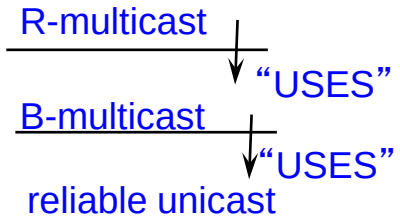
Lecture 29

- Administration

R-multicast



- Reliable multicast
  - Requirements: integrity, validity, *agreement*
  - Implementation:
    - *Received := {};*
    - ***R-multicast(g, m)** at process p: B-multicast(g, m);*
    - *On B-deliver(m) at process q*

      if(m ∉ Received)

      Received := Received ∪ {m};

      if(q ≠ p) B-multicast(g, m);

      R-deliver(m);

      end if

⇒ Inefficient: each message is sent |g| times to each process

Reliability

Correct processes: those that never fail.

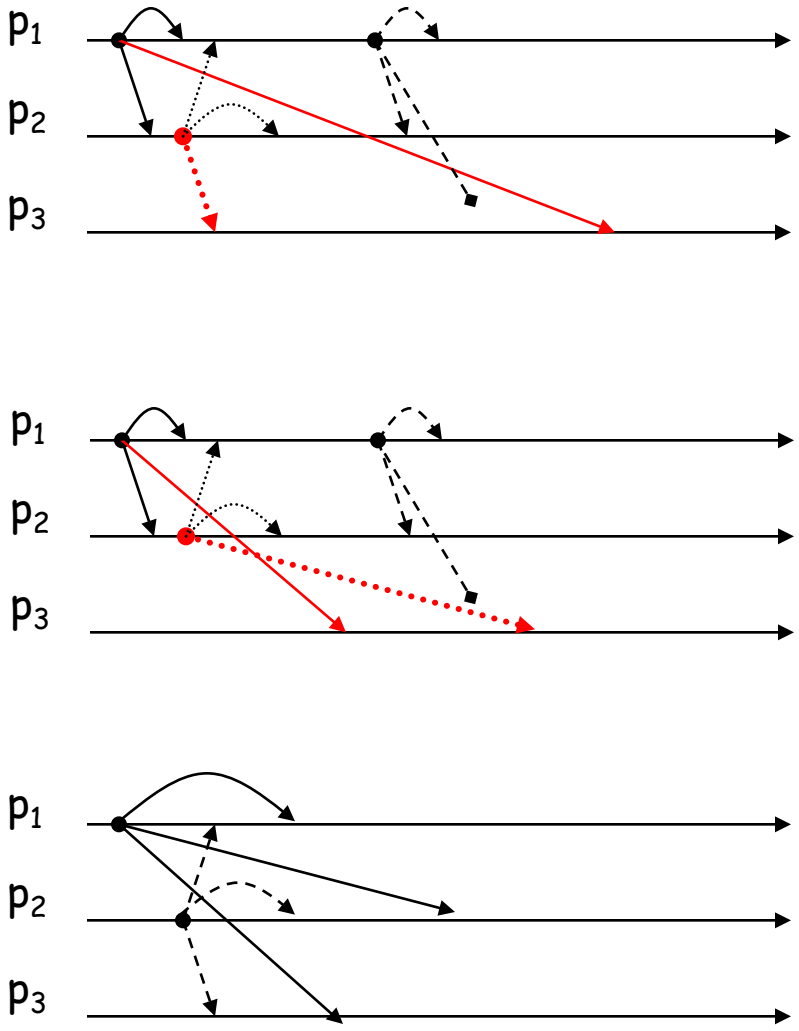
- Integrity
  - A *correct* process *delivers* a message at most once.
- Validity
  - A message from a *correct* process will be *delivered* by the process eventually.
- Agreement
  - A message *delivered* by a *correct* process will be *delivered* by all other *correct* processes in the group.

$\Rightarrow$ Validity + Agreement = Liveness

Ordered Reliable Multicasts

Assumptions: a process belongs to at most one group.

- FIFO
  - if $m_p \rightarrow m'_p$, all correct processes *that deliver* m'_p will deliver m_p before m'_p.
- Causal
  - if $m \rightarrow m'$, all correct processes *that deliver* m' will deliver m before m'.
- Total
  - if a correct process delivers m before m', all other correct processes *that deliver* m' will deliver m before m'.



Which multicast semantics? – Why?

item	From	Subject
23	A. Hanlon	Mach
24	G. Joseph	Microkernels
25	A. Hanlon	Re: Microkernels
26	T.L. Heureux	RPC performance
27	M. Walker	Re: Mach

Total Order Specification

- Validity

  If a correct process TO-broadcasts a message m, then it eventually TO-delivers m.

- Uniform Agreement

  If a process TO-delivers a message m, then all correct processes eventually TO-deliver m.

- Uniform Integrity

  For any message m, every process TO-delivers m at most once, and only if m was previously TO-broadcast by sender(m).

