Lecturer: Dr. Michael J. May
Kinneret College

Application Level Multicasting and Epidemics

24 November 2024
Lecture 4

Slide Credits: Maarten van Steen

Topics for Today

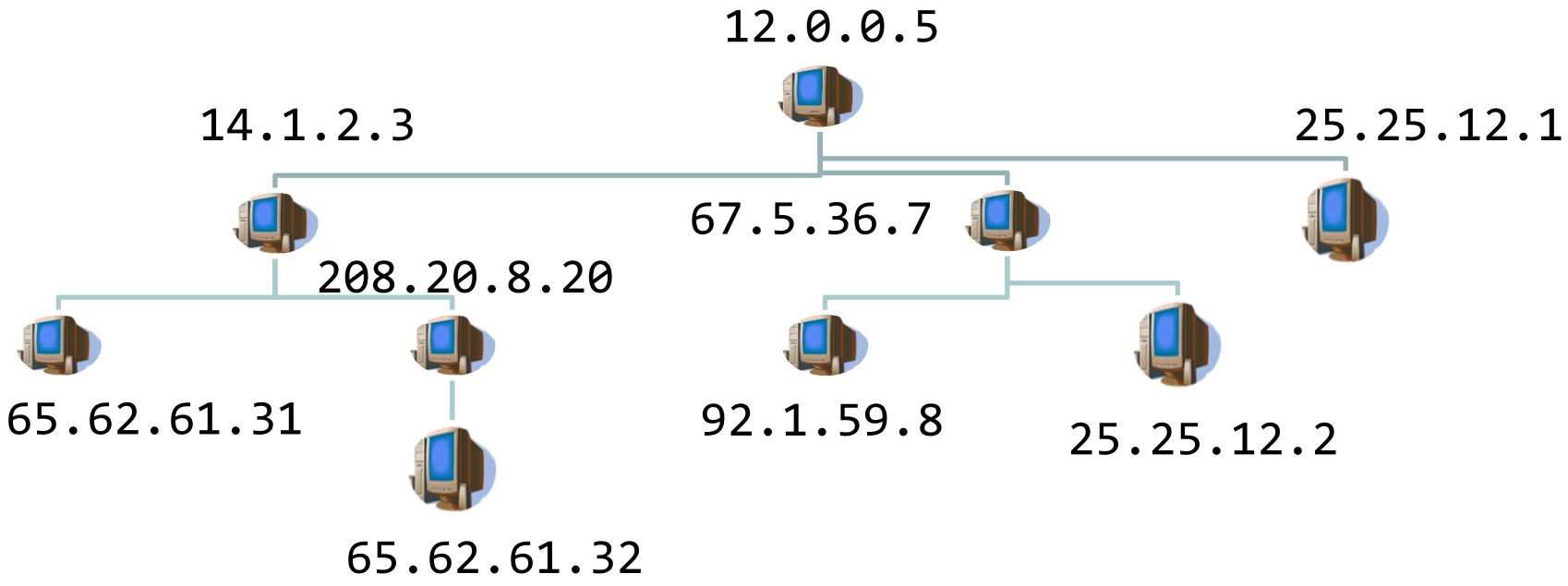
- Application Level Multicasting
- Epidemic Algorithms
- Cyber analysis: C&C

Source: TvS 5.1 – 5.4

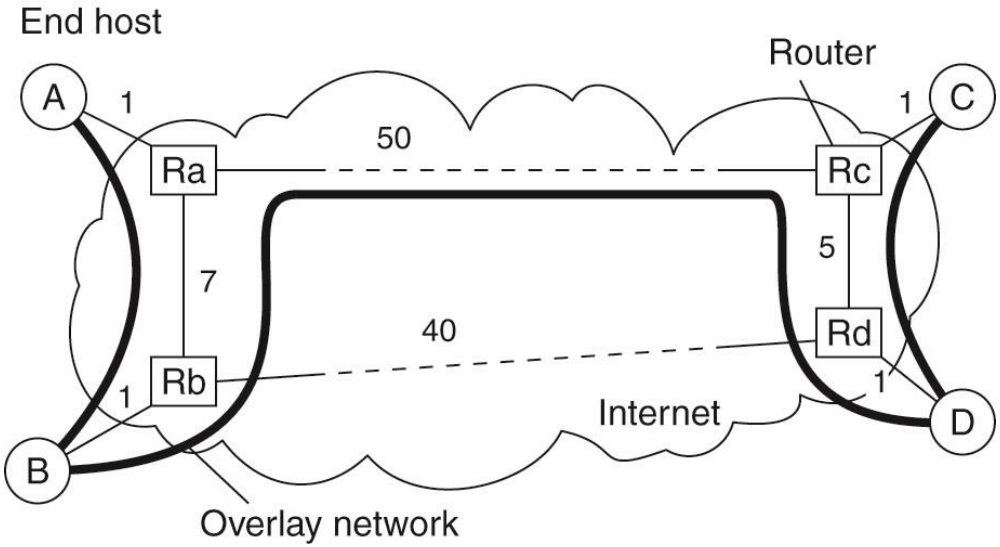
Application-Level Multicasting

Essence: Organize nodes of a distributed system into an **overlay network** and use that network to disseminate data.

Discern: Logical neighbors and physical neighbors



ALM: Some costs



- **Link stress**: How often does an ALM message cross the same physical link? **Example**: message from A to D needs to cross $\langle Ra, Rb \rangle$ twice.

- **Stretch**: Ratio in delay between ALM-level path and network-level path. **Example**: messages B to C follow path of length 71 at ALM, but 47 at network level → stretch = 71/47.

