Using Epidemics and Diffusion for Decentralized Monitoring and Control of Fully Distributed Systems

Márk Jelasity

Department of Computer Science

University of Bologna

Italy

Project funded by the Future and Emerging Technologies arm of the IST Programme

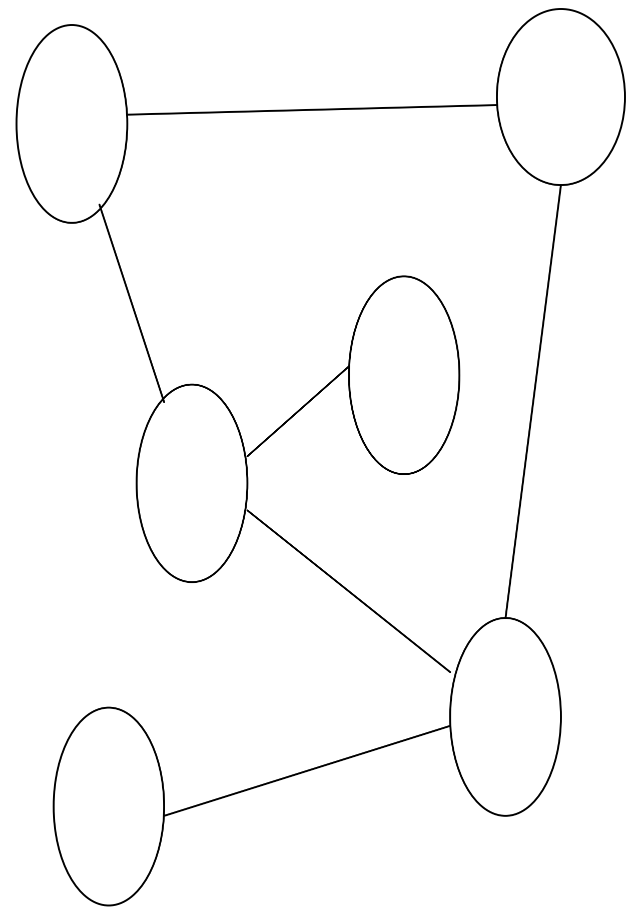
Outline

- System Model
- Epidemics
- Diffusion
- Applications

System Model

Components

- Nodes
- Random Communication Topology
  - Neighbors (“knows about” relation)
  - Maintained by specific protocols



Push-Pull Protocol Skeleton

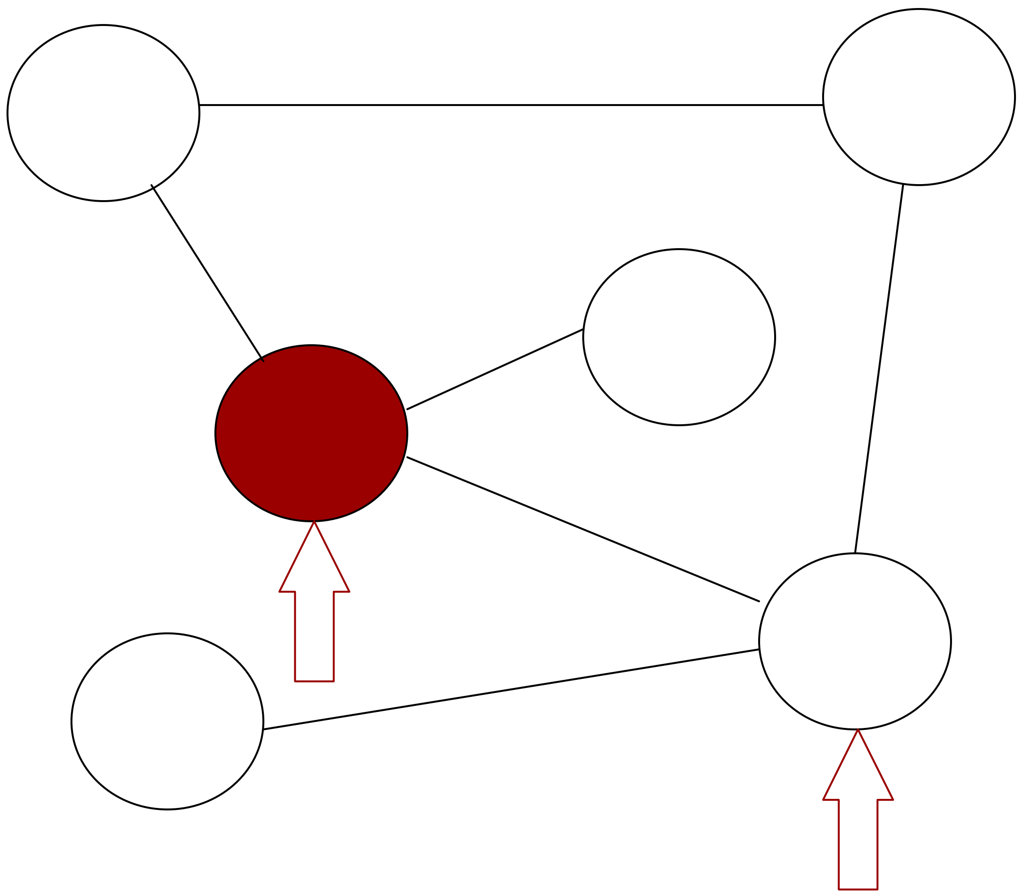
```
// active thread
do forever
   wait(T time units)
   peer = selectRandomNeighbor()
   send state to peer
   receive peer.state from peer
   state = updateState(state,peer.state)

// passive thread
do forever
   (peer,peer.state) = waitMessage()
   send state to peer
   state = updateState(state,peer.state)
```