- Uniform Total Order

  If processes p and q both TO-deliver messages m and m', then p TO-delivers m before m', if and only if q TO-delivers m before m'.

Definition of Uniform

- A multicast is *uniform* if:

  If any process delivers the multicast, all group members that don't fail will deliver it (even if the initial recipient fails immediately after delivery).

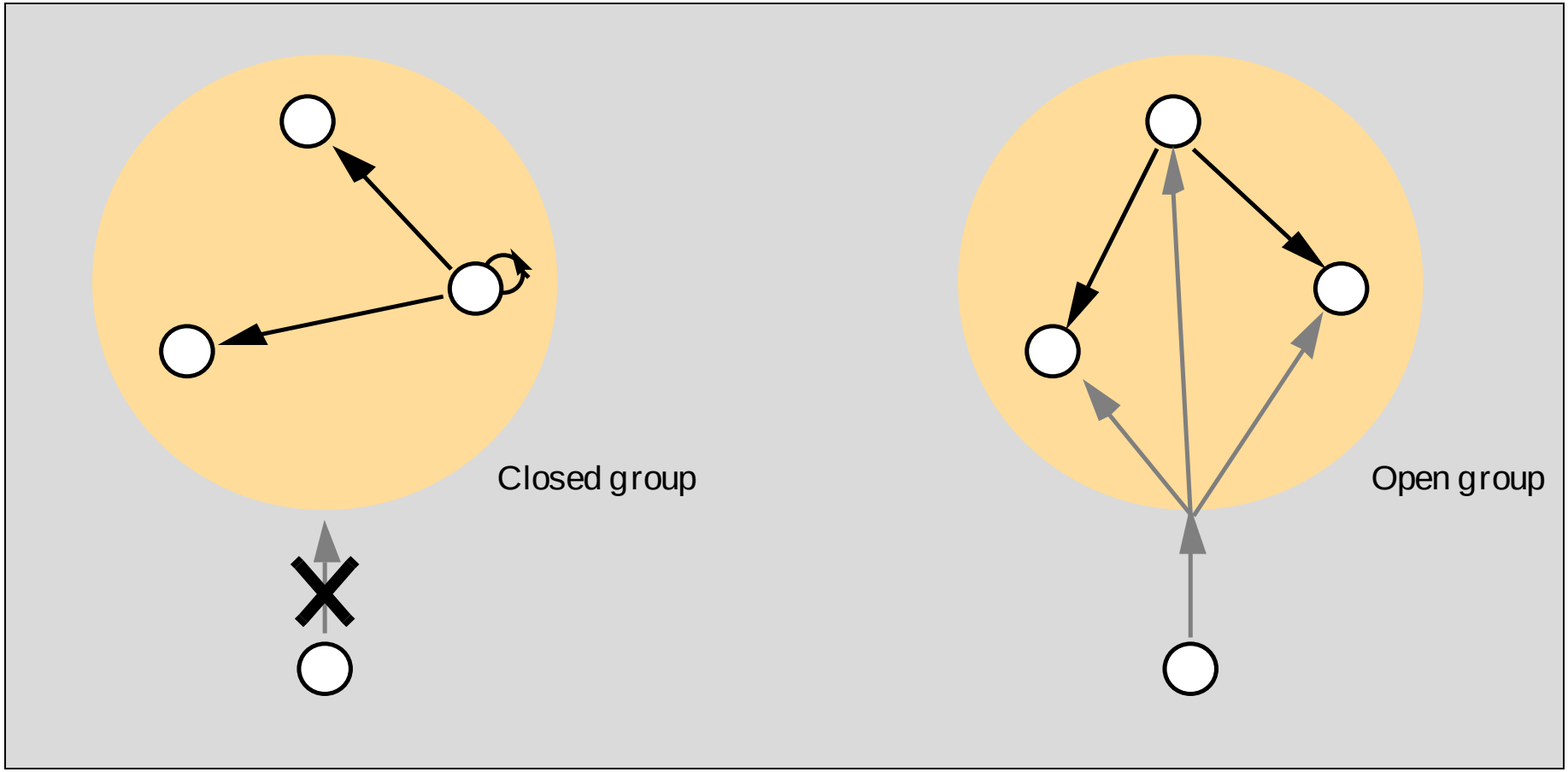
- Otherwise we say that the multicast is "not uniform"

Group Communication

Multicast Communication

- Group Communication is challenging even when all members of the group are static and aware of each other. Dynamic groups, with processes joining and leaving the group, are even more challenging. Most of the challenges are concerned with efficiency and delivery guarantees.
- Efficiency concerns include minimizing overhead activities and increasing throughput and bandwidth utilization.
- Delivery guarantees ensure that operations are completed.

