CSE 586: DISTRIBUTED SYSTEMS
COURSE PROJECT 2

UB ID: Anjali Bachani (abachani)(50336863)
Lakshmi Chaitanya Chapala
(lakshmic)(50290974)

Agreement Property:

Agreement property states that no two nodes/processes can decide two different values. Agreement property should be satisfied at all rounds even when we have more failure nodes than half of the nodes present in the model. This can be satisfied as when failure nodes are more than working nodes then it won't be able to make a decision as majority is required to propose a p2v value. So, failure nodes will have -1 decision value as they won't be able to take a decision and hence still the agreement property will be satisfied as decision value for them is same.

Progress Property:

To attain a progress property, all processes should come to a decision. Having same p1v values for the nodes will give guaranteed progress property as all nodes see that majority value and propose it to p2v and hence all node make a decision in phase 2 for that round. On the other hand, if we have different values for p1v for the nodes, it may not have the progress property as nodes might not see a majority value to propose to p2v and hence p2v gives the default -1 value to the next phase and decision may be postponed for any amount of time so this will lead to progress property to be violated.

Minority Report:

Minority Report claims that it is not possible for the nodes to decide a value that is not in majority in the given initial input. In some scenarios, model checker can show a minority consensus value in the input values, this happens when there is one faulty node, so model might not find any majority in phase 1 and propose default -1 value to the next round, now while selecting the random values for the nodes, there might be a case when model might give the value that was in minority to the nodes, so even when the faulty node comes back online, it sees the majority value and proposes that value to the next phase so we end up having all the nodes with the value which was initially a minority in the input set.

So, when we have a faulty node in our model then we may violate the Minority Report, as random values selected could be a minority value from the input set, leading to that value being a majority in later phase.

Task-1: Checking our specification for scenario- Agreement and Progress Tests

Scenario 1: F <- 0| N<-4 | Input <- <1,1,1,1> | MAXROUND <-4

No Errors

Explanation: In this scenario, we again have same values in the Input set. So, we have all values of “1” values, so “1” is proposed to the next phase and hence all the nodes/processes are able to make a decision and also there is no failure so all the nodes reach same decision because all nodes propose same value, so Agreement and Progress property are maintained.

Scenario 2: F <- 0 | N<-4 | Input <- <0,0,0,0> | MAXROUND <-4

No Errors

Explanation: Both Agreement property and progress property are not violated. Progress property is usually violated when we have different values in the Input Set, which is not the case in this scenario. And for agreement property, no two nodes/processes decide two different values and it is also maintained in this scenario. So, all the nodes will make same decision for all states and will end up coming to the consensus so properties are not violated in this scenario.

Scenario 3: F <- 0 | N<-4 | Input <- <0,0,1,1> | MAXROUND <- 4

Temporal Properties were violated (Progress)

States:

```
1 /\ decided = <<-1, -1, -1, -1>>
/\ p1Msg = { <<1, 1, 0>>,
  <<1, 2, 0>>,
  <<1, 3, 0>>,
  <<1, 4, 1>>,
  <<2, 1, 0>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 0>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<3, 3, 1>>,
  <<3, 4, 0>>,
  <<4, 1, 1>>,
```

```
  <<4, 2, 1>>,
  <<4, 3, 0>>,
  <<4, 4, 1>> }
/\  p1v = <<1, 0, 1, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, -1>>,
  <<1, 3, -1>>,
  <<1, 4, -1>>,
  <<2, 1, -1>>,
  <<2, 2, -1>>,
  <<2, 3, -1>>,
  <<2, 4, -1>>,
  <<3, 1, -1>>,
  <<3, 2, -1>>,
  <<3, 3, -1>>,
  <<3, 4, -1>>,
  <<4, 1, -1>>,
  <<4, 2, -1>>,
  <<4, 3, -1>>,
  <<4, 4, -1>> }
/\  p2v = <<-1, -1, -1, -1>>
/\  pc = <<"Done", "Done", "S", "S">>
/\  r = <<5, 5, 5, 5>>
```

