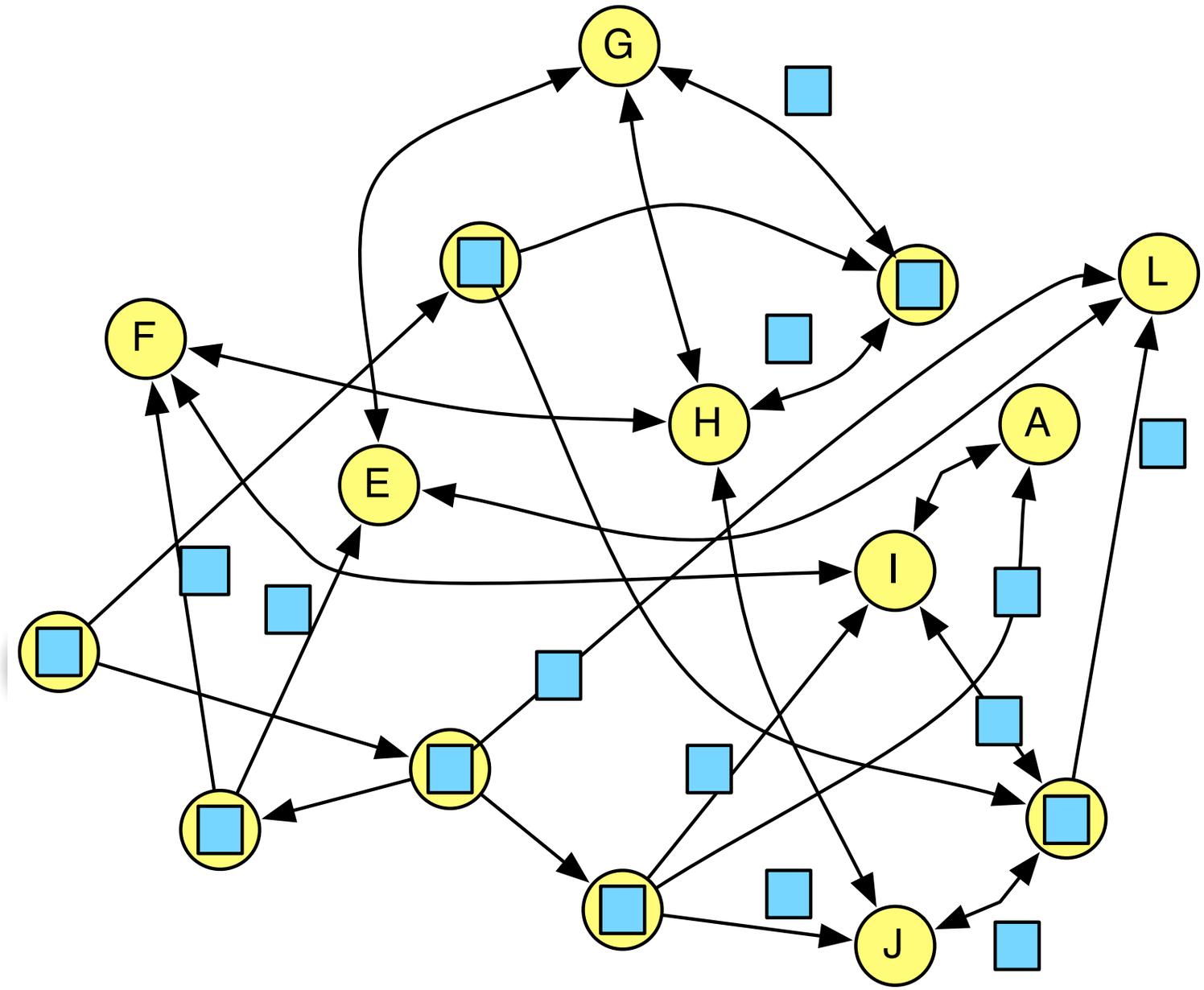
(Inactive Links not shown)

