Ordered slicing of very large scale overlay networks

Mark Jelasity
University of Bologna, Italy

Anne-Marie Kermarrec
INRIA Rennes/IRISA, France

Motivation

- Potential of P2P technology
- Several applications running on the same infrastructure
- Need for network partitioning
  - Decentralized
  - Robust
  - Handle dynamics
  - Customizable

Gossip-based approach to “slice” the network

Outline

1. Introduction
2. Background of gossip-based membership algorithms
3. Ordered slicing
4. Simulation results
5. Conclusion

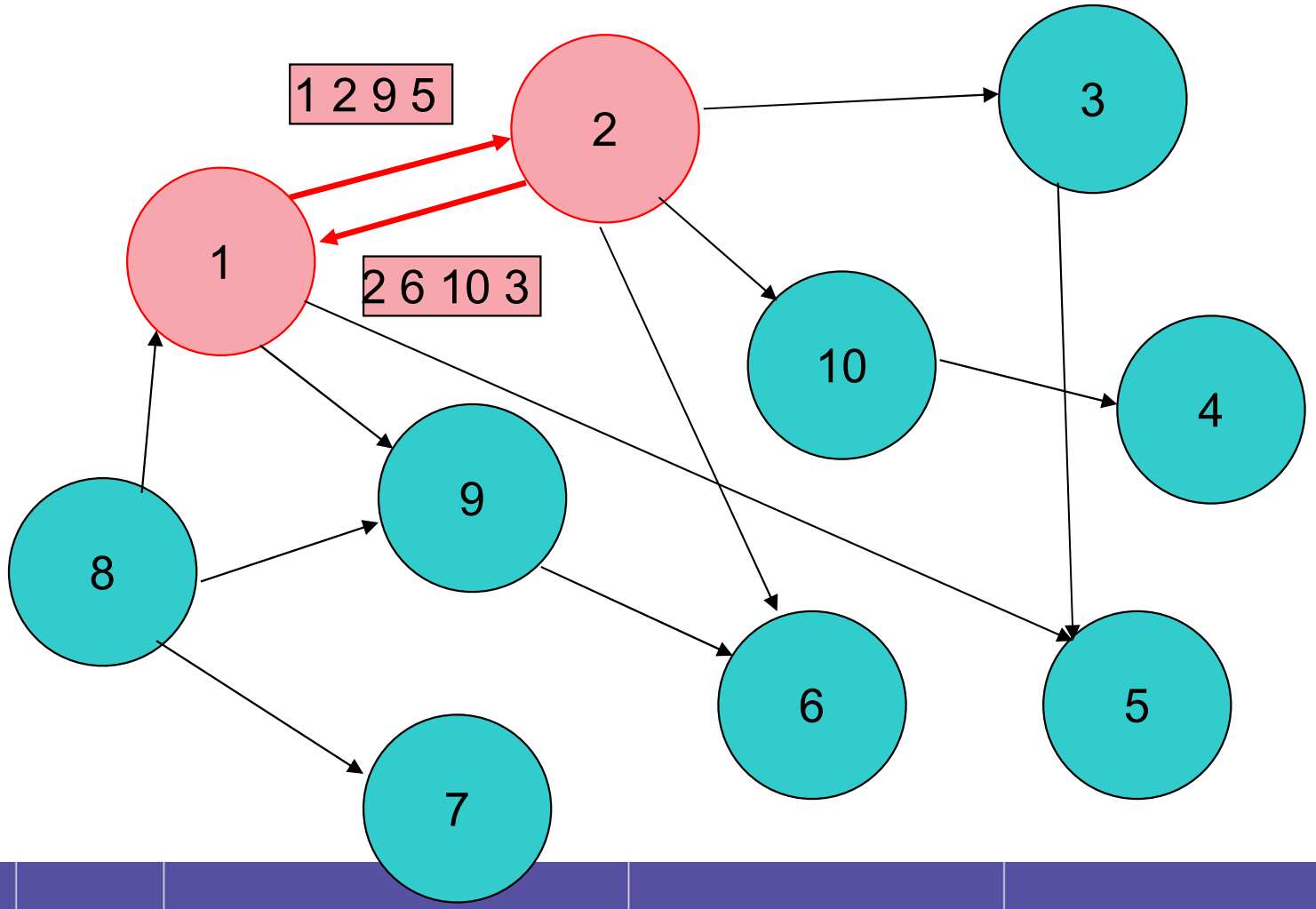
Objective

- Targeted applications
  - Deskstop grids
  - Testbed platforms (PlanetLab)
  - Telco applications
- Resource assignment
- Objective
  - Create and maintain partitions (slices) as subsets of the network in a fully decentralized manner
  - Ordered nature: along a single attribute (memory, bandwidth, computing power)
- Our approach: Use a gossip-based approach to estimate to which partition a node belongs
    - Scalable
    - Robust
    - Based on local knowledge

