



Group Communication

Zhong, Rongrong
rzhong@cs.rochester.edu



What & Why

❖ What is group communication

- A message sent to a group should be received by every member of that group

❖ Why we need group communication

- Replication
- Secure conferencing
- Military



What are the assumptions?

- ❖ We assume they are all good citizens and are fail-stop

- ❖ Processes are assumed to operate correctly
 - Reliable = every group member receives the message

- ❖ Processes might fail unexpectedly
 - Reliable = a message is delivered to either every member or none of them



Protocols we will cover today

- ❖ Scalable Reliable Multicasting (SRM)
- ❖ Virtual Synchrony
- ❖ Epidemic Protocols

Basic reliable-multicasting schemes

❖ Best-effort (assumes no faulty processes)

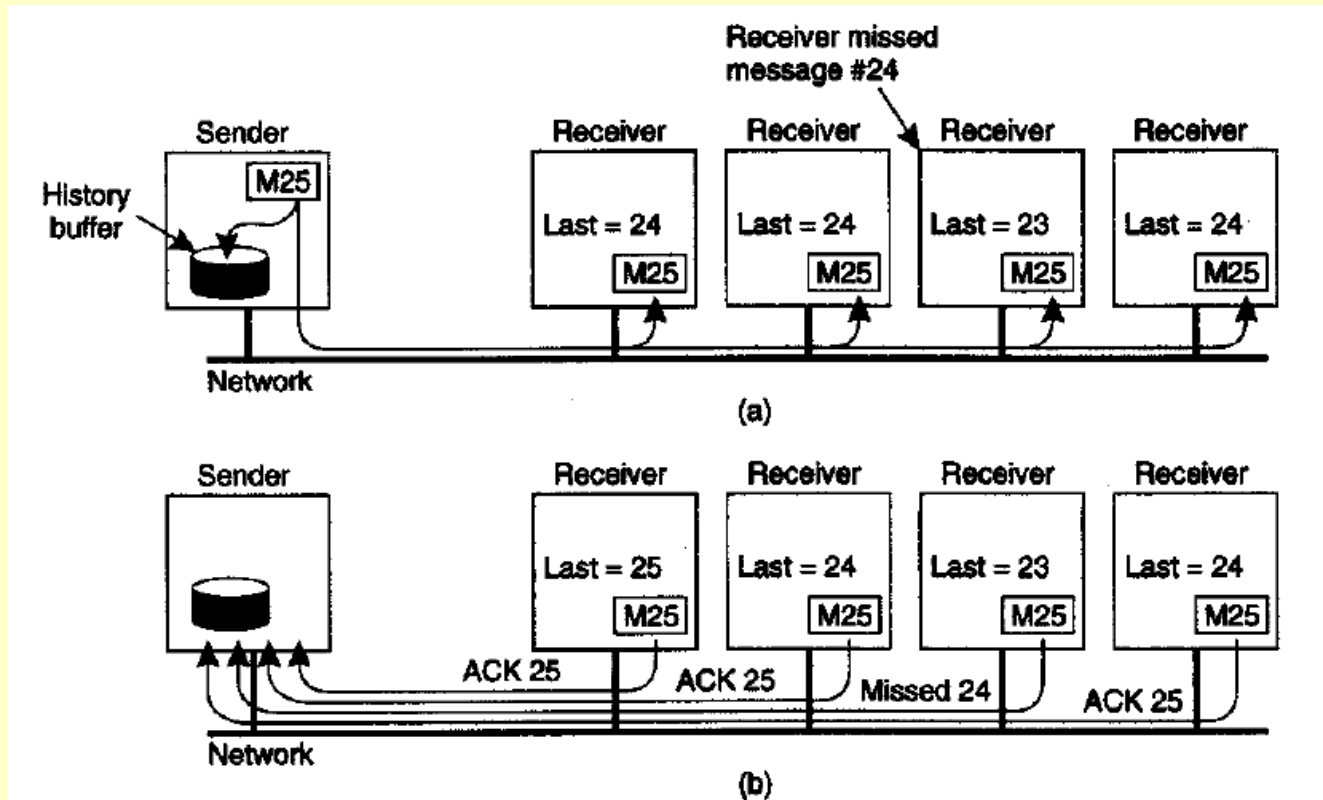


Figure 7-8. A simple solution to reliable multicasting when all receivers are known and are assumed not to fail. (a) Message transmission. (b) Reporting feedback.

Figure from Tanenbaum and van Steen



Basic reliable-multicasting schemes

❖ Problems

- Too many messages

❖ Trade-offs

- Piggyback ACKs with other messages
- Retransmitting with point-to-point communication

Scalable Reliable Multicasting (SRM)

- ❖ Best-effort
- ❖ No ACKs, only NACKs
- ❖ Feedback suppression, retransmission requests are sent after some random delay