Some comments

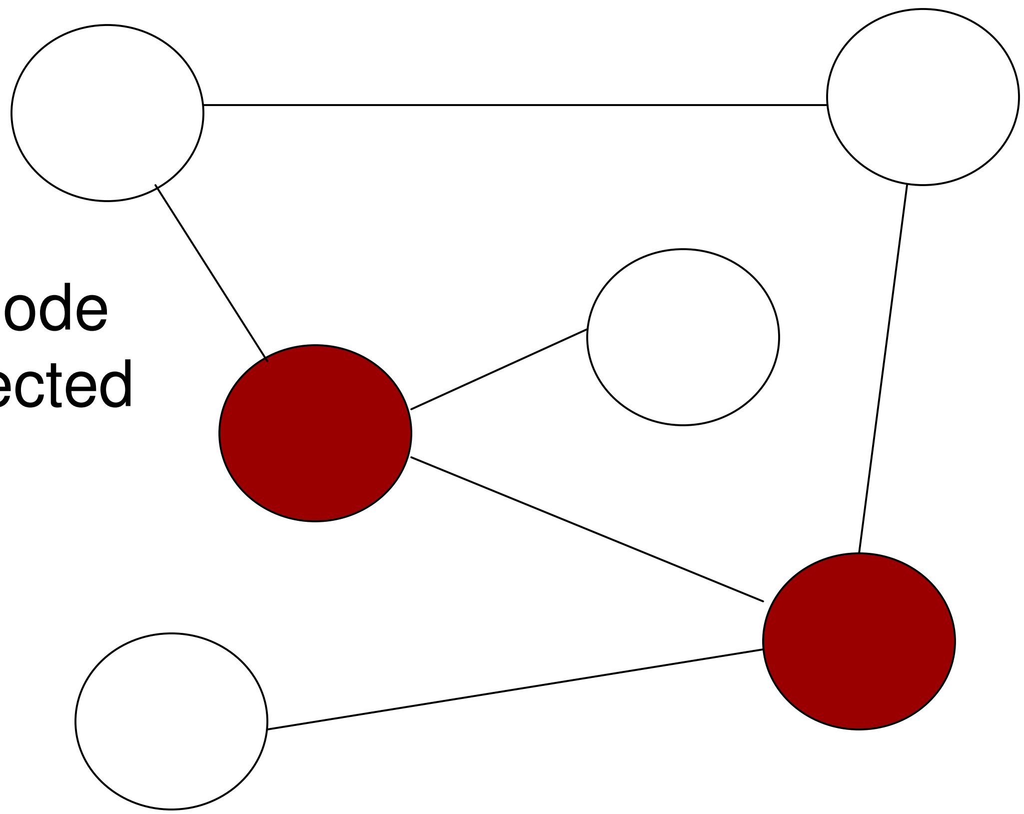
- A cellular automaton-like model
  - cycles (each T time units interval)
  - state updates based on neighborhood state
- But: topology is
  - random (not regular, low diameter)
  - can be dynamically changing over time
  - maintained by protocols that can deal with
    - node failure
    - new nodes joining the network

