Improving Scalability for Reliable Group Communication

Sam Shaojun Zhao

Outlook

- Group Communication
- Scalability Issue
- Previous Approaches
- Gossip-Based Protocol
- Security Issues
- Tools and Applications
- Conclusion

Group Communication

- Why **group**
  - Organizing several identical processes into a group for **fault tolerance**
- Property
  - When a message is sent to a group, all members receive it
  - One process fails, others will take over
- Groups are analogous to social organizations

Group Communication

- Flat vs. hierarchical groups
  - voting or coordinating
- Changing membership
  - join or leave
  - overhead

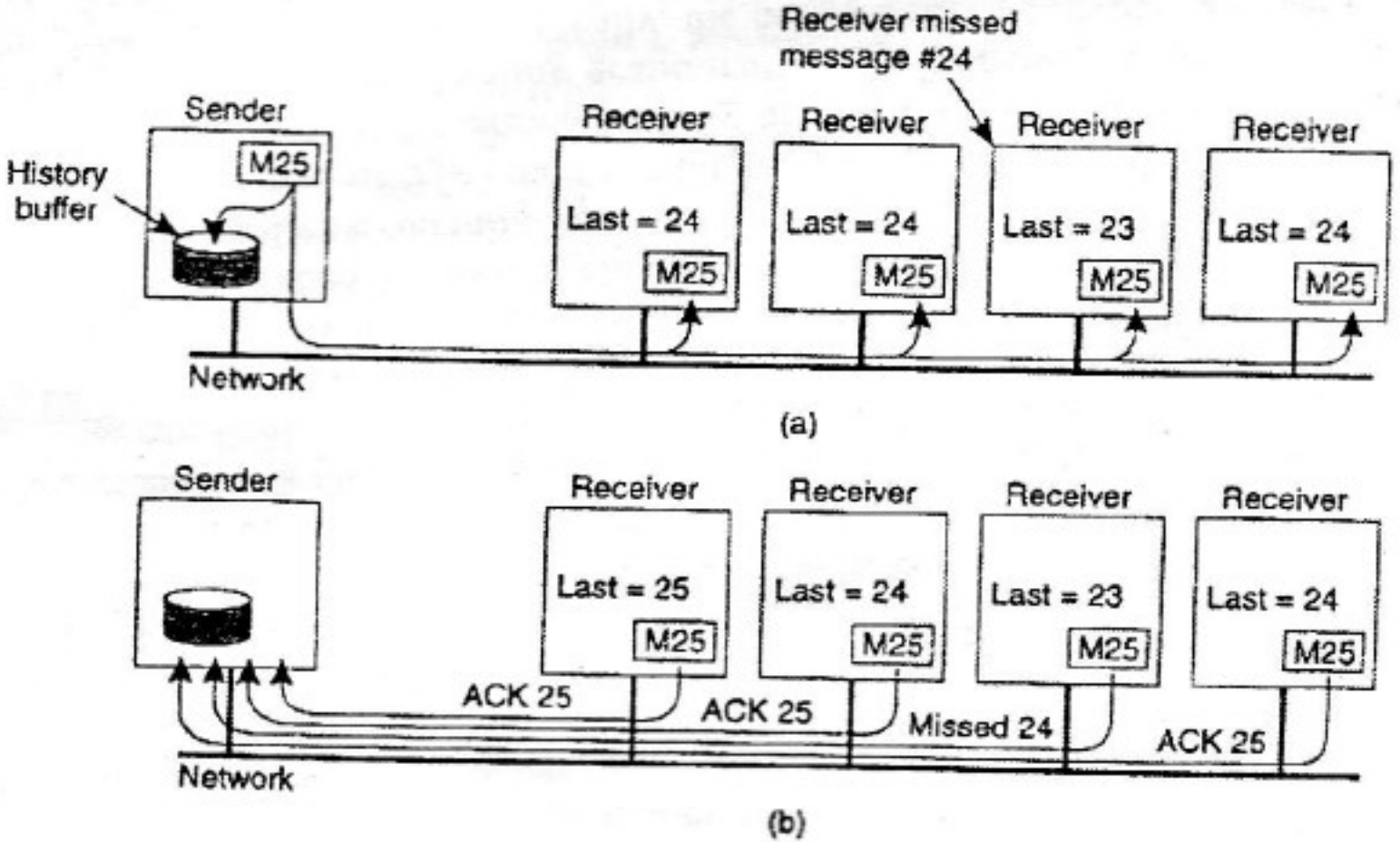


Figure 7-8. A simple solution to reliable multicasting when all receivers are known and are assumed not to fail. (a) Message transmission. (b) Reporting feedback.