BRISA in action



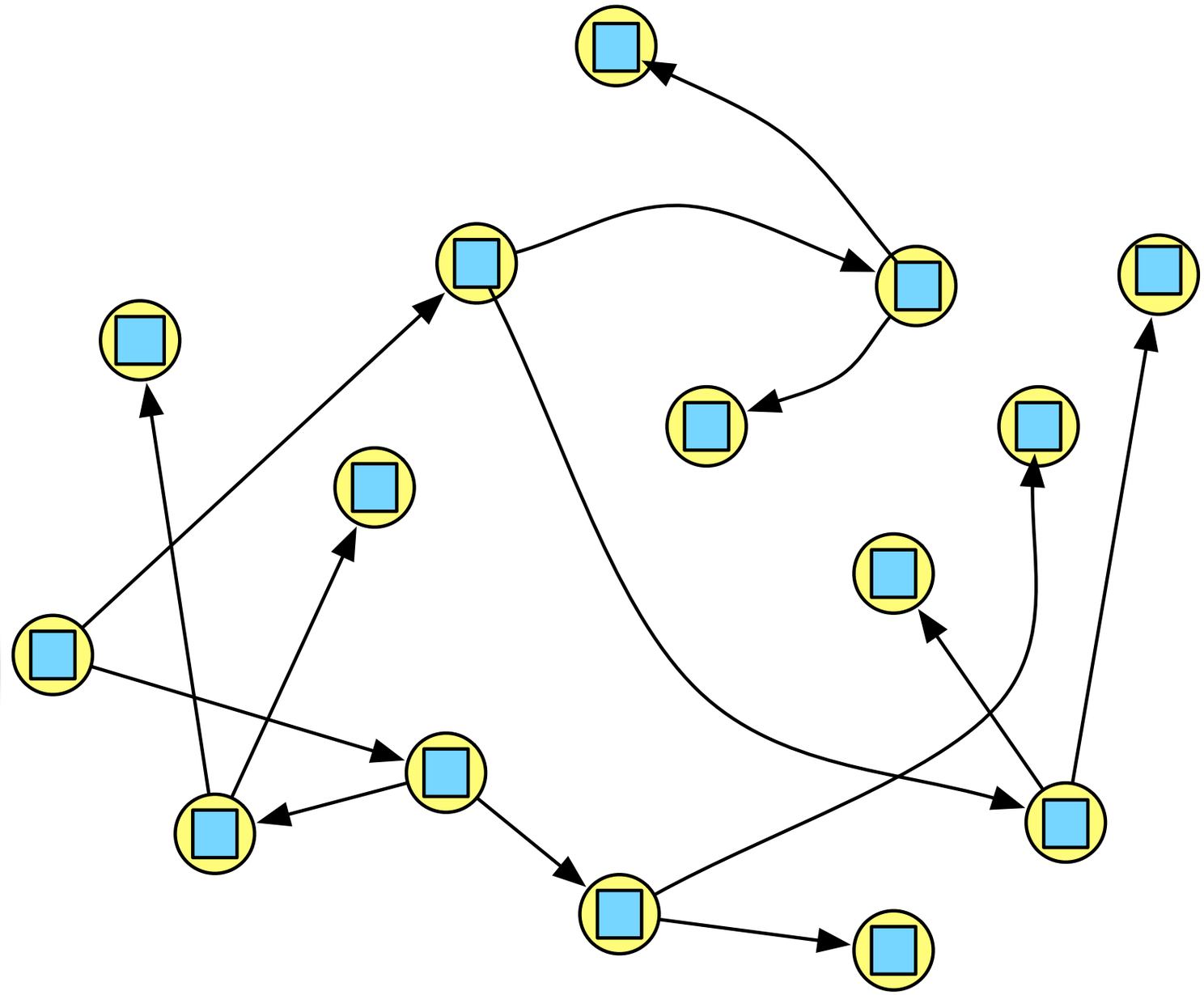
(Inactive Links not shown)

BRISA in action



(Inactive Links not shown)