Epidemics

Basic operation

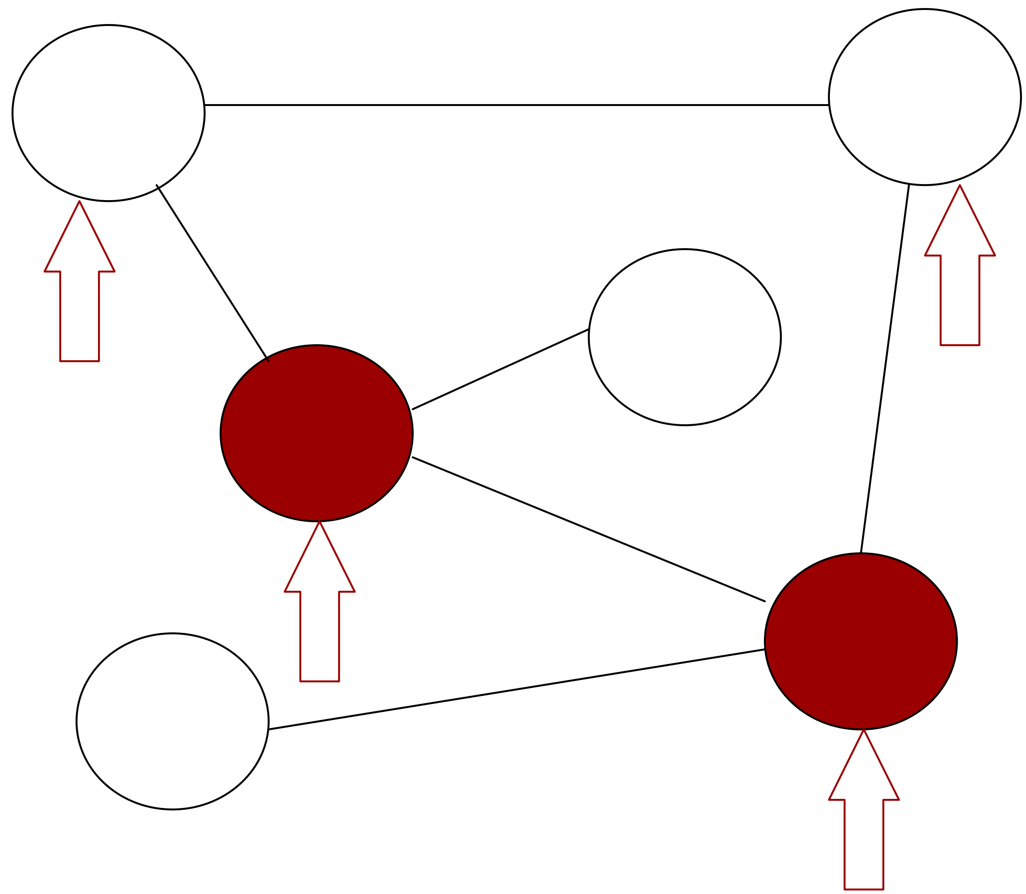


Basic operation

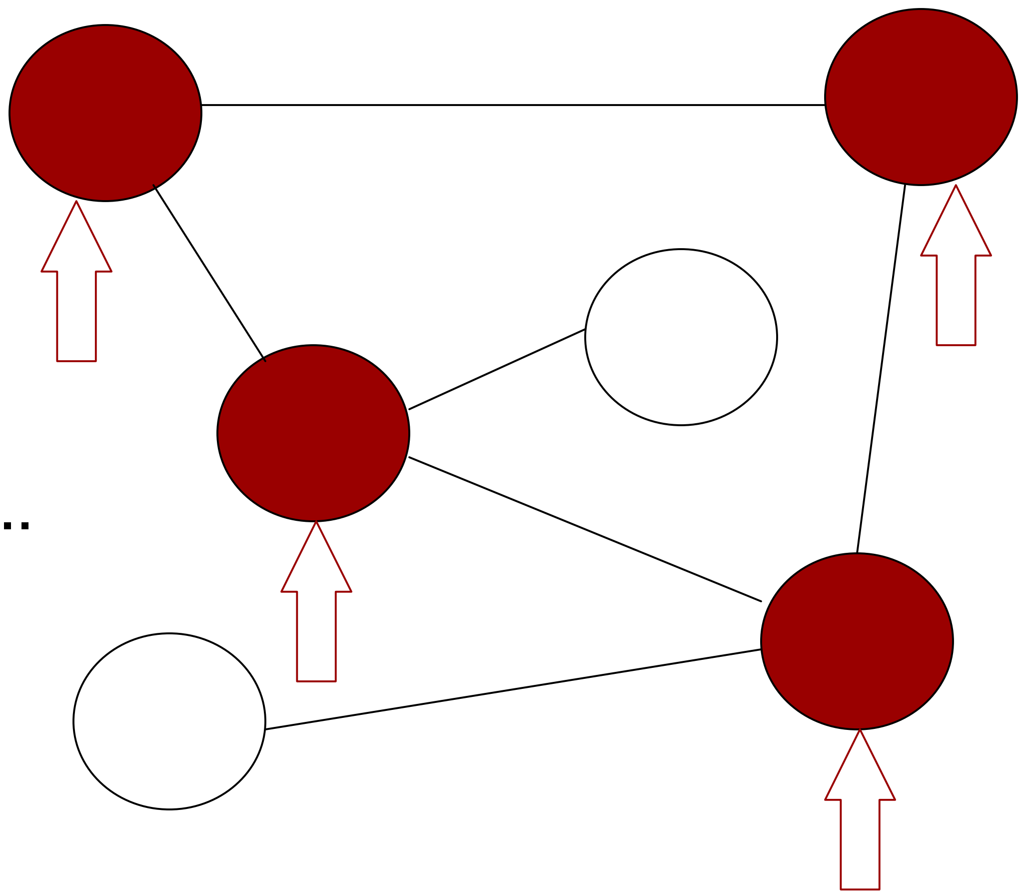


A new node
gets infected

Basic operation



Basic operation



And so on...

