Exercise Session 7
Consensus (part II) – Solutions

Problem 1

Give the four properties of consensus. Give four executions, each of which violates exactly one of the consensus properties.

Answer:
C1. Validity: Any value decided is a value proposed.
Validity violation: \( p_1 \) and \( p_2 \) propose 1. \( p_1 \) and \( p_2 \) decide 0.
C2. Agreement: No two processes decide differently.
Agreement violation: \( p_1 \) proposes 1 and \( p_2 \) proposes 0. \( p_1 \) decides 1 and \( p_2 \) decides 0.
C3. Termination: Every correct process eventually decides.
Termination violation: \( p_1 \) proposes 1 and \( p_2 \) proposes 0. \( p_1 \) decides 1 and \( p_2 \) never decides.
C4. Integrity: No process decides twice.
Integrity violation: \( p_1 \) proposes 1 and \( p_2 \) proposes 0. \( p_1 \) decides 1. \( p_2 \) decides 1 twice.

Problem 2

Algorithm 1 implements a consensus protocol using a perfect failure detector and best effort broadcast (beb). Assume you have to change this Algorithm 1 in order to obtain a uniform consensus protocol.

Explain these changes and rewrite the algorithm accordingly.

Answer:
In Algorithm 1, a process decides in its corresponding round. First of all, we make the processes only broadcast their current value and not decide on its corresponding round. Secondly, the processes decide after exactly \( n \) round to ensure the uniform consensus. So, we go to the next round till \( \text{round} == n \) and the process has not decided yet (indicated as a new parameter \text{decided} ). The changes are made in Algorithm 2.
Algorithm 1 Consensus Using a Perfect Failure Detector and Beb

Upon event < Init > do
1: suspected = Ø
2: round = 1
3: currentProposal = nil
4: broadcast = false
5: delivered[] = false

Upon event < Crash, pi > do
1: suspected = suspected ∪ {pi}

Upon event < Propose, v > do
1: if currentProposal == nil then
2: currentProposal = v
3: end if

Upon event < bebDeliver, p_round, value > do
1: currentProposal = value
2: delivered[round] = true

Upon event delivered[round] == true or p_round ∈ suspected do
1: round = round + 1

Upon event p_round == self and broadcast == false and currentProposal ≠ nil
1: trigger < Decide, currentProposal >
2: trigger < bebBroadcast, currentProposal >
3: broadcast = true
Algorithm 2 Uniform Consensus Using a Perfect Failure Detector and Beb

Upon event < Init > do
1: suspected = Ø
2: round = 1
3: currentProposal = nil
4: decided = false
5: broadcast = false
6: delivered[] = false

Upon event < Crash, p_i > do
1: suspected = suspected ∪ {p_i}

Upon event < Propose, v > do
1: if currentProposal == nil then
2: currentProposal = v
3: end if

Upon event < bebDeliver, p_round, value > do
1: currentProposal = value
2: delivered[round] = true

Upon event delivered[round] == true or p_round ∈ suspected do
1: if round == n and decided == false then
2: trigger < Decide, currentProposal >
3: decided = true
4: else
5: round = round + 1
6: end if

Upon event p_round == self and broadcast == false and currentProposal ≠ nil
1: trigger < bebBroadcast, currentProposal >
2: broadcast = true