

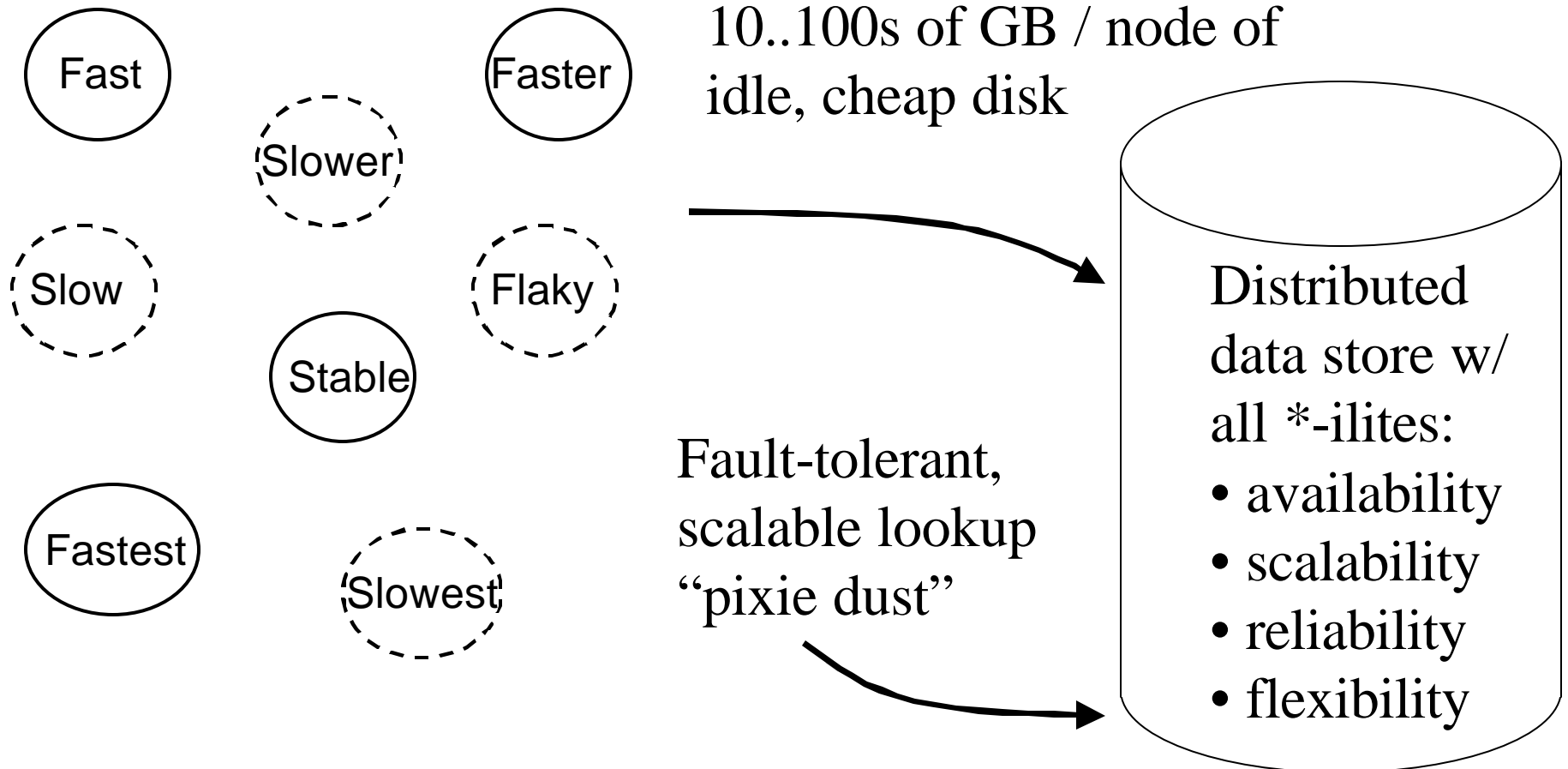
# The Cost of Membership Dynamics

Rodrigo Rodrigues

MIT Computer Science and Artificial Intelligence Laboratory

Joint work with Charles Blake, Barbara Liskov

# The P2P Dream



How realistic is this dream?