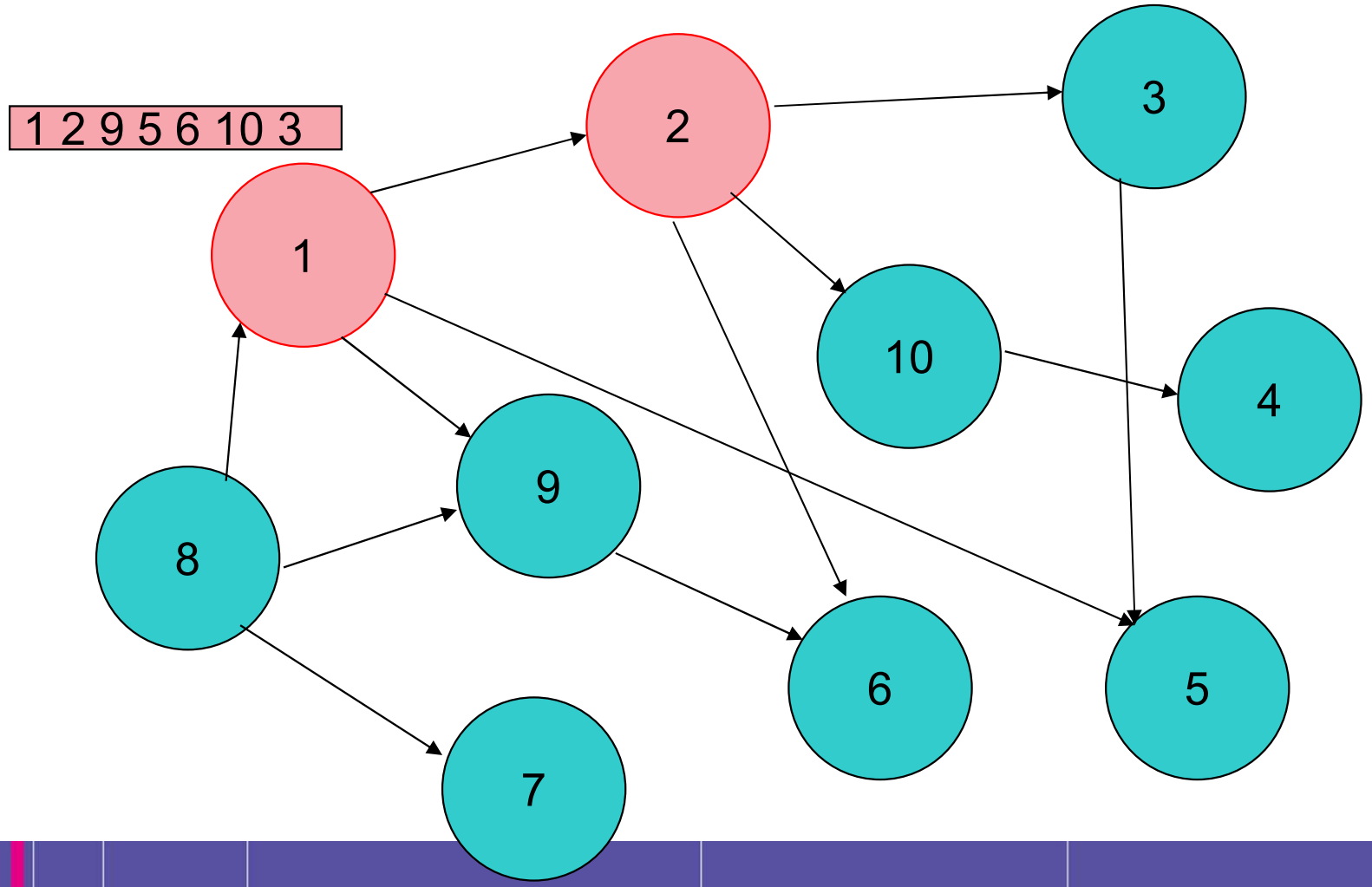
Gossip-based protocols

- Unstructured peer to peer networks
  - Highly resilient to failure and dynamics
- Gossip-based membership protocols
  - Periodic exchange of information between nodes
- Basic functionality: Peer sampling
  - Provide a sample of peers given a metric
  - Random sampling
- Applications
  - Event dissemination
  - Recovery protocols
  - Aggregation

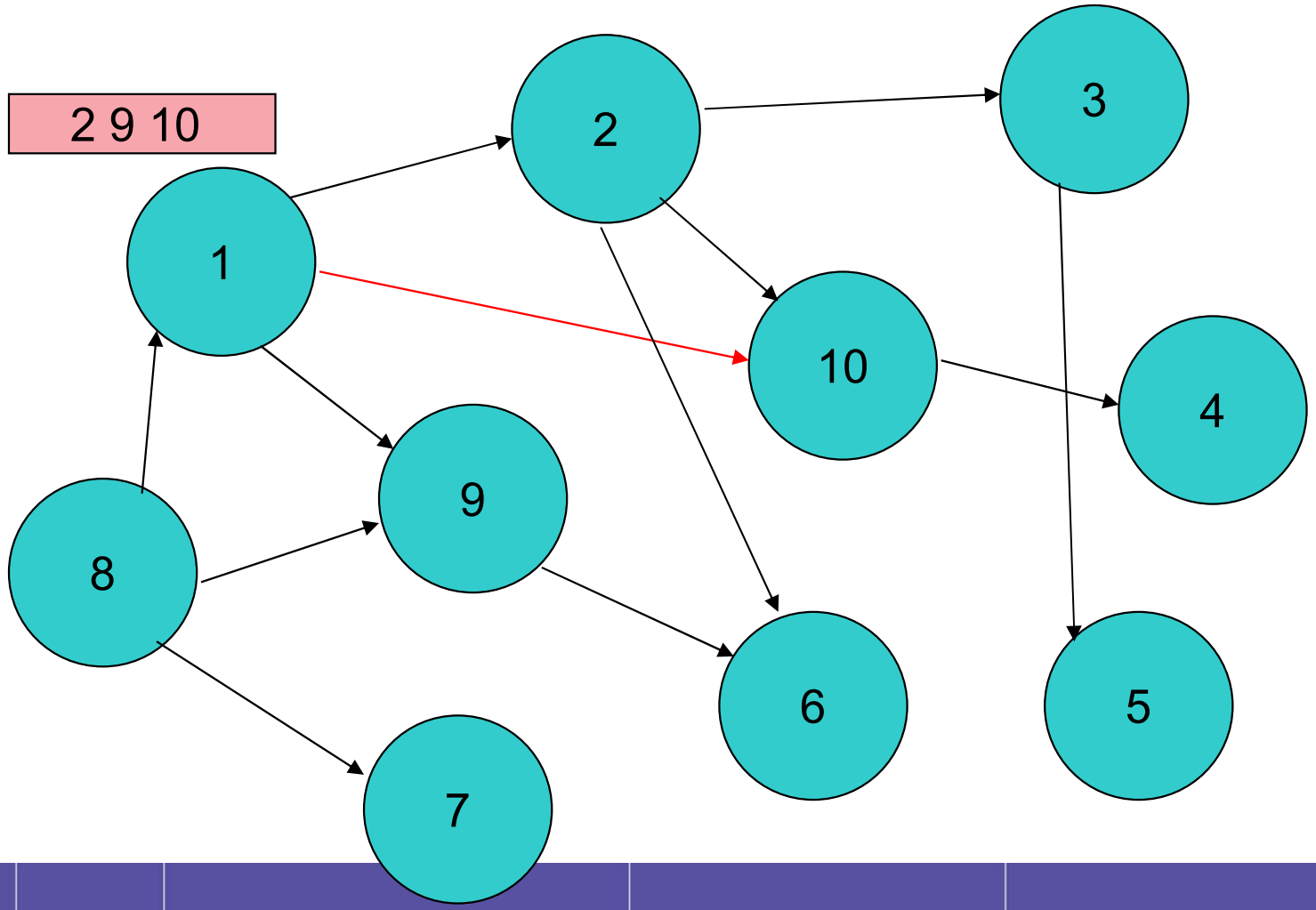
Gossip-based generic protocol



Gossip-based generic protocol



Gossip-based generic protocol



Design space

- Periodically each peer initiates communication with another peer
- **Peer selection**