BRISA in action

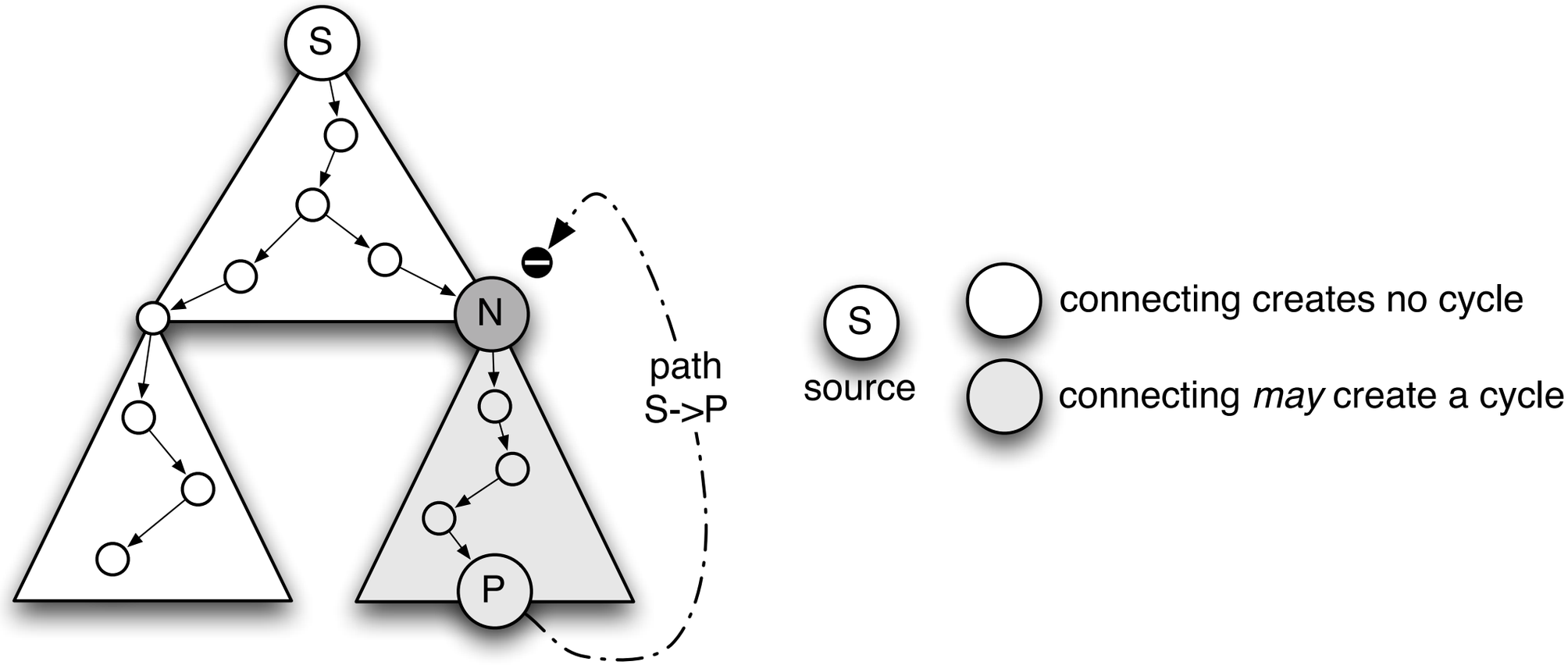


(Inactive Links not shown)

Failure Detection: RPSS level

- Parent failure implies reactivating a link
  - Send *activate* to one peer
  - Remember that connections are persistent
- Problem:
  - Which peer to reactivate ?
    - Avoid creating cycles
- Cycle detection mechanism
- Selection between eligible peers
  - Performance criteria
  - Criteria on tree structure

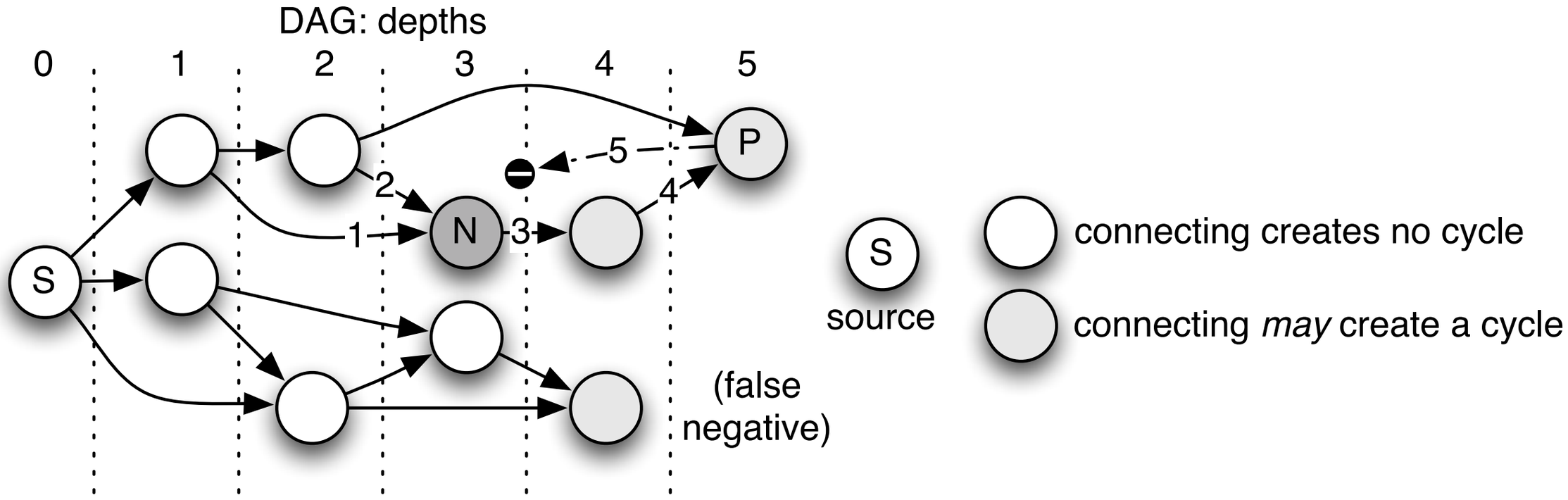
Avoiding cycles



Parent selection strategies

- Selection of peer to reactivate among eligible peers
- Simplest: first-come first picked
- Performance criteria
  - Delay-aware
- Criteria on tree structure
  - Heterogeneity-aware
    - Favors peer with most available bandwidth
- Load-balancing
  - Seeks to reduce out-degree variance
- Gerontocratic
  - Favors peers with highest uptime
  - Observations show that these peers are more stable