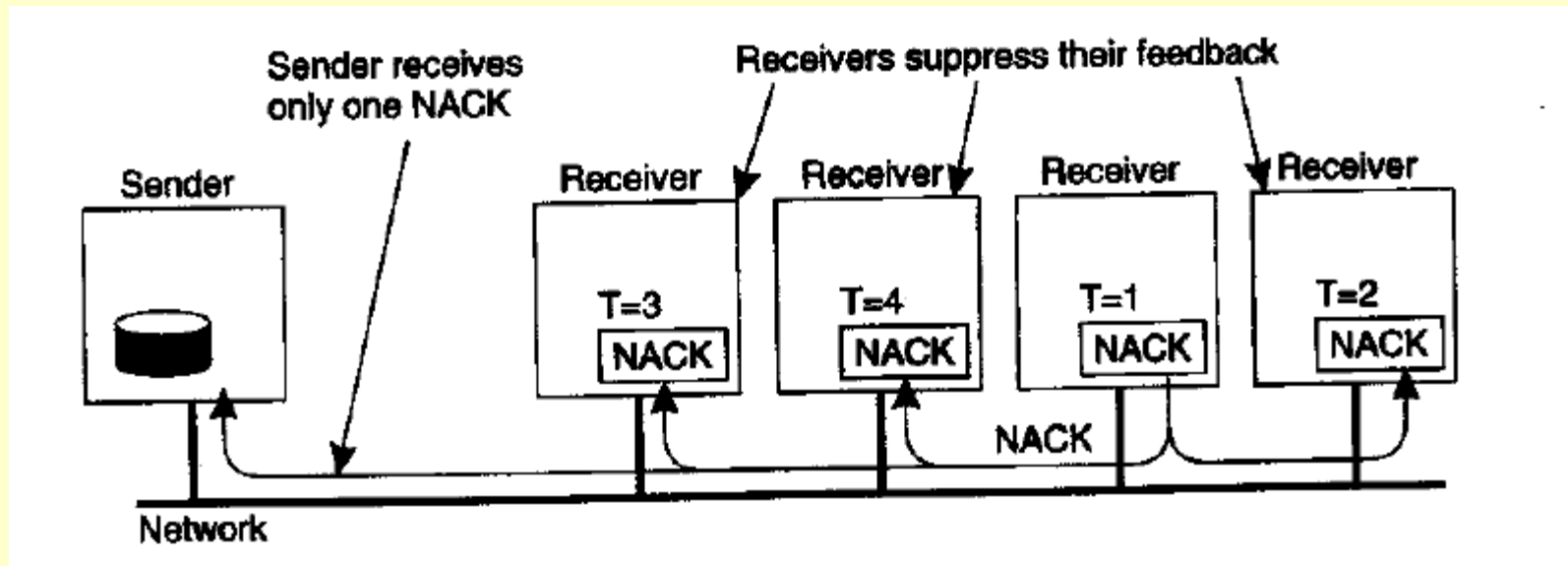


Figure from Tanenbaum and van Steen



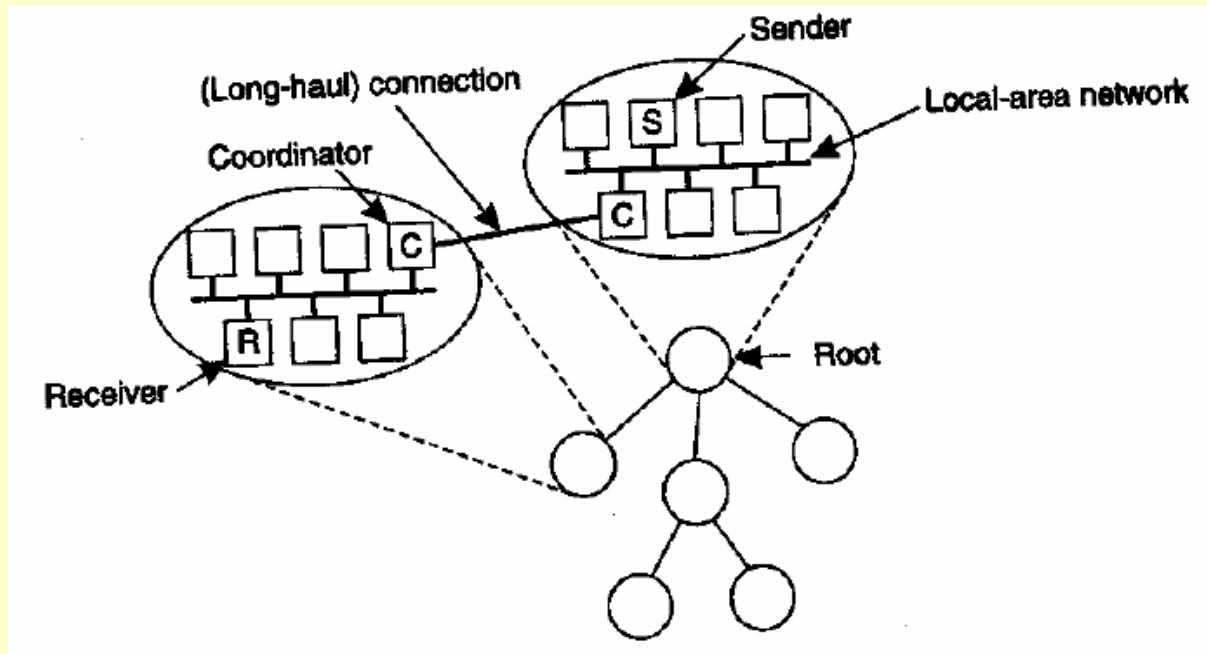
Scalable Reliable Multicasting (SRM)

❖ Problems

- In theory, sender is forced to keep its messages in history buffer forever
- Interrupts processes who have received the message successfully
- It is not easy to ensure only one request for retransmission