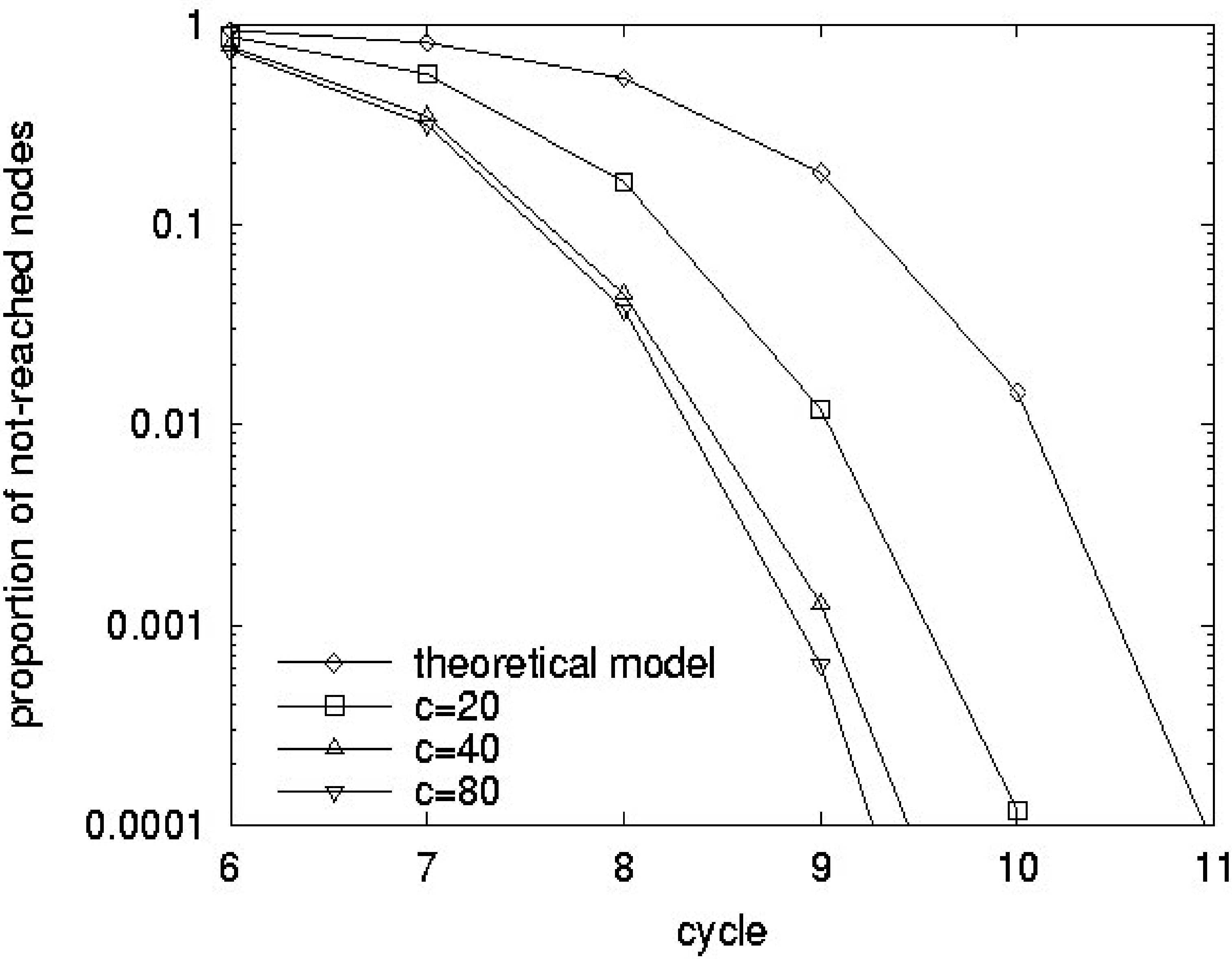
Some Examples

- Epidemics for database updates
  - state: infected or not. Infected state means knowing a new piece of information
  - updateState(s1, s2): if any of s1 or s2 is infected, new state is infected
- Epidemics for finding a maximal value
  - state: the maximal value seen so far
  - updateState(s1,s2):= max(s1,s2)

Convergence speed

In the push pull model the follwing is a good approximation of the superexponential convergence speed

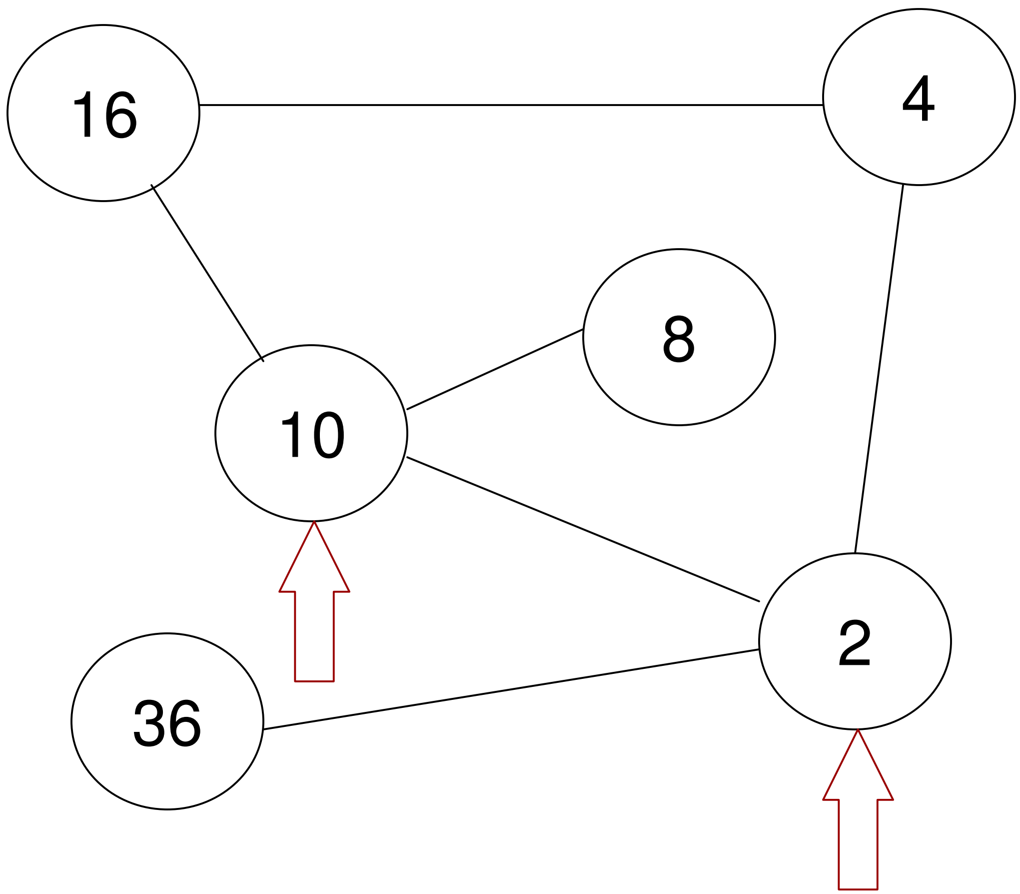
$$p_{i+1} = p_i p_i \left(1 - \frac{1}{N}\right)^{N(1-p_i)} < p_i^2 < p_0^{2^{i+1}} = \left(1 - \frac{1}{N}\right)^{2^{i+1}}$$



1
0.1
0.01
0.001
0.0001
6
7
8
9
10
11
cycle
proportion of not-reached nodes
theoretical model
c=20
c=40
c=80

