# CS4103 - Distributed Systems

200007413

## 1 Implementation

The basic implementation begins with the `Proposer` instances initializing through sending `prepare` messages to all `Acceptor` instances with their initial unique ballot ID. It is acceptable to send the same ballot ID to all acceptors, because each `Acceptor` is independent, and it allows us to track the max returning ballot ID for that proposal round.

The `Acceptor` instances receive these proposals and keep track of the maximum ballot ID they have seen through the `maxBallotId` attribute. If the incoming ballot ID is greater than the current `maxBallotId`, this max value is updated, and an acknowledgment sent back.

When a prepare message has a smaller ballot ID than the current max seen ballot ID, the basic Paxos implementation silently ignores them. However, to enable proposers to make new proposals if higher numbered proposals cannot make a consensus, the `prepare_reject` message type enables acceptors to notify proposers. Upon receiving a majority `prepare_reject` messages from acceptors, the proposer will re-submit a proposal with a higher ballot ID. This ID is calculated by taking the current maximum seen ballot ID by acceptors, and multiplying that value by 100, and adding the original ballot ID. This keeps values unique as each ballot ends with their unique ballot ID, and is larger than any previous ballots made by that proposer.

Proposers are designed to complete once their proposals have been accepted. In the basic implementation, proposers would stop processing messages once they send an `accept` message. However, to handle message failures, a `reject` message type was implemented to notify proposers when their ballot was rejected. This includes the proposal of the previously accepted proposal, and the proposer updates it's proposal to that value.

Message unreliability was implemented through an unreliability attribute, which defines the proportion of failed messages. A wrapper function for sending messages `trySendMessages` uses a random number generator to simulate these failures following the unreliability probability.

To handle message unreliability, a timeout was created in the proposers and learners to re-try making proposals and learn requests if they do not receive responses after a certain number of iterations. This is important to add to learners due to the optimisation of sending just a single initial learn request that that registers the learner with an acceptor to receive the value message. If this message fails, then the learner may never learn a majority of values, and thus consensus not reached. Therefore, adding a timeout enables learners to send not too many learn requests, but recover if some

messages fail.

# 2 Testing

## 2.1 Approach

Java tests were created initializing the `ParallelRunner` with different configurations to test the system's reliability for finding consensus. The tests were run with different numbers of proposers, acceptors, and learners to see how the system behaves under different conditions. Additional tests were conducted to see how the system behaves when processes fail, or when particular messages have a change to fail.

## 2.2 Results

Appendix A gives the trace for the system with the default settings given in the starter code. The system was able to reach consensus on the value 'third' after 100 steps with 3 proposers, 3 acceptors, and 2 learners. Logs stopped being printed before the system terminated, suggesting consensus was reached in significantly fewer than 100 steps.

Where a minority of acceptors fail the inherent fault tolerance of the system enabled consensus to be reached (see Appendix C and Appendix D).

Where a majority of acceptors fail, none of the acceptors are able to reach consensus (see Appendix B). The traces show that no `accept` messages were sent from proposers before timeout.

Where proposers are unaware of all acceptors, as long as each proposer is aware of a majority of working acceptors, a consensus is reached (see Appendix E). Where some proposers are unaware of a majority of working acceptors, a majority is not reached (see Appendix F)

## 2.3 Discussion

A success of this implementation is that it is able to reach consensus on a value in a reasonable number of steps. This is likely due to the lack of redundant messages that are sent. Proposers send just their initial proposals to acceptors in their initialization, instead of sending them repeatedly when they do not receive any messages. Additionally, learn requests cause acceptors to store the ID of the learner, and send the value once accepted, instead of sending values when learn requests are received, which would require more frequent learn requests as it takes time for acceptors to accept values.

