The University of Melbourne
School of Computing and Information Systems
COMP90020 Distributed Algorithms

Tutorial Week 7: Multicast

Notes

Exercises

30. How, if at all, should the definitions of integrity, agreement and validity for reliable multicast change for the case of open groups?

31. Consider the following algorithm from the textbook:

```
On initialization
   Received := {};

For process p to R-multicast message m to group g
   B-multicast(g, m);          // p ∈ g is included as a destination

On B-deliver(m) at process q with g = group(m)
   if ( m ∉ Received )
   then
            Received := Received ∪ { m } ;
            if ( q ≠ p ) then B-multicast(g, m); end if
            R-deliver m;
   end if
```

Consider reversing the order of the lines:

if (q ≠ p) then B-multicast(g,m) end if
R-Deliver m

(a) Why would this change make the algorithm no longer satisfy uniform agreement?

(b) Does the reliable multicast algorithm based on IP multicast satisfy uniform agreement?

32. Show that the FIFO-ordered multicast algorithm does not work for overlapping groups, by considering two messages sent from the same source to two overlapping groups, and considering a process in the intersection of those groups. Adapt the protocol to work for this case. Hint: processes should include with their messages the latest sequence numbers of messages sent to all groups.

33. In a group, there are eight members, of which at most one can crash at any time. You have no idea about which process can crash. The topology is a completely connected network.

(a) In the worst case, what is the smallest number of messages required to guarantee reliable atomic multicast from a given process to the entire group?

(b) How does this change if up to four members can crash at any time?

34. Consider the following algorithm from the textbook:

1. Algorithm for group member p

On initialization: $r_g := 0$;

To TO-multicast message m to group g
  B-multicast($g \cup \{sequencer(g)\}$, <*m, i*>);

On B-deliver(<*m, i*>) *with* $g = group(m)$
  Place <*m, i*> in hold-back queue;

On B-deliver(m_{order} = <"order", *i, S*>) *with* $g = group(m_{order})$
  wait until <*m, i*> in hold-back queue and $S = r_g$;
  TO-deliver m; // (after deleting it from the hold-back queue)
  $r_g := S + 1$;

2. Algorithm for sequencer of g

On initialization: $s_g := 0$;

On B-deliver(<*m, i*>) *with* $g = group(m)$
  B-multicast(*g*, <"order", *i*, s_g >);
  $s_g := s_g + 1$;

(a) Show that, if the basic multicast that we use in the algorithm is also FIFO-ordered, then the resultant totally-ordered multicast is also causally ordered.

(b) Is it the case that any multicast that is both FIFO-ordered and totally ordered is thereby causally ordered?

35. Consider a multiparty game of quiz involving five teams.

> *Each team poses a question and a member of another team has to answer it. The team that answers the question first scores a point. If no team can answer a question within 30s, then no one scores any point. The clocks are synchronized, so who answered first is decided by the time when the reply was posted. The teams can pose questions at any time and in no particular order.*

What kind of multicast is appropriate here?

36. [EXTENSION] Consider an election in a state. The citizens cast their votes at the individual polling centers. At the end of the day when the poll closes, counting begins. Each count recorded at a center is multicast to all the other centers, so that all of them exactly know the latest count at any time when the counting is in progress. The interconnection topology of the network connecting the polling centers is a completely connected graph.

(a) Assume that at any time failures can bring down some of the communication lines without creating a partition. What kind of multicast will guarantee that all centers are able to record every vote?

(b) Assume that communication failure partitioned the network for an hour. Apparently the counting must stop. What would you recommend so that the progress of counting is not affected even if the network temporarily partitions?