# Potential Problem

- Dynamic membership implies redundancy maintenance traffic
- Bandwidth is limited
  - Particularly in peer-to-peer deployments
- How dynamic can membership be?

# Basic Model for Data Maintenance Cost

- $N$  nodes ( $N$  probably  $\gg 10,000$ )
  - Placed randomly about the Internet
  - Using similar bandwidth and disk
  - Cooperatively serving  $DBsize$  bytes of unique data
  - Redundancy factor  $redun$  (replication or stretch)
    - $\therefore Space/N = redun \times DBsize / N$
- Assume average system size,  $N$ , stable
- Join = Leave forever rate =  $N / Lifetime$

# Basic Model for Data Maintenance

- Leaves induce redundancy replacement  
replacement size x replacement rate
- Joins cost the same

$\therefore$  Total BW  $>$  2 x *redun x DBsize / N* x Join rate

Space/node

Space/node  $<$   $\frac{1}{2}$  BW/node x Lifetime

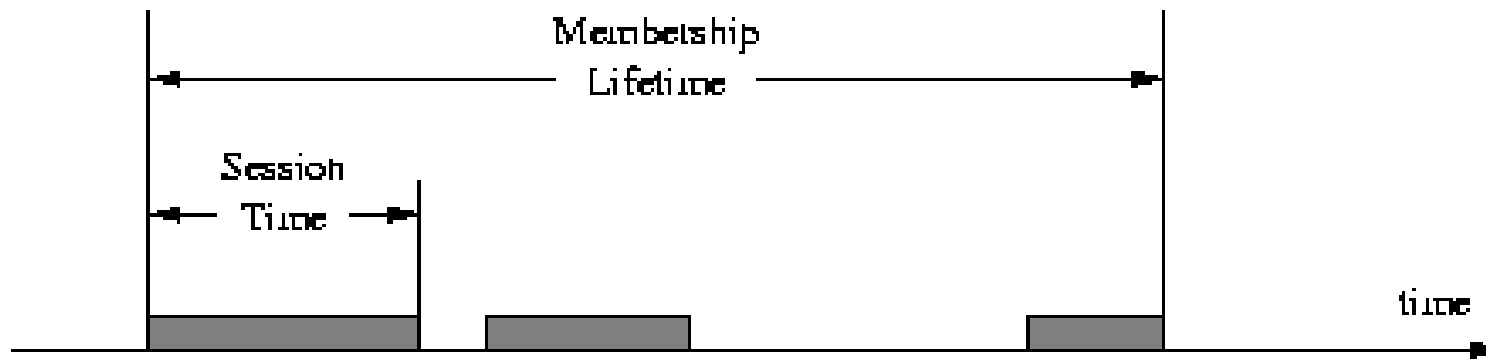
# This Scaling is a Problem

- Maintenance bw = 50 kbps
- Lifetime = Median 2001 Gnutella session = 1 hour
- served space = 22 MB/node  
<< donatable storage

# This bw Cost is Optimistic

- We assume served data is totally static
- Serious “promise” → worst case
- Identical space & bandwidth
- Fixed population
- Load-balance, Popular data more available Additional redundancy → more BW
- Downtime isn't leaving forever