Extension to DAGs



- DAG = more than one parent
  - Control the number of received duplicates
  - Allows supporting higher churn levels
  - Lazy parent replacement
    - Service continuity if at least one parent remains
- Path-embedding is impossible for DAGs
  - Use node depth instead

Evaluation Setup

- Evaluation using a prototype
  - Supported by the Splay distributed system evaluation framework
- Two testbeds
  - 15 nodes cluster with Splay lightweight virtualization
    - 512 nodes
    - Uses Splay's churn replay mechanism
  - 128 nodes on PlanetLab

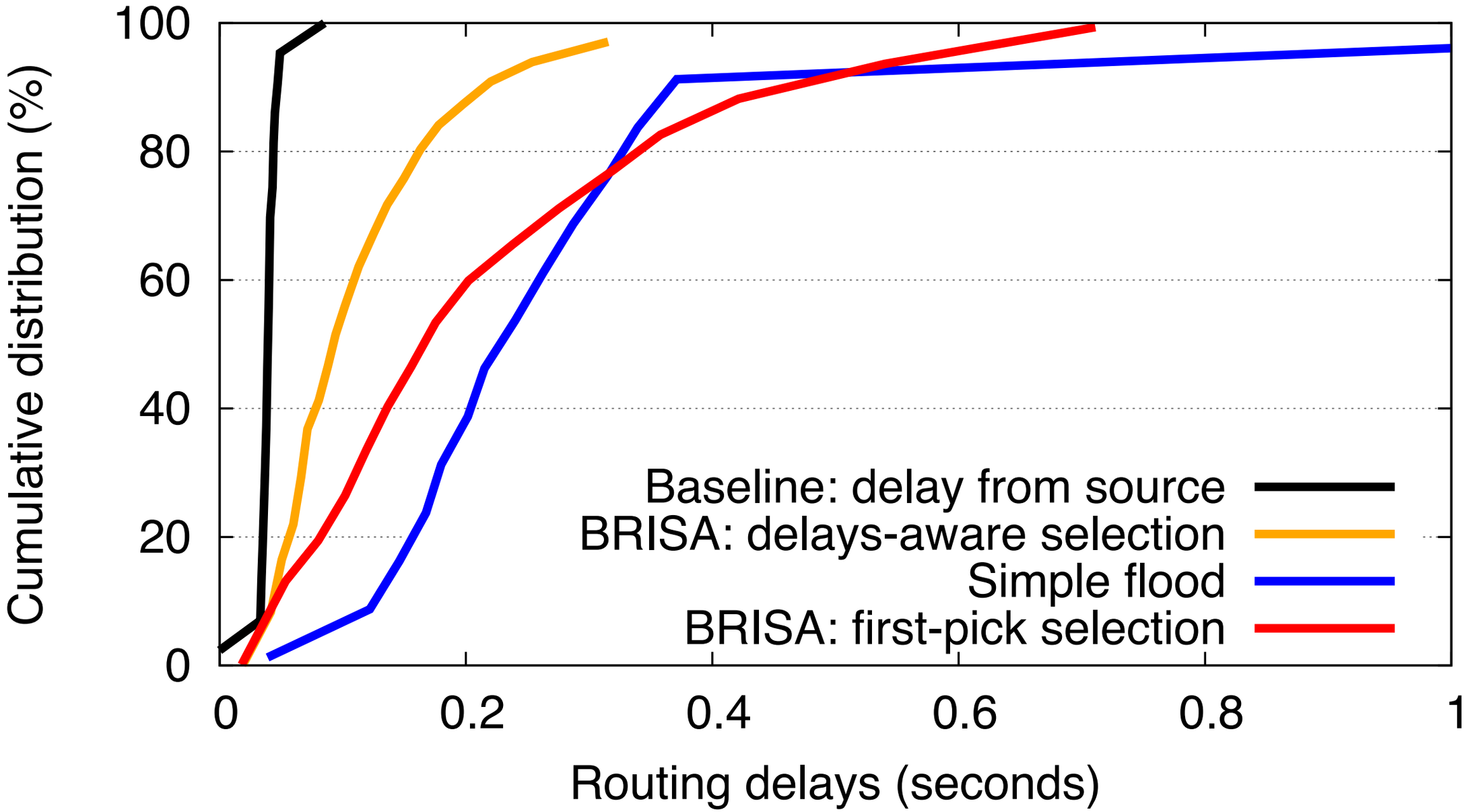
http://www.splay-project.org/

Evaluation Setup

- SimpleTree: statically constructed tree-based protocol
- SimpleGossip: traditional gossip-based protocol
- TAG [Liu & Zhou]
  - Closest related work
  - Another protocol that combines trees and epidemic overlays
    - Overlay used to build and repair the tree
    - No cycle detection: data is pulled from the overlay and from the tree
    - Tree repairing done by traversing the overlay