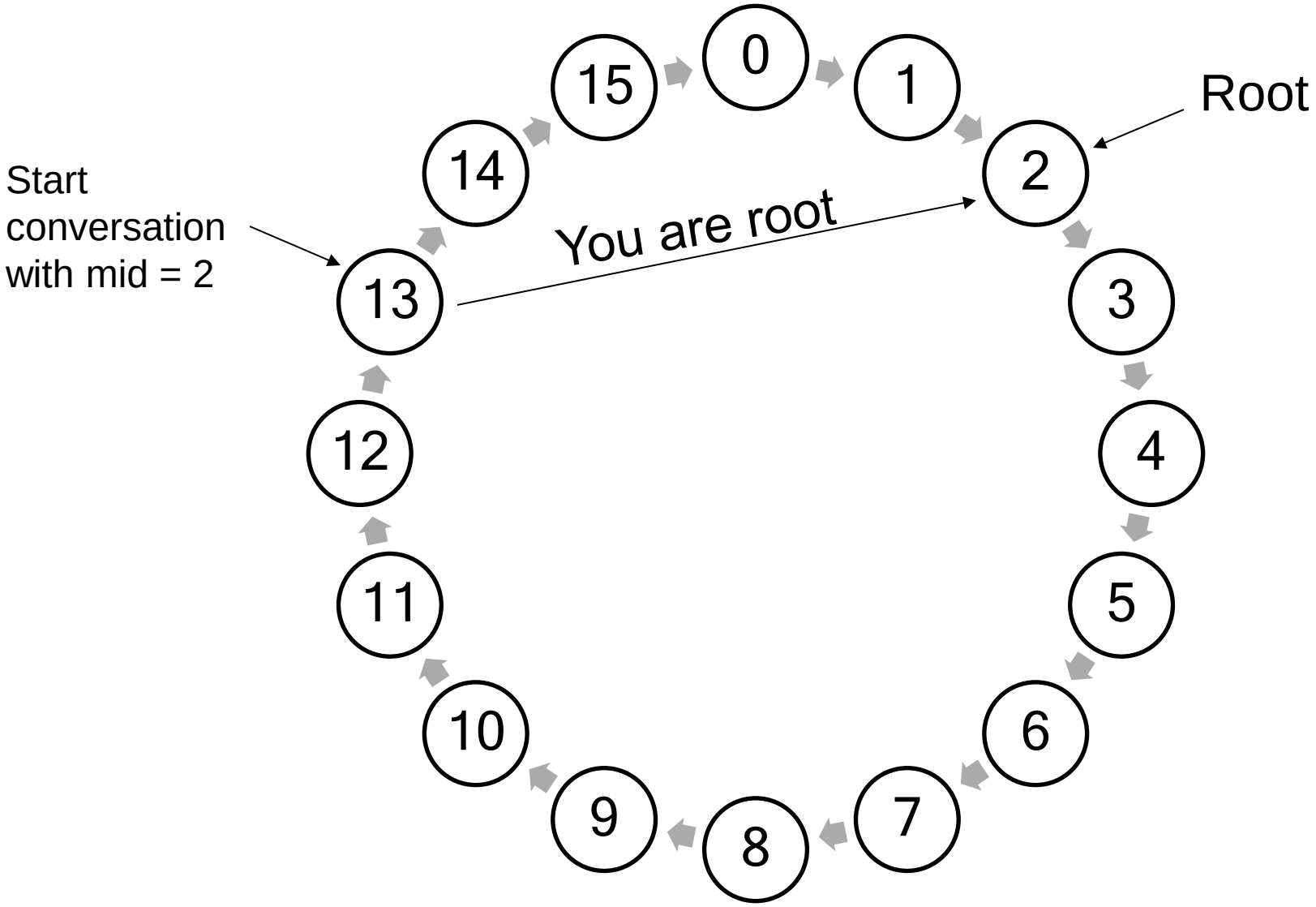
Implementing Multicast

Example 1: A multicast dissemination tree

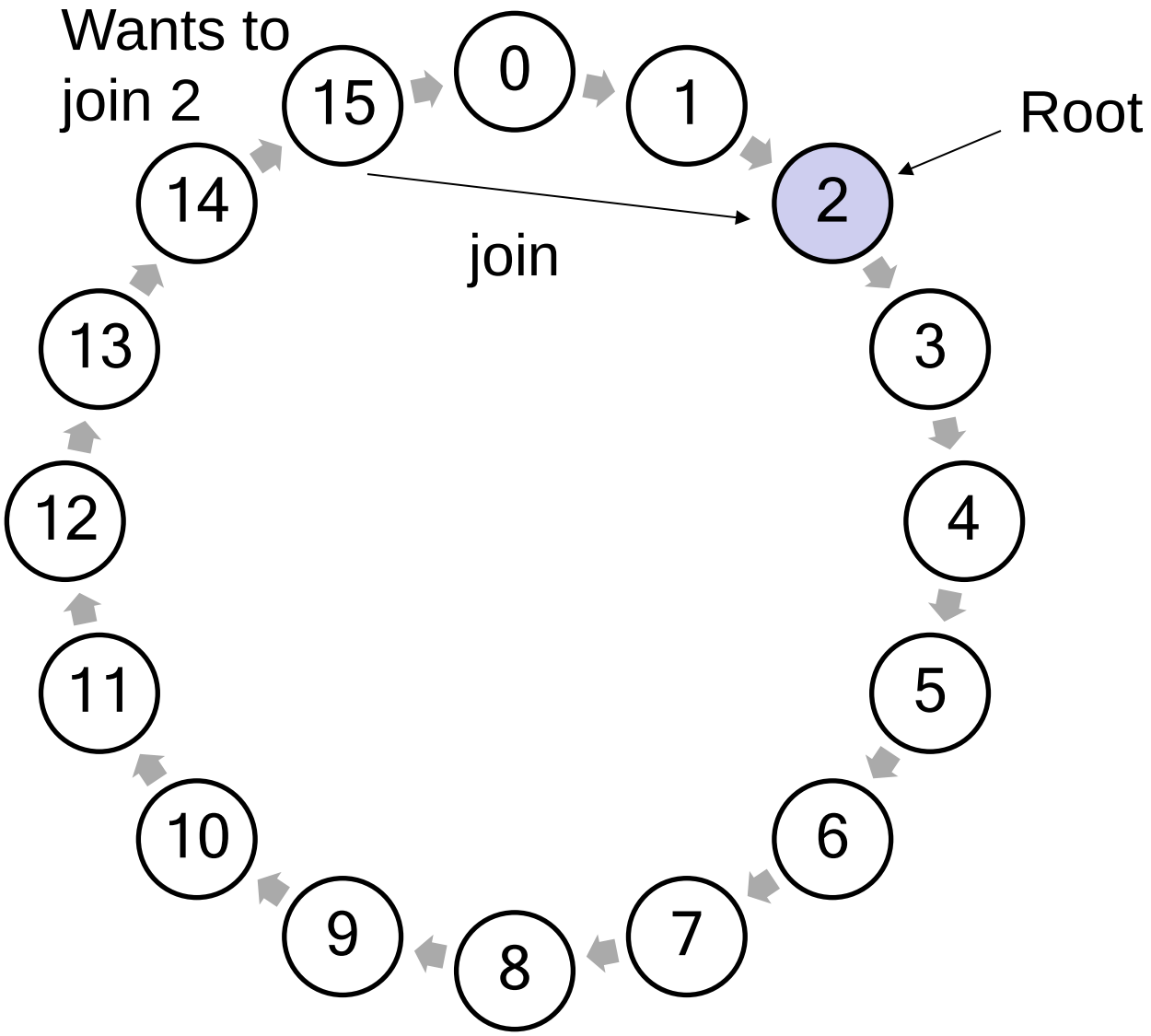
Example 2: Based on a Distributed Hash Table (Chord) peer-to-peer system:

1. Initiator generates a **multicast identifier** mid.
2. Lookup $succ(mid)$, the node responsible for mid.
3. Request is routed to $succ(mid)$, which will become the **root**.
4. If P wants to join, it sends a **join** request to the root.
5. When request arrives at Q:
   - Q has not seen a join request before ➔ it becomes **forwarder**; P becomes child of Q. Join request continues to be forwarded.
   - Q knows about tree ➔ P becomes child of Q. No need to forward join request anymore.