The Naive Implementation

- Feedback Implosion
  - Reduce the number of feedback messages
- No abstraction

Scalable Reliable Multicasting

- No ACK
  - Feedback implosion is still possible
- Multicast NACK and ask for retransmission
  - Random delay

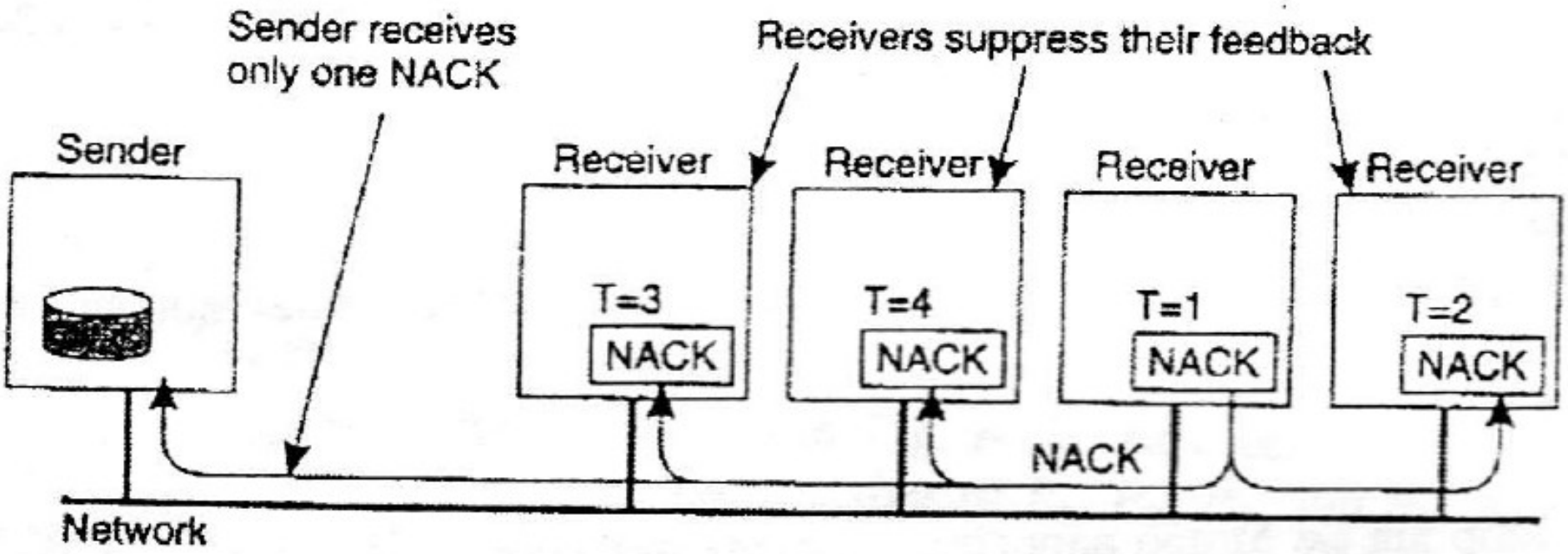


Figure 7-9. Several receivers have scheduled a request for retransmission, but the first retransmission request leads to the suppression of others.