2.

```
/\  decided = <<-1, -1, -1, -1>>
/\  p1Msg = { <<1, 1, 0>>,
  <<1, 2, 0>>,
  <<1, 3, 0>>,
  <<1, 4, 1>>,
  <<2, 1, 0>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 0>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<3, 3, 1>>,
  <<3, 4, 0>>,
  <<4, 1, 1>>,
  <<4, 2, 1>>,
  <<4, 3, 0>>,
  <<4, 4, 1>> }
/\  p1v = <<1, 0, 1, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, -1>>,
```

<<1, 3, -1>>,
<<1, 4, -1>>,
<<2, 1, -1>>,
<<2, 2, -1>>,
<<2, 3, -1>>,
<<2, 4, -1>>,
<<3, 1, -1>>,
<<3, 2, -1>>,
<<3, 3, -1>>,
<<3, 4, -1>>,
<<4, 1, -1>>,
<<4, 2, -1>>,
<<4, 3, -1>>,
<<4, 4, -1>> }
/\ p2v = <<-1, -1, -1, -1>>
/\ pc = <<"Done", "Done", "Done", "S">>
/\ r = <<5, 5, 5, 5>>

3. /\ decided = <<-1, -1, -1, -1>>
/\ p1Msg = { <<1, 1, 0>>,
<<1, 2, 0>>,
<<1, 3, 0>>,
<<1, 4, 1>>,
<<2, 1, 0>>,
<<2, 2, 1>>,
<<2, 3, 1>>,
<<2, 4, 0>>,
<<3, 1, 1>>,
<<3, 2, 0>>,
<<3, 3, 1>>,
<<3, 4, 0>>,
<<4, 1, 1>>,
<<4, 2, 1>>,
<<4, 3, 0>>,
<<4, 4, 1>> }
/\ p1v = <<1, 0, 1, 1>>
/\ p2Msg = { <<1, 1, -1>>,
<<1, 2, -1>>,
<<1, 3, -1>>,
<<1, 4, -1>>,
<<2, 1, -1>>,
<<2, 2, -1>>,
<<2, 3, -1>>,
<<2, 4, -1>>,
<<3, 1, -1>>,
<<3, 2, -1>>,
<<3, 3, -1>>,

```
  <<3, 4, -1>>,
  <<4, 1, -1>>,
  <<4, 2, -1>>,
  <<4, 3, -1>>,
  <<4, 4, -1>> }
/\  p2v = <<-1, -1, -1, -1>>
/\  pc = <<"Done", "Done", "Done", "Done">>
/\  r = <<5, 5, 5, 5>>
```

Explanation: To maintain a progress property, all nodes should be able to make a decision. In this scenario, we have equal number of “0” and “1” in the Input set. So, nodes/processes are not able to reach a decision and hence pass -1 default value to the next phase. As we have no faulty nodes in this scenario, we need 3 nodes to have same value to make a majority. On receiving these default values, model checker will assign random values to the nodes and among these 4 nodes, 3 should have same random value to have a majority, but the model checker assigns the values in such a way that nodes never get the majority of one value, and the decision is postponed indefinitely, which would lead to progress property being violated as nodes/processes will never be able to make a decision based on the “random” values.

Scenario 4: F <-1 | N <-4 | Input <- <1,1,1,1> | MAXROUND <- 4

No Errors

Explanation: In this scenario, we have same values in the Input set, although one node is failed, we will be still able to have majority of 3 nodes with same value. So, each node proposes the same value for the next phase and also it’s a majority, so each node is able to make a decision and each node has the same decision even with one failure node because of same values, so agreement property and progress property are not violated in this scenario.