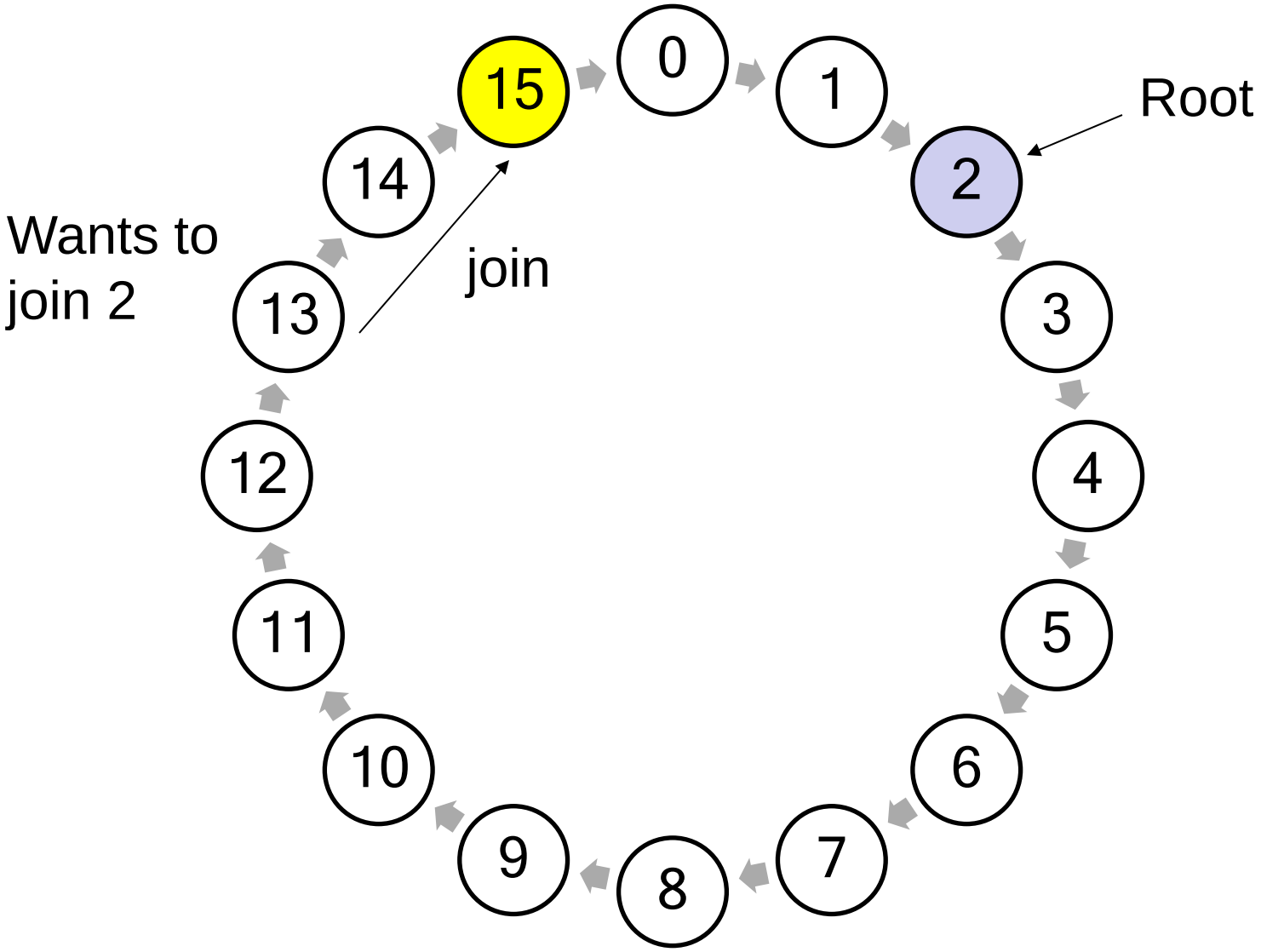
Chord Multicast Example



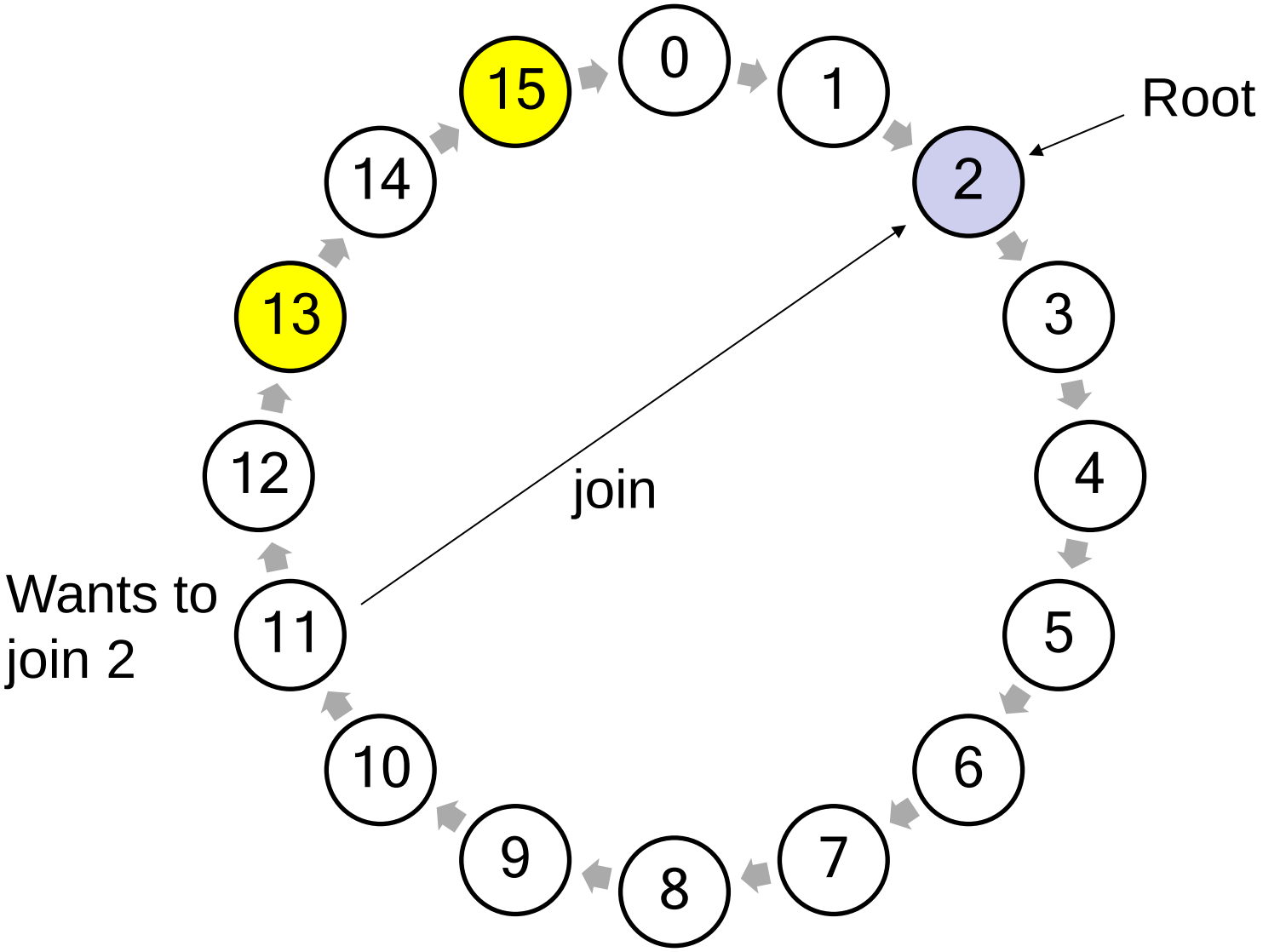
Chord Multicast Example



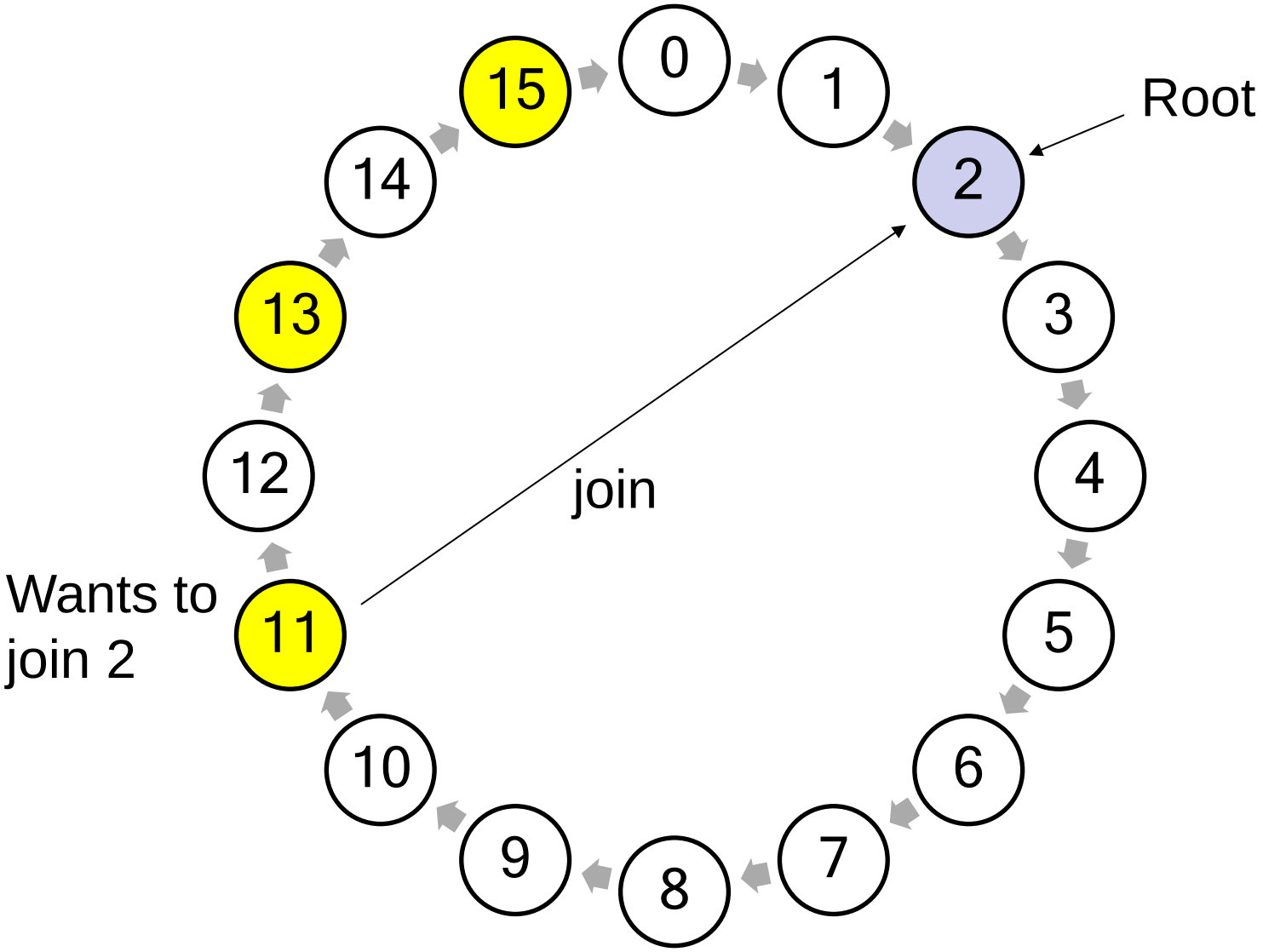
Chord Multicast Example



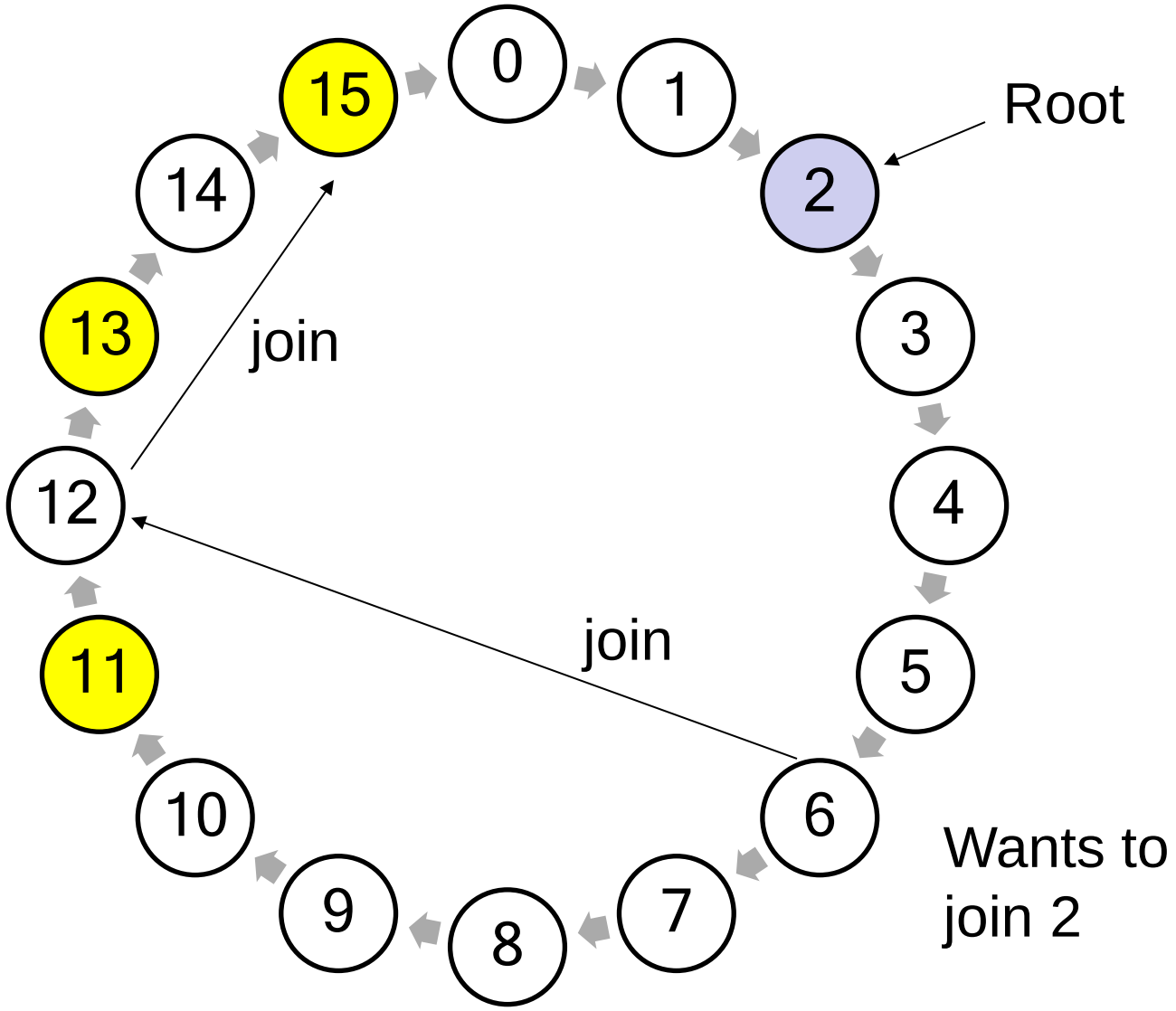
Chord Multicast Example



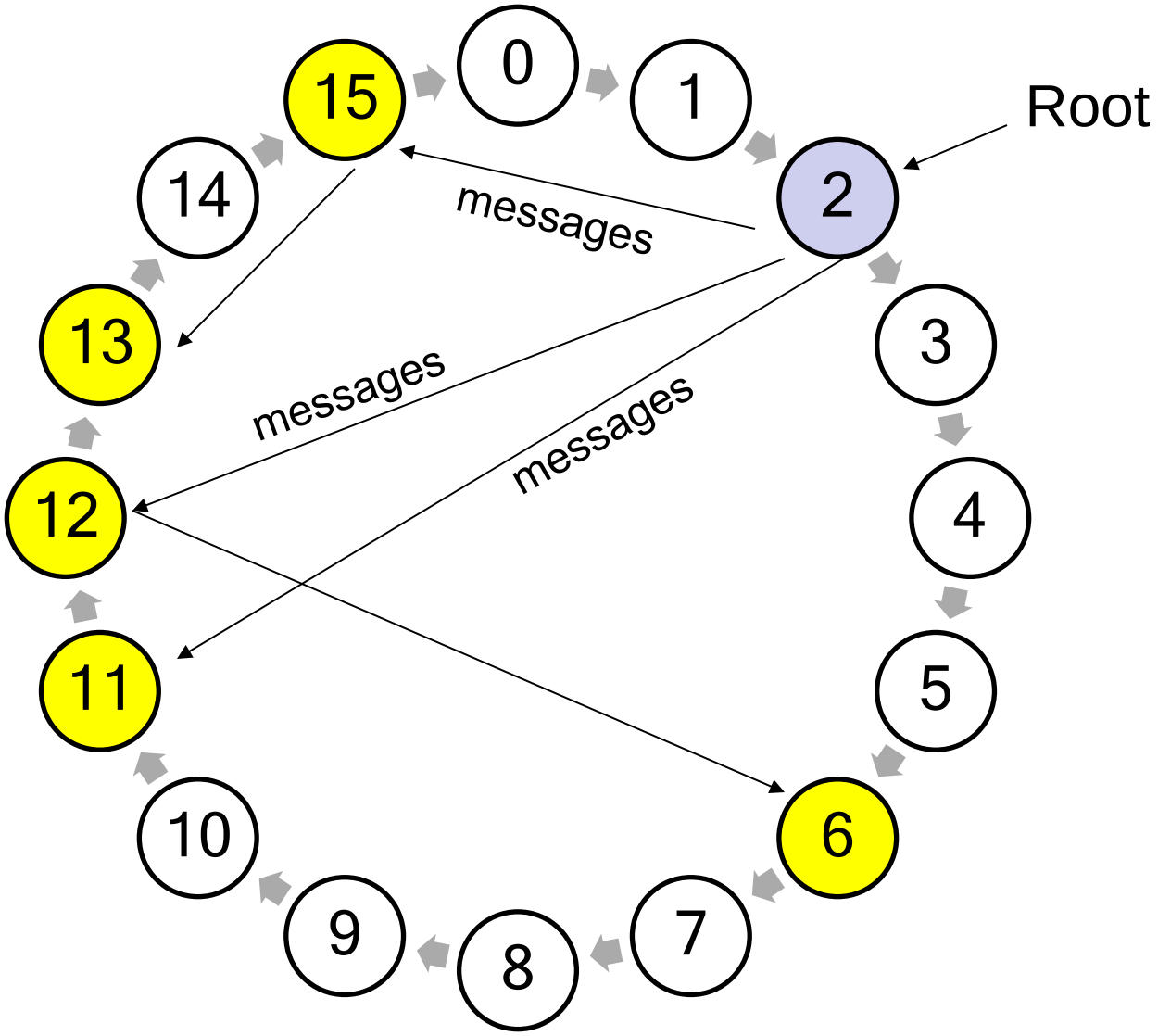
Chord Multicast Example



Chord Multicast Example



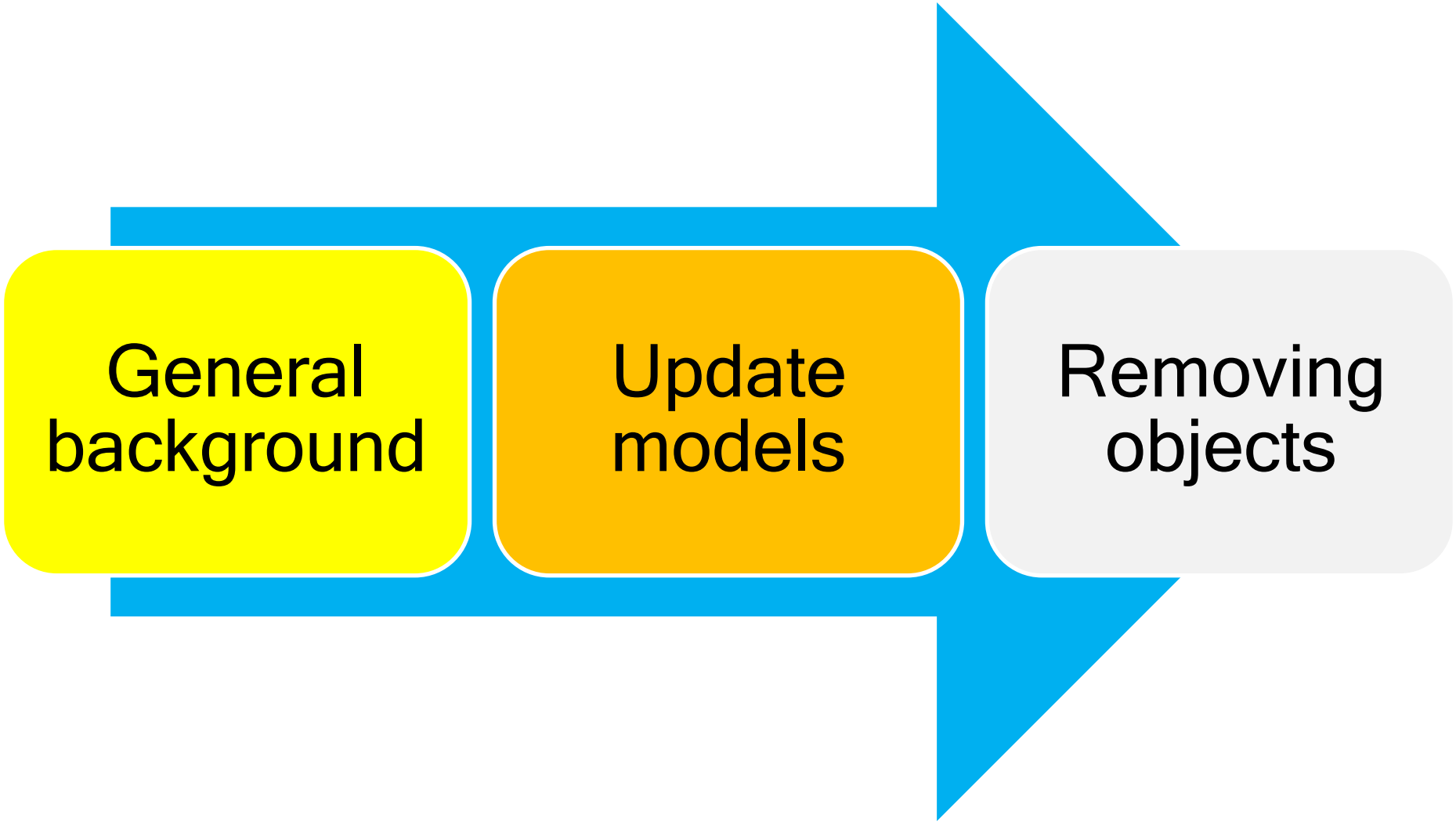
Chord Multicast Example