SRM

- Problem
  - Random delay
    - Birthday paradox?
  - Overhead inside the group
  - No consistent group view
- Note
  - Local recovery
    - faulty process

Virtual Synchrony Model

- For all members, there is an agreement on the membership

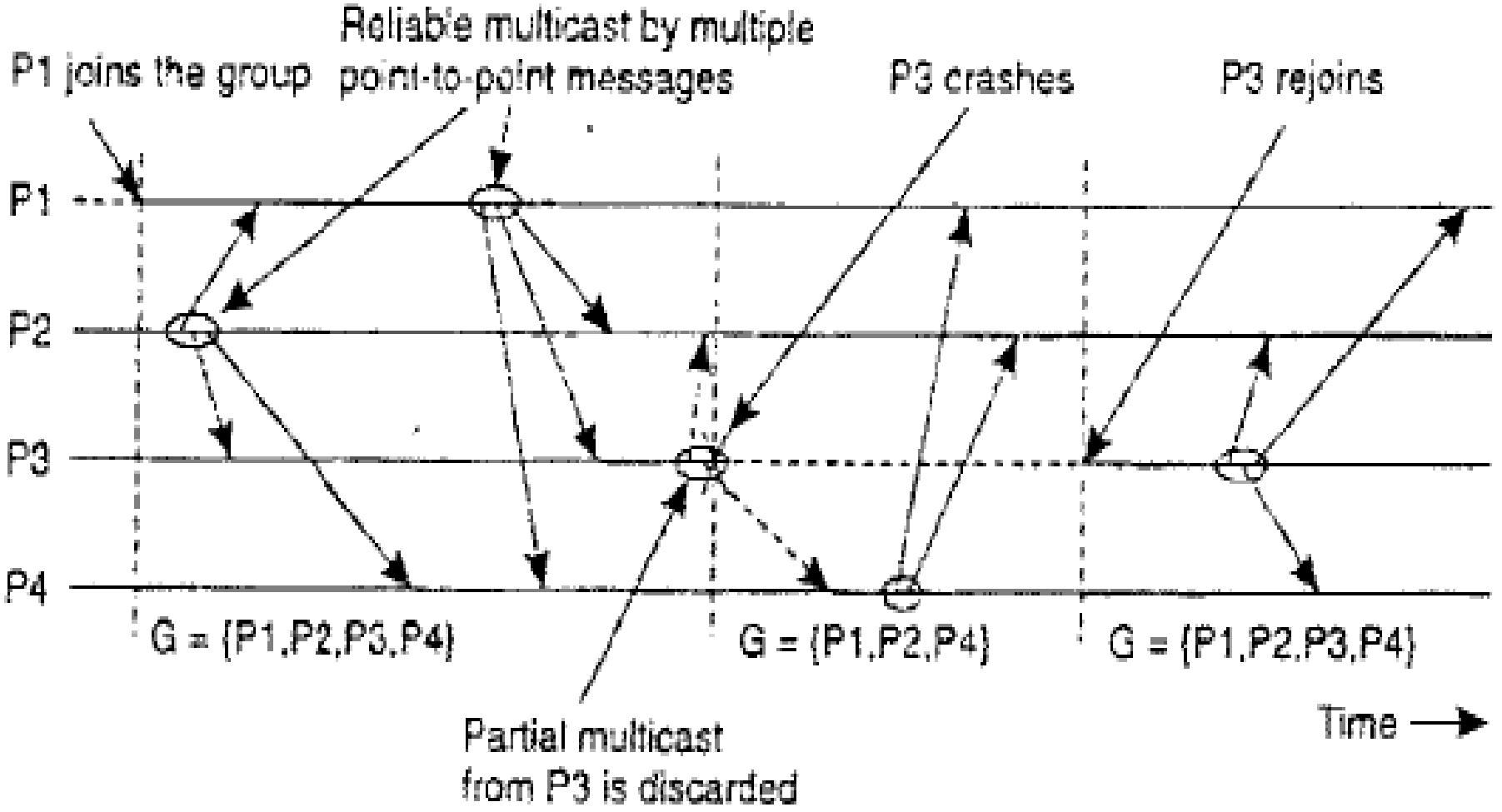


Figure 7-12. The principle of virtual synchronous multicast.

