Constructing Overlay Networks through Gossip

Márk Jelasity
Università di Bologna

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

The Four Main Theses

1: Topology (network structure) is a key abstraction in distributed systems

2: Gossip protocols can maintain topologies in a scalable and robust manner

3: In particular, random topology and

4: structured topologies (ring, torus, binary tree).

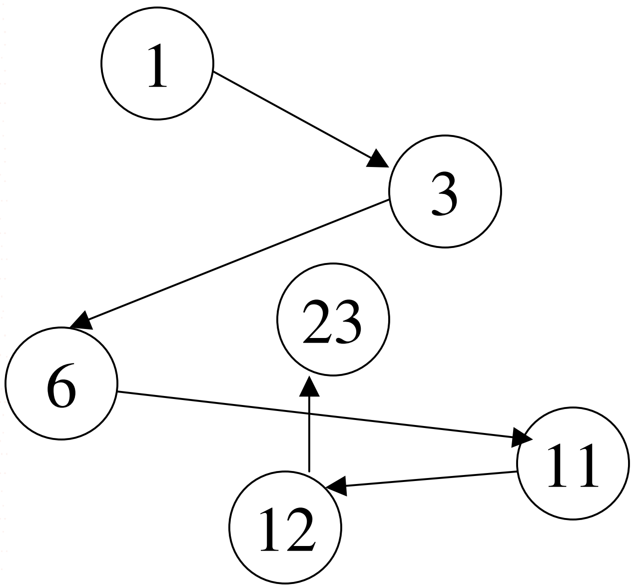
Topology as key abstraction (1)

- New field of compex networks: topology is key to explain robustness and function not only in computer science
  - spreading of epidemics and information (gossip): (human societies (sexual and other contact networks))
  - tolerance to random damage or attacks (food chains, chemical reactions (DNA), power grid)
  - search (strong and weak ties, 6 degrees of separation, small worlds)

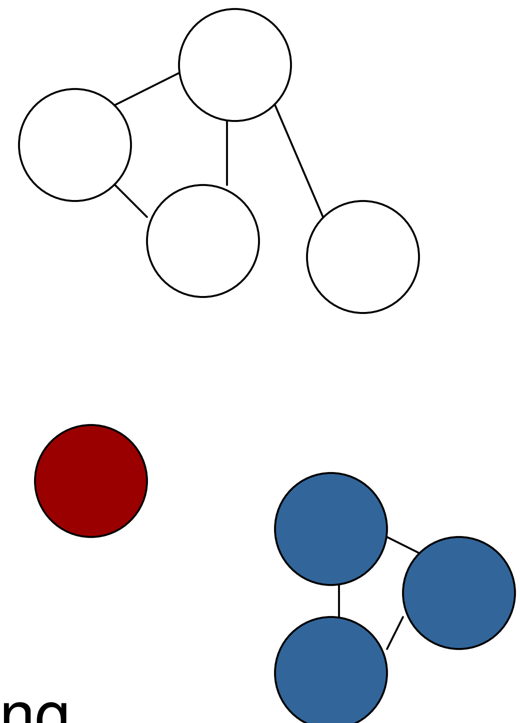
Topology as key abstraction (2) Communication Topology in CS

- Examples: WWW, Internet, P2P overlay networks (FastTrack, Gnutella, DHTs)
- Topology is responsible for
  - performance: function dependent: can be small diameter, randomness, hierarchy
  - robustness: proximity, redundancy, no hotspots

Topology as key abstraction (3) Other uses of Topology in CS



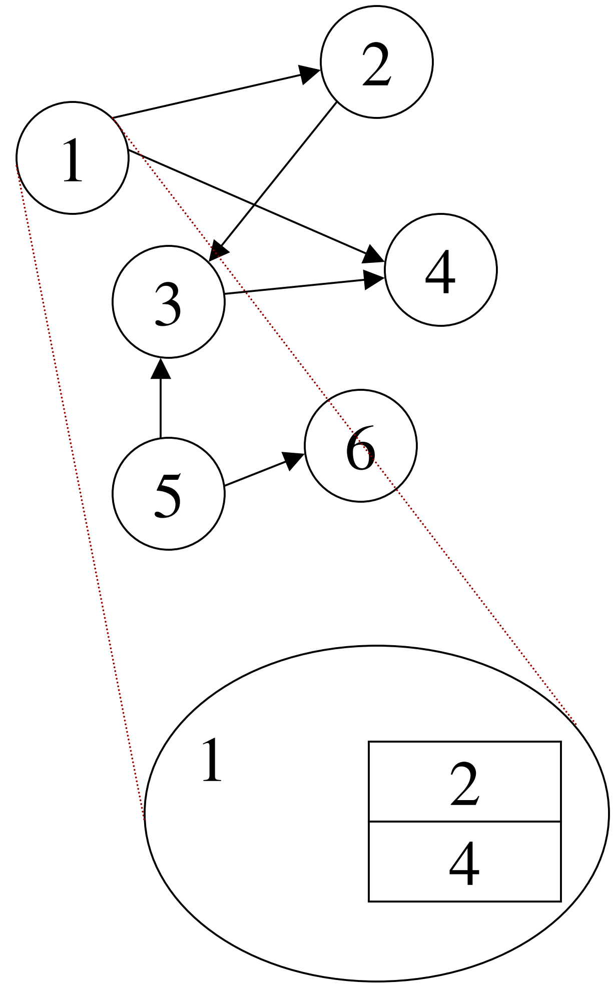
Sorting



Clustering

Topology as key abstraction (4) Using graph language

- Nodes: components of a large distributed system
- Neighbor: if x ``knows about'' y, y is a neighbor of x
- view: list of neighbors



Topology as key abstraction (5) Goal

- In some cases topology is given or is evolving out of control of any entity (WWW, Internet)
- In other cases it can be designed explicitly and built in a static manner (computer architectures, LAN)
- Dynamic and large distributed systems (overlay networks): efficient and robust protocols for topology maintenence are needed to support several different topologies (communication, sorting, clusterig).

Gossip protocols for topology management (1)

- push pull gossip scheme: information exchange is symmetric
- many applications
  - broadcast: state is info update
  - data aggregation: state is numeric value
  - topology: state is the view

active thread

```
do once in each T time units
at a random time
       p = selectPeer()
       send state to p
       receive state_p from p
       state = update(state_p)
```

passive thread

```
(p,state_p) = waitMessage()
send state to p
state = update(state_p)
```

Gossip protocols for topology management (2)

active thread

```
do once in each T time units at a random
time
        p = selectPeer()
        buffer = merge(view, {myDescriptor})
        send buffer to p
        receive view_p from p
        buffer = merge(view, view_p)
        view = selectView(buffer)
```

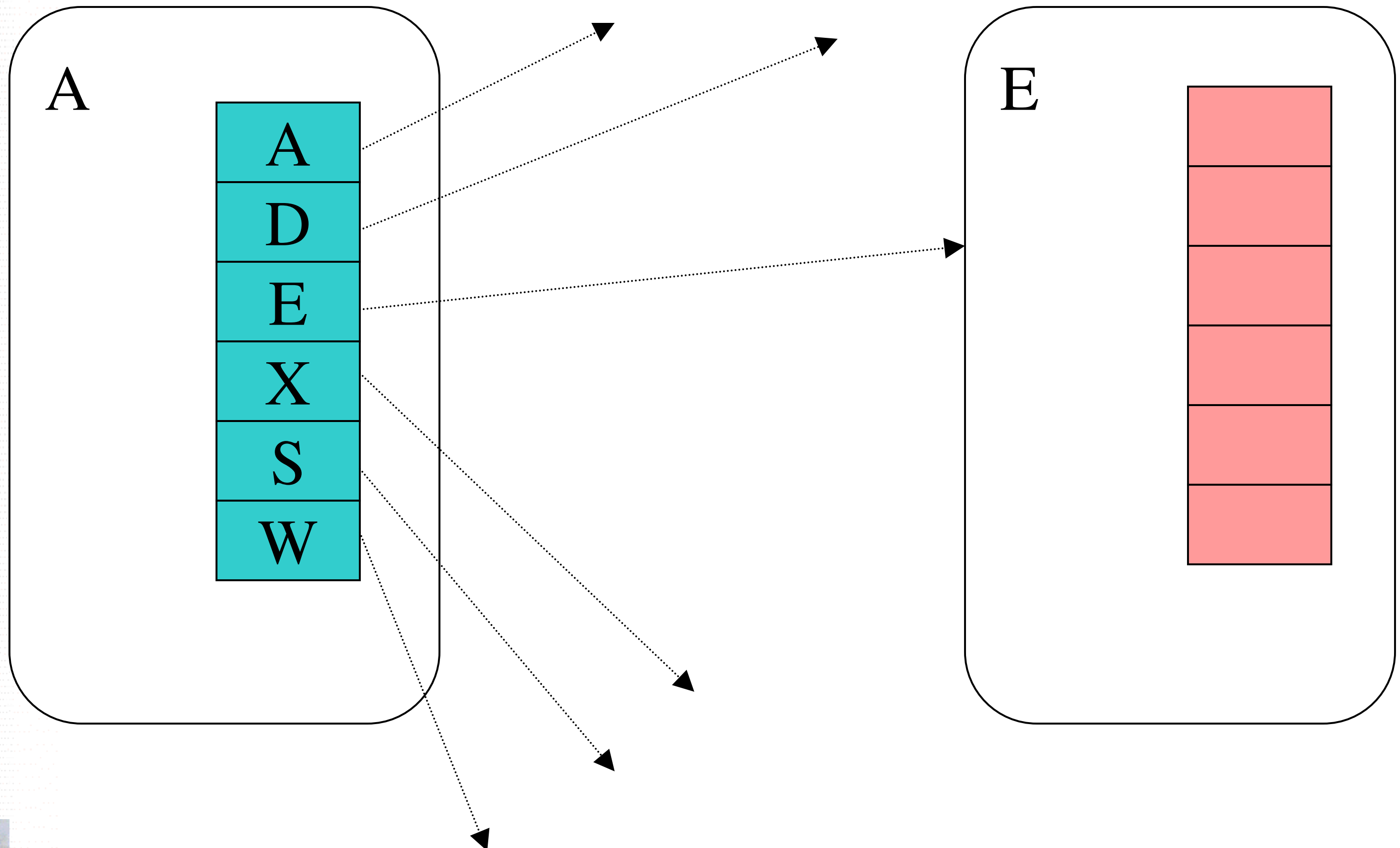
passive thread

```
(p,view_p) = waitMessage()
buffer = merge(view, {myDescriptor})
send view to p
buffer = merge(view, view_p)
view = selectView(buffer)
```