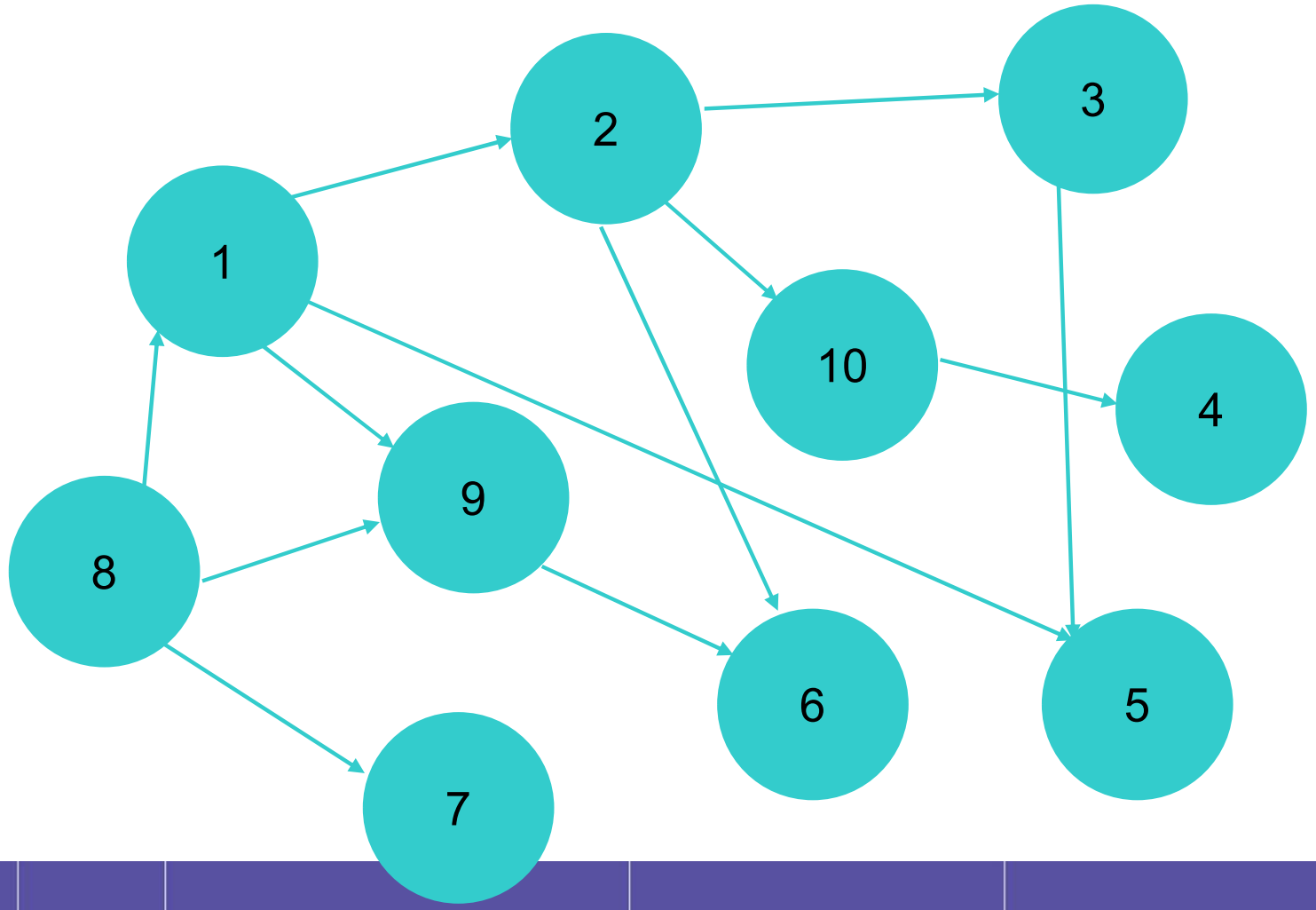
  Selectpeer(): return the IP@ of an alive peer
- **View propagation**

  How peers exchange their membership information
- **View selection**: Select (c, buffer)
  - c: size of the resulting view
  - Buffer: information exchanged

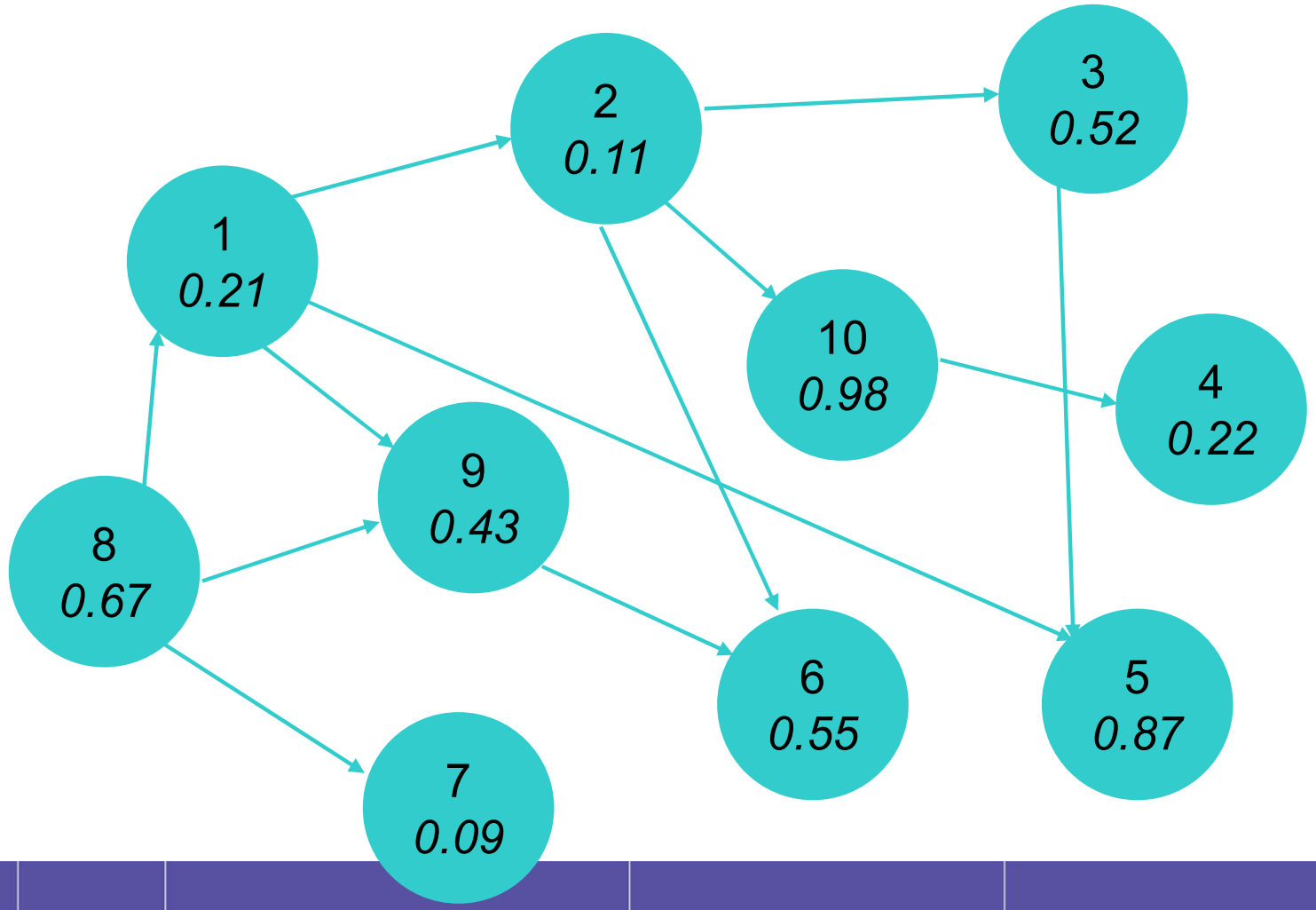
Peer sampling algorithm

- Common framework for existing gossip-based protocols
  - Lpbcast [Eugster & al, ACM TOCS 2003]
  - Cyclon [Voulgaris & al JNSM 2005]
  - Newscast [Jelasity & van Steen, 2002]
    - Peer selection: Select a peer at random from the view
    - View propagation: Pushpull
    - View selection: Select the freshest entries