Message Ordering

- Un-ordered multicast
- FIFO-ordered
- Causally-ordered
- Totally-ordered

Reliability Group Communication

- Virtual Synchrony Model
  - A message is delivered to either all (non-faulty) processes or none
    - For consistency purpose
- atomic mulitcast problem
  - A message is delivered in the same **order** for all processes
- Implementation for VSM
  - let every process in a group keep message *m* until it knowns that all members have received it

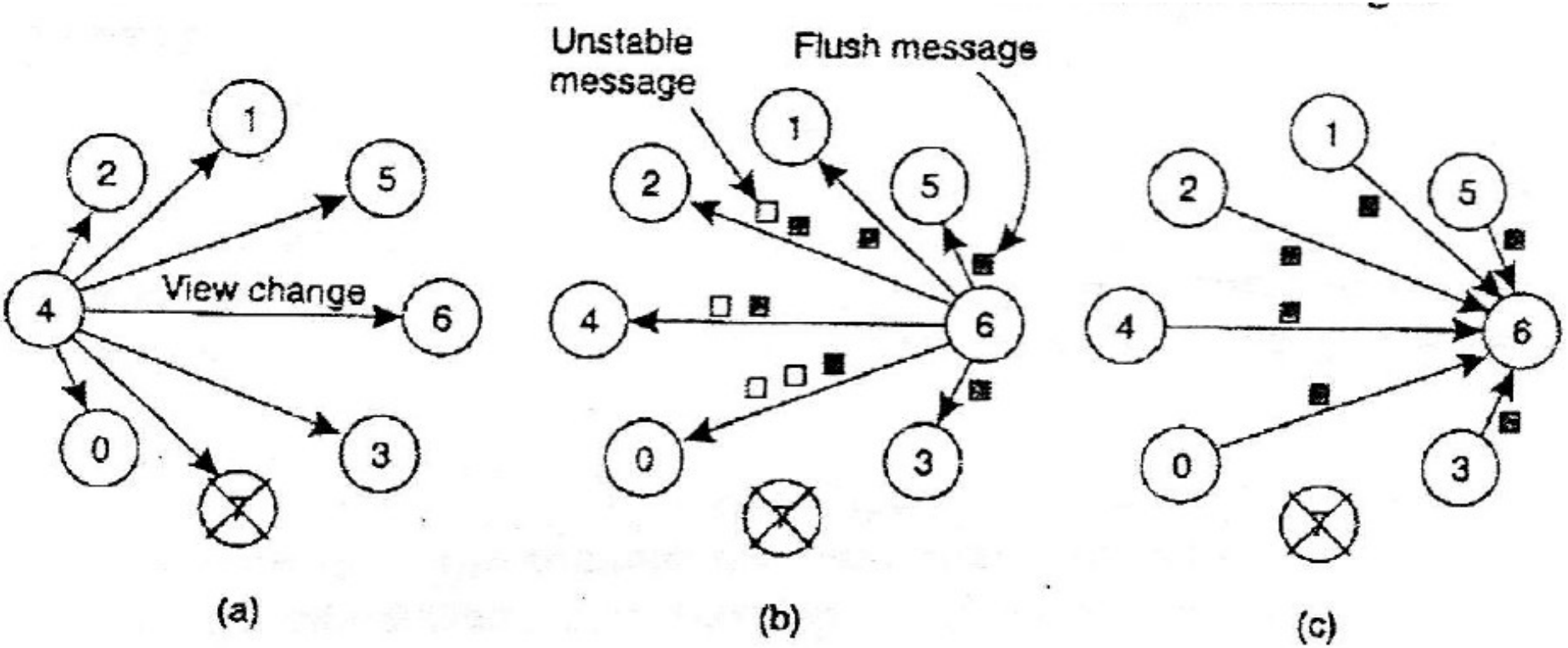


Figure 7-16. (a) Process 4 notices that process 7 has crashed and sends a view change. (b) Process 6 sends out all its unstable messages, followed by a flush message. (c) Process 6 installs the new view when it has received a flush message from everyone else.