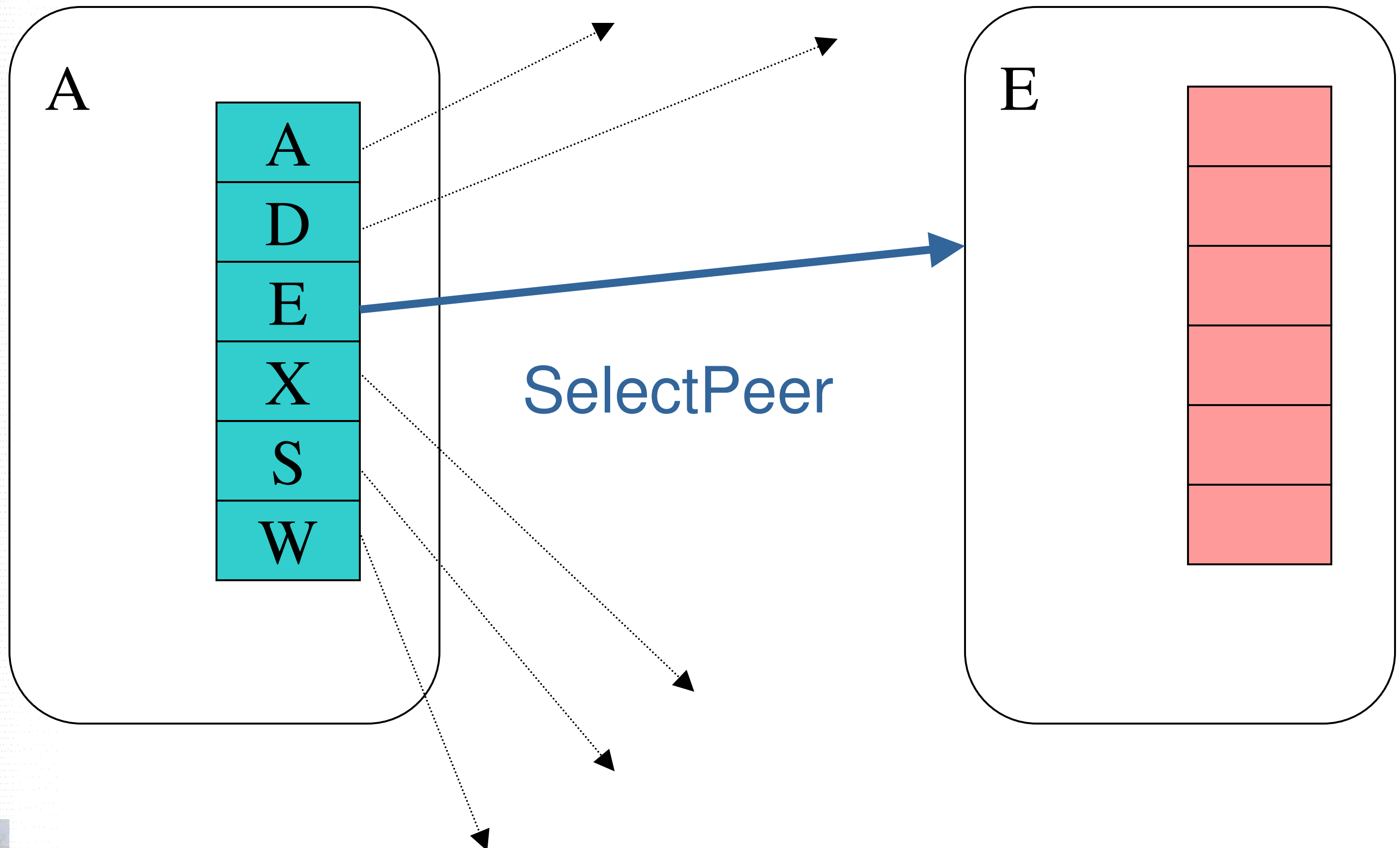
Gossip protocols for topology management (3)

- merge: implements set operation merge
- myDescriptor: contains address and application dependent information
- selectPeer: uses the actual view to select a peer to contact
- selectView: based on the information contained in the descriptors constructs the next view

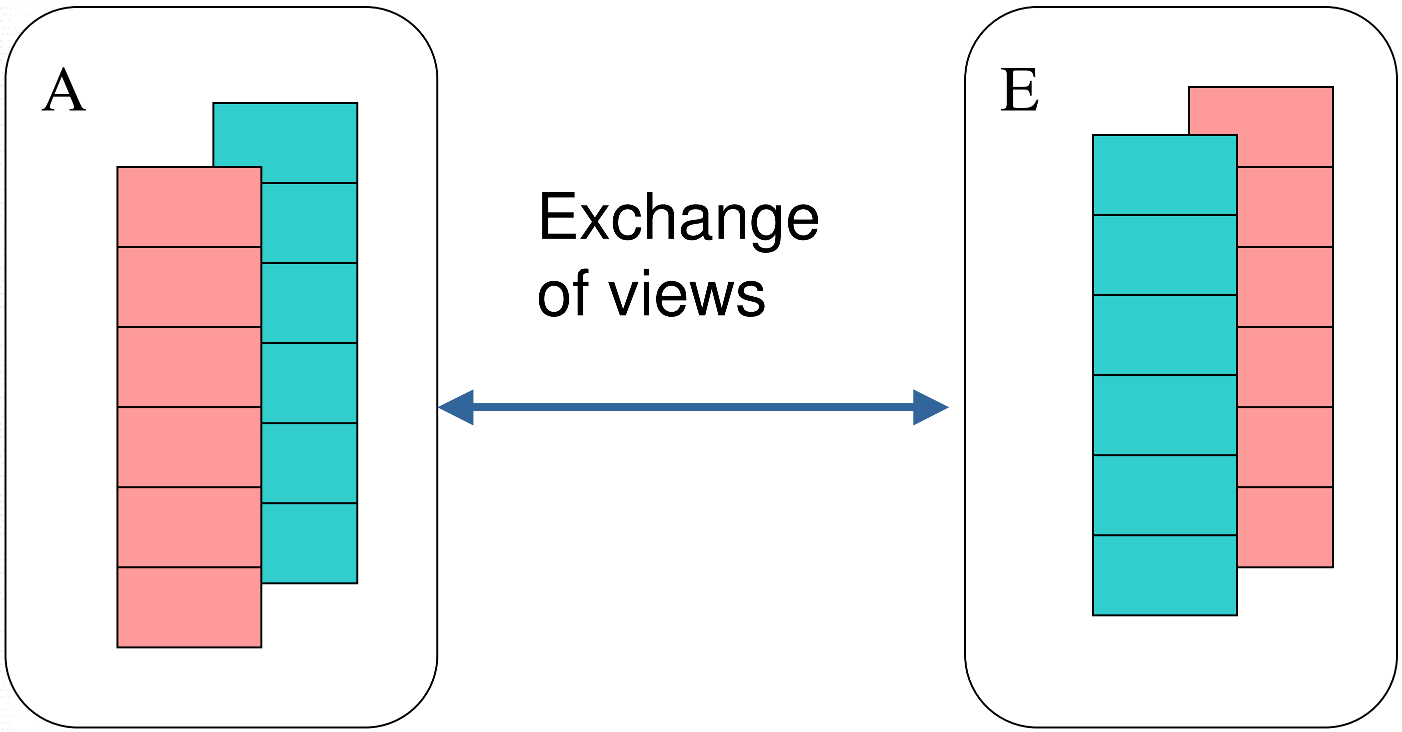
Gossip protocols for topology management (4)



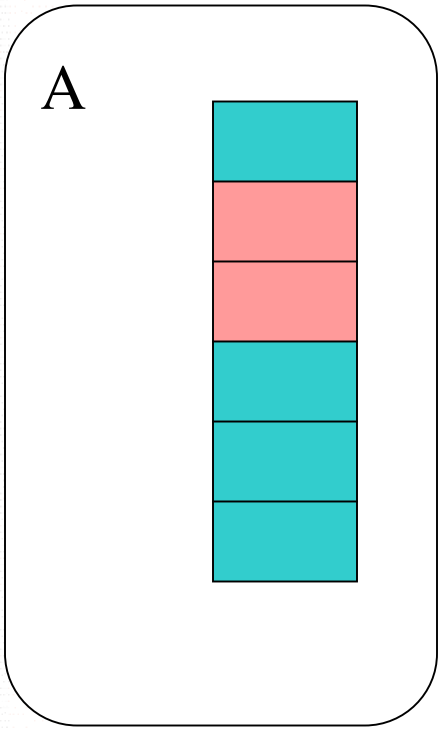
Gossip protocols for topology management (4)



Gossip protocols for topology management (4)