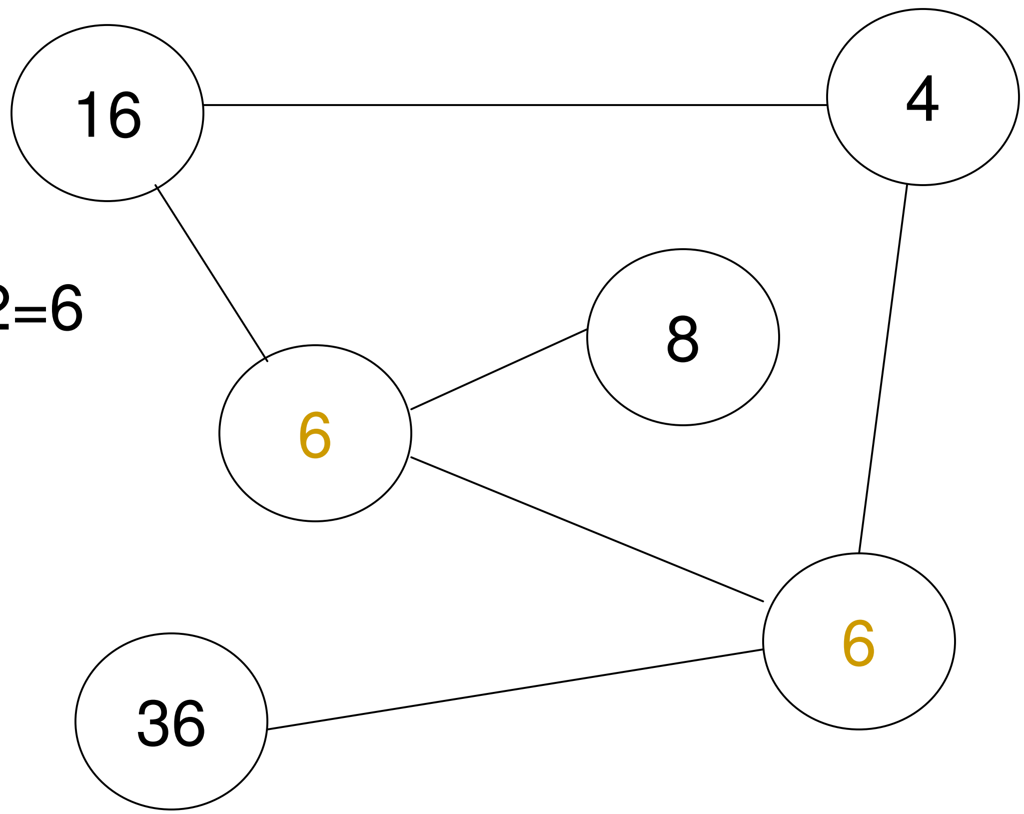
Diffusion

Basic operation

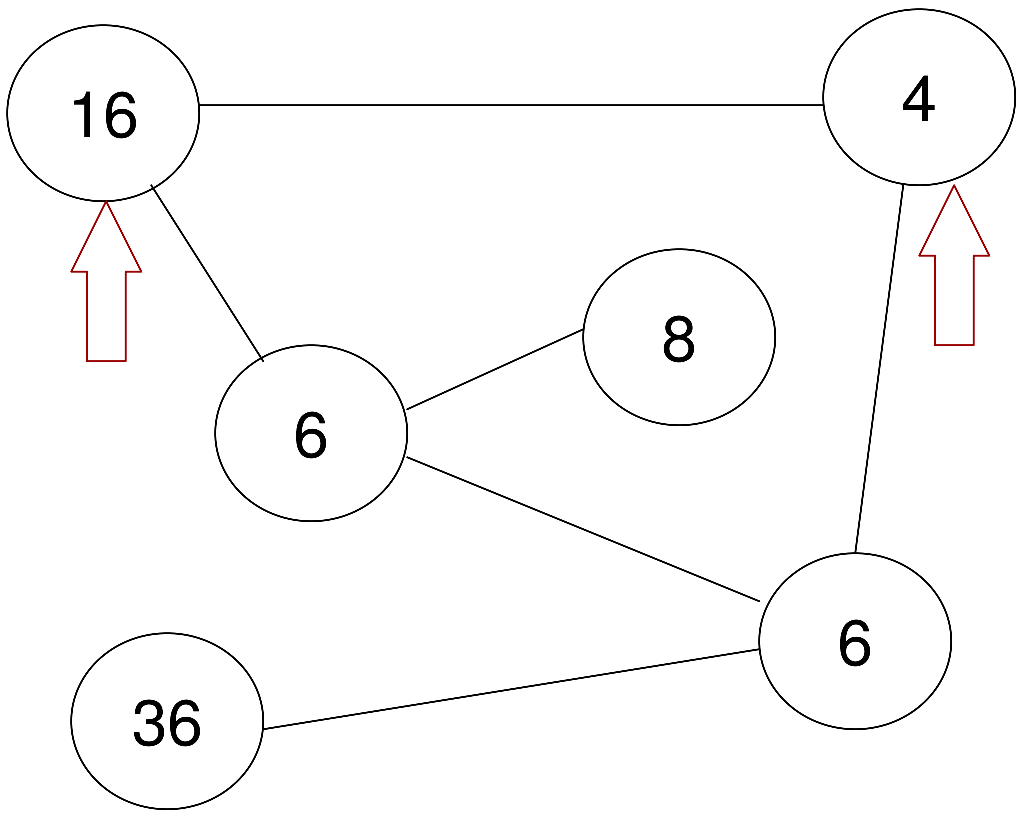


Basic operation

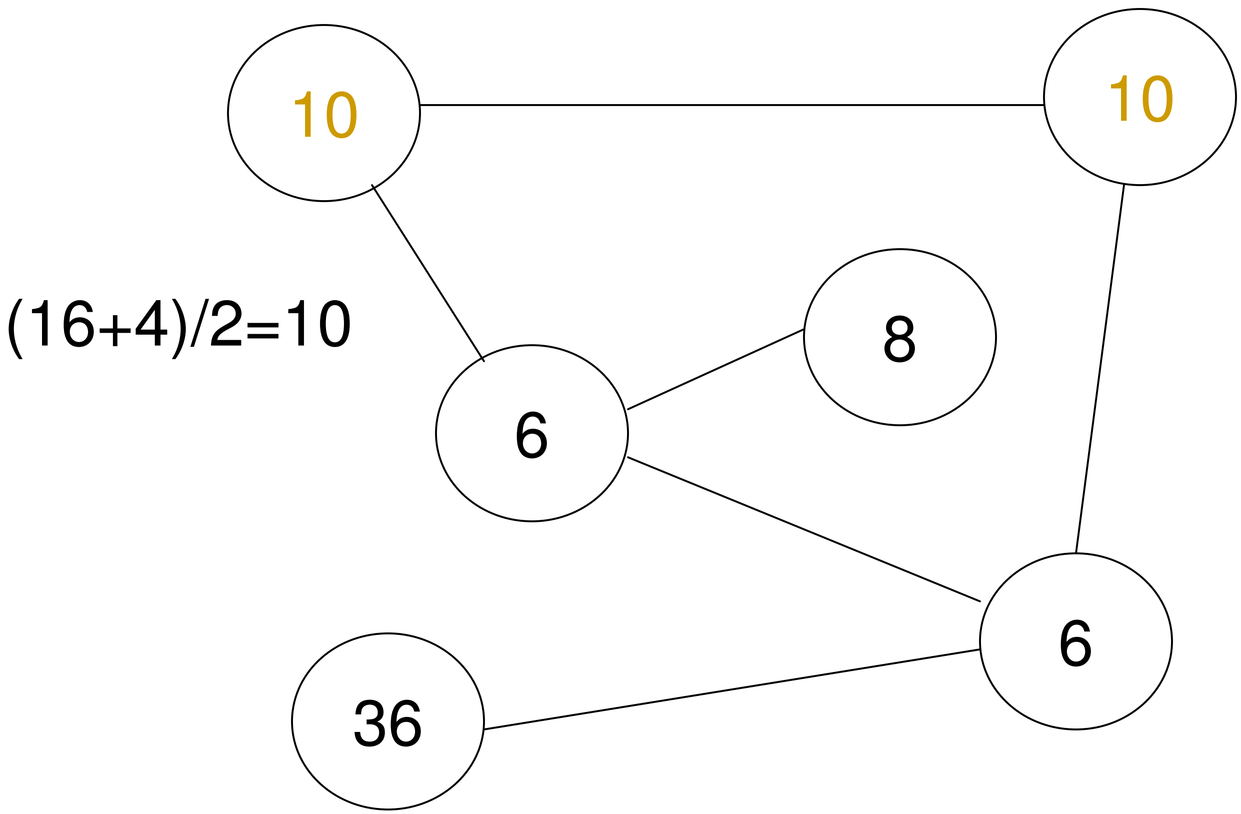
(10+2)/2=6



Basic operation



Basic operation



Some Examples

- Diffusion for calculating the average
  - state: current approximation of average in the whole system
  - updateState(s1, s2):= (s1+s2)/2
- Diffusion has lots of other applications including
  - network size estimation
  - calculating variance (or any moments)

Some Comments

- Different from load balancing due to lack of constraints
- Diffusion is normally studied on regular topologies (grid)