Hierarchical feedback control

- ❖ Each subgroup has a local coordinator which has its own history buffer



❖ Problems

- Constructing the tree

❖ Request and retransmission storms

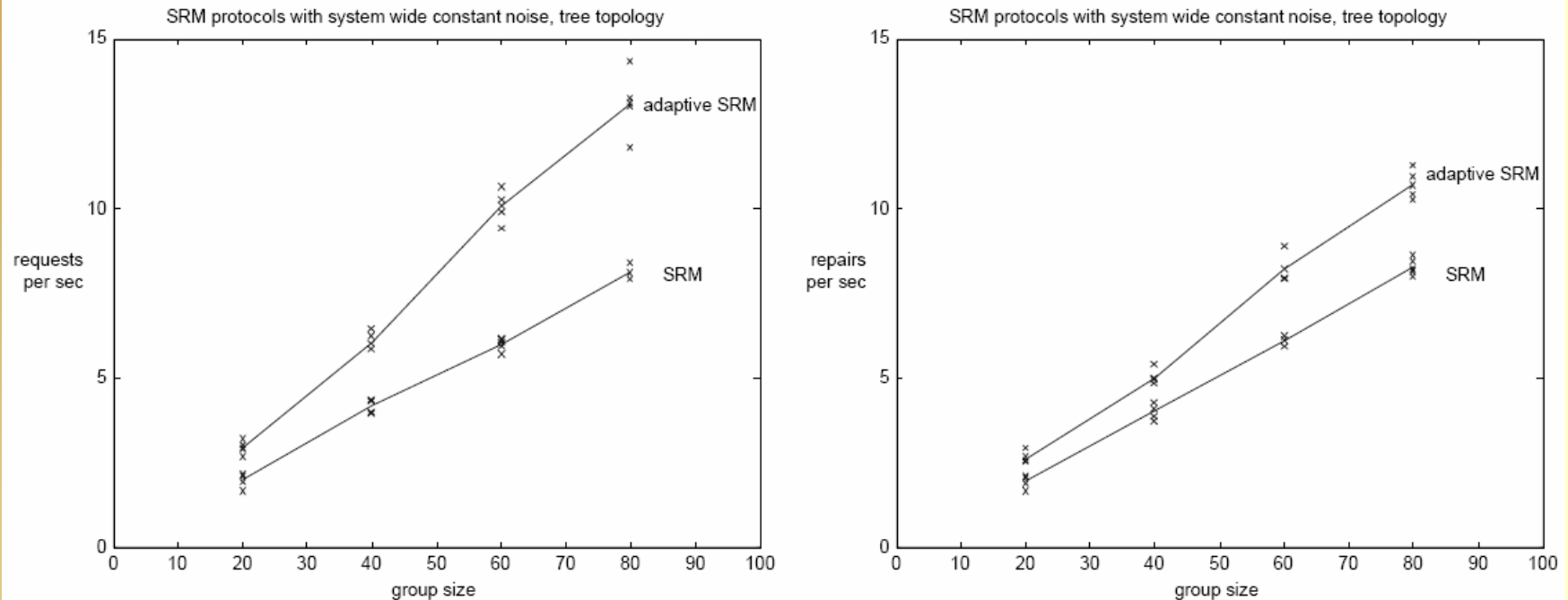


Figure 3: As the size of the group increases, a low level of background noise (0.1% in this case) can trigger high rates of requests (left) and retransmissions (right) for the SRM protocols. Most of these are duplicates. Notice that the data rate is being held constant; only the size of the group is increased in these experiments.



When there are faulty processes

❖ Atomic multicast

- Guarantees that a message is delivered to either all processes or to none at all
- messages are delivered to all processes in the same order (total ordering)