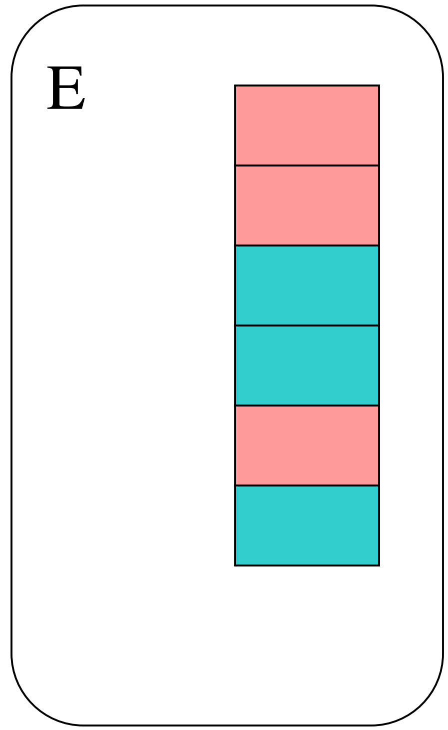


Gossip protocols for topology management (4)



Both sides apply selectView

thereby redefining topology



Gossip protocols for topology management (5)

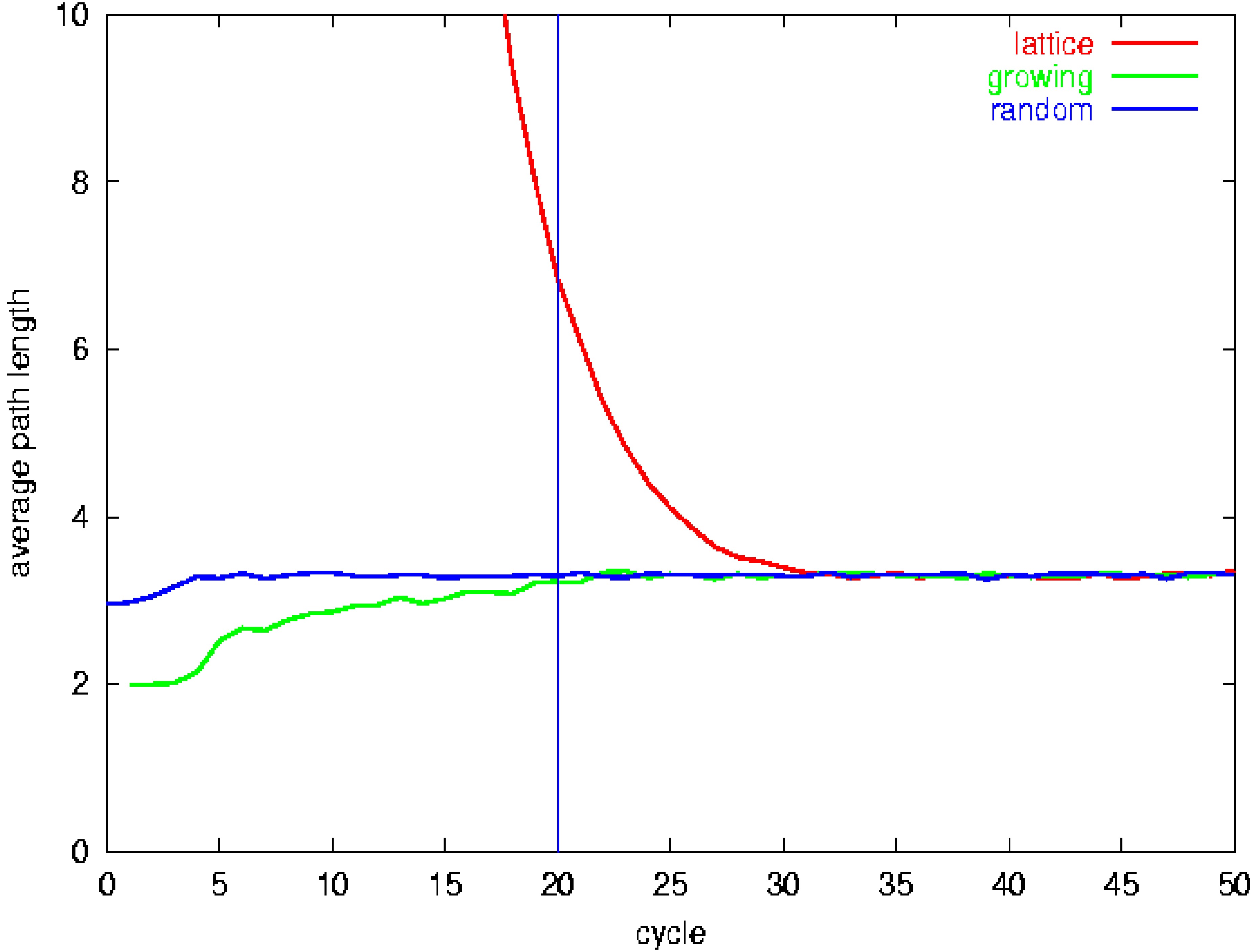
- Notion of cycle: any time interval when on average a node performs one view exchange: T/2 time units
- selectPeer and selectView are crucial: different implementations result in different topologies
  - newscast: random networks
  - T-Man: structured networks

Newscast: a gossip protocol for random topologies (1)

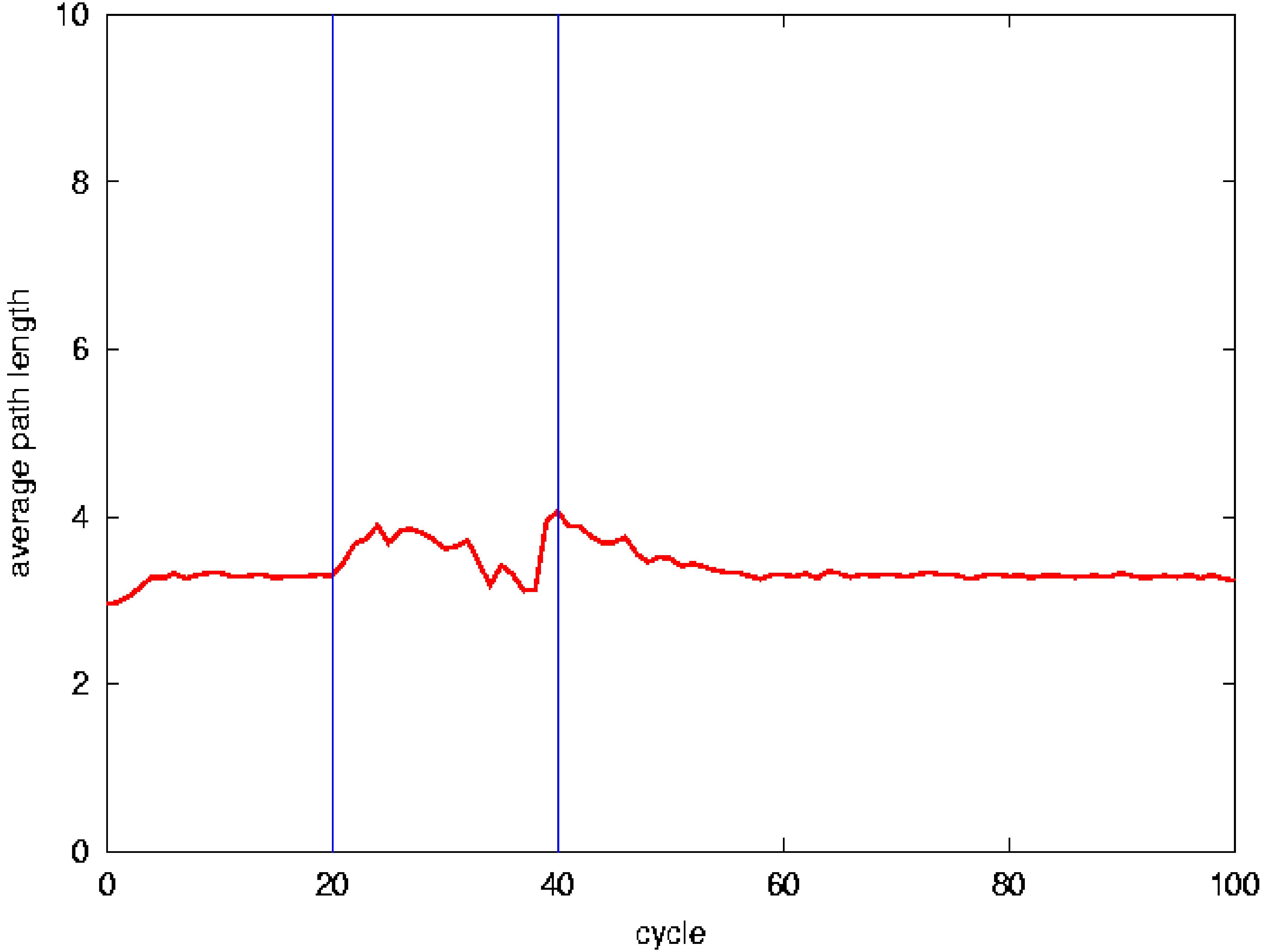
- descriptor: contains timestamp of creating the descriptor
- selectPeer: randomly selects a neighbor from the view
- selectView: fills the view with the freshest descriptors. New information gradually replaces old information: high robustness and adaptivity.