The implementation successfully demonstrated the inherent fault tolerance of the Paxos

algorithm to acceptor failure, with the system still able to reach consensus with 3 functional acceptors, and 2 failed acceptors. The rule that a majority of acceptors must be functional is shown where the system is unable to reach consensus with 2 functional acceptors and 3 failed acceptors. This is likely the case because the same number of steps was allocated for these tests as their associated minority acceptor failure tests. In addition, no accept messages were ever sent from the proposers, due to the requirement due to all proposers being aware of all acceptors, meaning that half of acceptors must acknowledge their requests, which is not possible if they are not functional. This is a fundamental property of the Paxos algorithm, and the implementation successfully demonstrates this.

# A Default Setup Trace

```
threads set up
Acceptor 1: message <learn> received from process 6
Acceptor 0: message <prepare> received from process 3
Acceptor 2: message <prepare> received from process 5
Proposer 3: message <acknowledge> received from process 0
Proposer 5: message <acknowledge> received from process 2
Acceptor 0: message <learn> received from process 7
Acceptor 2: message <learn> received from process 6
Acceptor 1: message <learn> received from process 7
Acceptor 2: message <prepare> received from process 3
Acceptor 0: message <prepare> received from process 4
Acceptor 2: message <learn> received from process 7
Acceptor 1: message <prepare> received from process 4
Proposer 4: message <acknowledge> received from process 0
Acceptor 0: message <prepare> received from process 5
Acceptor 1: message <prepare> received from process 5
Acceptor 2: message <prepare> received from process 4
Proposer 5: message <acknowledge> received from process 0
Acceptor 1: message <accept> received from process 5
Acceptor 1: accepted value: third
Proposer 4: message <acknowledge> received from process 1
Acceptor 0: message <accept> received from process 4
Acceptor 0: rejected value: second
Acceptor 2: message <accept> received from process 4
Acceptor 2: rejected value: second
Acceptor 1: message <accept> received from process 4
Acceptor 1: rejected value: second
Acceptor 2: message <accept> received from process 5
Acceptor 2: accepted value: third
Acceptor 0: message <accept> received from process 5
Acceptor 0: accepted value: third
Acceptor 1: message <prepare> received from process 3
Learner 6: message <value> received from process 2
Learner 7: message <value> received from process 2
Acceptor 0: message <learn> received from process 6
Learner 6: message <value> received from process 0
Learner 6: consensus value: third
Learner 7: message <value> received from process 0
Learner 7: consensus value: third
executed 100 steps, processes terminating
done
```

# B Majority Broken Acceptor Trace

```
threads set up
Acceptor 12: message <prepare> received from process 16
Acceptor 11: message <learn> received from process 20
Proposer 16: message <acknowledge> received from process 12
Acceptor 12: message <prepare> received from process 13
Acceptor 11: message <prepare> received from process 17
Proposer 17: message <acknowledge> received from process 11
```

```
Acceptor 12: message <prepare> received from process 15
Acceptor 12: message <prepare> received from process 14
Acceptor 11: message <prepare> received from process 14
Acceptor 12: message <prepare> received from process 17
Acceptor 11: message <learn> received from process 19
Acceptor 11: message <learn> received from process 18
Acceptor 12: message <learn> received from process 19
Proposer 17: message <acknowledge> received from process 12
Acceptor 12: message <learn> received from process 18
Acceptor 11: message <prepare> received from process 15
Acceptor 12: message <learn> received from process 20
Acceptor 12: message <learn> received from process 21
Acceptor 11: message <prepare> received from process 13
Acceptor 11: message <prepare> received from process 16
Acceptor 11: message <learn> received from process 21
executed 500 steps, processes terminating
done
```

# C Minority Broken Acceptor Trace (small)

```
    threads set up
    Acceptor 2: message <learn> received from process 7
    Acceptor 1: message <prepare> received from process 5
    Proposer 5: message <acknowledge> received from process 1
    Acceptor 2: message <learn> received from process 6
    Acceptor 1: message <prepare> received from process 3
    Acceptor 2: message <prepare> received from process 3
    Acceptor 1: message <prepare> received from process 4
    Proposer 3: message <acknowledge> received from process 2
    Acceptor 1: message <learn> received from process 7
    Acceptor 2: message <prepare> received from process 5
    Acceptor 1: message <learn> received from process 6
    Acceptor 2: message <prepare> received from process 4
    Proposer 5: message <acknowledge> received from process 2
    Acceptor 1: message <accept> received from process 5
    Acceptor 1: accepted value: third
    Acceptor 2: message <accept> received from process 5
    Acceptor 2: accepted value: third
    Learner 7: message <value> received from process 1
    Learner 6: message <value> received from process 2
    Learner 7: message <value> received from process 2
    Learner 7: consensus value: third
    Learner 6: message <value> received from process 1
    Learner 6: consensus value: third
```