❖ Definitions

- Message receipt vs. message delivery

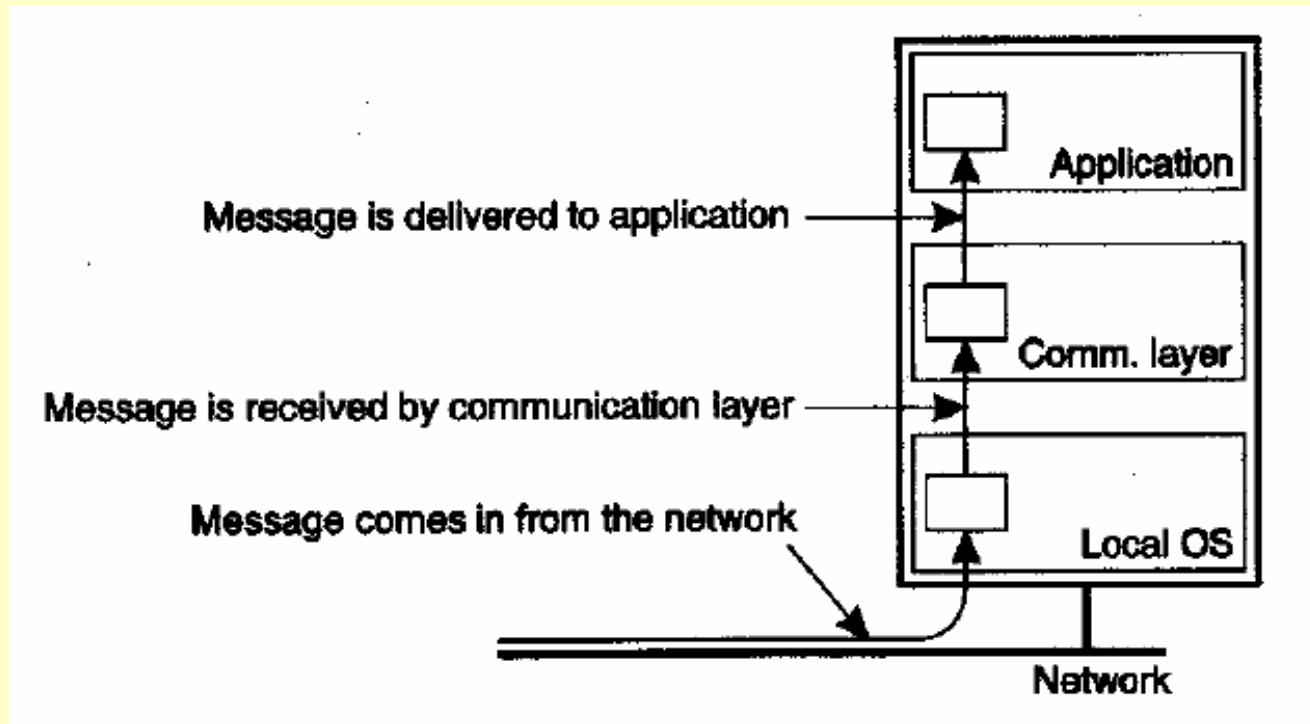


Figure from Tanenbaum and van Steen

❖ Definitions

- Message ordering
 - Unordered multicasts
 - FIFO-ordered multicasts
 - Causally-ordered multicasts

Multicast	Basic Message Ordering	Total-Ordered Delivery?
Reliable multicast	None	No
FIFO multicast	FIFO-ordered delivery	No
Causal multicast	Causal-ordered delivery	No
Atomic multicast	None	Yes
FIFO atomic multicast	FIFO-ordered delivery	Yes
Causal atomic multicast	Causal-ordered delivery	Yes

Table from Tanenbaum and van Steen



Virtual Synchrony

❖ Definitions

- **Group view:** the view on the set of processes contained in the group at the time a sender multicast a message. All group members should agree on this view.
- **View change:** a multicasting message announcing the joining or leaving of a process
- **Stable:** a message m is stable if it has been received by all members in the group