- We are interested in realistic topologies: random, small-world, scale-free, etc.
- Diffusion is often the basis of biological self-organization like aggregation (2nd sense) and regeneration

Some Observations

- The procedure is convergent if the graph is connected
- Each node converges to the average of the original values

Summary of Our Theoretical Results

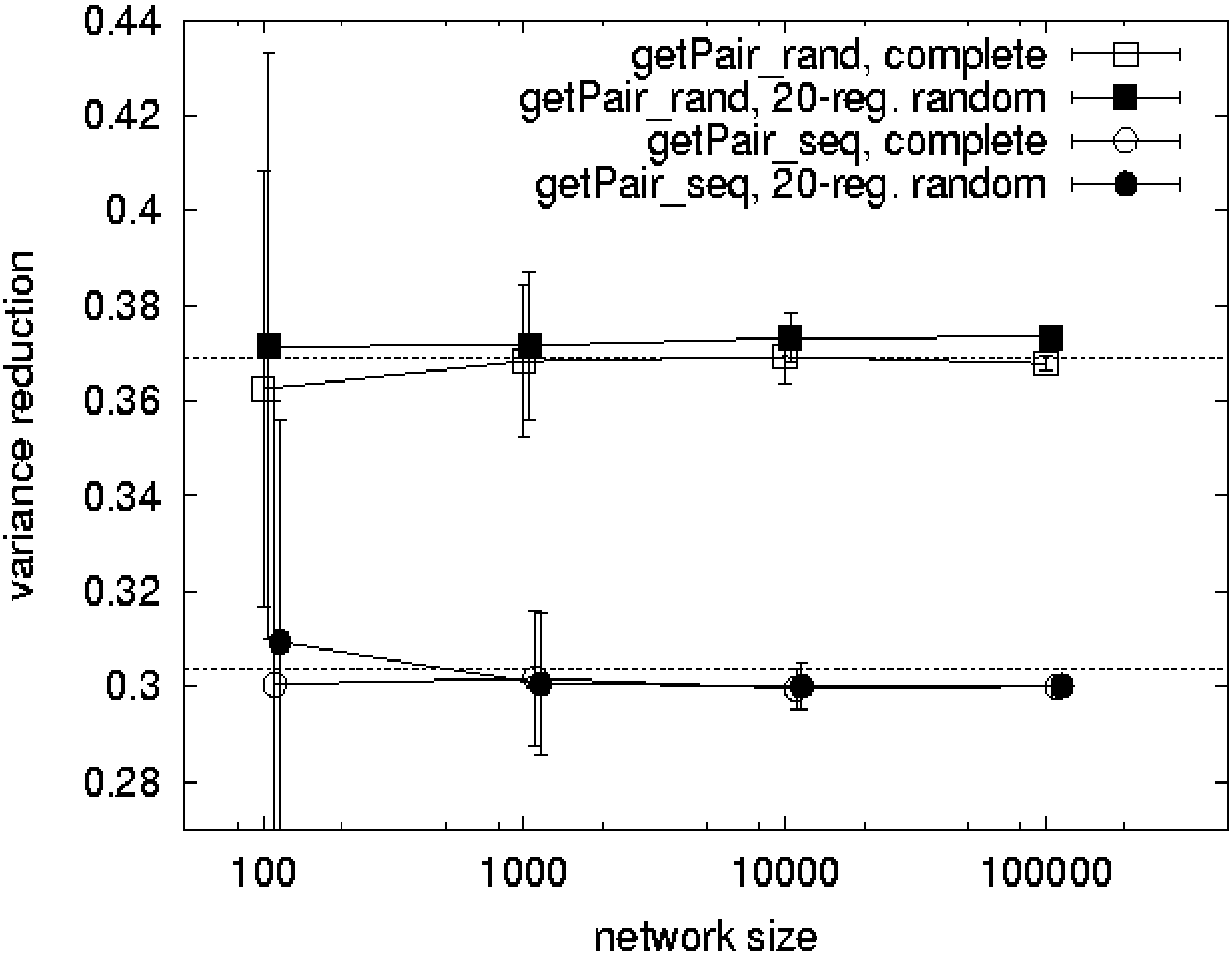
- On the fully connected topology convergence speed is exponential.
- On a random topology it is practically exponential.
- Node failure can destroy convergence above a theoretically described threshold.
- Dropping messages is not critical.