Newscast: a gossip protocol for random topologies (2)

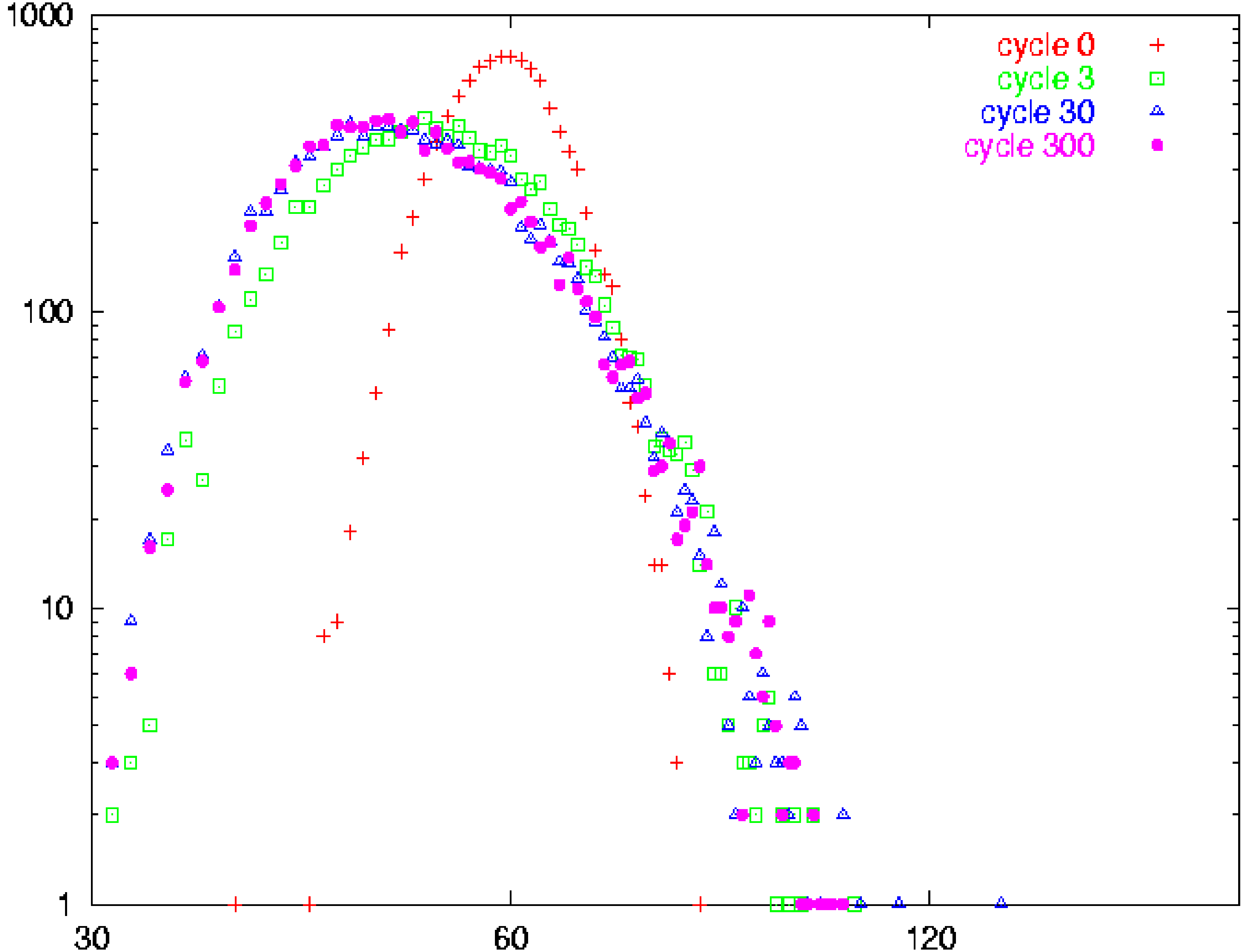
- simulation results: N=100 000, view size c=30
- scenarios:
  - growing: start with no nodes, add 5000 nodes each cycle (connecting them to first node only)
  - lattice: start with regular linear lattice (each node konnected to k nearest nodes)
  - random: start with random topology
  - dynamic: random, but between cycle 20 and 40 replace 10% of nodes each cycle



lattice
growing
random
average path length
cycle
0
2
4
6
8
10
0
5
10
15
20
25
30
35
40
45
50



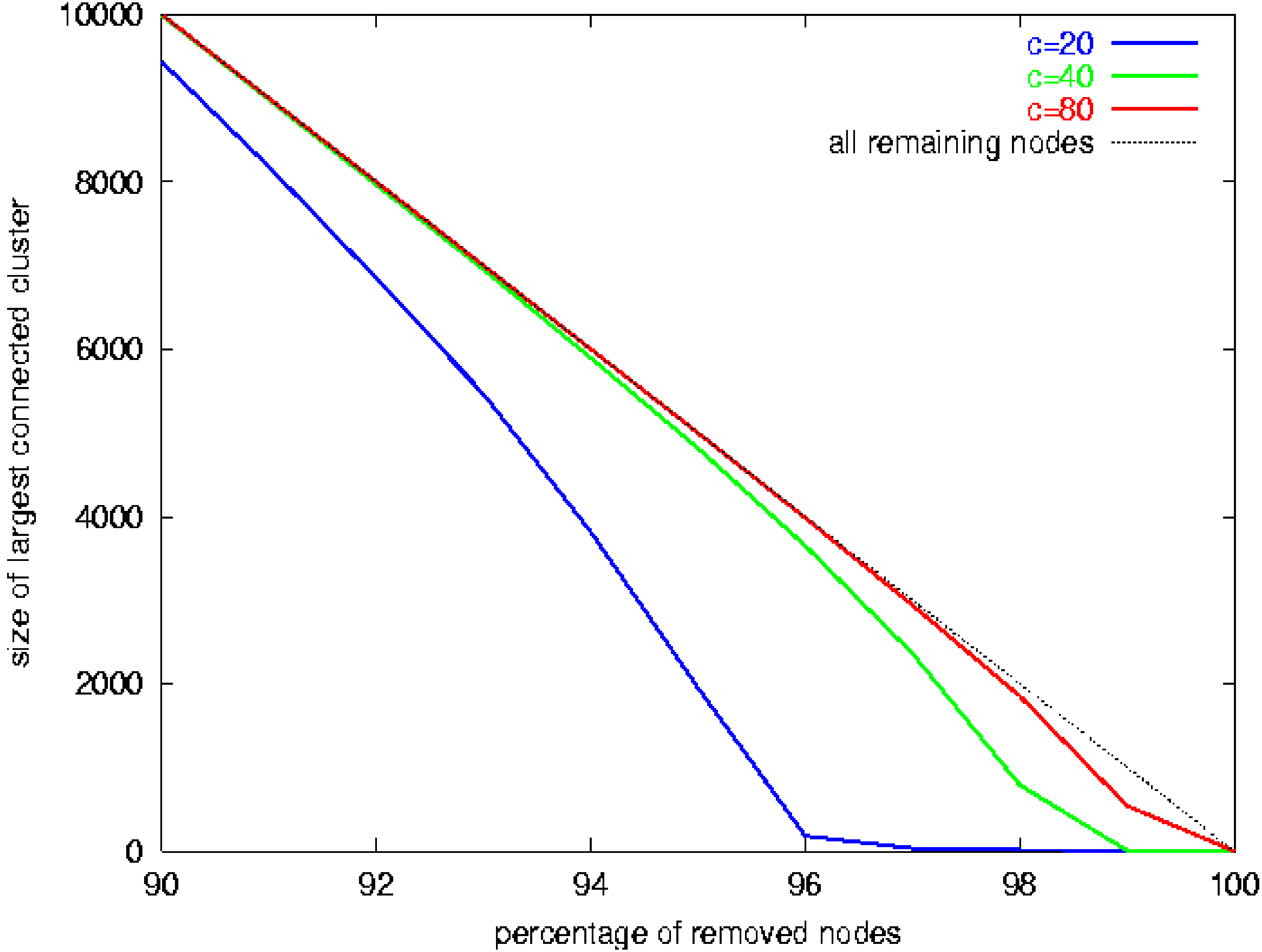
average path length
cycle
0
2
4
6
8
10
0
20
40
60
80
100



1000
100
10
1
30
60
120
cycle 0
cycle 3
cycle 30
cycle 300

Newscast: a gossip protocol for random topologies (3)

- robustness to failure simulation results: N=100 000, view size c=20,40 and 80
- allowed the newscast topology to converge
- size of largest connected cluster after removing a given proportion of random nodes


c=20
c=40
c=80
all remaining nodes
size of largest connected cluster
percentage of removed nodes
0
2000
4000
6000
8000
10000
90
92
94
96
98
100

Newscast: a gossip protocol for random topologies (4)

- newscast generates a closely random topology irrespective of starting conditions
- the convergence is fast, a few cycles
- scalable (though not demonstrated here)
- robust to node dynamism (churn)
- robust to node failure
- a reliable source of random nodes from the whole system, a service many applications need, including T-Man

T-Man: a gossip protocol for structured topologies (1)

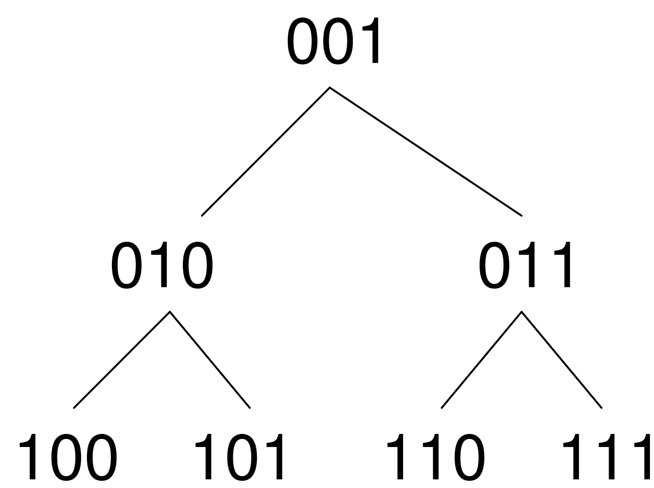
- Ranking function: $R(y,\{x_1,\ldots,x_m\})$ ranks a set of nodes $\{x_1,\ldots,x_m\}$ with respect to a base node y
- For a fixed base node y and node set $S=\{x_1,\ldots,x_m\}$ the ranking function defines an ordering relation ($<=$) over S
- if d is a distance function over the set of all nodes $\{x_1,\ldots,x_N\}$ then it can be used to define a ranking function (backwards not true)

T-Man: a gossip protocol for structured topologies (2)

- Example 1: ring and line Let the nodes be real numbers. Let the ranking function be defined by the distance d(a,b)=|a-b|. For the ring, apply periodic boundary conditions, assuming nodes are from [0,N].
- Example 2: mesh and torus Let the nodes be two dimensional real vectors. Similarly to the ring, let the Manhattan distance define the topology

T-Man: a gossip protocol for structured topologies (3)

- Example 3: binary tree
  Let the nodes be binary strings of length m. Let the ranking function be defined by the distance given by the hop count in the binary undirected rooted tree as illustrated



T-Man: a gossip protocol for structured topologies (4)

- Example 4: sorting Let <= be a total ordering over the nodes. Let the ranking function apply a distance function consistent with <= separately to those < and > than the base node, and merge the ranked two subsets
- For example R(10,{1,2,4,100, 300}) could return (4,100,2,300,1). No distance function over the set of nodes generates this ranking function!