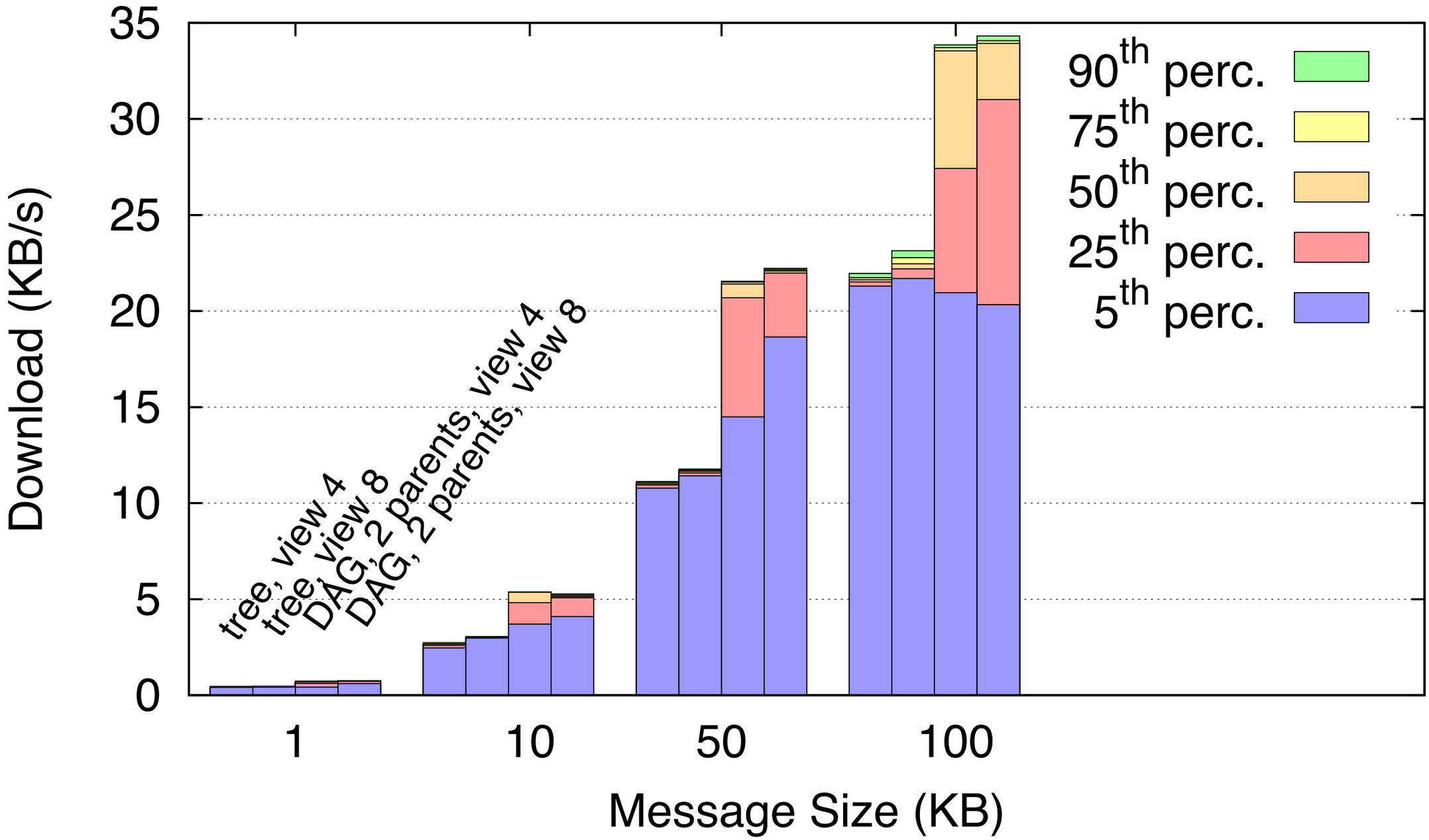
BRISA Dissemination Delay

150 PlanetLab nodes, 1KB



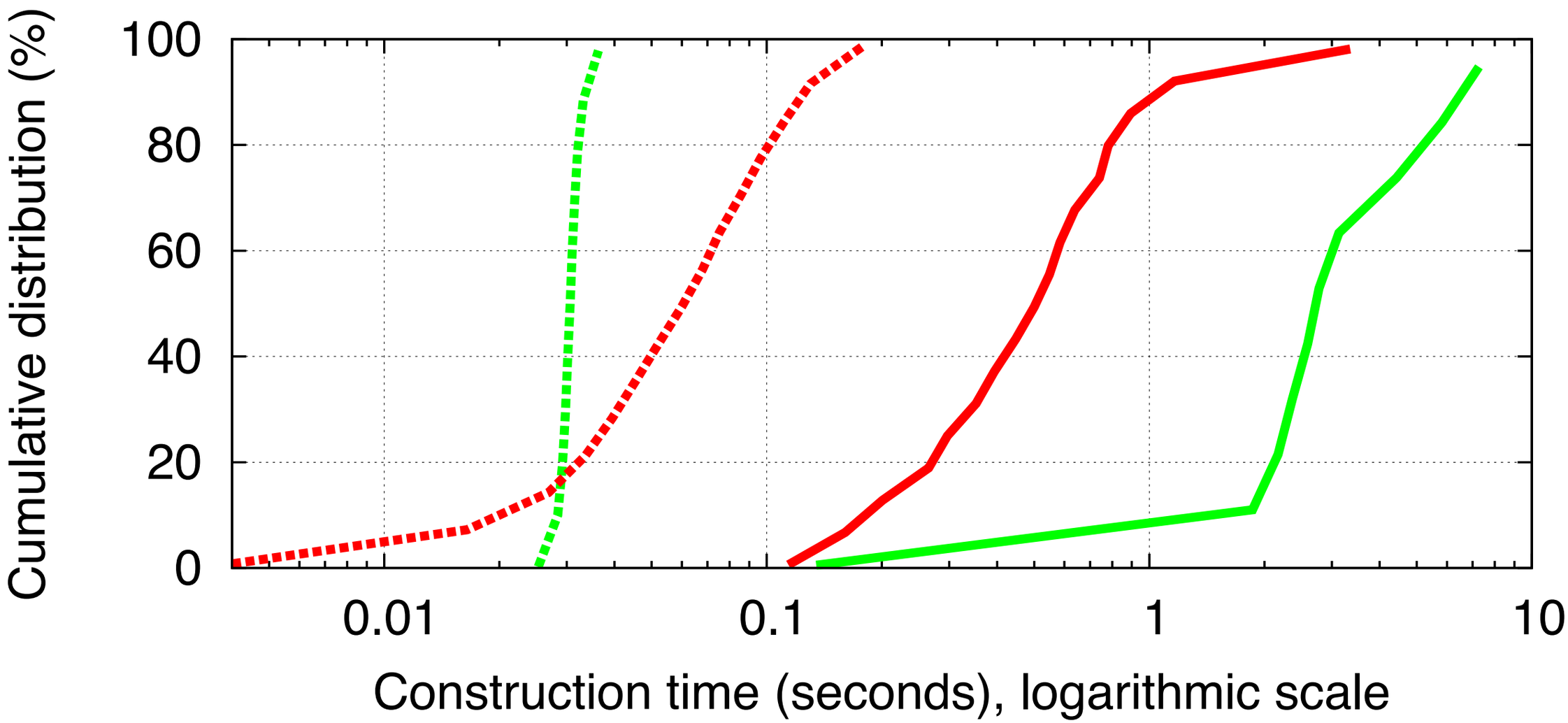
BRISA Bandwidth usage: download

512 cluster nodes



Structure construction time

512 nodes



Dissemination Latency

512 cluster nodes, 500 x 1KB message, 5 msg / sec.

Protocol	*Latency (seconds)*	*Overhead*
SimpleTree	**100.02**	**-**
BRISA	**106.59**	**+6%**
SimpleGossip	**128.23**	**+28%**
TAG	**200.48**	**+100%**