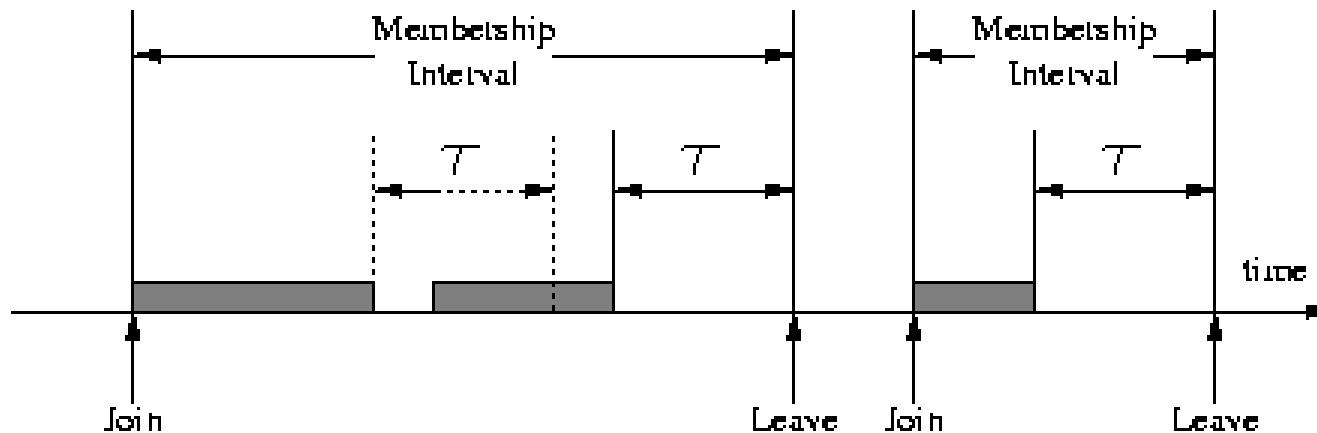
# Distinguishing Sessions and Lifetime



- Important to distinguish them (avoid redundancy maintenance)
- But requires future knowledge

# Delayed Response to Failures

- Wait  $t$  to declare unreachable member gone



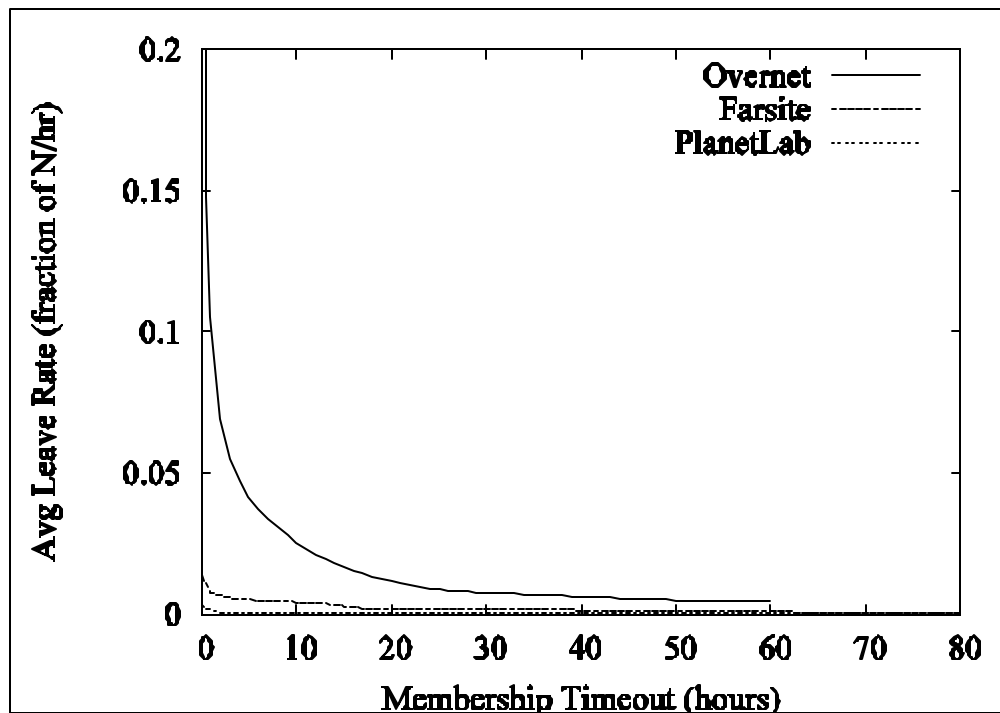
# Consequences of $t$

- $N$  increases – more peers, some down
- $upfrac = P(\text{typical node is up})$
- Redundancy must be higher
  - Replication:  $redun = \log(e)/\log(1-upfrac)$