Case studies: realistic case

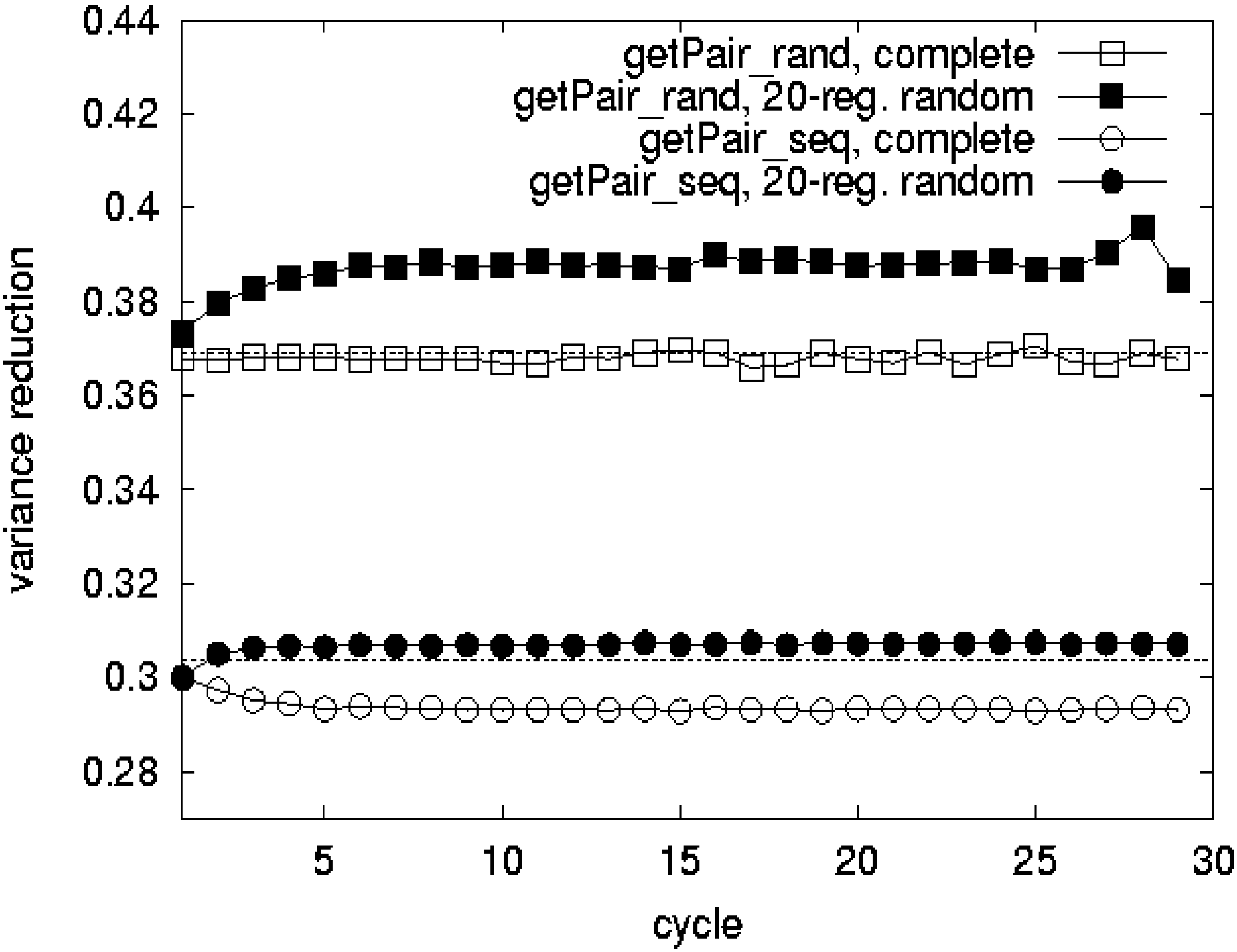
The rate of convergence is given by the formula

$$E(\sigma_{i+1}^2) \approx \frac{E(\sigma_i^2)}{2\sqrt{e}}$$

Where σ_i^2 is the empirical variance of the set of the approximations at the nodes in cycle i.