Churn

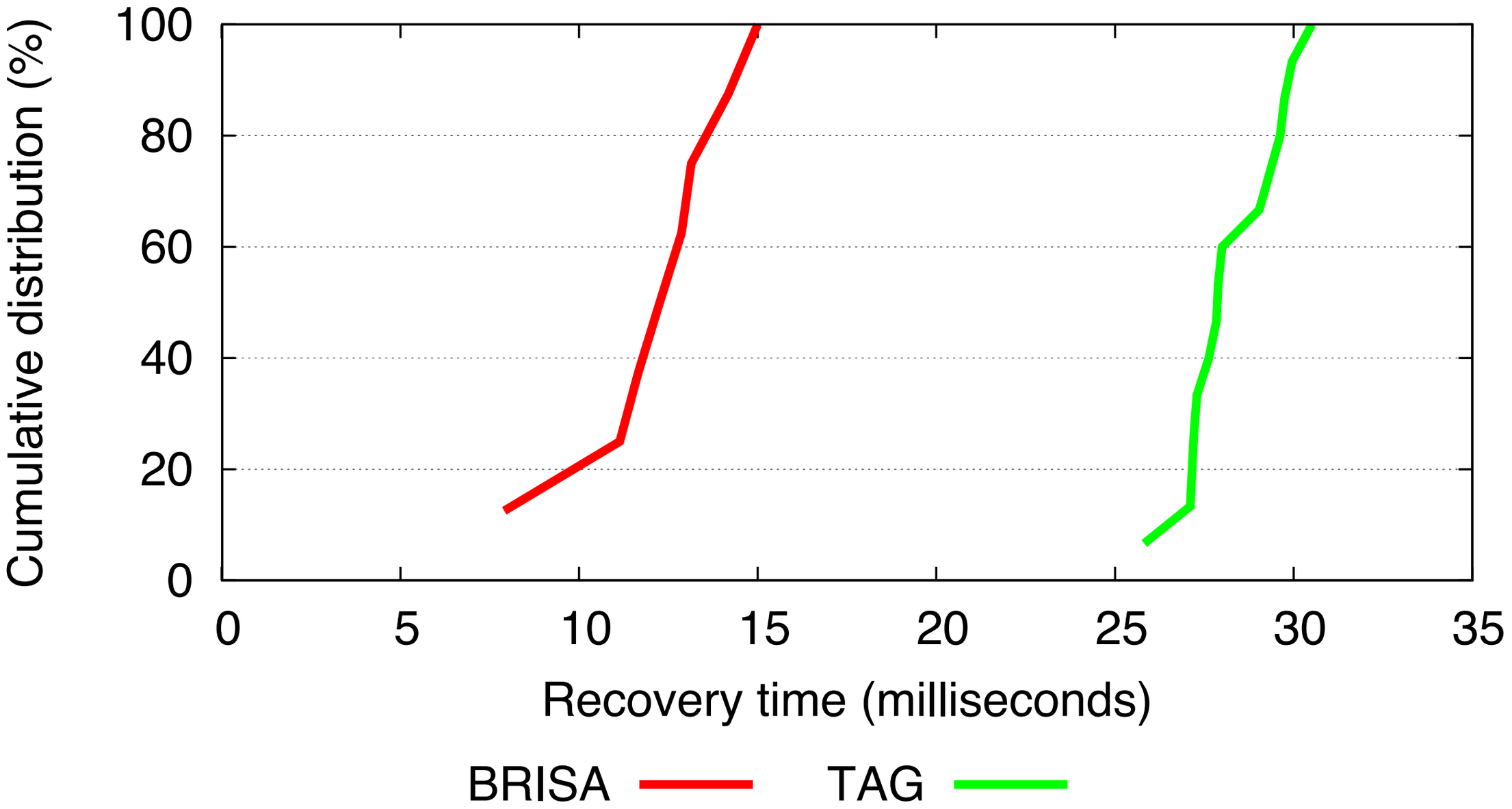
128 cluster nodes, 10 minutes

	Churn rate	*Parents lost/min*	*Orphans/ min*	*Direct fallback on RPSS link*	*Tree rejoin*
Tree	***3% replacement per minute***	**2.3**	**2.3**	**87%**	**13%**
DAG, 2 parents		**4**	**0.2**	**92.5%**	**7.5%**

(average per minute)

Parent recovery delay (tree rejoins)

128 cluster nodes; churn rate 3%



Conclusion & perspectives

- BRISA *combines* advantages of tree-based dissemination with advantages of epidemic dissemination
  - Robust to faults and churn
  - Quick failure recovery
  - Low overhead
  - Low latency
- Perspectives
  - Use DAGs as base for multiple trees
  - Low latency and overhead makes BRISA suitable for video streaming

BRISA:
Combining Efficiency and Reliability in Epidemic Data Dissemination

Miguel Matos[*], Valerio Schiavoni[+], Pascal Felber[+]
Rui Oliveira[*] and Etienne Rivière[+]

[*]INESC TEC & University of Minho, Portugal
[+]Université de Neuchâtel, Switzerland

miguelmatos@di.uminho.pt

International Parallel & Distributed Processing Symposium
24 May, 2012

HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

UNIVERSITÉ DE NEUCHÂTEL

Backup slides

Future Perspectives

- More elaborate parent selection strategies
- Use DAGs as base for multiple trees
- Low latency and overhead makes it suitable to video streaming