Group Communication

- Issues
  - Scalability
  - Security

Scalability

- Perturbation example
  - Select a single group member, and force it to sleep for randomly selected 100ms intervals, with probability from 0 to 0.9
  - Note: between ideal and failure

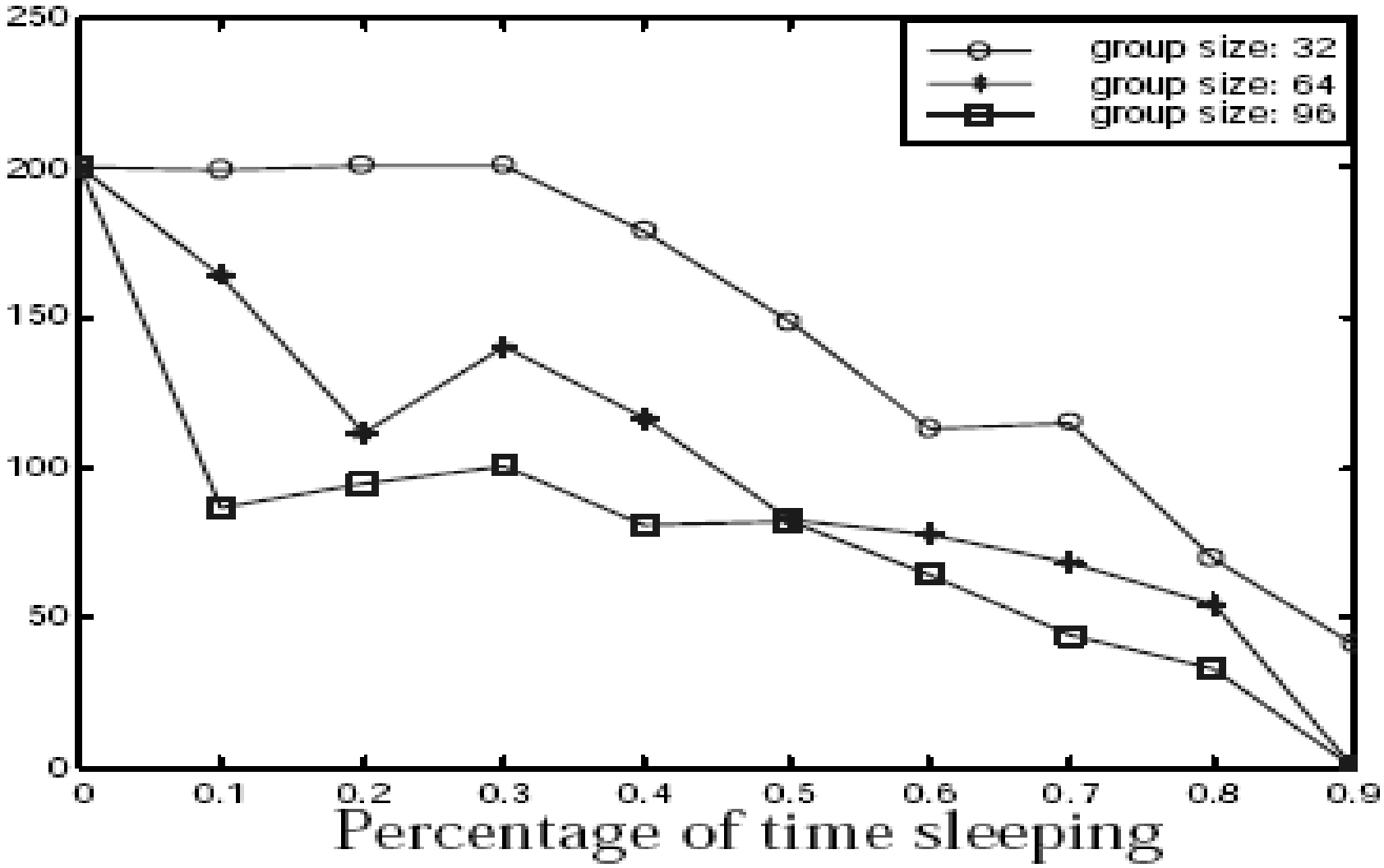


Figure 1: Sustainable throughput (multicasts/sec) drops as group size increases [3]

Previous Approaches

- Virtual Synchrony Model
  - Horus
- Receiver-Driven Model
  - SRM

Virtual Synchrony Model

- It offers **strong** fault-tolerance
  - Automated tracking of group membership
  - Reporting of membership changes to the members
  - Fault-tolerance multicast
  - Various ordering properties

Virtual Synchrony Model

- We just saw its performance
  - This model forces the sender to buffer messages until all members in the group acknowledge receipt
- Suggestion for improvement?