T-Man: a gossip protocol for strucured topologies (5)

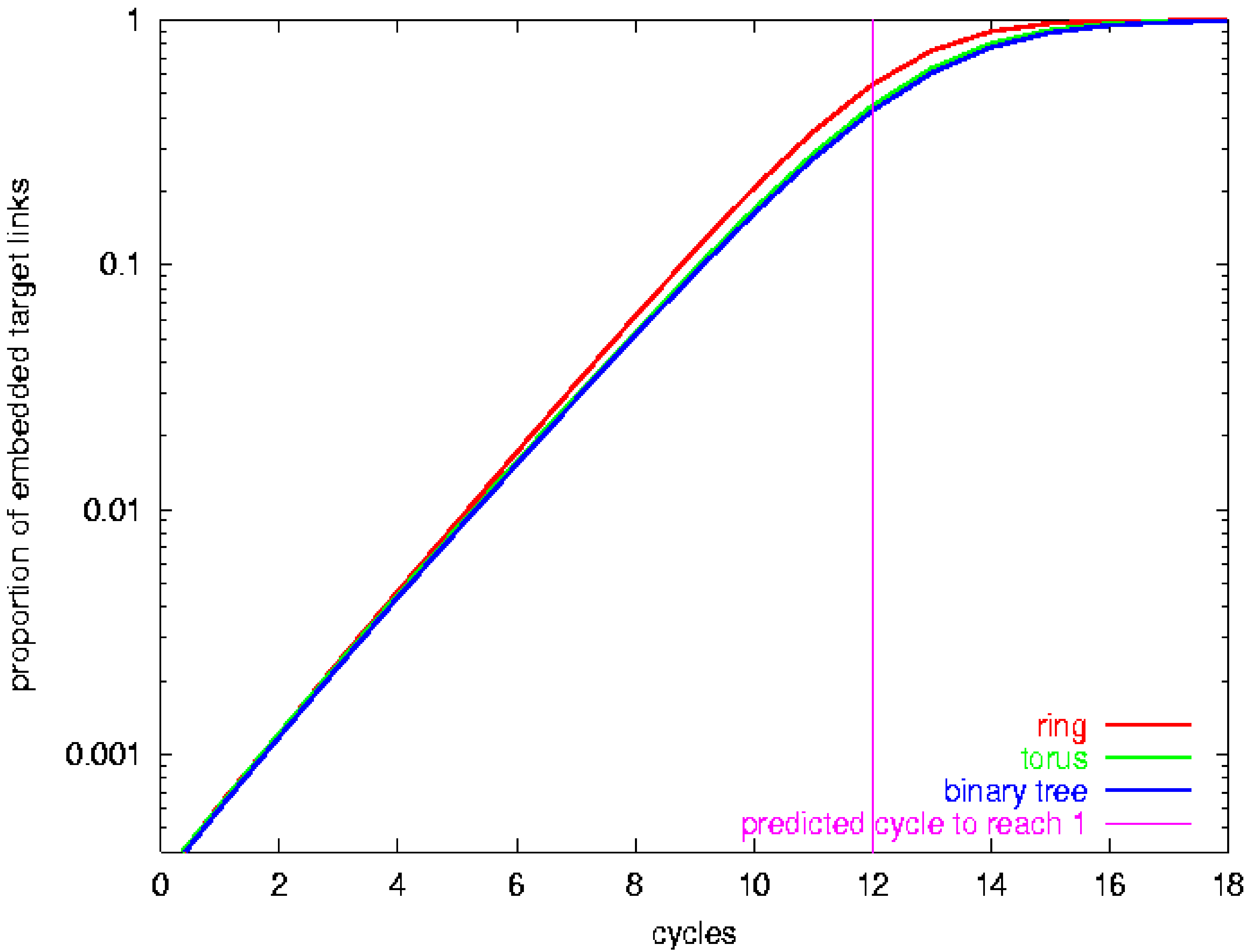
- Problem 1(construction) Construct the view of node x such that they contain the highest ranked elements when ranking is applied over the whole node set with x as base node
- Problem 2 (embedding) Construct the view of node x such that it contains at least k elements which have a highest rank when ranking over the whole nodes set with x as base node

T-Man: a gossip protocol for structured topologies (6)

- descriptor: contains profile of the node (real number, 2d vector, etc)
- selectPeer: Ranks view and selects a neighbor from the first half according to ranking
- selectView: fills the view with the lowest rank descriptors
- View initializtion: random initial nodes are desirable: newscast is used

T-Man: a gossip protocol for structured topologies (7)

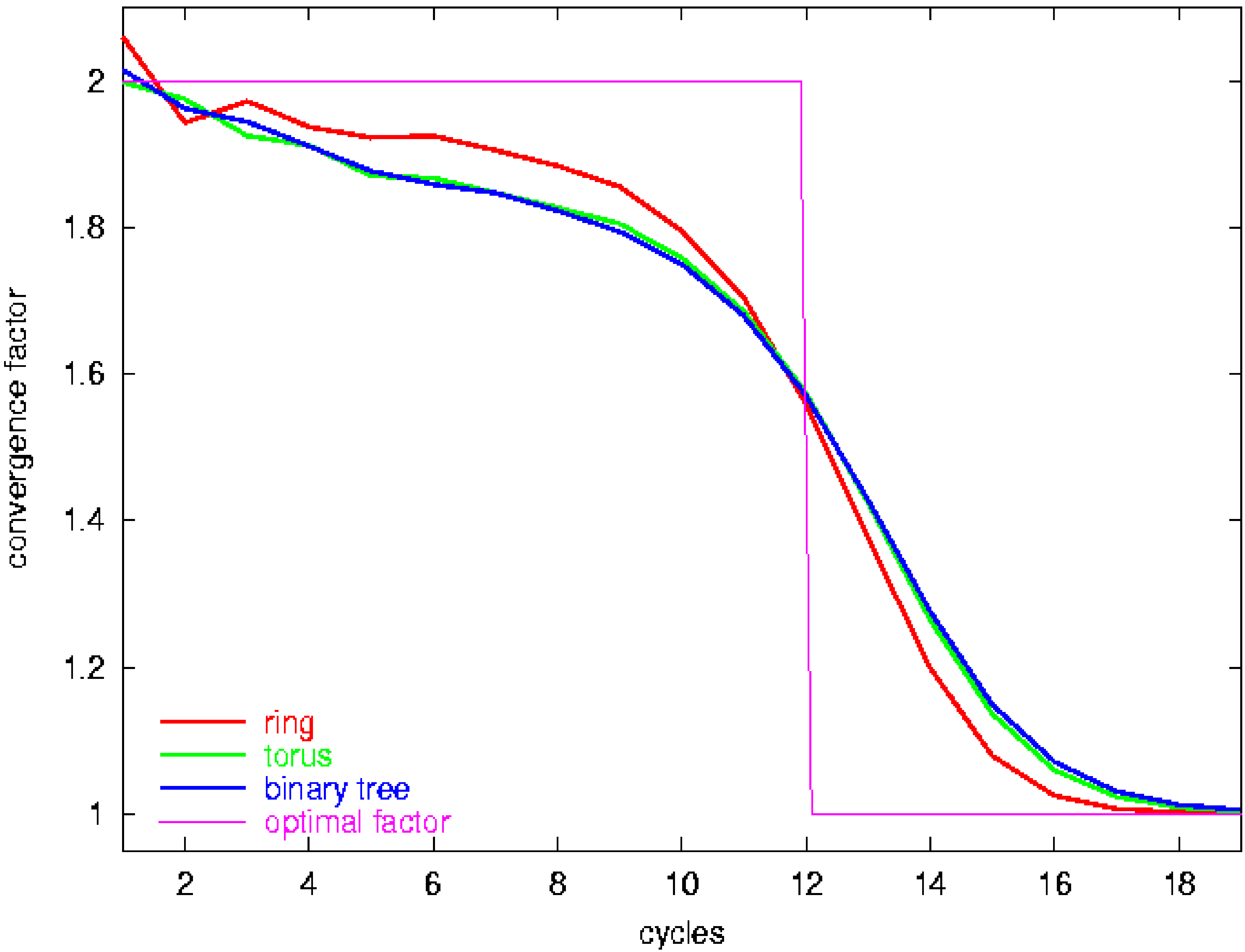
- irregular initialization (nodes are random, with possible gaps in the distribution): clustering and sorting can be achieved.
  - using the line ranking function: clustering
  - using the ordering ranking function: sorting
- regular initialization (nodes are 1,…,N for ring, similarly for other topologies): rest of the results about this case.
  - $N=2^{14},2^{17},2^{20}$; $c=20,40,80$; ranking function is ring, 2-d torus and binary tree.



1
0.1
0.01
0.001
0
2
4
6
8
10
12
14
16
18
cycles
proportion of embedded target links
ring
torus
binary tree
predicted cycle to reach 1

T-Man: a gossip protocol for structured topologies (8)

- Observed exponential behavior can be explained using an intuitive approximate model:
  - 1) the view after the first contact is the first (lowest ranking) half of a random sample twice as large as the view
  - 2) the extension of this idea is that after the ith contact the view is the first c elements of $c2^i$ random samples
- this simple model predicts exponentially increasing probability of embedding good links, in particular, it doubles in each cycle (after each contact).