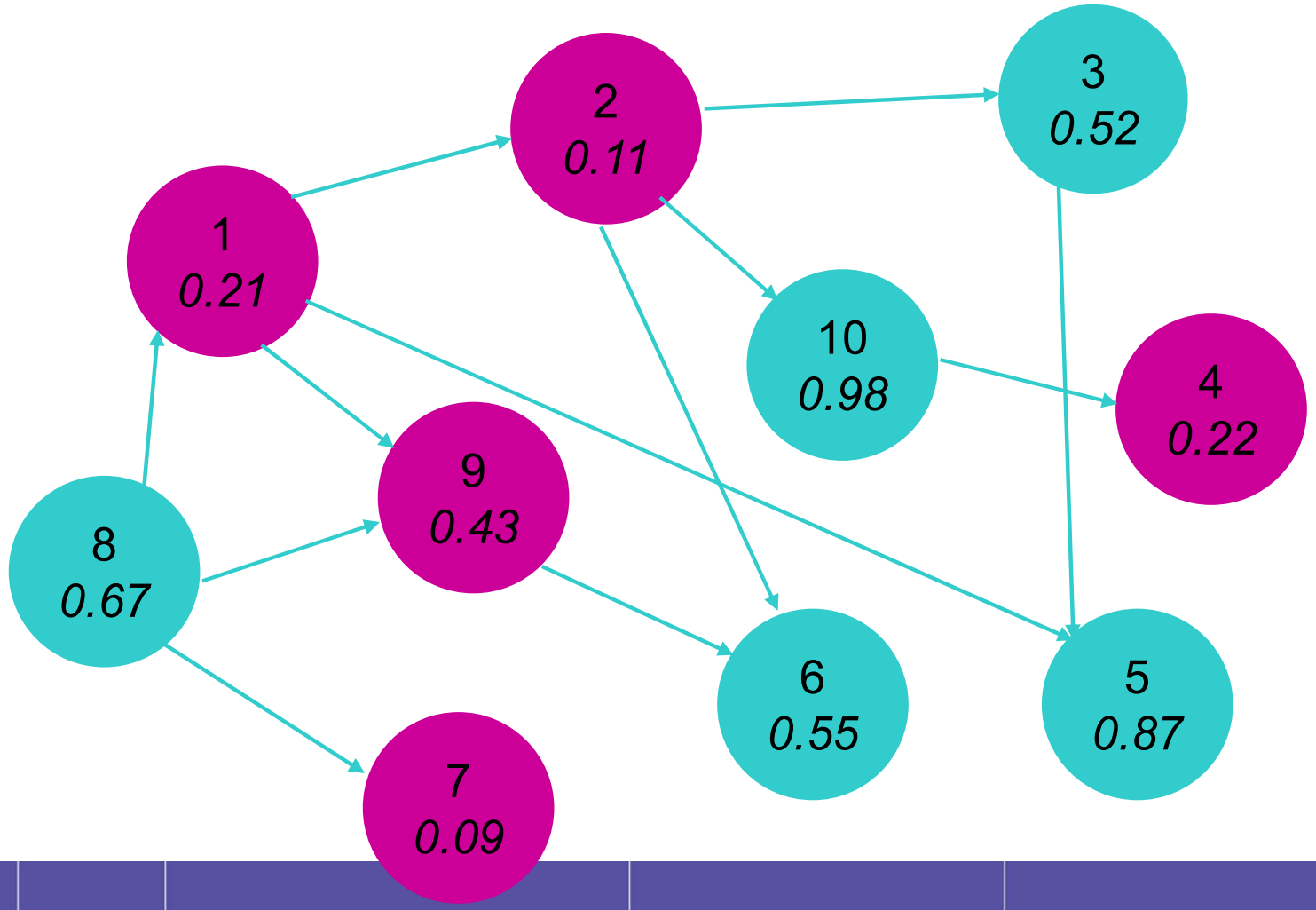
Random slices



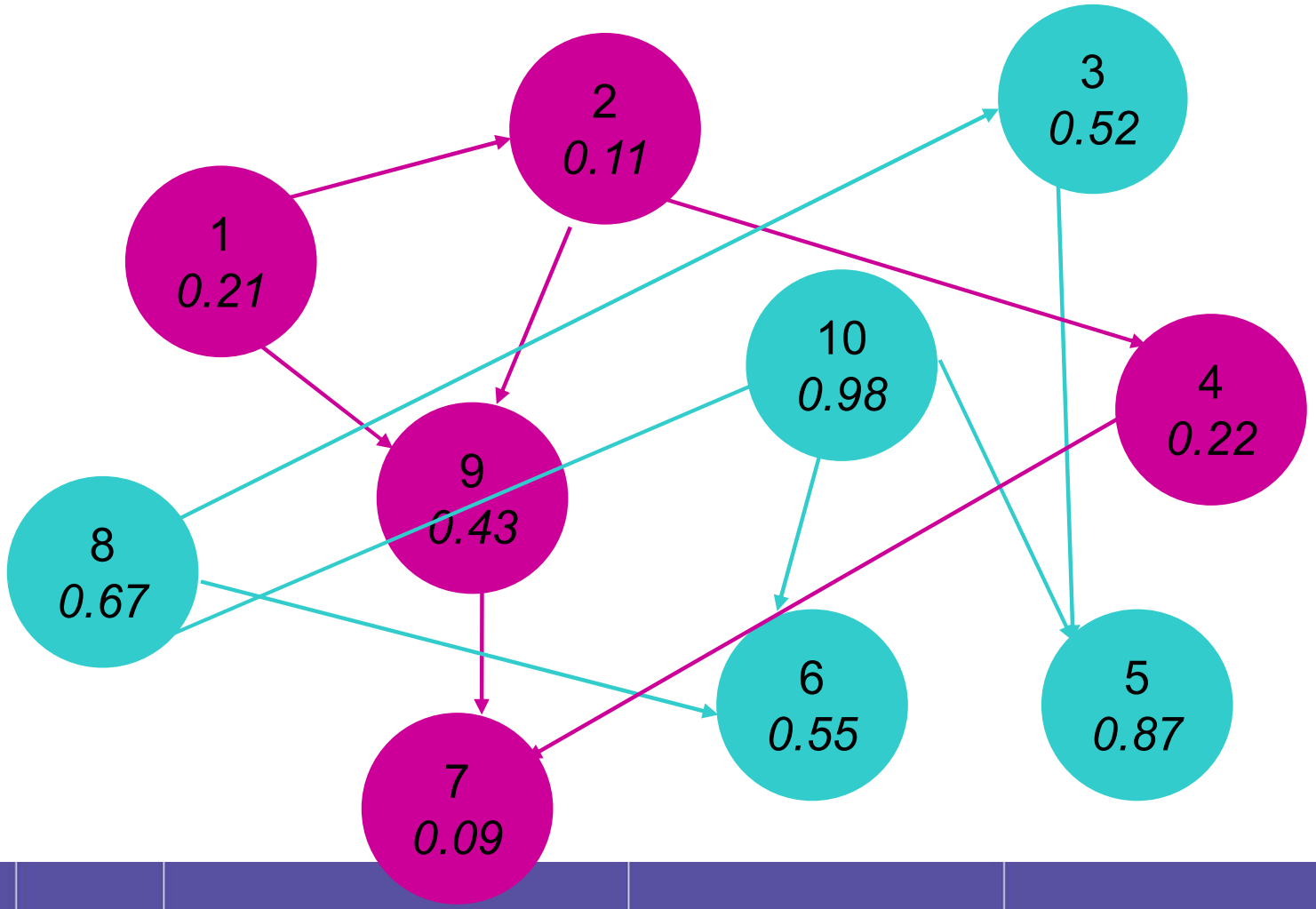
Random slices



Random slices



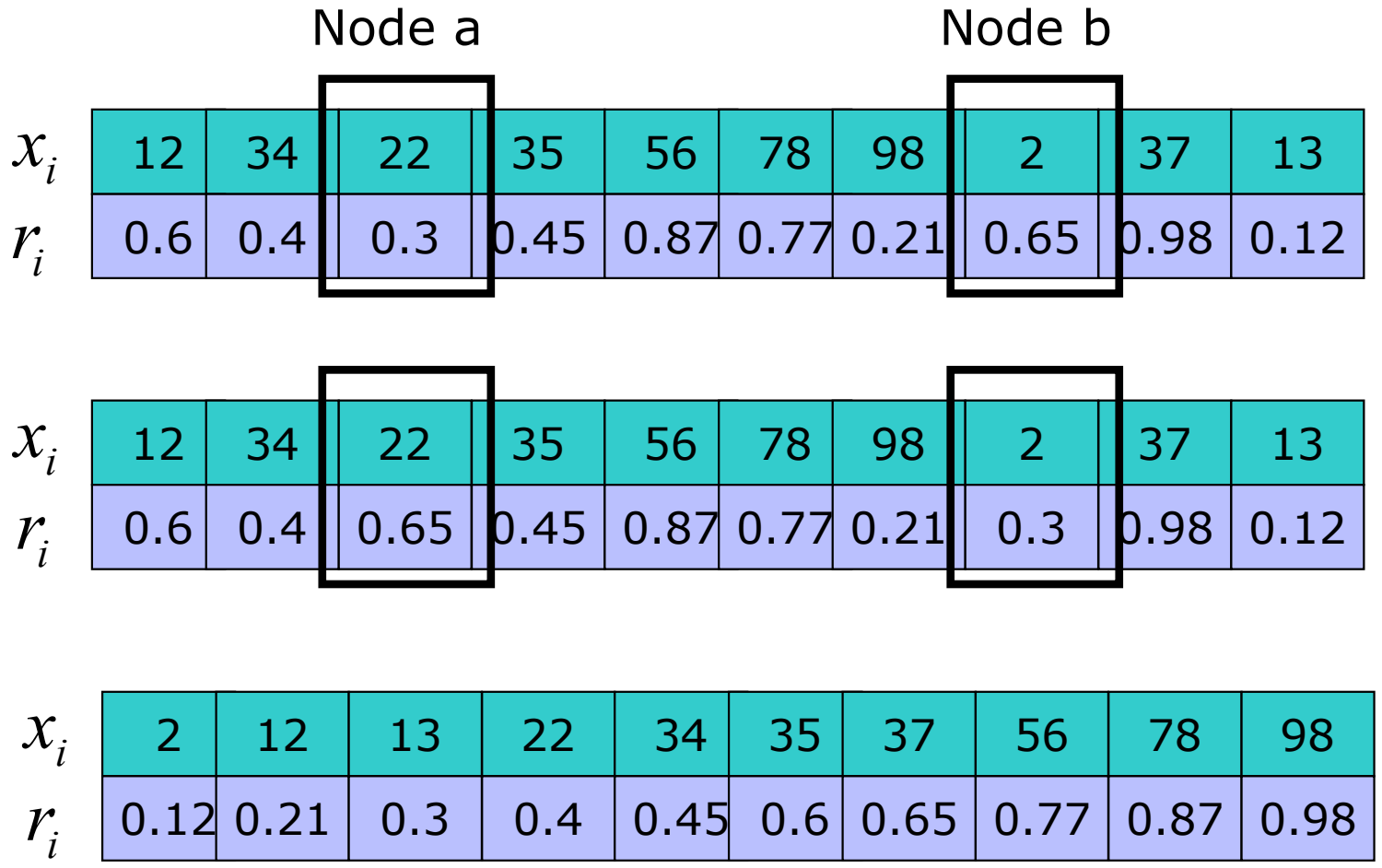
Random slices



Ordered slices: System model

- Problem
  - Automatically assign a node to a slice
  - Along an attribute metric
- System model
  - Crash-only nodes
  - Each node i has
    - an attribute x_i
    - a random number r_i
    - a view of *c* entries (peer sampling)
    - a time stamp
  - Each node belongs to one slice

Ordered slicing algorithm: basic operation



Ordered slicing algorithm

Wait (t)

p<- random element from view

buffer<-*view* U {(my@, my_stamp, x_q, r_q)}

Send buffer to *p*

Receive buffer(*p*)