Virtual Synchrony Model

- Increase the sender side buffer size?
  - Have to increase the buffer size at least linear in the group size for scalability

Virtual Synchrony Model

- Knock out the slow receiver? (Micropartitions)
  - Have to adjust membership, and report the change to the members
  - “Thrashing” if considering rejoining

Virtual Synchrony Model

- Hierarchical Model?
  - Message “convoys”
  - Data becomes bursty

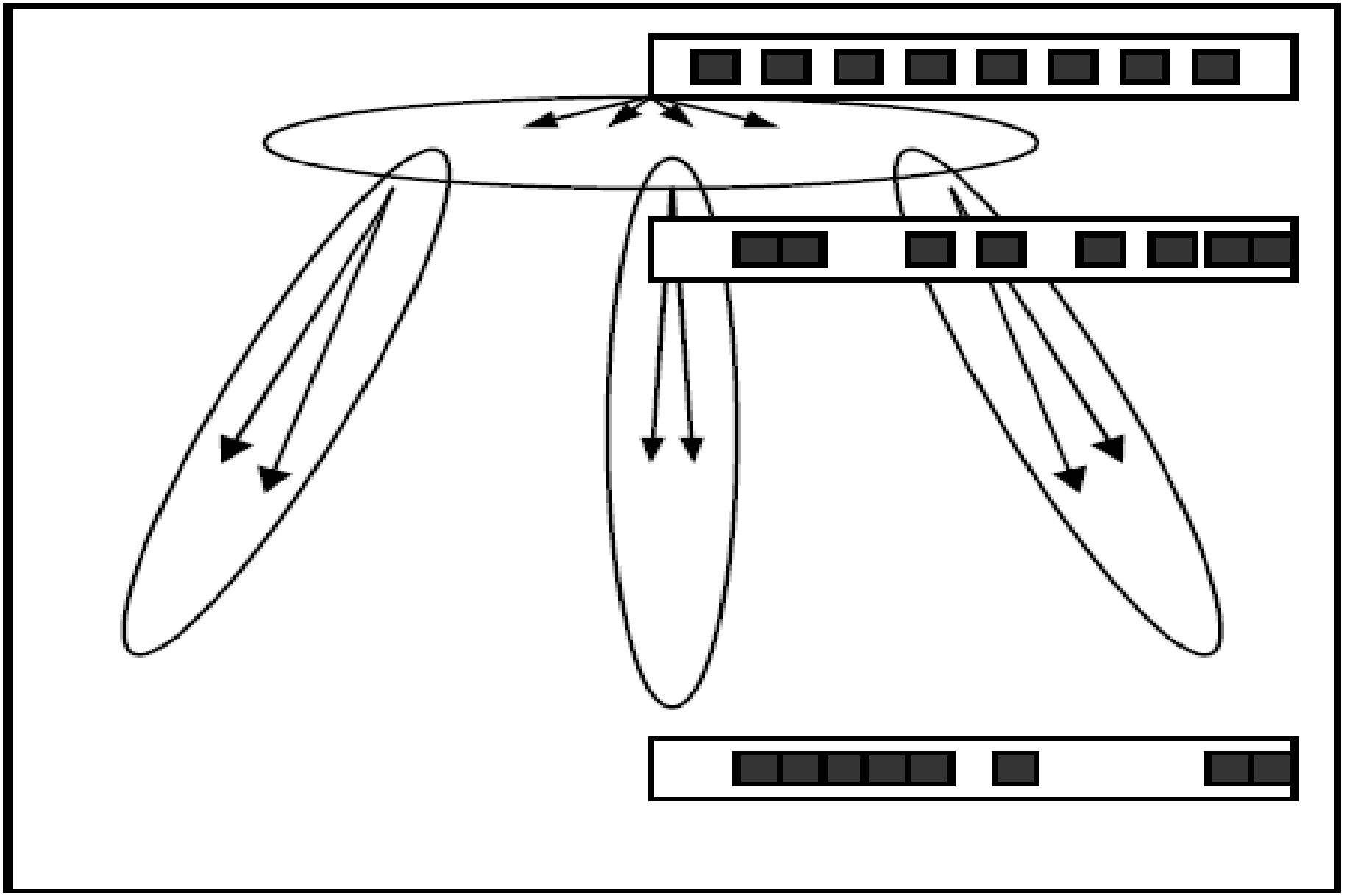
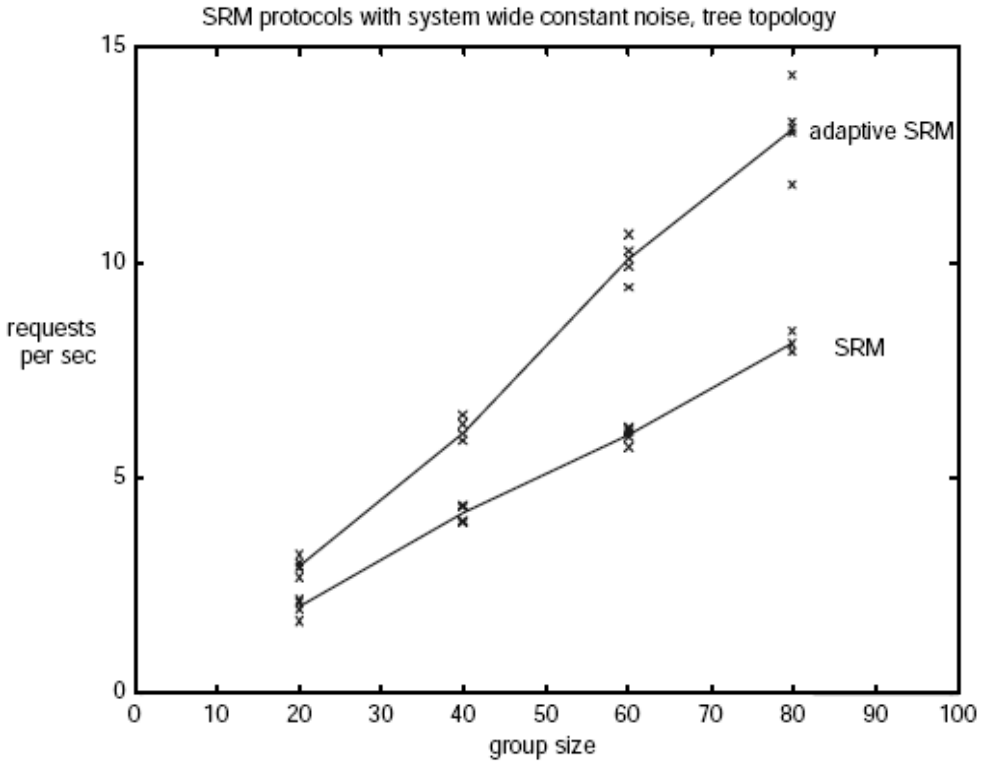