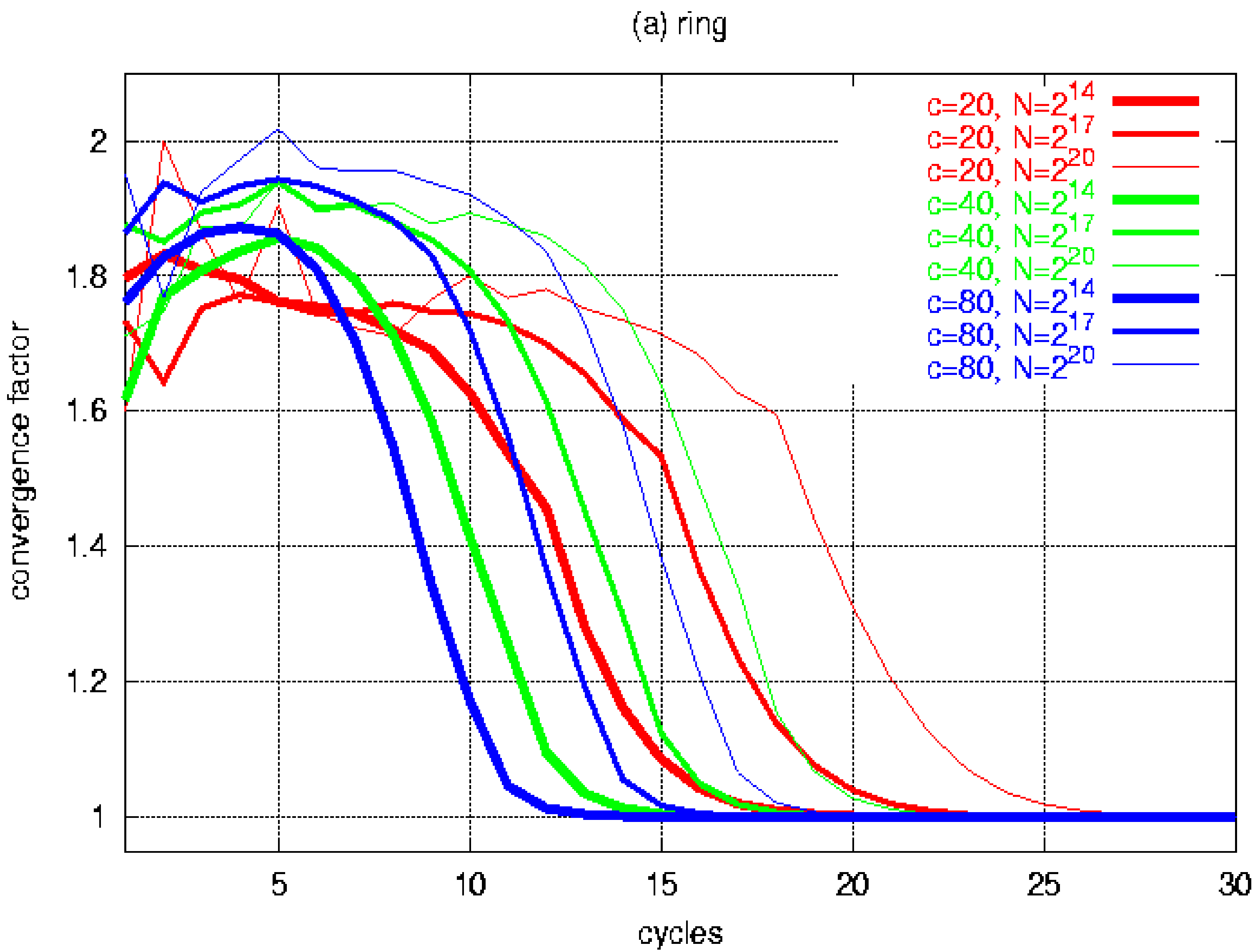


2
1.8
1.6
1.4
1.2
1
2
4
6
8
10
12
14
16
18
convergence factor
cycles
ring
torus
binary tree
optimal factor

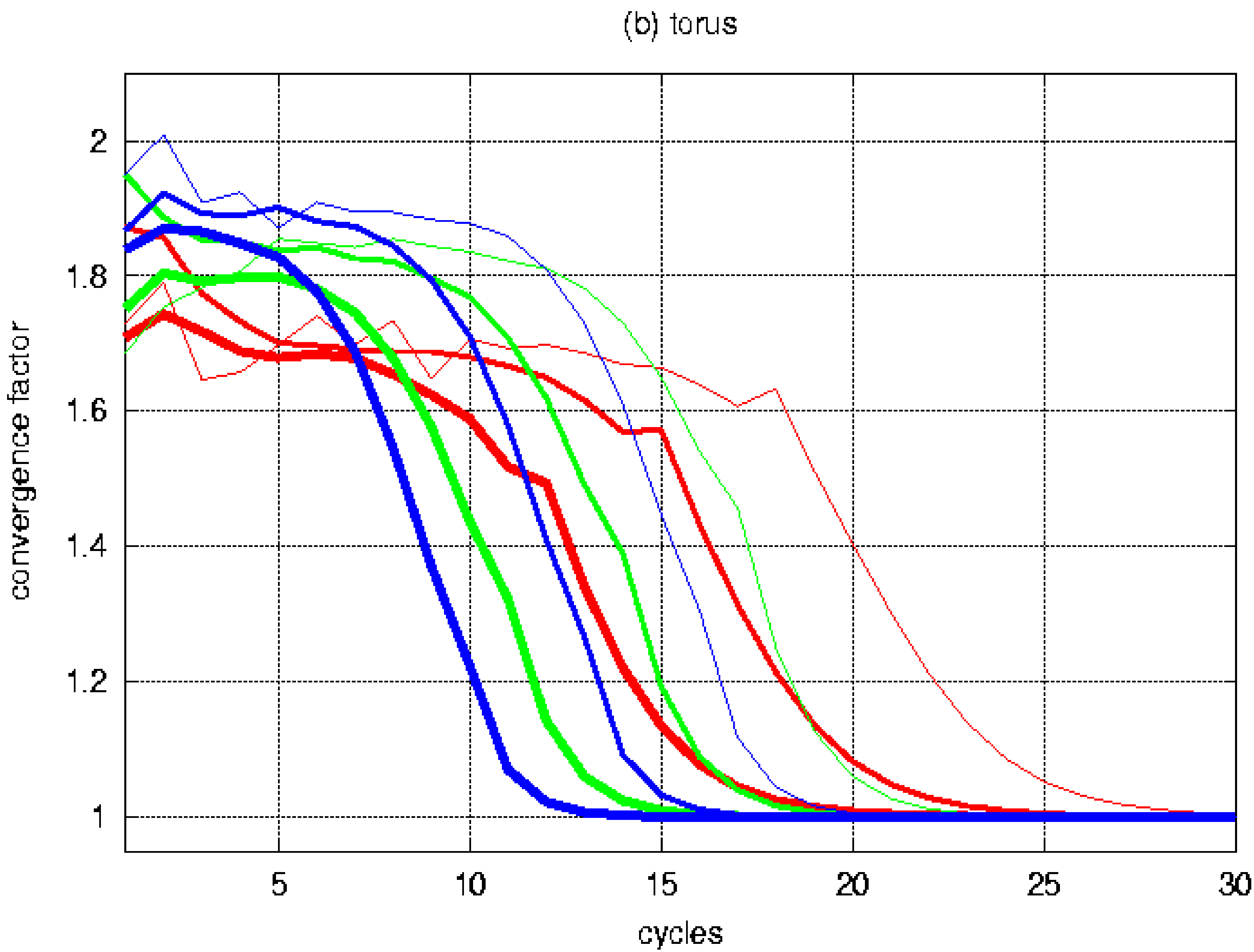
T-Man: a gossip protocol for structured topologies (9)

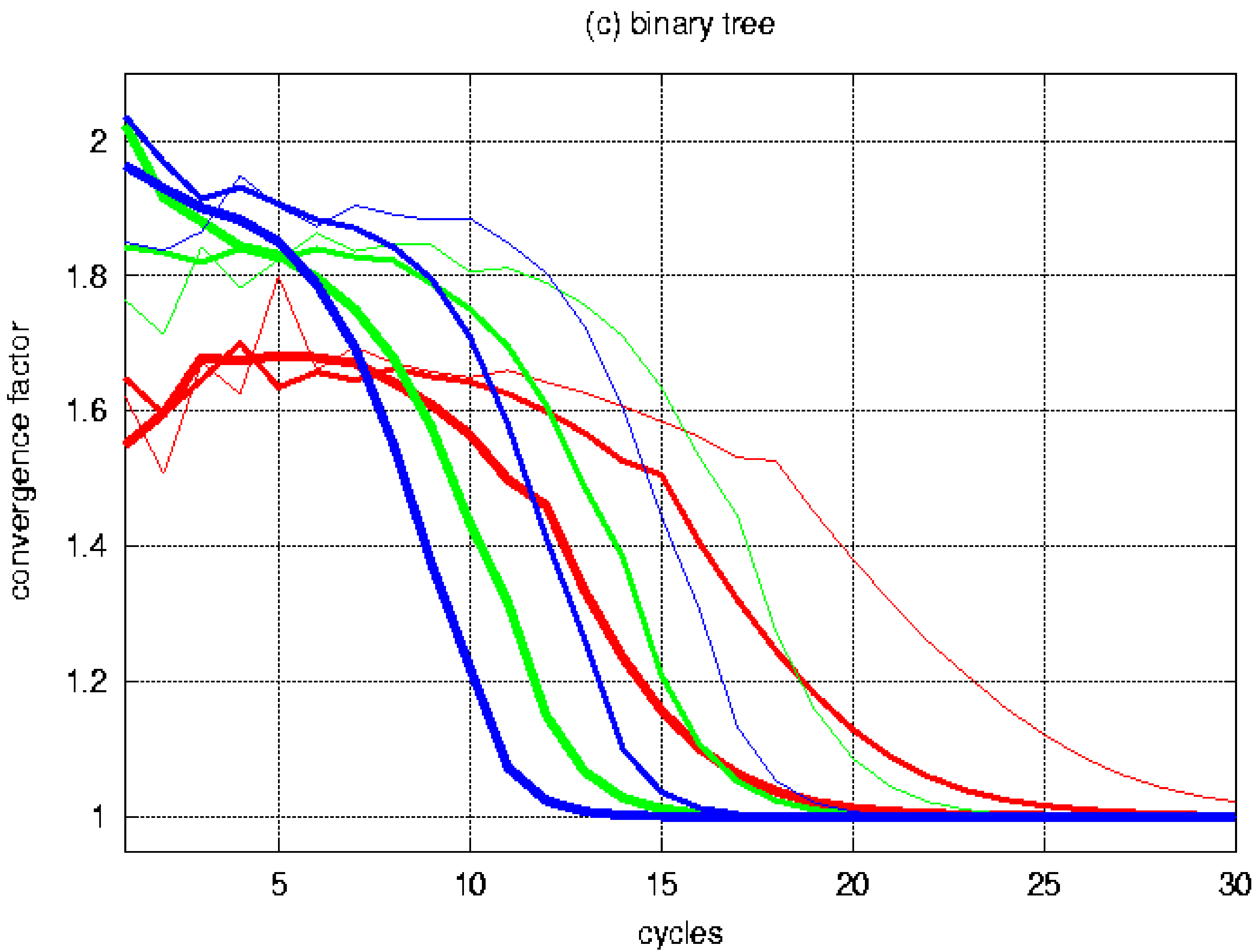
We predict the time of embedding by calculating the time for reaching probability one for the inclusion of the target links, according to the simple model which gives the formula $i < \log_2(N-1) - \log_2 c$. This yealds the following predictions:

	C=20	C=40	C=80
$N=2^{14}$	10	9	8
$N=2^{17}$	13	12	11
$N=2^{20}$	16	15	14



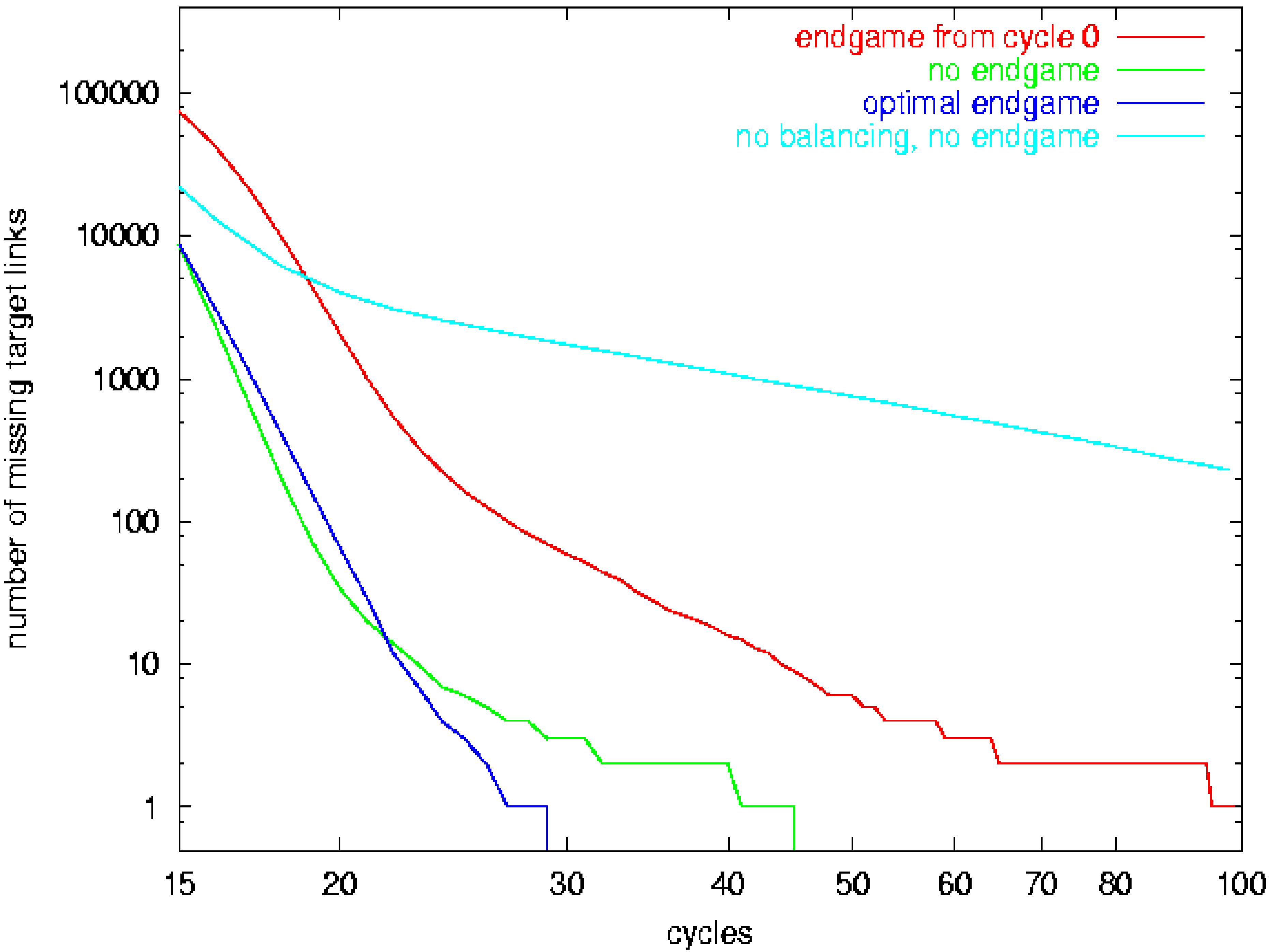
(a) ring
c=20, N=2^14
c=20, N=2^17
c=20, N=2^20
c=40, N=2^14
c=40, N=2^17
c=40, N=2^20
c=80, N=2^14
c=80, N=2^17
c=80, N=2^20
convergence factor
cycles
2
1.8
1.6
1.4
1.2
1
5
10
15
20
25
30


(b) torus
convergence factor
cycles
2
1.8
1.6
1.4
1.2
1
5
10
15
20
25
30

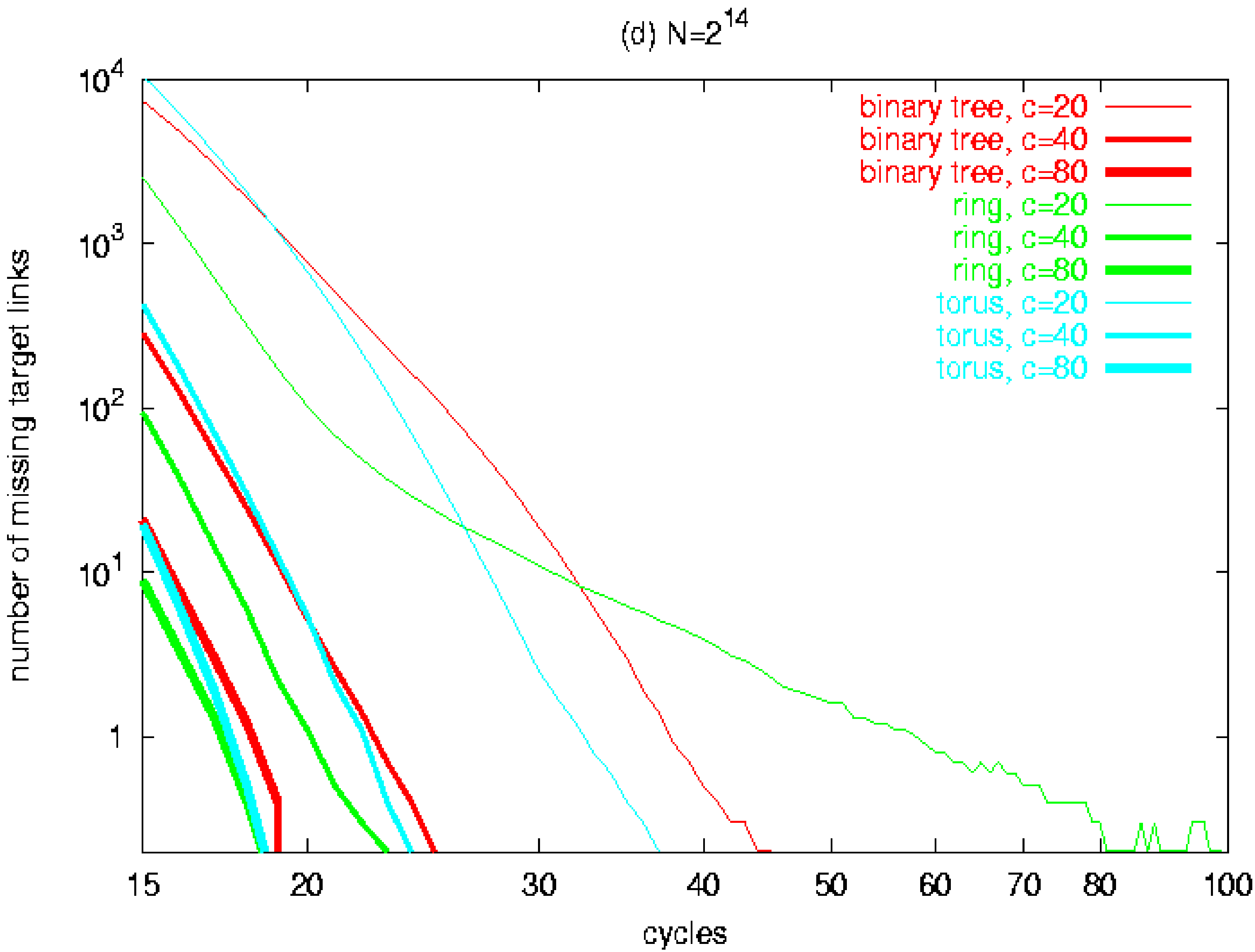

(c) binary tree
convergence factor
cycles
2
1.8
1.6
1.4
1.2
1
5
10
15
20
25
30

T-Man: a gossip protocol for strucured topologies (10)