Virtual Synchrony

❖ Virtual synchronous multicast

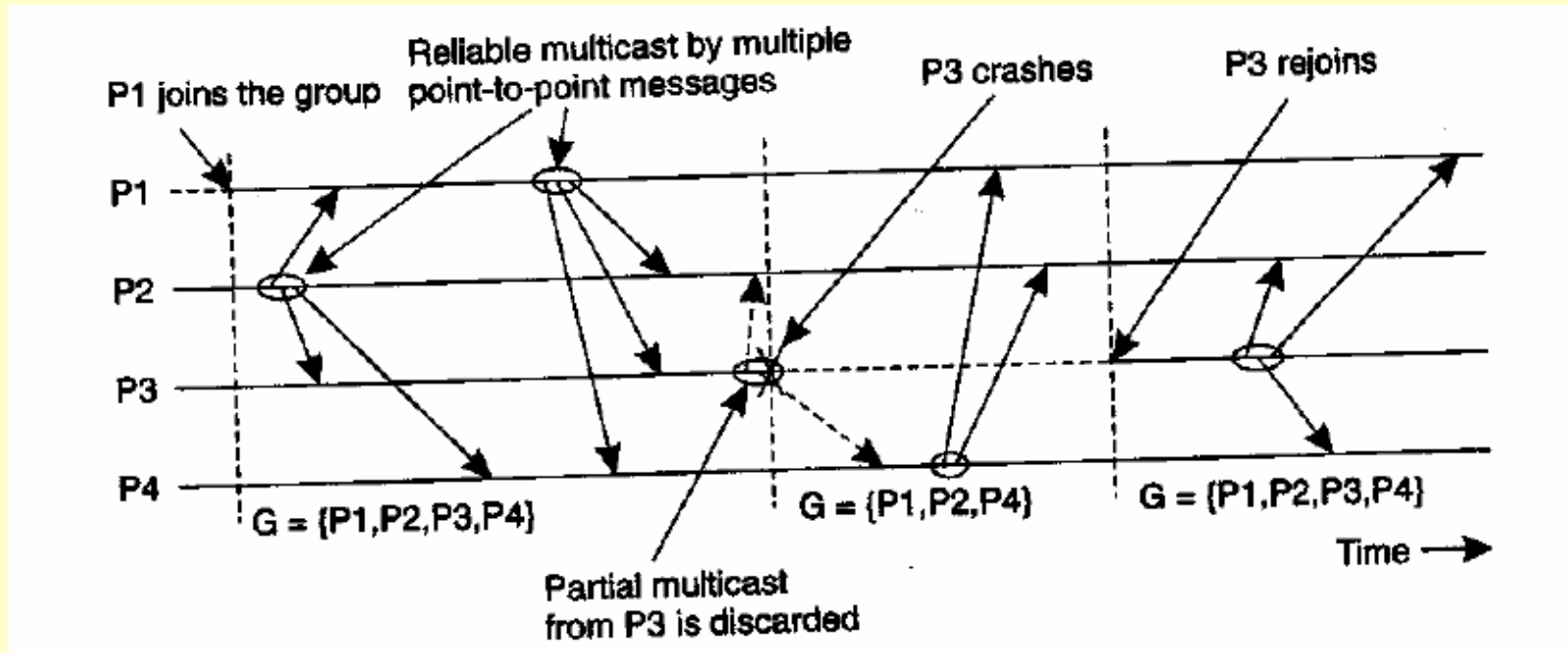


Figure from Tanenbaum and van Steen

❖ Implementation

- Use reliable point-to-point communication, TCP
- View change

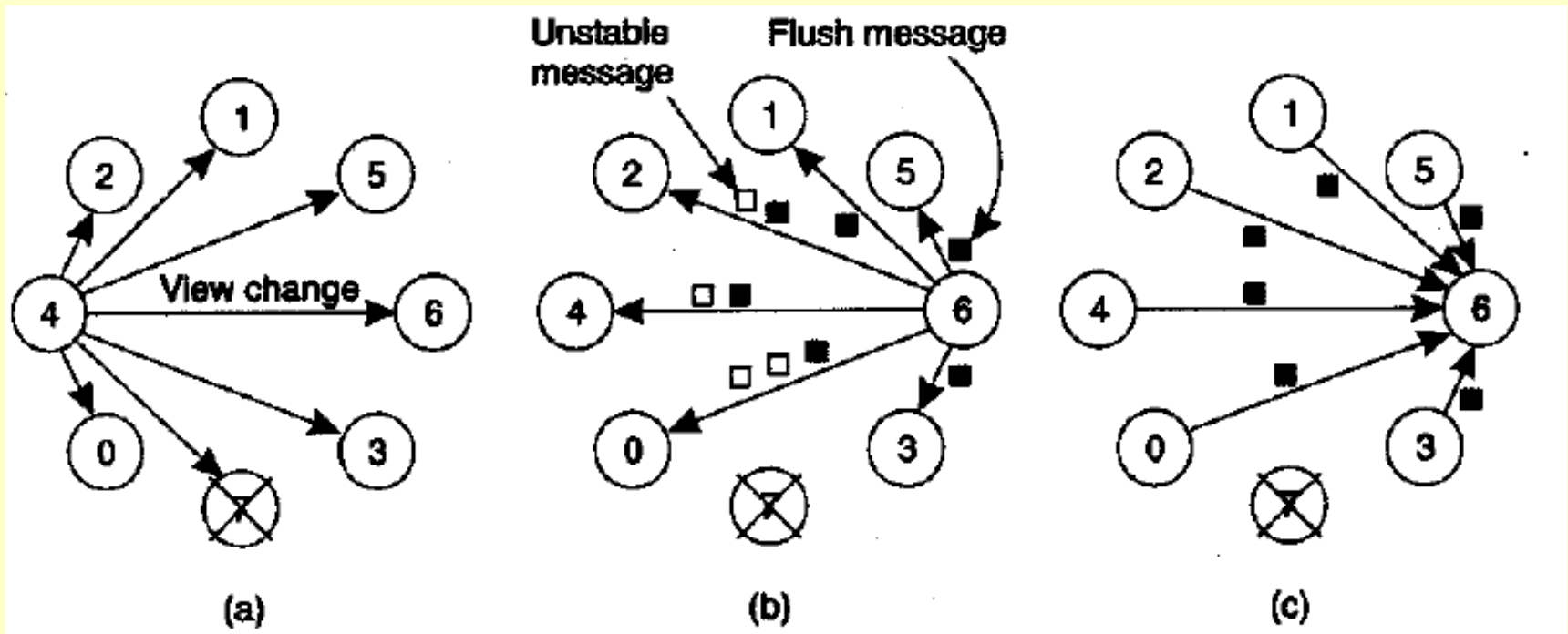


Figure from Tanenbaum and van Steen



Virtual Synchrony

❖ Good reliability

- Automated tracking of group membership
- Reporting of membership changes to the members
- Fault-tolerance multicast

❖ What about scalability?