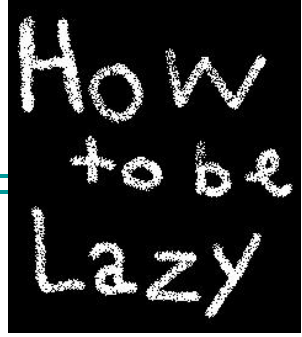
So Far

- Application Level Multicasting
- Epidemic Algorithms
- Cyber analysis: C&C

Epidemic Algorithms



Principles



Basic idea:

Assume there are no write-write conflicts:

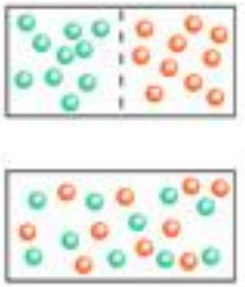
- Update operations are initially performed at one or only a few replicas
- A replica passes its updated state to a limited number of neighbors
- Update propagation is lazy, i.e., not immediate
- Eventually, each update should reach every replica

Two forms of epidemics

Anti-entropy: Each replica regularly chooses another replica at random, and exchanges state differences, leading to identical states at both afterwards.

Gossiping: A replica which has just been updated (i.e., has been **contaminated**), tells a number of other replicas about its update (contaminating them as well).

Anti-Entropy



Principle Operations:

- A node P selects another node Q from the system at random.
- **Push**: P only sends its updates to Q
- **Pull**: P only retrieves updates from Q
- **Push-Pull**: P and Q exchange mutual updates (after which they hold the same information).

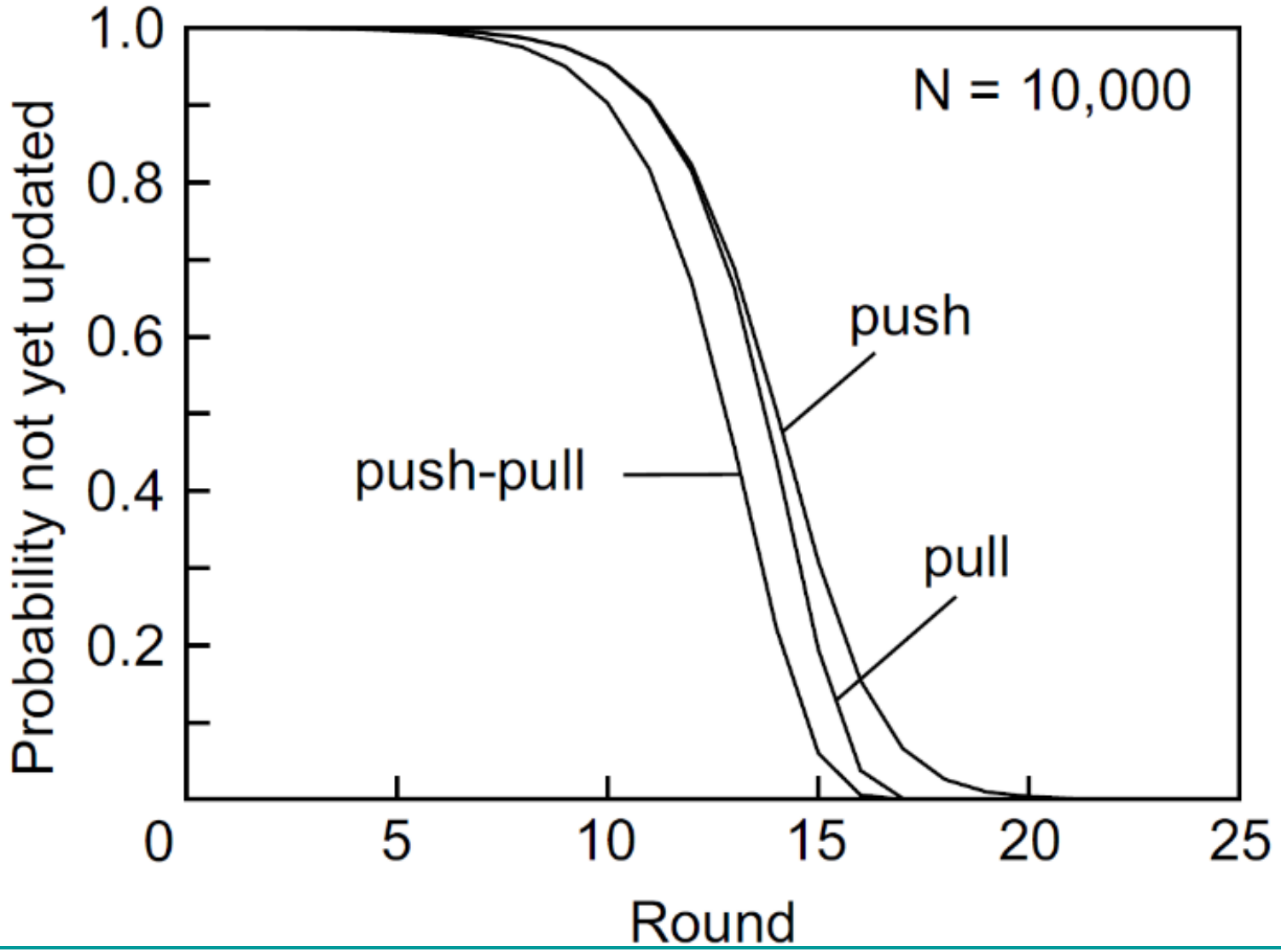
Observation:

For push-pull it takes $O(\log(N))$ rounds to disseminate updates to all N nodes (**round** = when every node as taken the initiative to start an exchange).