view<-freshest *c* entries from {buffer(*p*) U *view*}

i<- peer such that $(x_i - x_q)(r_i - r_q) < 0$

Send (x_q, r_q) to *i*

$r_q = r_i$

Active thread at peer *q*

Receive buffer(*q*) from *q*

buffer<-*view* U {(my@, my_stamp, x_p, r_p)}

Send buffer to *q*

view<-freshest *c* entries from {buffer(*p*) U *view*}

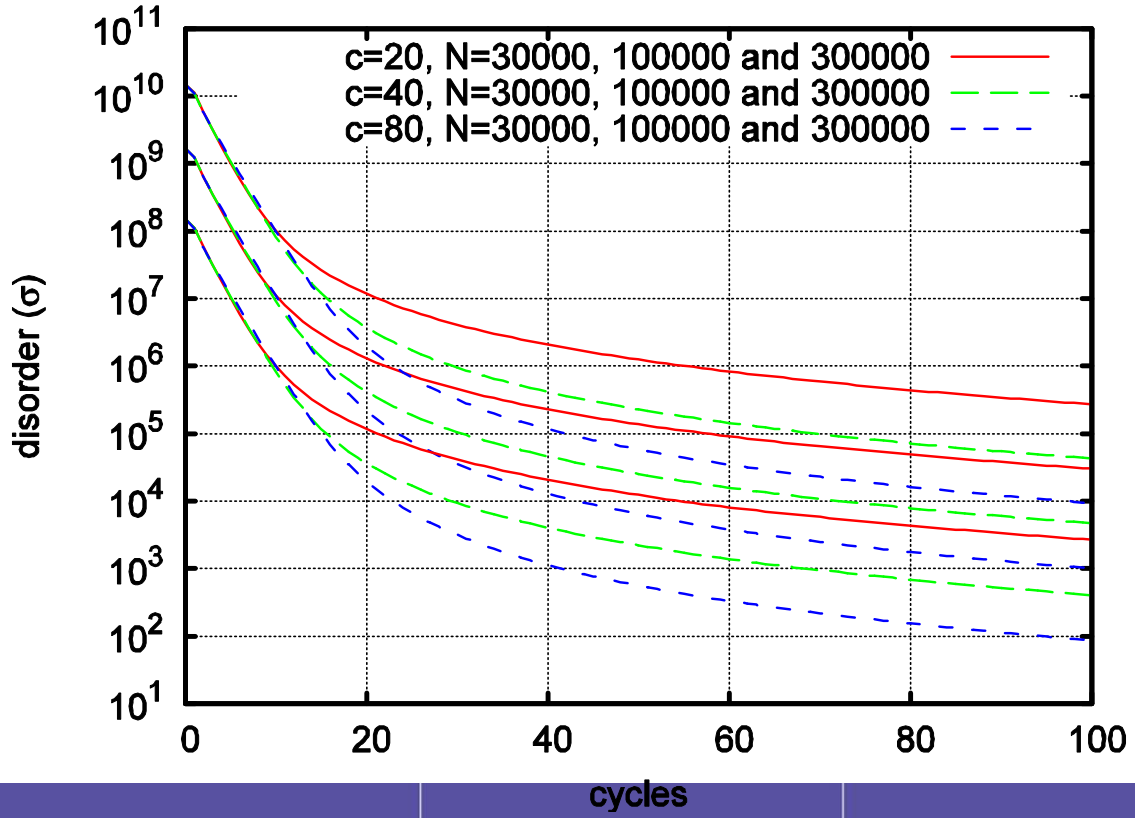
Passive thread at peer *p*

Ordered slicing algorithm: maintenance

- New nodes discovered using the random peer sampling
- Random number ensures uniform spread
- Once the order stabilizes: each node knows to which slice it belongs
- Example
  - A peer with a number <0.5 knows in the first 50% of the nodes according to the metric
- Slice creation and maintenance

Disorder measure

$$\sigma(t, r_{i_1(t)}, \ldots, r_{i_N(t)}) = \sigma(t) = 1/N \sum_{j=1}^{N} (j - i_j(t))^2$$