❖ Throughput instability

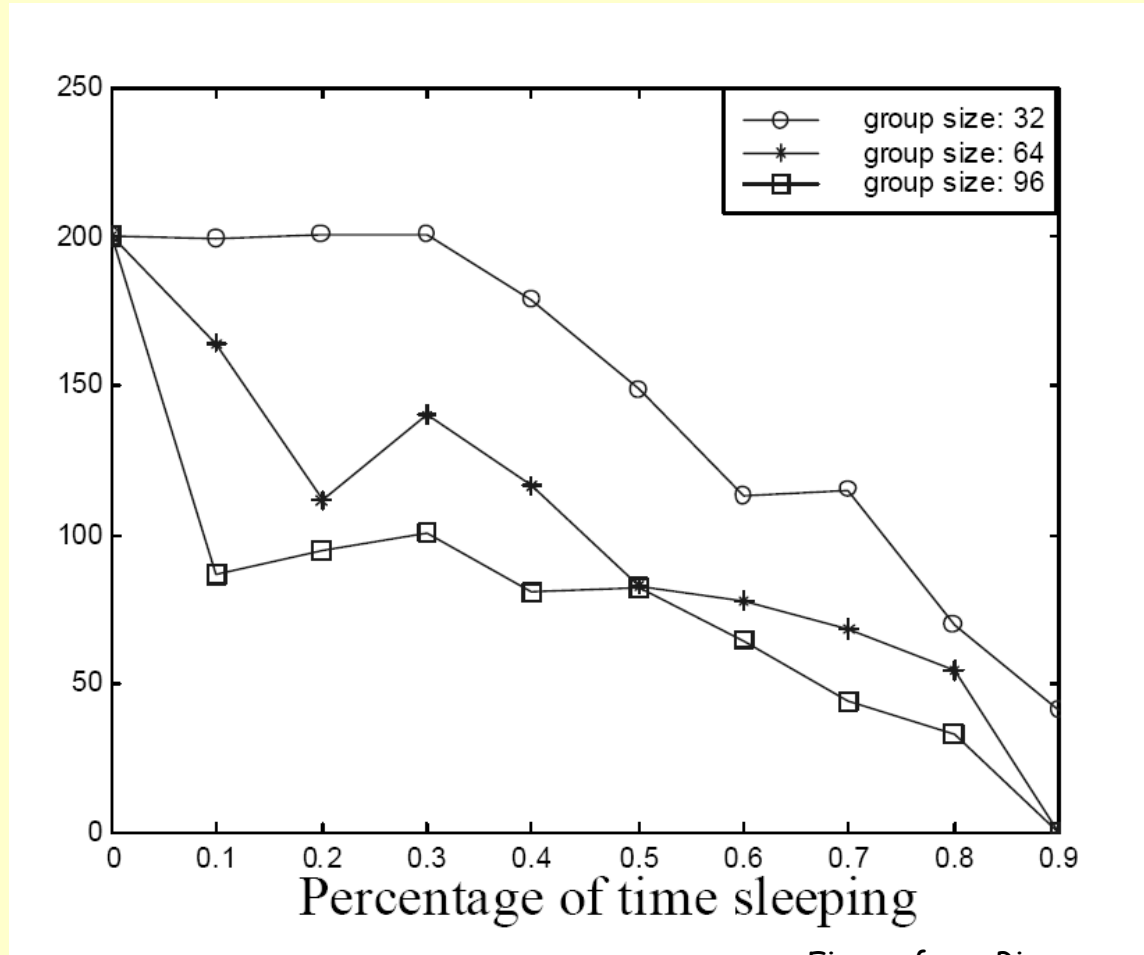


Figure from Birman, van Renesse and Vogels



Scalability issues

❖ Micropartitions

- Set failure detection threshold aggressively to knock out the slow receivers
- Costly leave/rejoin

Scalability issues

❖ "Convoys" with hierarchies

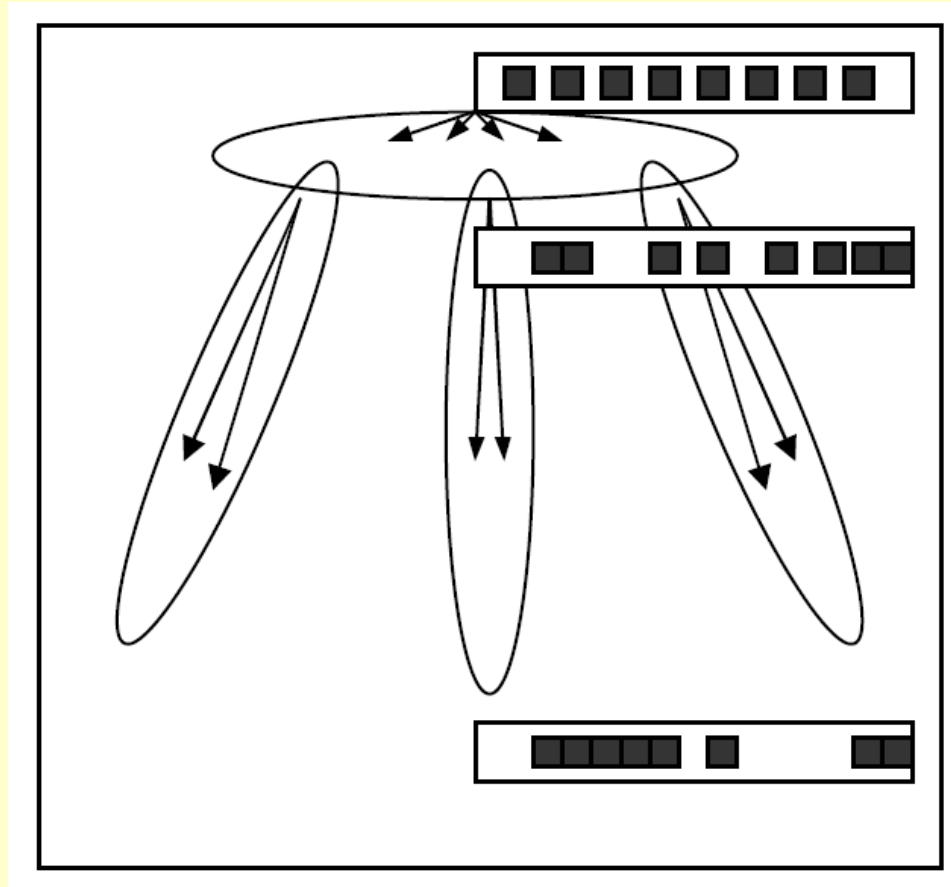


Figure from Birman, van Renesse and Vogels



Reliability vs. Scalability

- ❖ Reliability guarantees come at a cost
- ❖ Can we scale better while providing reliability most of the time?