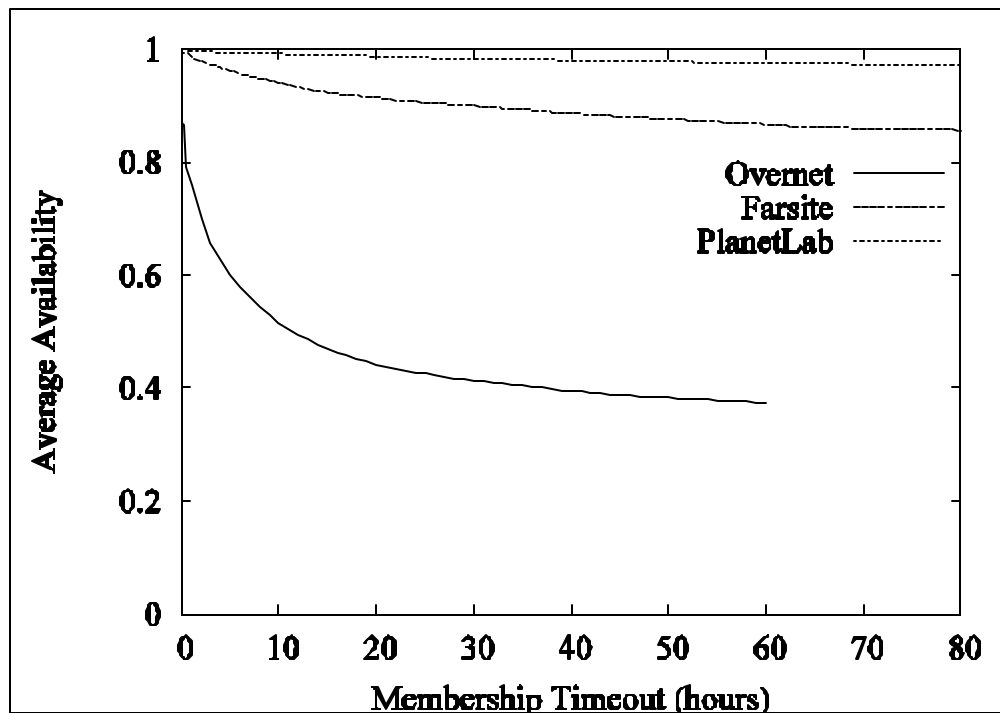
# How to set $t$ ?

- Depends on distribution of node session times
- Analyzed three traces:
  - Overnet
  - Farsite
  - PlanetLab

