Distributed Computing Problem Set #2

Kai Sun (孙锴)
5110309061

Problem 1

In round 1:

Sender sends m to all processes

In round 2 to round f+2:

Every process runs the consensus algorithm which tolerates f crash failures to make a decision, and delivers it in round f+2.

Termination: By the termination of consensus, we know that the consensus will make a decision eventually. And we know that the decisions are what the processes deliver in round f+2, so every correct process delivers eventually.

Agreement: By the agreement of consensus, we know that the decision made by the consensus is the same, and we know that the decisions are what the process delivers in round f+2, so every correct process delivers the same message eventually.

Validity: If the sender is correct and broadcasts a message m, then all processes propose m. By the validity of consensus, all correct processes eventually decide m. And we know that the decisions are what the process delivers in round f+2, so every correct process eventually delivers m.

Integrity: We can see that process delivers only in round f+2, so every correct process delivers at most one message. And if it delivers m≠SF, then the decision made by the consensus must be m and by integrity of consensus, some process must have proposed m. So we know that m must have been broadcasted by some processes.

Problem 2

In round 1:

Sender sends m to all process

In round 2:

Sender sends m to all process

Every process records whether it receives the message from the sender.

In round 2 to round f+1:

Every process i runs the consensus algorithm which tolerates f-1 crash failures to get a decision Di.

In round f+1:

If process i received the message m from the sender in round 2, then it delivers the message(m).

If process i did not receive the message from the sender in round 2, then it delivers Di.

The idea is that if SF happens in round 1, then it is clear that the problem can be solved with consensus algorithm which only tolerates f-1 crash failures. If SF does not happen in round 1, then in round 2, if process i receives the message from the sender, it knows that every process has received the message in round 1 so process i can deliver m; if process i does not receives the message from the sender, then it knows that there exists a failure in SF, so it knows the TRB can

be solved with consensus algorithm which only tolerates f-1 crash failures. So in both cases, the problem can be solved with consensus algorithm which only tolerates f-1 crash failures.

The main difference between the algorithm in problem 1 and the algorithm in problem 2 are the number of failures which the consensus algorithm can tolerate. So the proof here is just the same as the proof of Termination, Agreement, Validity and Integrity in problem 1 except that we need to consider two cases:

(1) Process i received the message from the sender in round 2.

(2) Process i did not receive the message from the sender in round 2.

Here we illustrate the change of the proof by Termination. The change of Agreement, Validity and Integrity is just similar.

Termination: By the termination of consensus, we know that the consensus will make a decision eventually. What the process i delivers are either (1) those decisions(Di) or (2)the message m which is received in round 2 , which depends on whether process i received the message from the sender in round 2. And we know that either those decisions or Di are what the processes deliver in round f+1, so every correct process delivers eventually.

Problem 3

1. algorithm:

```
In round 1, sender sends m to all processes
For every processes R in round i where 1<=i<=t+1 do
    If R receives some message m in round i-1
        R sends m to all processes
        Goto label_end
    End
End
Label_end:
Do nothing until round t+2
If R has received some message m in some rounds
    Send m to all processes
Else
    Send SF to all processes
End
If R received a majority of some message m
    Deliver m
```

Proof:

It is easy to see that for the first t+1 rounds, the only difference between the above algorithm and the t+1-round TRB algorithm is that the above algorithm does not deliver the message immediately. So we can draw the following properties of the above algorithm from the proof that the t+1-round TRB algorithm satisfies Termination, Validity, Agreement and Integrity for general omission failures:

(1) Every correct process sends a message in round t+2

(2) If a correct sender S broadcasts a message m, then S sends m in round t+2

(3) If a correct process sends a message m in round t+2, then all correct processes send m in

round t+2

(4) If a correct process sends a message m(m is not SF) in round t+2, the sender broadcast m.

With the above properties, we can proof Termination, Validity, Uniform Agreement and Integrity as follows:

Termination: With property (1) and (3), we know that every correct process sends the same message m in round t+2. And we know n>2t, then every correct process receives a majority of m and delivers m.

Validity: If a correct sender broadcasts a message m, then from property (2) and (3) we know that all correct processes send m in round t+2. And we know n>2t, so the correct processes will receive a majority of m and deliver m.

Uniform Agreement: If some process delivers a message m in round t+2, then message m must be sent by a majority of processes without omissions, so all correct processes will receive a majority of m and deliver m.

Integrity: If a correct process delivers a message m(m is not SF), that message must be sent by a majority of processes in round t+2 and at least one of them is correct(Because n>2t). From (4) we know that m is broadcasts by the sender.

2. We prove it by contradiction.

Assume that there exists an algorithm. We partition the processes into two groups A and B and each group is of size at most t. WLOG, we assume the sender is in A. Now we consider two runs:

Run 1: Every process of A crashes and every process of B is correct. Then we know that all processes in B deliver SF.

Run 2: Every process of A is correct and omission failure occurs for all messages from A to B. Then we know that all processes in B deliver SF because for each process of B, it can not distinguish the difference between Run1 and Run2. Then we know that every process of A also deliver SF by Uniform Agreement. But this circumstance violates Integrity!

Problem 4

Termination: From lemma 2 and lemma 3 shown in class we know that Vp contains at least one value which is not $\perp$, so if a process reaches phase 3, it will decide a value. And it is clear that if a process does not reach phase 3, it must be stuck in line 6 and line 12. But that cannot be true because our failure detector satisfies strong completeness.

Agreement: By lemma 1 and lemma 2 shown in class, we know that there are not two correct processes which decide differently.

Validity: From phase 3 we know that the decision is made from Vp's first non $\perp$ value. And from line 8 we can see that all non $\perp$ values in Vp are proposed by some processes.

Problem 5

I provide the following counterexample:

Assume there are 5 processes and the maximum number of faults which can be tolerated is 2.

Round 1:

	P1	P2	P2	P4	P5
Phase 1	a=1	a=1	a=1	a=0	a=0

Phase 1	Receive 1 1 1 b=1	Receive 1 1 1 b=1	Receive 1 1 1 b=1	Receive 1 0 0 b=⊥	Receive 1 0 0 b=⊥
Phase 2	Receive 1 1 1 a=1 **decide 1**	Receive1⊥⊥ a=0 (0 is chosen randomly)	Receive1⊥⊥ a=0 (0 is chosen randomly)	Receive1⊥⊥ a=0 (0 is chosen randomly)	Receive1⊥⊥ a=0 (0 is chosen randomly)

Round 2:

	P1	P2	P3	P4	P5
Phase 1	Have decided 1	a=0	a=0	a=0	a=0
Phase 1		Receive 0 0 0 b=0	Receive 0 0 0 b=0	Receive 0 0 0 b=0	Receive 0 0 0 b=0
Phase 2		Receive 0 0 0 a=0 decide 0	Receive 0 0 0 a=0 decide 0	Receive 0 0 0 a=0 decide 0	Receive 0 0 0 a=0 decide 0

So after the above 2 rounds, not all processes decide the same value, which does not satisfy the agreement.