Anti-Entropy: Staying Ignorant

- A single source is propagating its update. p_i is the probability that a node **has not received** the update after the i^{th} round.
- With pull, $p_{i+1} = (p_i)^2$: the node was not updated during the i^{th} round and should contact another ignorant node during the next round
- With push, $p_{i+1} = p_i \left(1 - \frac{1}{N}\right)^{N(1-p_i)} \approx p_i e^{-1}$ (for small p_i and large N): the node was ignorant during the i^{th} round and no updated node chooses to contact it during the next round.
- With push-pull:$(p_i)^2 \times (p_i e^{-1})$

Anti-Entropy: Staying Ignorant



Anti-Entropy (Initial)

10.0.0.1

File	Version
File1.txt	1
File2.docx	2
File3.pdf	2
File4.pdf	3

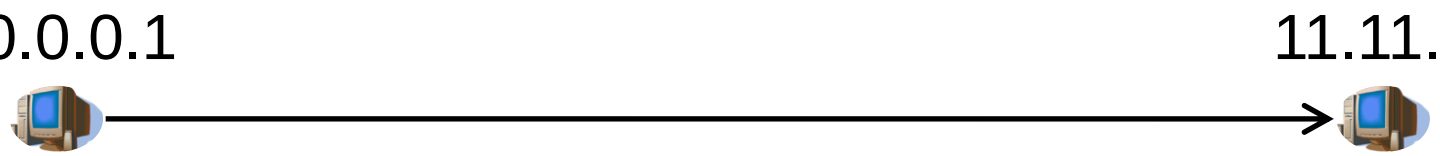
11.11.0.11

File	Version
File1.txt	1
File3.pdf	4
File5.pdf	1

12.12.0.12

File	Version
File2.docx	1
File3.pdf	2
File5.pdf	2

Anti-Entropy (Round 1)



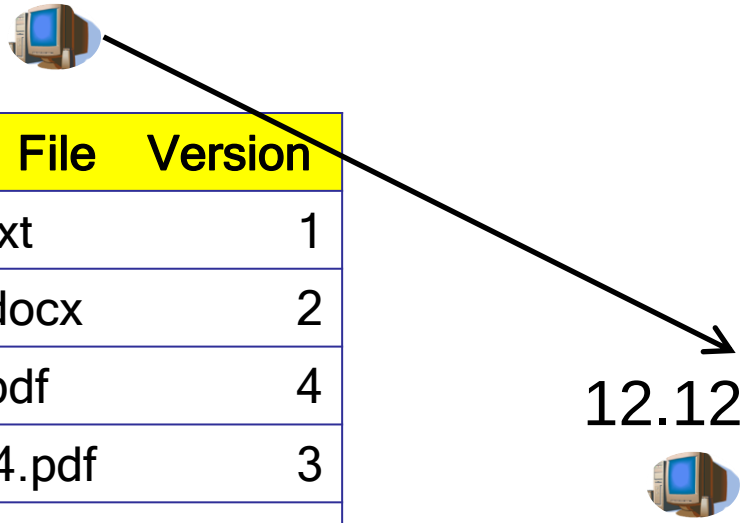
File	Version
File1.txt	1
File2.docx	2
File3.pdf	4
File4.pdf	3
File5.pdf	1

File	Version
File1.txt	1
File2.docx	2
File3.pdf	4
File4.pdf	3
File5.pdf	1

12.12.0.12

File	Version
File2.docx	1
File3.pdf	2
File5.pdf	2

Anti-Entropy (Round 1)

