Reliable Broadcast

Exercise 1 What happens in the reliable broadcast and uniform reliable broadcast algorithms if the: (a) accuracy, (b) completeness property of the failure detector is violated?

Solution

Reliable Broadcast.

a Suppose *accuracy* is violated. Then, the processes might be relaying messages when this is not really necessary. This wastes resources but does not impact correctness.

b Suppose now *completeness* is violated. Then, the processes might not be relaying messages they should be relaying. This may violate *agreement*. For illustration, assume that only a single process p_1 *bebDelivers* (and, hence *rbDelivers*) a message m from a crashed process p_2. If a failure detector (at p_1) does not ever suspect p_2 no other correct process will deliver m (i.e., *agreement* is violated).

Uniform Reliable Broadcast. Consider a system of three processes p_1, p_2 and p_3. Assume that p_1 *urbBroadcasts* the message m.

a First, suppose that *accuracy* is violated. Assume that p_1 falsely suspects p_2 and p_3 to have crashed. p_1 eventually *urbDelivers* m. Assume that p_1 crashes afterwards. It may happen that p_2 and p_3 never *bebDeliver* m and have no knowledge about m: *uniform agreement* is violated.

b Suppose now *completeness* is violated. Then, p_1 might never *urbDeliver* m if either p_2 or p_3 crashes and p_1 never detects their crash or *bebDelivers* m from p_2 and p_3. Hence, p_1 would wait indefinitely for p_2 and p_3 to relay m. In this case: *validity* property is violated.

Exercise 2 Implement a reliable broadcast algorithm without using any failure detector (i.e., using only *BestEffort-Broadcast* (beb)).

Solution We can circumvent the need for a failure detector in our reliable broadcast algorithm by adapting the following scheme: every process that gets a message relays it immediately. Recall that in the original algorithm, processes were relaying messages from a process p only if p crashes.

Exercise 3 Assume a majority of processes is correct. Modify the uniform reliable broadcast algorithm presented in the class, such that it *does not* use any failure detector.

Solution In the slide 35, substitute $\langle correct \subset ack[m] \rangle$ with $\langle |ack[m]| > N/2 \rangle$, where N denotes the total number of processes.

Exercise 4 The reliable broadcast algorithm presented in the class has the processes continuously fill their different buffers without emptying them. Modify it to remove unnecessary messages from the buffers: (a) *from*[p_i], and (b) *delivered*.

Solution

a The array *from* is used only to store messages that are relayed in the case of a failure. Therefore, messages from the array *from* can be removed as soon as they are relayed.

b Messages from the buffer *delivered* cannot be removed. If a process crashes and its messages are retransmitted by two different processes, then a process might *rbDeliver* the same message twice if it empties the *delivered* buffer in the meantime. This would violate the *no duplication* property.