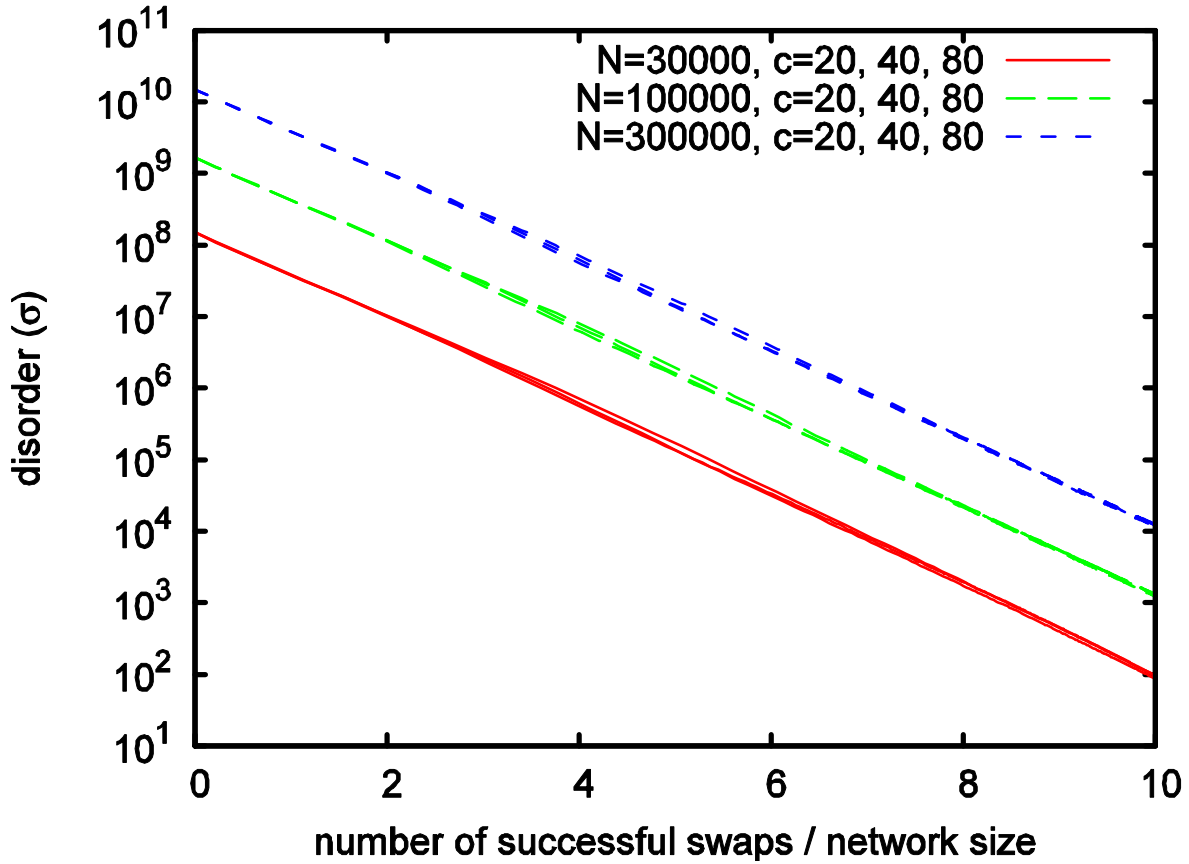
Analogy with average

- Weight conserving property

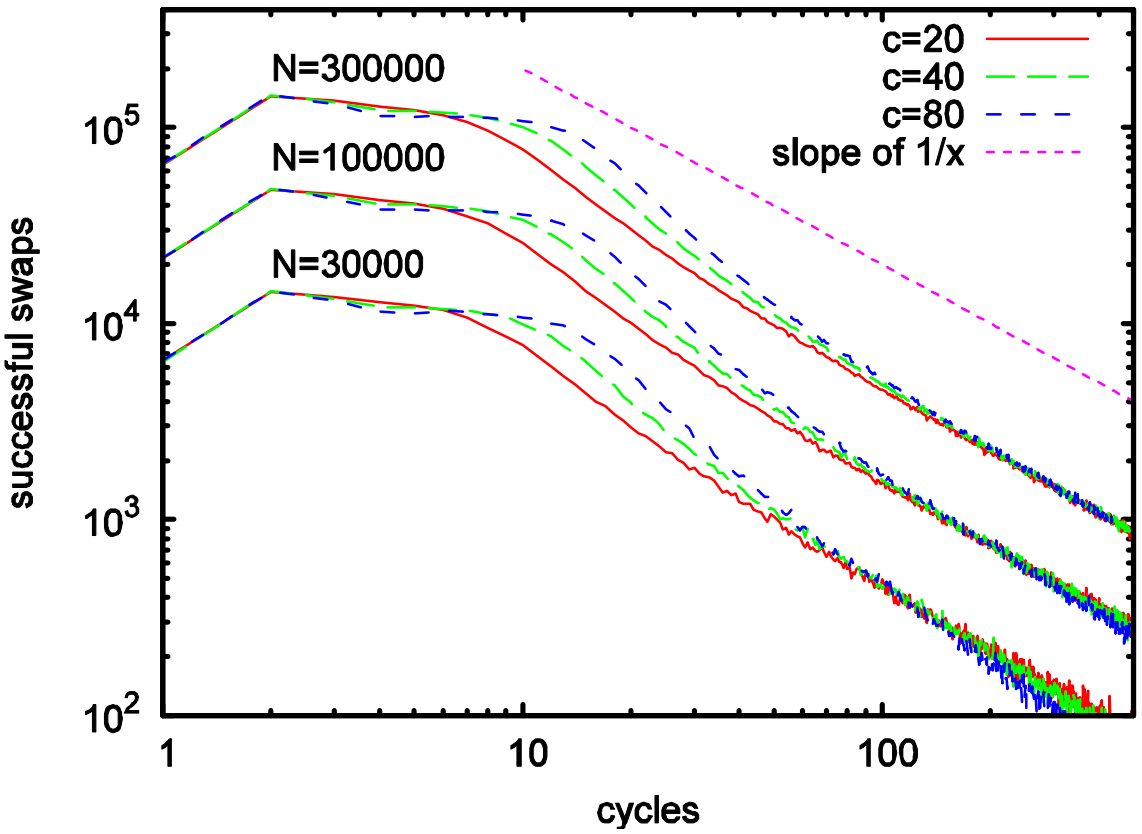
$$1/N\sum_{j=1}^{N} j - i_j(t) = 1/N\sum_{j=1}^{N} j - 1/N\sum_{j=1}^{N} i_j(t) = 0$$

- The swapping does not influence this value (=0) but always reduces the disorder value

Exponential decrease of the disorder



Swaps over time

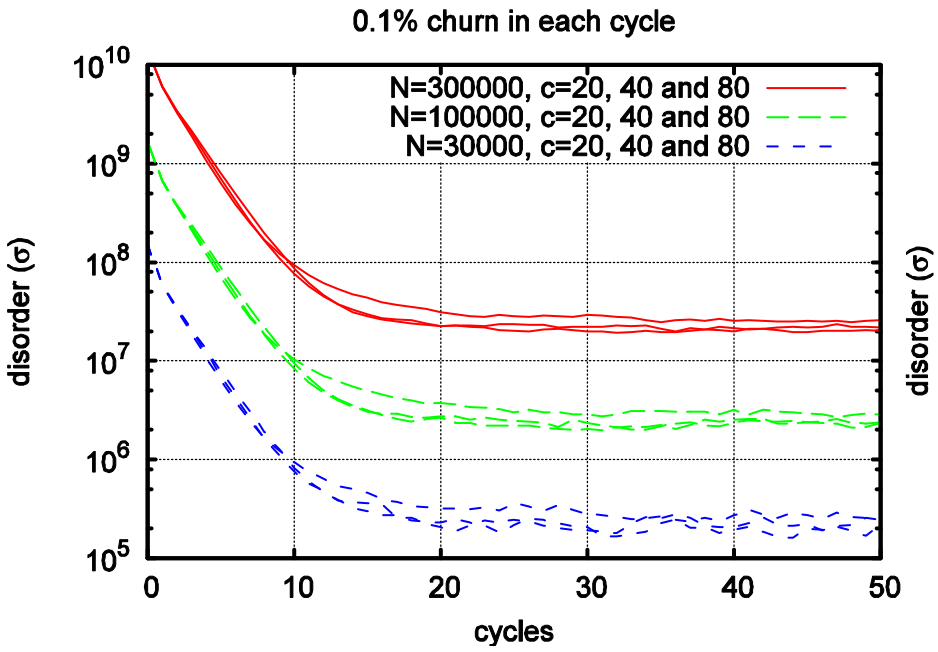


Simulation set-up

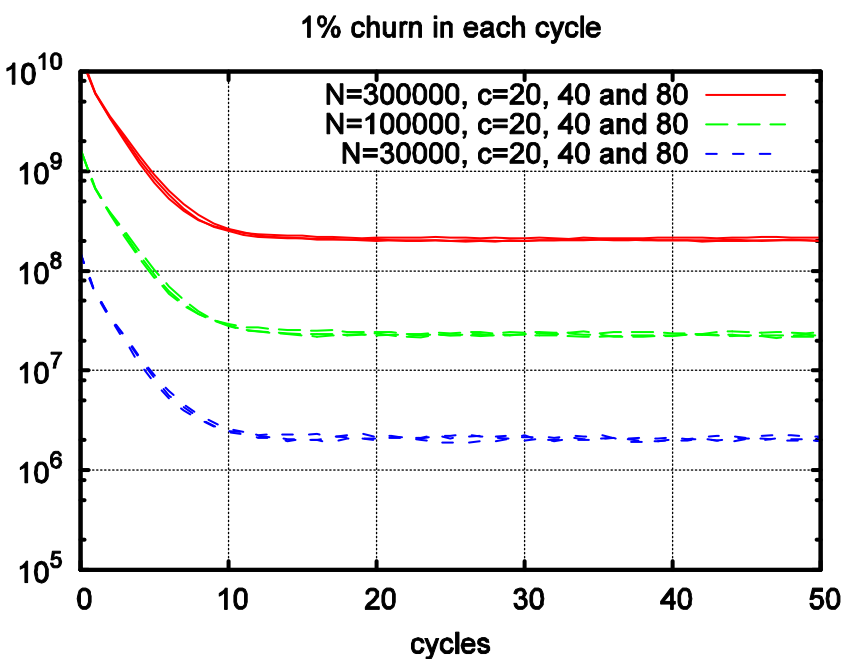
- PeerSim simulator
- Configurations
  - Network size: 30,000 – 100,000 – 300,000 node overlay
  - Views: c=20, 40 and 80
  - Unreliable communication channels
  - Churn
  - Age bias: peer selection based on age similarity
- Metric: disorder