Epidemic protocols

- ❖ Try to “infect” all members in the group with new updates as fast as possible



Epidemic protocols

- ❖ **Infective:** server that holds updates and is willing to spread out
- ❖ **Susceptible:** server that has not been updated yet
- ❖ **Removed:** server that is not willing or not able to spread its update

Anti-entropy

- ❖ A server P picks up a server Q at random
 - P only *pushes* its own updates to Q
 - Spreads slowly
 - P only *pulls* in new updates from Q
 - Works better when most servers are infective
 - P and Q send updates to each other (*push-pull*)

Gossiping

- ❖ If P is updated with data item x , it will contact an arbitrary server Q and tries to push the updates to Q
- ❖ If Q has already got the update, P with a probability of $1/k$ would lose interest in spreading it further



Bimodal Multicast

- ❖ **Gossip-based protocol**

- ❖ **Two sub-protocols**
 1. Unreliable data distribution protocol
 2. "gap-repairing" protocol



Bimodal Multicast

❖ Unreliable data distribution protocol

- Upon arrival, a message enters the receiver's message buffer
- Messages are delivered to the application layer in FIFO order and are garbage collected out of the message buffer after some period of time



Bimodal Multicast

- ❖ “Gap-repairing” protocol – Repair gaps in the message delivery record
 - Each process in the system maintains a list containing some random subset of the full system membership
 - Prefers nearby processes
 - Each participant select one process randomly and send a digest message of its current message buffer contents periodically
 - Pull missing messages by sending retransmission solicitations
 - push missing messages by sending unsolicited retransmissions



Bimodal Multicast

- ❖ Gossip is primarily done to nearby processes over low-latency links
- ❖ Use “gossip pull” for “young” messages and “gossip push” for “old” ones
- ❖ Don't buffer every message at every process
 - Using a hash scheme to spread the buffering load around the system

❖ Reliability

- Tunable reliability
 - Increasing reliability by increasing the time length before a message is garbage collected
- Reliability guarantees are midway between SRM and virtual-synchrony



Bimodal Multicast

❖ Scalability

- Constant loads
- Each gossip round = 1 message sent + 1 message received (with high probability) + retransmit a bounded amount of data

Bimodal multicast

❖ Scalability

- Pbcast: bimodal multicast
- 8 processes

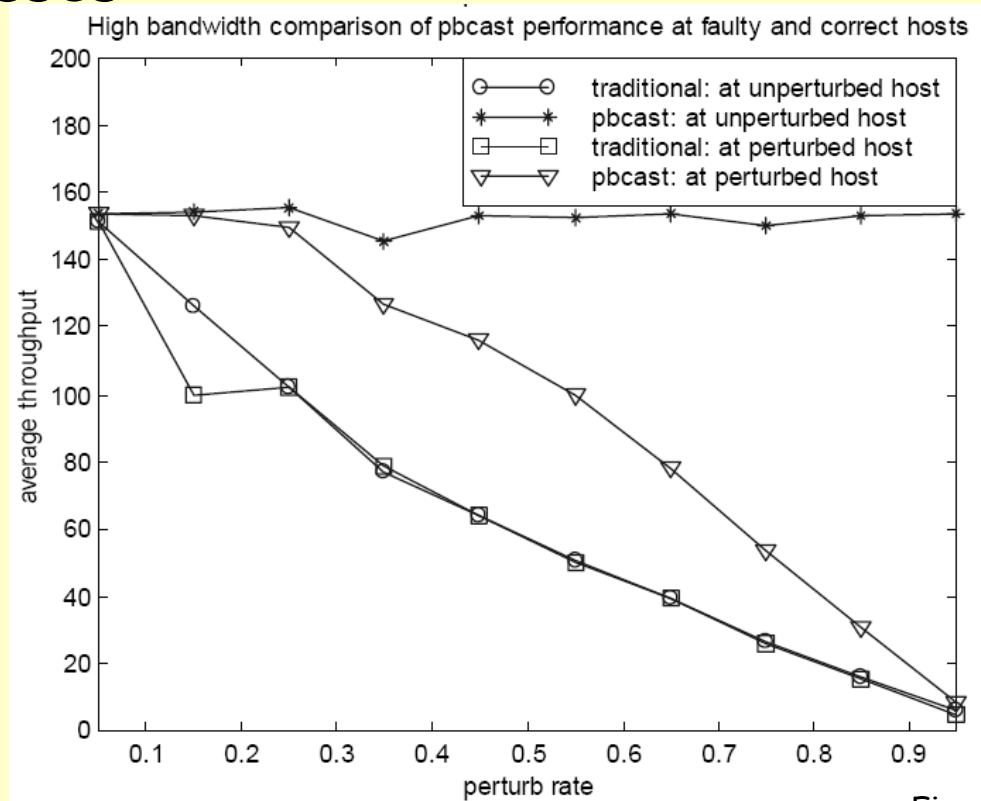


Figure from Birman et al.