getPair_rand, complete
getPair_rand, 20-reg. random
getPair_seq, complete
getPair_seq, 20-reg. random
variance reduction
network size
0.44
0.42
0.4
0.38
0.36
0.34
0.32
0.3
0.28
100
1000
10000
100000



getPair_rand, complete
getPair_rand, 20-reg. random
getPair_seq, complete
getPair_seq, 20-reg. random
variance reduction
cycle
0.44
0.42
0.4
0.38
0.36
0.34
0.32
0.3
0.28
5
10
15
20
25
30

Conclusions

- Scalability: results independent of N
- Efficiency: convergence is very fast
- Robustness: the algorithm is highly robust to both node and message failure (not discussed in the present talk)

Applications

Epidemics: some examples

- Critical Event Monitoring
  - All nodes monitor their environment (temperature, amount of communication, available storage, etc)
  - Critical events are treated as
    - database updates (when all are interesting)
    - maximization problem (when the most critical is important)
- Control
  - All nodes forward commands
  - Commands are treated as database updates

Diffusion: some examples

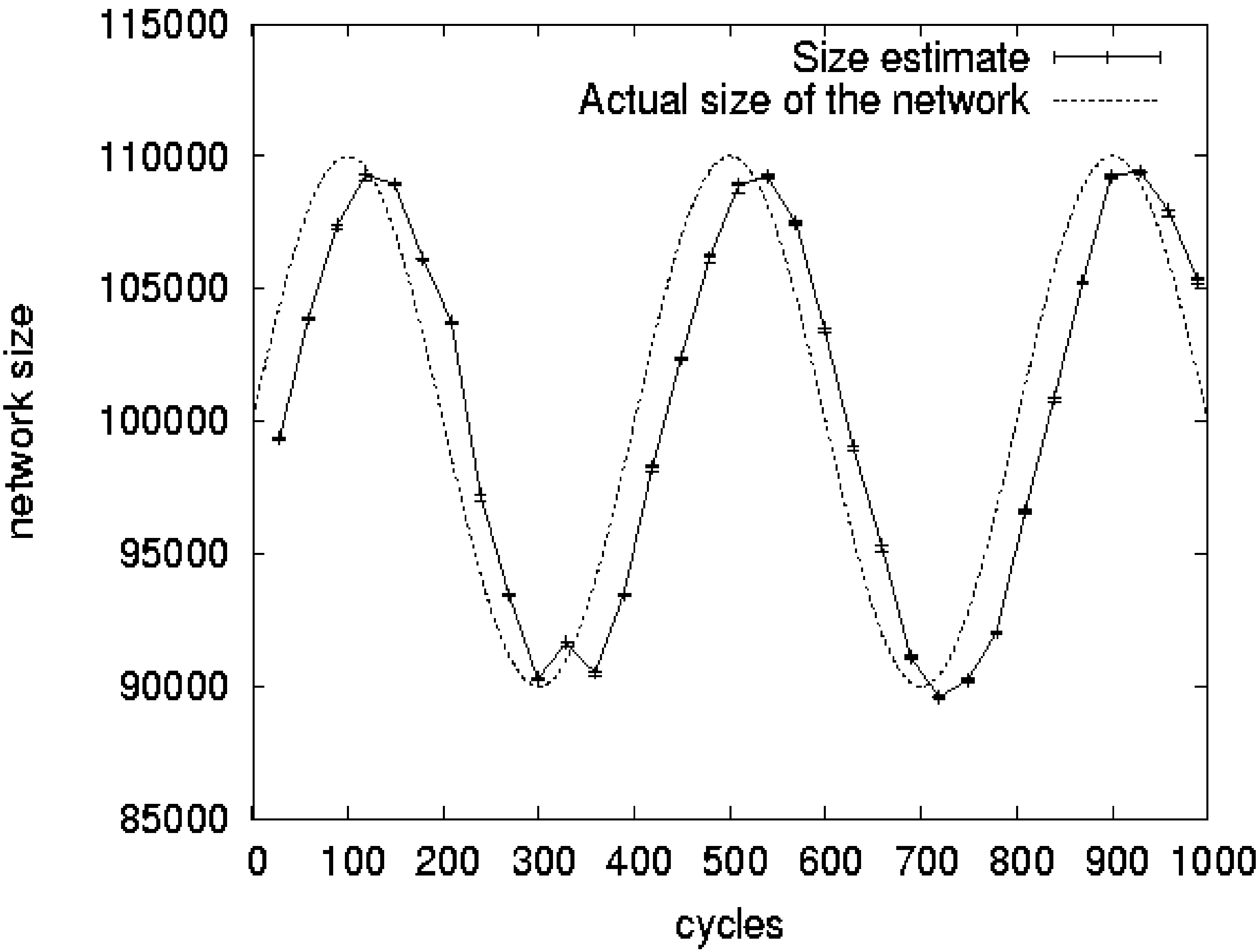
- Calculating variance
  - calculate the average of the squares of the values and use it along with the average to approximate variance
- Calculating sum of values
  - calculate the average and multiply it by the size of the network
- Calculating network size
  - we will focus on this example in the following

Diffusion: a case study

- Network size estimation
  - one node is assigned value 1, all the others are assigned 0
  - the average is calculated which is a=1/N.
  - the estimation of the size is 1/a
- Practical Extensions
  - restarting in regular intervals (epochs): to make the protocol adaptive
  - initial value assignment

Initial Value Assignment

- We need to make sure exactly one node starts with 1, the rest with 0. Solution: parallel execution of approximations
  - With a probability P each node can start an approximation process. P is a previous approximation of 1/N
  - The initiator node starts with 1 and assigns a unique ID to the approximation process. The other (passive) nodes will assume 0 initial value for all IDs not initiated by themselves.



115000
110000
105000
100000
95000
90000
85000
network size
0
100
200
300
400
500
600
700
800
900
1000
cycles
Size estimate
Actual size of the network