Churn

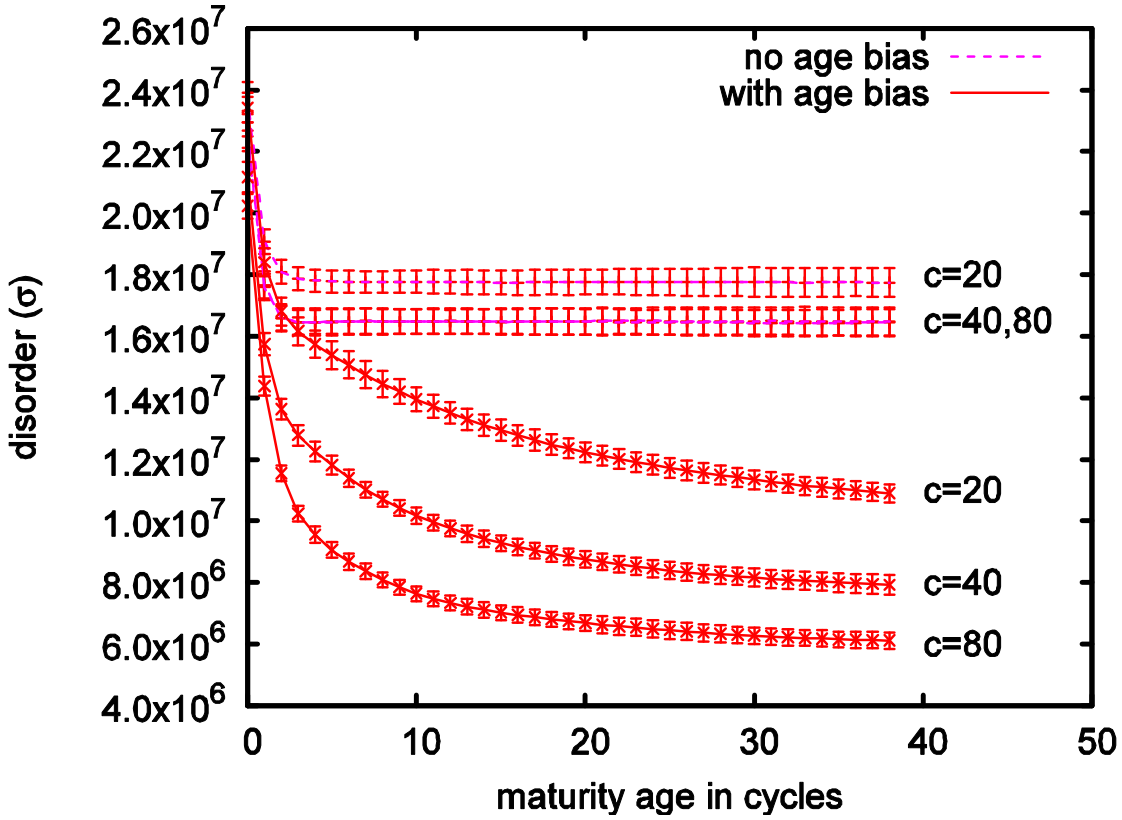
0.1%


0.1% churn in each cycle
N=300000, c=20, 40 and 80
N=100000, c=20, 40 and 80
N=30000, c=20, 40 and 80
disorder (σ)
cycles
10^{10}
10^{9}
10^{8}
10^{7}
10^{6}
10^{5}
0
10
20
30
40
50

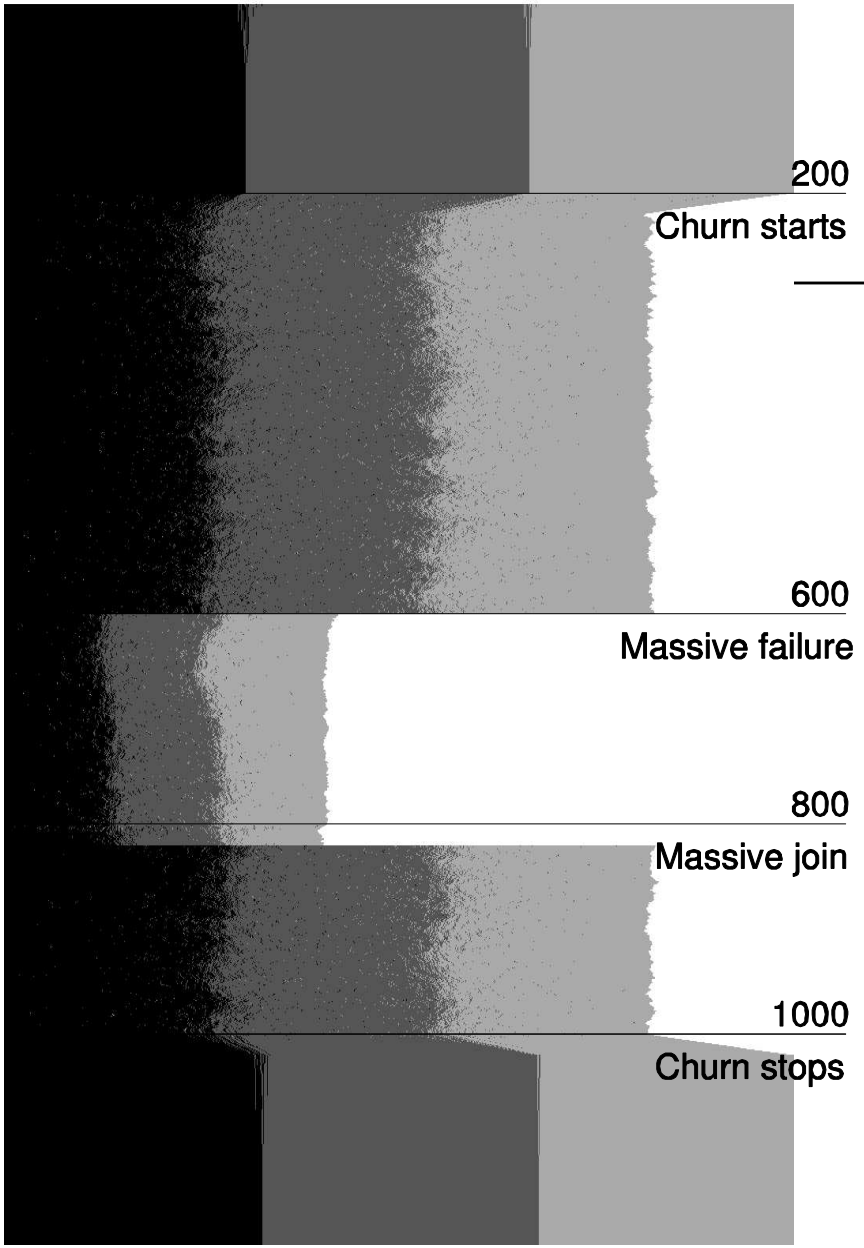
1%


1% churn in each cycle
N=300000, c=20, 40 and 80
N=100000, c=20, 40 and 80
N=30000, c=20, 40 and 80
cycles
10^{10}
10^{9}
10^{8}
10^{7}
10^{6}
10^{5}
0
10
20
30
40
50

Age-based technique



Young nodes disordered
Old nodes protected



Example

- Quick stabilization
- Relatively well-defined slices
  - constant churn
  - massive failure
  - massive join
- Stabilizes as soon as churn stops

Conclusion & future work

- Gossip-based solution to partition a dynamic network according to a given metric
- Resilient to churn
- Future work
  - Slices maintenance
  - Improve the peer selection

Thank you !