Scenario 5: F <-1 | N <-3 | Input <- <0,1,1> | MAXROUND <- 4

Temporal Properties were violated (Progress)

States:

1.
```
/\  decided = <<-1, -1, -1>>
/\  p1Msg = { <<1, 1, 0>>,
  <<1, 2, 1>>,
  <<1, 3, 0>>,
  <<1, 4, 0>>,
  <<2, 1, 1>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 1>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<3, 3, 1>>,
  <<3, 4, 1>> }
/\  p1v = <<1, 1, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, -1>>,
  <<1, 3, -1>>,
  <<1, 4, -1>>,
  <<2, 1, 1>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 1>>,
  <<3, 1, -1>>,
  <<3, 2, -1>>,
  <<3, 3, -1>>,
  <<3, 4, -1>> }
/\  p2v = <<-1, 1, -1>>
/\  pc = <<"Done", "CP2", "Done">>
/\  r = <<5, 4, 5>>
```

2.
```
/\  decided = <<-1, -1, -1>>
/\  p1Msg = { <<1, 1, 0>>,
  <<1, 2, 1>>,
  <<1, 3, 0>>,
  <<1, 4, 0>>,
  <<2, 1, 1>>,
```

```
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 1>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<3, 3, 1>>,
  <<3, 4, 1>> }
/\  p1v = <<1, 1, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, -1>>,
  <<1, 3, -1>>,
  <<1, 4, -1>>,
  <<2, 1, 1>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 1>>,
  <<3, 1, -1>>,
  <<3, 2, -1>>,
  <<3, 3, -1>>,
  <<3, 4, -1>> }
/\  p2v = <<-1, 1, -1>>
/\  pc = <<"Done", "S", "Done">>
/\  r = <<5, 5, 5>>

3./\  decided = <<-1, -1, -1>>
/\  p1Msg = { <<1, 1, 0>>,
  <<1, 2, 1>>,
  <<1, 3, 0>>,
  <<1, 4, 0>>,
  <<2, 1, 1>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 1>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<3, 3, 1>>,
  <<3, 4, 1>> }
/\  p1v = <<1, 1, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, -1>>,
  <<1, 3, -1>>,
  <<1, 4, -1>>,
  <<2, 1, 1>>,
  <<2, 2, 1>>,
  <<2, 3, 1>>,
  <<2, 4, 1>>,
  <<3, 1, -1>>,
```

```
  <<3, 2, -1>>,
  <<3, 3, -1>>,
  <<3, 4, -1>> }
/\  p2v = <<-1, 1, -1>>
/\  pc = <<"Done", "Done", "Done">>
/\  r = <<5, 5, 5>>
```

Explanation: In this scenario, we have just 3 nodes, out of which one node is faulty, and as we have mixture of “0” and “1” in the set, it might be the case that we have different values and hence no majority is seen in phase 1, so both the nodes pass “-1” default value to the next phase as it is not able to see a majority, when random values are assigned by the model checker, it will assign in a way that it is able to violate the progress property(i.e., one 0 and one 1), and so decision will be pending for indefinite amount of time as random values would never reach a majority. Although the failed node comes back and then we see majority in phase-1 later it might not be able to decide as the decision for that round is already made and the node which saw majority will take it to next round and this case might repeat leading to indefinite pending of decision violating Progress Property.

TASK 2: Checking our specification for the scenario- Agreement, Minority Report and Progress Tests

Scenario 1: F <- 0 | N <- 4 | Input <- <0,1,1,1> | MAXROUND <- 4

No Errors

Explanation: As suggested in the minority report, it claims that the value which is in the minority in the input set cannot be selected as the consensus value. In this case, when we have no faulty nodes, the model gets the majority value which is “1” in this case, and hence propose “1” for the later phase and decide a value. So, we get no error and Minority Report is not violated.

Scenario 2: F <- 1 | N <- 4 | Input <- <0,1,1,1> | MAXROUND <- 4

Results: Invariant MinorityReport is violated.