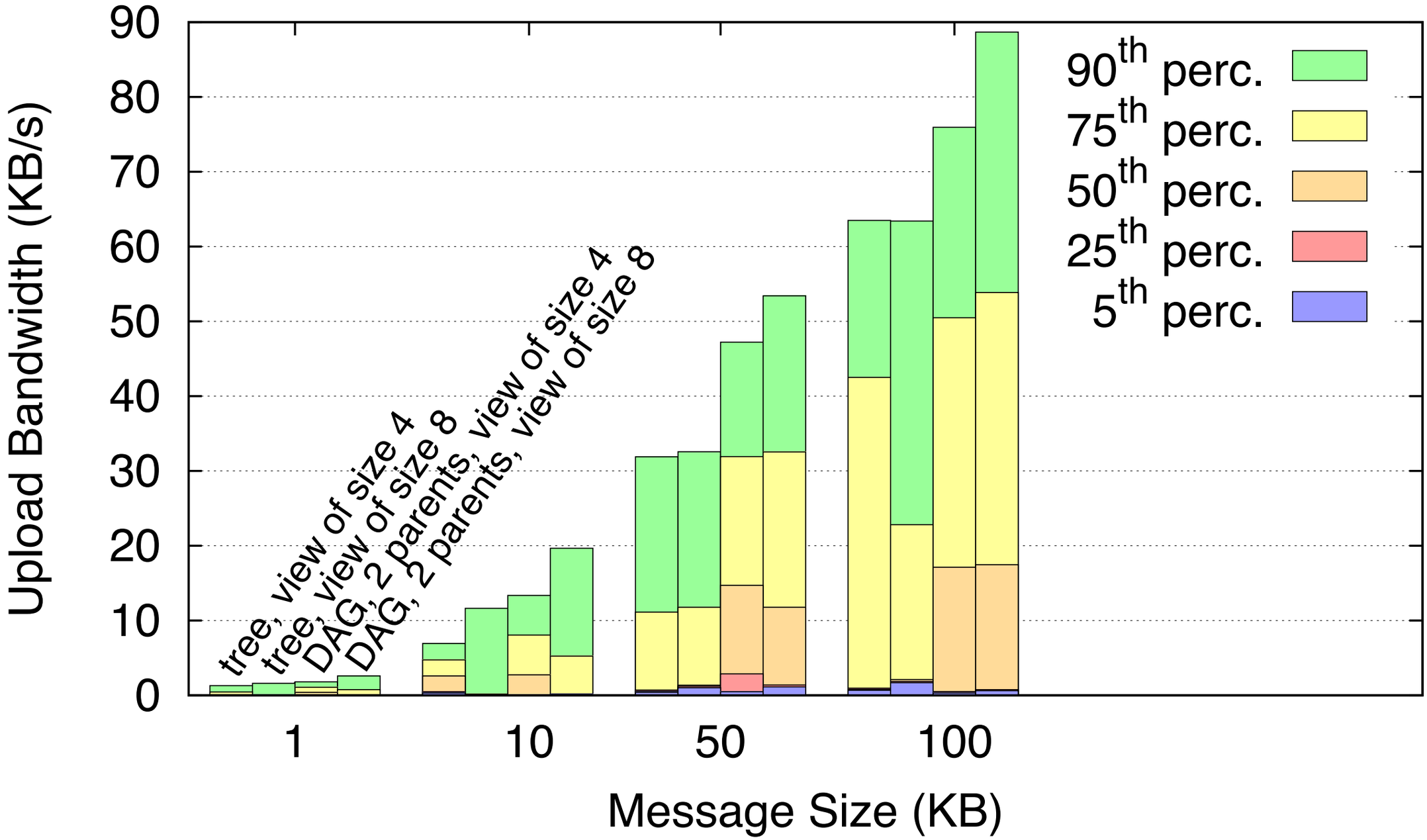
Failure Detection

- At the *Reactive Peer Sampling Service* (RPSS) level
  - Constant neighbors monitoring
    - Takes place on persistent connection
    - Also for links not used for dissemination
  - Classical failure detector
- "Informed" failure detector
  - Use hint from dissemination layer
  - Use deactivated links to send notifications of received message *ids*
    - Reception of a id for an (yet) unknown message
      => triggers instantaneous monitoring of parent
  - Allows reducing monitoring frequency while keeping reactiveness

➡ Detection of failed parent triggers link reactivation

BRISA Bandwidth usage: upload

512 cluster nodes



Goals

- Efficient: management overhead is low with respect to application data; low latency
- Robust: service continuity even under faults and churn
- Scalable: able to handle system growth sublinearly