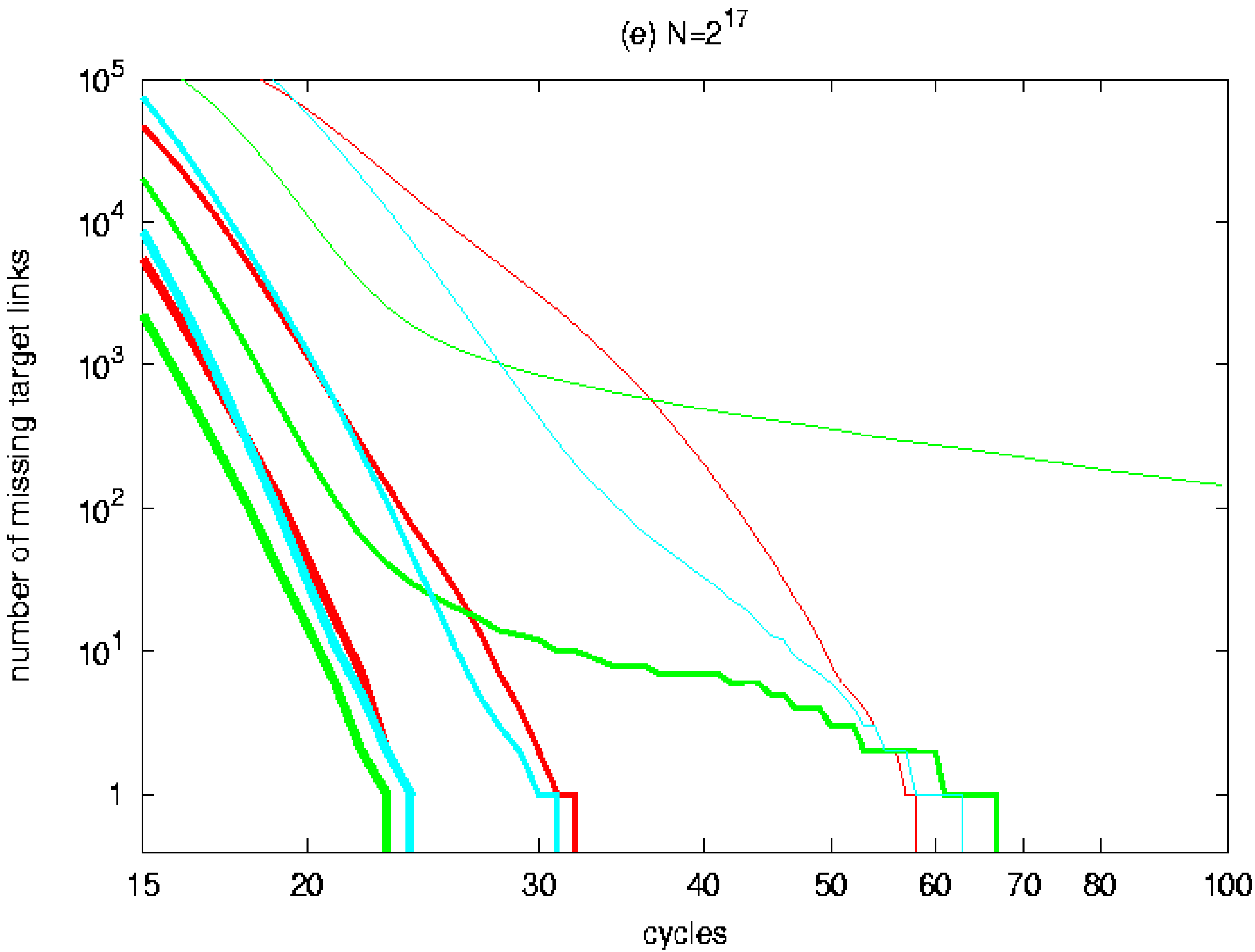
- End phase: due to deviations from the approximate model, a few nodes have to find their place, using the already converged structure, to reach perfection
- To improve end phase we apply
  - contact balancing: in cycle i find peer that communicated less than i times and reject connections if communicated i times already
  - endgame: more aggressive peer selection: select the closest peer (instead of random from first half)



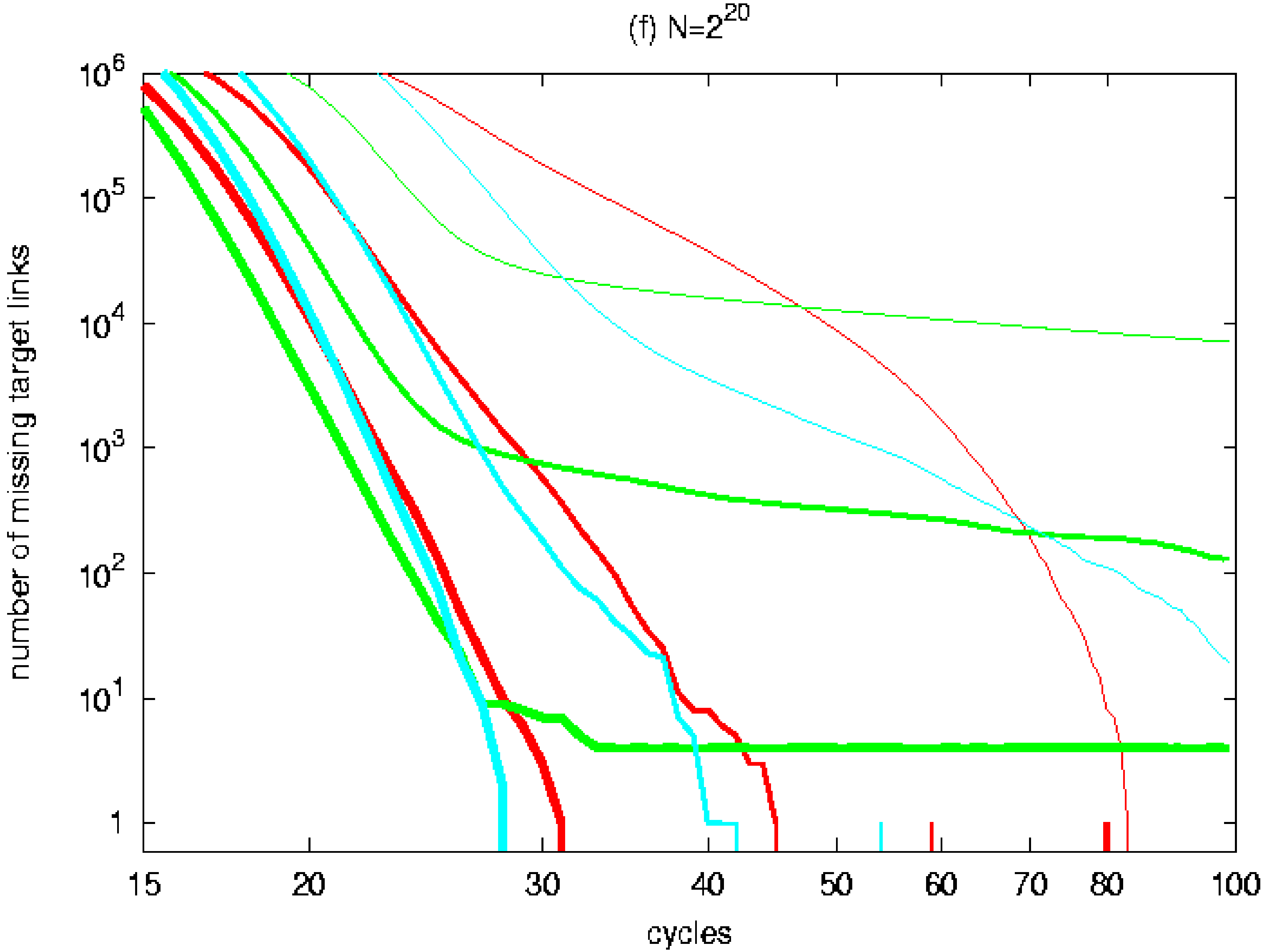
endgame from cycle 0
no endgame
optimal endgame
no balancing, no endgame
100000
10000
1000
100
10
1
15
20
30
40
50
60
70
80
100
cycles
number of missing target links



(d) N=2^14
number of missing target links
cycles
10^4
10^3
10^2
10^1
1
15
20
30
40
50
60
70
80
100
binary tree, c=20
binary tree, c=40
binary tree, c=80
ring, c=20
ring, c=40
ring, c=80
torus, c=20
torus, c=40
torus, c=80



(e) N=2^17
number of missing target links
10^5
10^4
10^3
10^2
10^1
1
15
20
30
40
50
60
70
80
100
cycles



(f) N=2^20
number of missing target links
10^6
10^5
10^4
10^3
10^2
10^1
1
15
20
30
40
50
60
70
80
100
cycles

T-Man: a gossip protocol for structured topologies (11)

- T-Man generates a wide range of topologies
- the convergence is fast
  - logarithmic in the number of nodes,
  - independently of the topology
- not only approximate, but perfect embedding can be achieved
- applications include communiction topology, sorting and clustering

Future Work

- fault tolerance: how T-Man reacts to node and link failures
- adaptivity: how T-Man adapts to a continuously changing node set
- our results with other gossip protocols suggest solutions
  - restarting
  - concurrently running protocol instances and usage of past results when restarting
- exploring applications

Contatct

For more info check out my homepage:

http://www.cs.unibo.it/~jelasity/