States:

1.
```
/\  decided = <<0, 0, 0, -1>>
/\  p1Msg = { <<1, 1, 0>>,
  <<1, 2, 0>>,
  <<2, 1, 1>>,
  <<2, 2, 0>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<4, 1, 1>>,
  <<4, 2, 1>> }
/\  p1v = <<0, 0, 0, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, 0>>,
  <<2, 1, -1>>,
  <<2, 2, 0>>,
  <<3, 1, -1>>,
  <<3, 2, 0>>,
  <<4, 1, 1>> }
/\  p2v = <<0, 0, 0, -1>>
/\  pc = <<"S", "S", "S", "CP1">>
/\  r = <<3, 3, 3, 2>>
```

2.
```
/\  decided = <<0, 0, 0, -1>>
/\  p1Msg = { <<1, 1, 0>>,
  <<1, 2, 0>>,
  <<2, 1, 1>>,
  <<2, 2, 0>>,
  <<3, 1, 1>>,
  <<3, 2, 0>>,
  <<4, 1, 1>>,
  <<4, 2, 1>> }
/\  p1v = <<0, 0, 0, 1>>
/\  p2Msg = { <<1, 1, -1>>,
  <<1, 2, 0>>,
```

<<2, 1, -1>>,
<<2, 2, 0>>,
<<3, 1, -1>>,
<<3, 2, 0>>,
<<4, 1, 1>>,
<<4, 2, 0>> }
/\ p2v = <<0, 0, 0, 0>>
/\ pc = <<"S", "S", "S", "CP2">>
/\ r = <<3, 3, 3, 2>>

3.

/\ decided = <<0, 0, 0, 0>>
/\ p1Msg = { <<1, 1, 0>>,
<<1, 2, 0>>,
<<2, 1, 1>>,
<<2, 2, 0>>,
<<3, 1, 1>>,
<<3, 2, 0>>,
<<4, 1, 1>>,
<<4, 2, 1>> }
/\ p1v = <<0, 0, 0, 0>>
/\ p2Msg = { <<1, 1, -1>>,
<<1, 2, 0>>,
<<2, 1, -1>>,
<<2, 2, 0>>,
<<3, 1, -1>>,
<<3, 2, 0>>,
<<4, 1, 1>>,
<<4, 2, 0>> }
/\ p2v = <<0, 0, 0, 0>>
/\ pc = <<"S", "S", "S", "S">>
/\ r = <<3, 3, 3, 3>>

Explanation: In minority report we state that a value which is in minority might not have the possibility to be decided as consensus value, but here as we have one faulty node, and for model to get the majority value 3 nodes must have “1” proposed (as it is the majority in the input set) but as one of the node is faulty(suppose it is the node having 1 as the p1v value) , it cannot decide on a majority as only 2 node have “1” proposing and hence sends -1 default value and at the end of round1, and for the next round all the two nodes might end up selecting 0 with random selection, and the minority value, which is 0 in this case might be assigned to these nodes and hence we have 3 nodes having 0 proposed the end of the phase1, so when the fourth node (faulty node) comes back online and completing it first round normally, it sees that the 0 is in majority in round2 so it takes that value as well and hence at the end all the nodes decide 0, which was initially a minority value in the input set. This is why the minority report property was violated in this case.

Conclusion

Ben-Or is a consensus algorithm which was proposed by Ben Or. It takes only binary input as it is not a leader-based algorithm so in order to avoid tie in system. The protocol is fault resilient as long as f < n/2 and it also takes the assumption that all the channels are FIFO. It consists of indefinite number repetition of "asynchronous rounds". The process which gets a round number are able to be processed for the rounds. These rounds are common for all "n-f" nodes. These "n-f" values are in the Input set, where "n" refers number of nodes/processes and "f" refers to number of faulty nodes. With each round, we have two phases, in first phase, every node/process is figuring out which is the value that is in majority amongst them so that it can propose that value for the next phase. If there is no majority amongst these values then default value "-1" is sent to next phase by all the nodes, in order to help the system towards making a decision so that the system can reach a consensus value in the next round. In second phase, based on the values received from the first phase, the node finalizes its decision if the same value is proposed by at least f+1 node. If a node sees a value from another node, it adopts the value for its p1v value. This is done so that at least one node sees a majority in the p1v values. Otherwise if none of this works for the system then it "randomly" selects value amongst 0 and 1 and assigns those values to its p1v values which would shift the system to have a majority and makes a decision. In this round, every node broadcast it’s p2v to p2Msg till there is at least “n-f” values in the channel.