Homework 2

Kunal Lad (UT EID: KL28697)
CS 388D - Distributed Computing

October 28, 2015

Slip days used(This assignment):0 Total Slip days used(for assignments):1

Problem 1. Given : Synchronous system that tolerates f crash failures and an algorithm that solves consensus in f+1 rounds for this system.
To Solve: Terminating Reliable Broadcast in f+2 rounds for this system.

Solution. Basic idea of our protocol is that sender broadcasts a message m in the 1st round. All processes then wait for receipt of m till end of round 1. After that all the processes run consensus algorithm with proposed value either m(if they received it) or SF(if they didn't receive it). After consensus algorithm terminates they deliver whatever is decided by consensus algorithm.

TRB Protocol:

1. If (p == Sender) broadcast message m.
2. If Received m, then set v = m
   Else set v = SF
   Propose v to Consensus Algorithm
3. Set v = Decision of Consensus Algorithm
   Deliver(v)

NOTE : Consensus Algorithm is started in step 2 and deliver(v) is executed in same round in which decision of consensus is reached. Thus total number of rounds is:
1(For Step1) + f+1(Execution of Consensus Algo and delivery of message) = f+2

Proof of Correctness:

- **Termination** Every correct process:
  1. Reaches round f+2 because every correct process terminates in Consensus Algorithm.
  2. Delivers m or SF whatever is decided by Consensus (Because correct processes don't crash they will be able to deliver the message in round f+2)

- **Integrity**

  1. **Every correct process delivers at most one message:**
     Deliver() is executed only once in step 3 and since it delivers only 1 message, at most one message is delivered by all correct processes.
  2. **Only if it was broadcasted:**
     Suppose m $\neq$ SF was delivered by some correct process.

     $\implies$ m was decided by Consensus Algorithm (Because all correct processes deliver the value decided by consensus)

     $\implies$ m was proposed by some process (By Integrity of Consensus)

     $\implies$ m was received by some process in step 1(By Protocol design a process proposes m in the Consensus only if it received m in step 1)

     $\implies$ m was broadcasted by some process in step 1(By uniform integrity of send and receive)

- **Validity** If a correct process broadcasts m.

  $\implies$ All process alive at the beginning of round 2 receive m (By uniform integrity of send and receive and synchronous model).

  $\implies$ All participants of Consensus propose m (By TRB Protocol design).

  $\implies$ All correct processes decide m in Consensus (By Validity of Consensus).

  $\implies$ All correct processes deliver m. (By TRP Protocol design)

- **Agreement** If a correct process delivers m.

  $\implies$ Decision of Consensus was m. (Since by protocol if a process delivers then it delivers whatever is decided by Consensus)

  $\implies$ All correct processes deliver m. (Since by protocol if a process delivers then it delivers whatever is decided by Consensus)

Problem 3 a. Protocol : Given a k-bit binary input, use k parallel instances of binary consensus (one for each bit) and decide the value to be equal to $d_1, d_2, \cdots, d_k$ where d_i is the decision of i-th instance of binary consensus.

Solution. Proposed protocol does not solve consensus. We prove this by giving a counter example in which INTEGRITY is not satisfied by above protocol.

Consider a scenario in which there 2 processes p_1 and p_2 (both correct) are participating in the protocol and the protocol is using the binary consensus algorithm which uses Min function to decide on the values (Consensus Algorithm discussed in class).

Then the consensus algorithm will decided 0 for both the bits. Thus the value decided by our protocol will be 00 by both the processes. This violates INTEGRITY property of consensus as no process had proposed 00 but all correct processes decided 00.

Problem 3 b. Protocol :

1. Each process broadcasts its value v.
2. Each process proposes a k-bit binary value which identifies its id uniquely.
3. k-instances of binary consensus (one for each bit) are used to decide on value of each bit.
4. Every process waits for delivery of the value from the process with id $d_1, d_2, \cdots, d_k$ where d_i is the value decided by binary consensus for i-th bit.
5. Each process decides the received value in above step.

Solution. Above protocol does not solve consensus. We prove this by giving counter example.

1. **Number of participating processes is not a perfect power of 2 :**

   In this case the process id chosen by the binary consensus on k-bit process ids may not exist. Thus the processes(who have not crashed) will time out in step 4 and will receive $\perp$.

   Eg: Consider the scenario in which 3 correct processes are participating in the protocol and protocol uses a binary consensus algorithm which uses Max function to decide on the proposed value. Then if the processes proposed the following values:

Process	ProposedId	Broadcasted Value(v)
p_0	00	00
p_1	01	01
p_2	10	10

   In this eg the process id chosen by binary consensus will be 11, but process 11 does not exist. Hence all the processes will timeout waiting for delivery of v from process with id 11.

2. **Number of participating processes is a perfect power of 2 :**

   In this case the process id chosen by the binary consensus on k-bit process ids may be faulty. Thus the processes(who have not crashed) will time out in step 4 and will receive $\perp$.

   Eg: Consider the scenario in which 3 correct processes(p_0, p_1, p_2) and 1 faulty process(p_3) are participating in the protocol and protocol uses a binary consensus algorithm which uses Max function to decide on the proposed value. Then if the processes proposed the following values:

Process	ProposedId	Broadcasted Value(v)
p_0	00	00
p_1	01	00
p_2	10	00
p_3	11	11

   In this eg the process id chosen by binary consensus will be 11, but process p_3 is faulty. Hence all the processes will timeout waiting for delivery of v from process p_3.