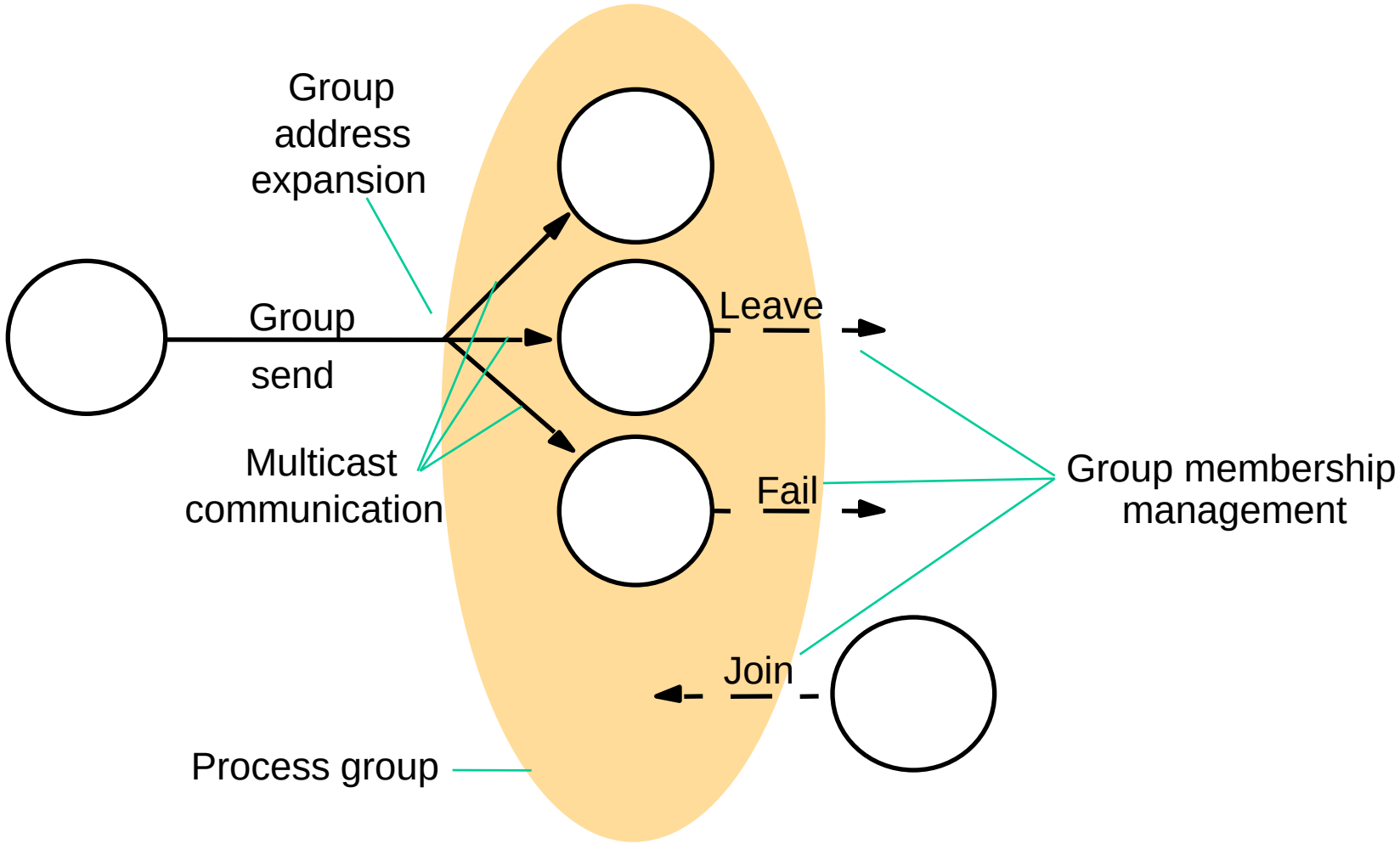
Open and closed groups



Closed and Open Groups

- For multicast communications, a group is said to be closed if only members of the group can multicast to it. A process in a closed group sends to itself any messages to the group.

- A group is open if processes outside the group can send to it. Some algorithms assume closed groups while others assume open groups.


Group
address
expansion
Group
send
Leave
Multicast
communication
Fail
Group membership
management
Join
Process group

Summary

- Multicast communication can specify requirements for reliability and ordering, in terms of integrity, validity and agreement
- B-multicast
  - a correct process will eventually deliver a message provided the multicaster does not crash
- reliable multicast
  - in which the correct processes agree on the set of messages to be delivered;
  - we showed two implementations: over B-multicast and IP multicast
- delivery ordering
  - FIFO, total and causal delivery ordering.
  - FIFO ordering by means of senders' sequence numbers
  - total ordering by means of a sequencer or by agreement of sequence numbers between processes in a group
  - causal ordering by means of vector timestamps
- the hold-back queue is a useful component in implementing multicast protocols