Figure 2: Message "convoys" often arise when groups are cascaded in a hierarchical manner

Receiver-Driven Model

- It offers weaker reliability
  - Receiver joins itself to the transmission group
  - Collects data
  - Requests retransmissions of mission data
- Better Scalability?



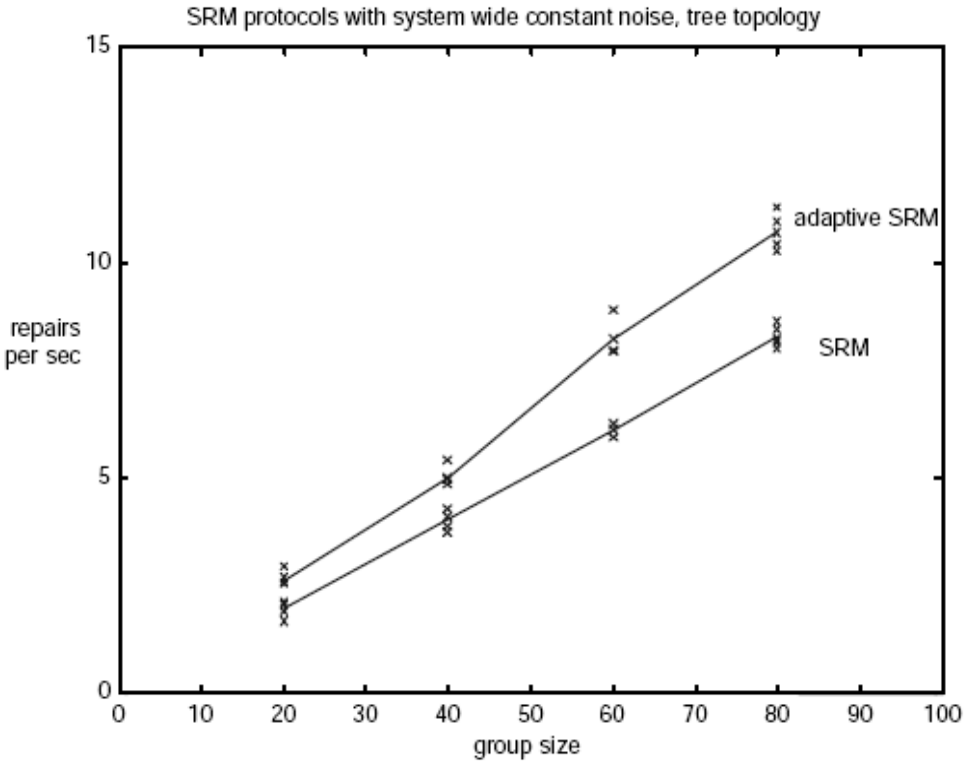


Figure 3: As the size of the group increases, a low level of background noise (0.1% in this case) can trigger high rates of requests (left) and retransmissions (right) for the SRM protocols. Most of these are duplicates. Notice that the data rate is being held constant; only the size of the group is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

Scalable Reliable Multicast

- As a network becomes large, the frequency of low probability events grows at least linearly with the size of the network
- As the network is scaled and the global frequency of these low probability events rises, one begins to observe growing numbers of requests for each multicast packet.

Conclusion so far

- Throughput degradation is not unique to reliability model
- Poor scalability arise mostly in technologies that provide reliability guarantees
- There is a trade off between scalability and reliability
  - *Providing reliability in a proper way*

Epidemic Protocols

- Bimodal multicast
  - Gossip-based protocol
  - Two sub-protocols

Bimodal multicast

- Unreliable data distribution protocol (vs. IP multicast)
  - 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
  - Each process in the system maintains a list containing some **random subset** of the full system membership
    - Prefers nearby processes
  - Sending **digest** message
    - pull/push missing messages

Advantage

- We do not buffer every message at every process
- A hash scheme is used to spread the buffering load around the system.
- Average message is buffered at **enough** processes to guarantee reliability
- Average buffering load on a participant **decreases** with increasing system size

Bimodal Multicast

- Impose **constant loads** on participants
  - During each gossip round, a process sends a single message, receives (with high probability) a single message, and may be asked to retransmit at most a bounded amount of data
- Tunable reliability
- Characteristics are preserved for larger system

Reliability

- Probabilistic guarantee
- Between the Virtual Synchrony Model and Scalable Reliable Multicast
- Virtual Synchrony over Bimodal Multicast

Bimodal Multicast

- Use gossip for
  - Multicast reliability
  - Tracking system membership

Tools

- Bimodal multicast
- Astrolabe (hierarchical tables)
- Gravitational Gossip (pub-sub)
- Anonymous Gossip (wireless)

Applications

- Joint Battlefield Infosphere
  - pub-sub

Security

- Session authentication
- Digital signatures

Conclusion

- The proposed approach is **scalable**
  - OSI enforces reliability and performs flow control low in the network
  - “inversion of ISO stack”
    - Low layers are gossip-based
    - Upper layers introduce stronger properties
  - Overcome infrequent disruptive problems with mechanisms having small, localized cost