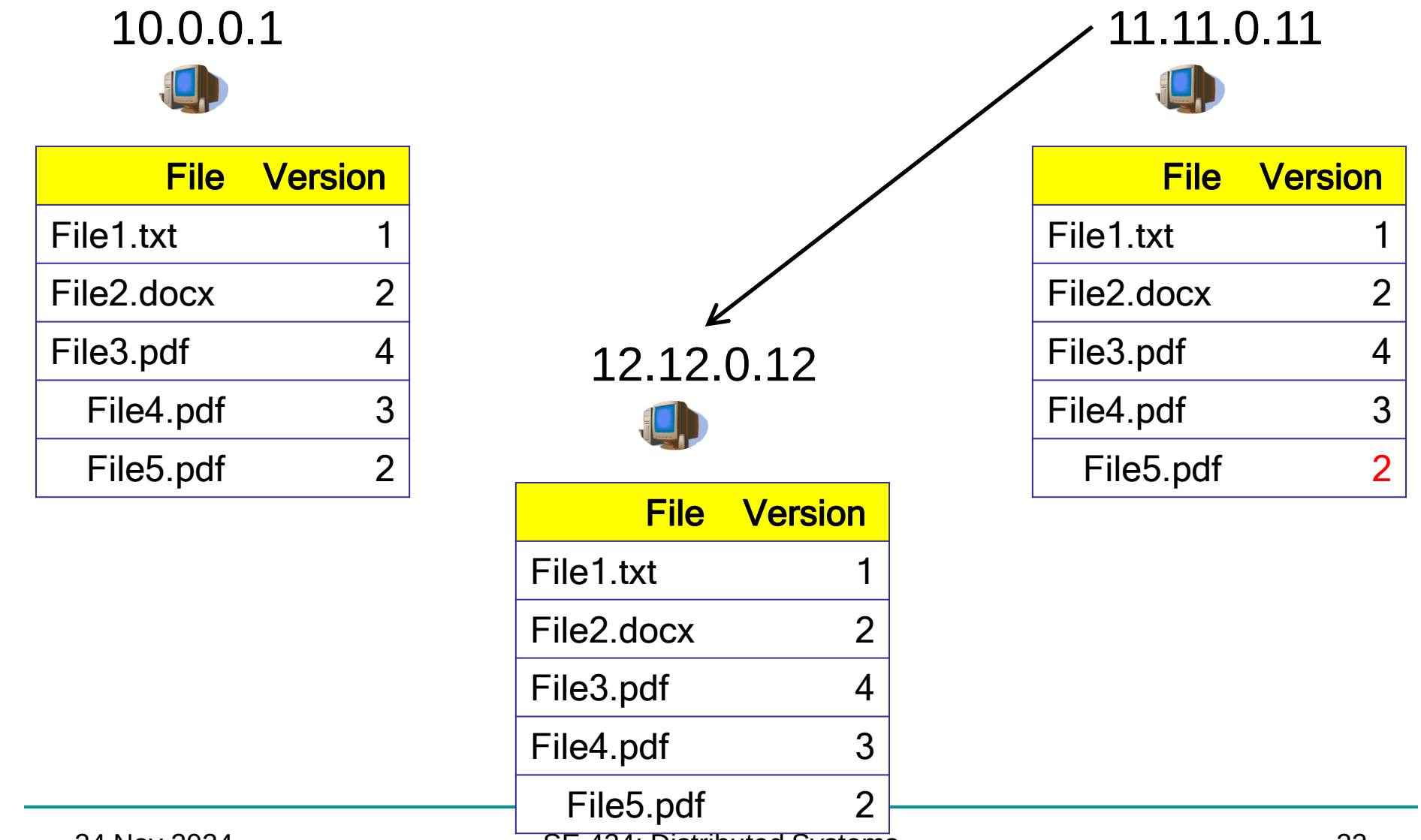


File	Version
File1.txt	1
File2.docx	2
File3.pdf	4
File4.pdf	3
File5.pdf	2

File	Version
File1.txt	1
File2.docx	2
File3.pdf	4
File4.pdf	3
File5.pdf	1

File	Version
File1.txt	1
File2.docx	2
File3.pdf	4
File4.pdf	3
File5.pdf	2

Anti-Entropy (Round 1)



Gossiping

Basic model:

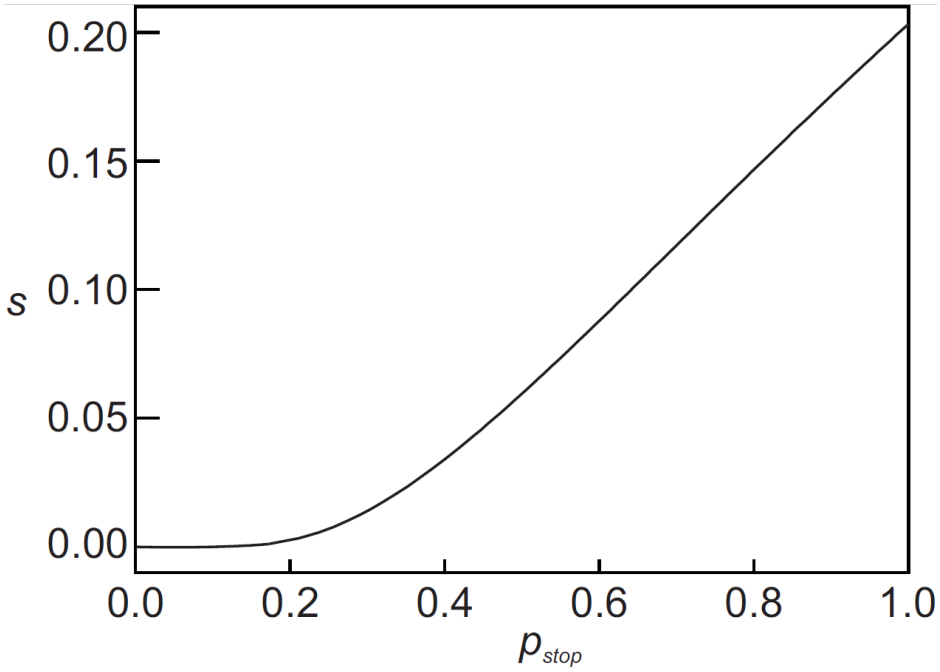
A server S having an update to report, contacts other servers. If a server is contacted to which the update has already propagated, S stops contacting other servers with probability p_{stop}

Observation:

If s is the fraction of ignorant servers (i.e., which are unaware of the update), it can be shown that with many servers

$$s = e^{-\left(\frac{1}{p_{stop}}+1\right)(1-s)}$$

Gossiping



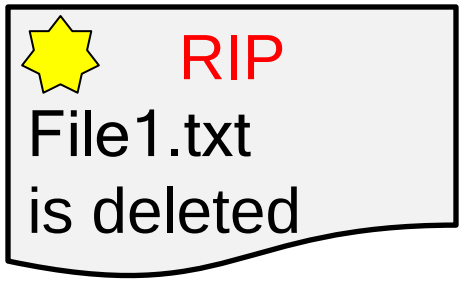
N=10,000 Nodes			
$\frac{1}{p_{stop}}$	p_{stop}	s Uninformed ratio	N_s Uninformed Total
1	1	0.203188	2032
2	0.50	0.059520	595
3	0.33	0.019827	198
4	0.25	0.006977	70
5	0.20	0.002516	25
6	0.16	0.000918	9
7	0.14	0.000336	3

Note: If we really have to ensure that all servers are eventually updated, gossiping alone is not enough

Deleting Values

Fundamental problem: We cannot remove an old value from a server and expect the removal to propagate. Instead, mere removal will be undone in due time using epidemic algorithms

Solution: Removal has to be registered as a special update by inserting a **death certificate**



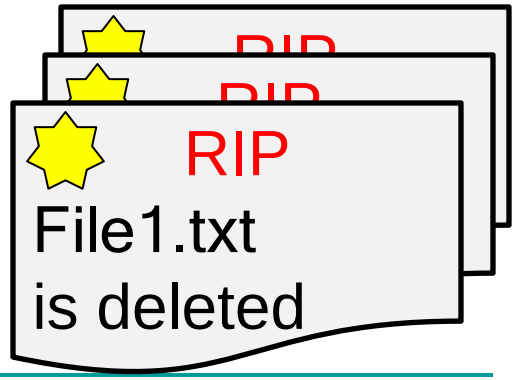
Deleting Values

Next problem: When to remove a death certificate (it is not allowed to stay for ever):

- Run a global algorithm to detect whether the removal is known everywhere, and then collect the death certificates (looks like garbage collection)
- Assume death certificates propagate in finite time, and associate a maximum lifetime for a certificate (can be done at risk of not reaching all servers)

Note: it is necessary that a removal actually reaches all servers.

Question: What's the scalability problem here?



Example Applications

Typical applications:

- **Data dissemination**: Perhaps the most important one. Note that there are many variants of dissemination.

- **Aggregation**: Let every node i maintain a variable x_i. When two nodes gossip, they each reset their variable to
  - $x_i, x_j \leftarrow \frac{x_i + x_j}{2}$
- Result: in the end each node will have computed the average

$$\bar{x} = \sum_i \frac{x_i}{N}$$

Question: What happens if initially $x_i = 1$ and $x_j = 0$, $j \neq i$?

So Far

- Application Level Multicasting
- Epidemic Algorithms
- Cyber analysis: C&C

Conclusion

- Application Level Multicasting
- Epidemic Algorithms
- Cyber analysis: C&C