❖ Scalability

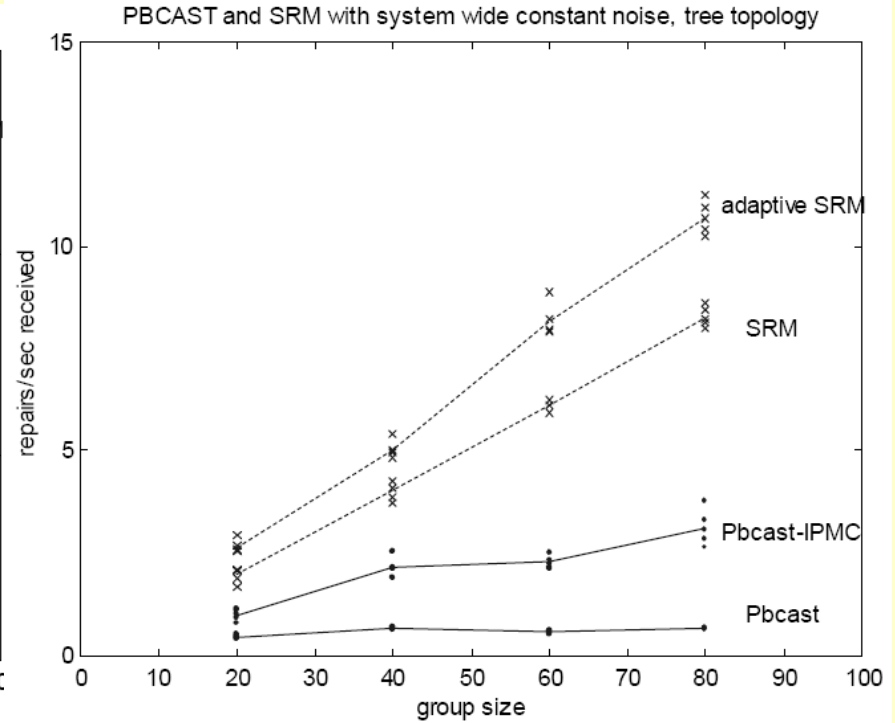
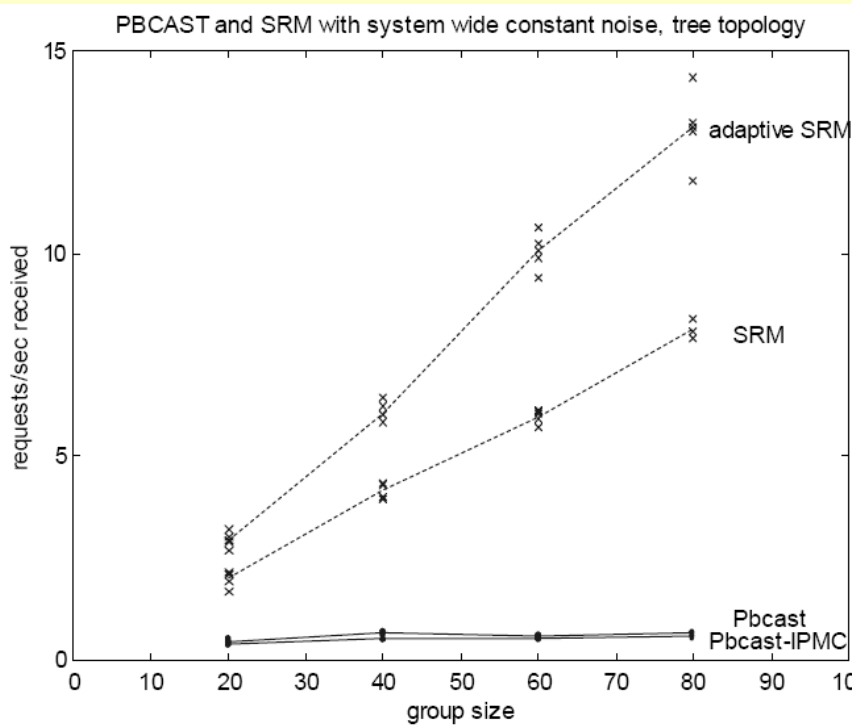


Figure from Birman et al.



Probabilistic Tools

- ❖ Bimodal multicast
- ❖ Astrolabe (hierarchical tables)
- ❖ Gravitational gossip (publish-subscribe)
- ❖ Anonymous gossip (mobile wireless device)

Conclusion

- ❖ SRM is a best-effort group communication protocol. Reliability is not guaranteed with faulty processes.
- ❖ Virtual synchrony is a reliable group communication protocol.
- ❖ Both SRM and virtual synchrony does not scale well
- ❖ Gossip-based protocols can provide good scalability while provide probabilistic reliability guarantees

References

- ❖ *Distributed Systems, Principles and Paradigms*, 2002 Edition: Andrew S. Tanenbaum and Maarten van Steen
- ❖ *Spinglass: Secure and Scalable Communications Tools for Mission-Critical Computing*, Kenneth P. Birman, Robbert van Renesse and Werner Vogels, International Survivability Conference and Exposition, DARPA DISCEX-2001, Anaheim, California, June 2001.
- ❖ *Bimodal Multicast*, Kenneth P. Birman, Mark Hayden, Oznur Ozkasap, Zhen Xiao, Mihai Budiu and Yaron Minsky, *ACM Transactions on Computer Systems*, Vol. 17, No. 2, May 1999, Pages 41-88.



Thank You !