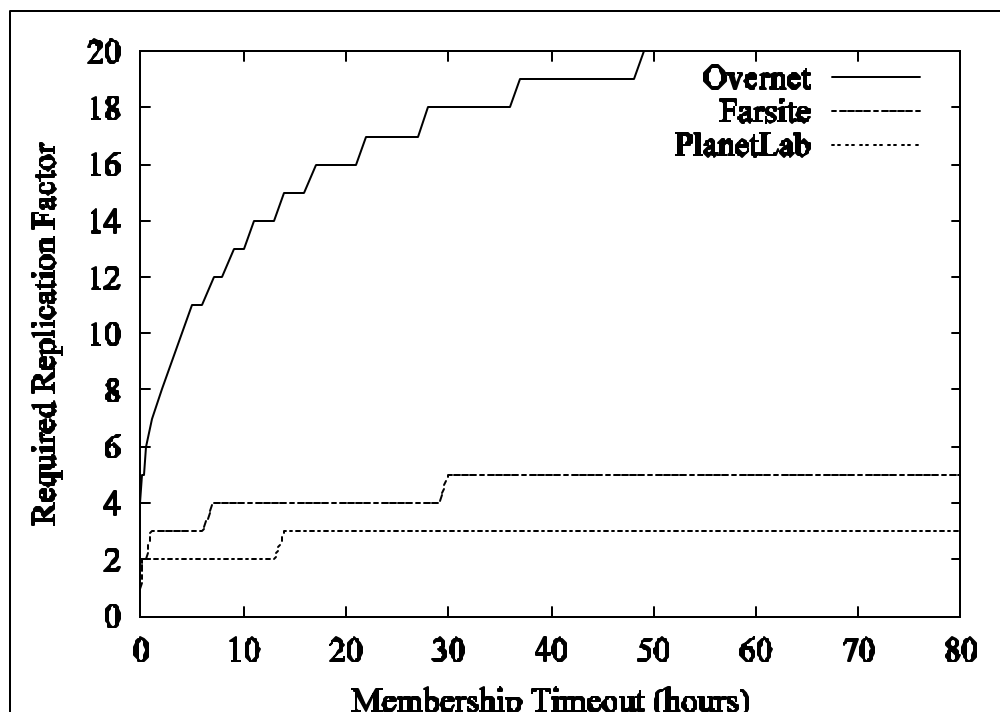
# Membership Dynamics



# Node Availability

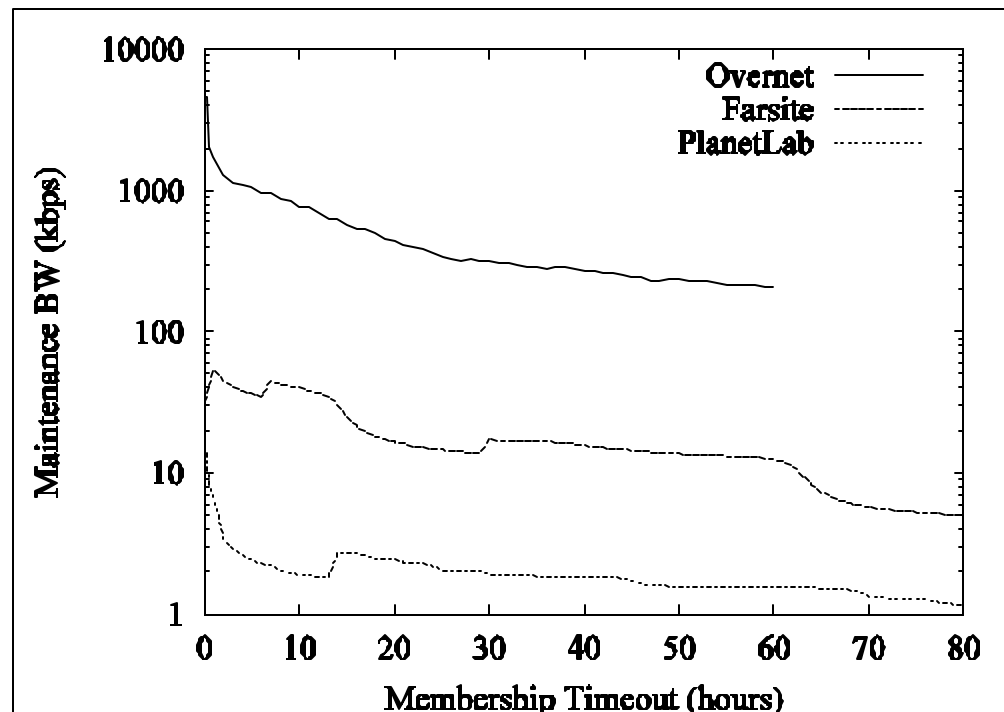


# Required Replication



- Using  $redun = \log(e)/\log(1-upfrac)$
- Targets 4 nines of per-object availability

# Required Bandwidth



- Assumes 10k nodes, 10 TB of unique data

# Erasure Coding

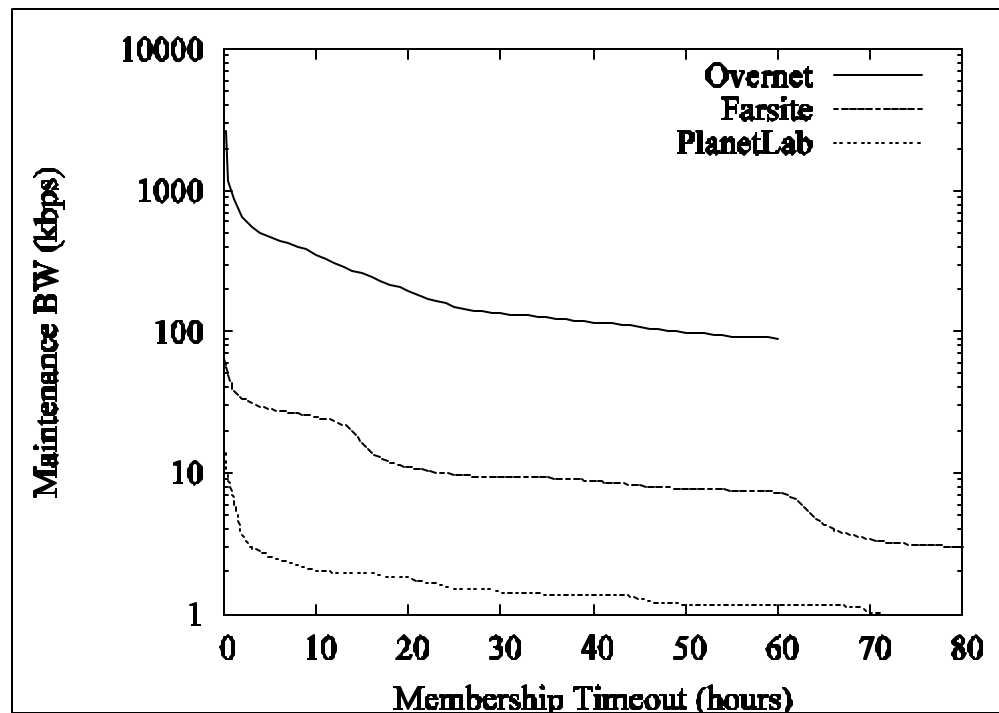
- Download  $b$  out of  $k.b$  fragments
- Reduces redundancy factors:

$$1-e = \sum_{i=b}^{k.b} \binom{i}{k.b} a^i (1-a)^{k.b-i}$$

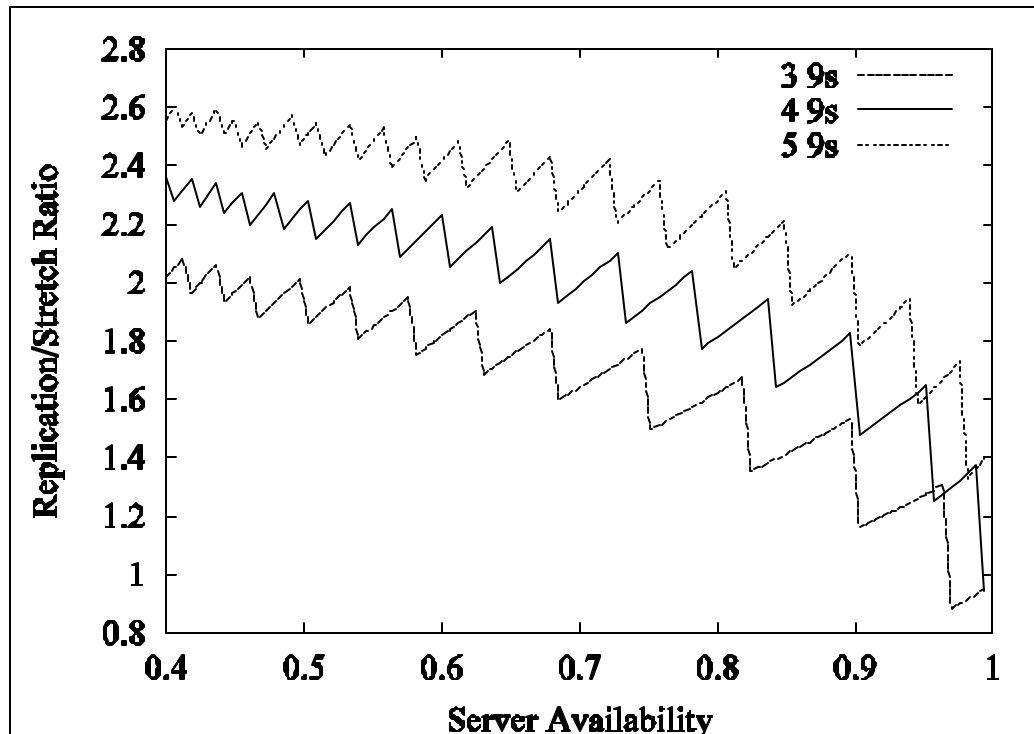
$$k = \left( \frac{\sigma_\epsilon \sqrt{\frac{\alpha(1-\alpha)}{b}} + \sqrt{\frac{\sigma_\epsilon^2 \alpha(1-\alpha)}{b} + 4a}}{2a} \right)^2$$

- Fragment reconstruction can be problematic

# Required Bandwidth (Coding)



# How much do we win?



- Redundancy gains depend on server availability, target availability

# Conclusions

- Cooperative storage requires stable membership
- Careful distinction of downtime vs. departure is important
- Erasure coding also help, but gains depend on deployment