# D Minority Broken Acceptor Trace

```
threads set up
Acceptor 24: message <learn> received from process 35
Acceptor 26: message <learn> received from process 33
```

```
Acceptor 25: message <prepare> received from process 27
Acceptor 24: message <prepare> received from process 27
Proposer 27: message <acknowledge> received from process 25
Acceptor 26: message <prepare> received from process 31
Acceptor 24: message <prepare> received from process 31
Proposer 31: message <acknowledge> received from process 24
Acceptor 26: message <prepare> received from process 30
Proposer 27: message <acknowledge> received from process 24
Acceptor 25: message <learn> received from process 35
Proposer 31: message <acknowledge> received from process 26
Acceptor 26: message <learn> received from process 34
Acceptor 24: message <learn> received from process 32
Acceptor 25: message <prepare> received from process 31
Acceptor 26: message <prepare> received from process 27
Proposer 31: message <acknowledge> received from process 25
Acceptor 24: message <accept> received from process 31
Acceptor 24: accepted value: fifth
Acceptor 25: message <accept> received from process 31
Acceptor 25: accepted value: fifth
Acceptor 25: message <prepare> received from process 28
Acceptor 26: message <accept> received from process 31
Acceptor 26: accepted value: fifth
Acceptor 24: message <prepare> received from process 28
Acceptor 25: message <learn> received from process 33
Acceptor 26: message <learn> received from process 32
Acceptor 24: message <learn> received from process 34
Acceptor 25: message <learn> received from process 32
Acceptor 26: message <prepare> received from process 29
Acceptor 24: message <prepare> received from process 29
Acceptor 25: message <prepare> received from process 30
Acceptor 26: message <prepare> received from process 28
Acceptor 24: message <prepare> received from process 30
Acceptor 25: message <prepare> received from process 29
Acceptor 26: message <learn> received from process 35
Acceptor 25: message <learn> received from process 34
Acceptor 24: message <learn> received from process 33
Learner 35: message <value> received from process 25
Learner 32: message <value> received from process 25
Learner 34: message <value> received from process 26
Learner 33: message <value> received from process 25
Learner 35: message <value> received from process 26
Learner 32: message <value> received from process 24
Learner 33: message <value> received from process 26
Learner 34: message <value> received from process 24
Learner 32: message <value> received from process 26
Learner 32: consensus value: fifth
Learner 35: message <value> received from process 24
Learner 35: consensus value: fifth
Learner 33: message <value> received from process 24
Learner 33: consensus value: fifth
Learner 34: message <value> received from process 25
Learner 34: consensus value: fifth
executed 500 steps, processes terminating
done
```

---

# E Proposer Partial Acceptor Awareness

```
Preparing Test
Test Partial Proposer Awareness
threads set up
Acceptor 4: message <learn> received from process 9
Acceptor 2: message <learn> received from process 9
Acceptor 3: message <learn> received from process 9
Acceptor 1: message <prepare> received from process 6
Acceptor 0: message <learn> received from process 8
Acceptor 4: message <prepare> received from process 7
Acceptor 2: message <prepare> received from process 5
Acceptor 3: message <learn> received from process 8
Proposer 7: message <acknowledge> received from process 4
Proposer 6: message <acknowledge> received from process 1
Proposer 5: message <acknowledge> received from process 2
Acceptor 2: message <prepare> received from process 6
Acceptor 1: message <learn> received from process 8
Acceptor 0: message <learn> received from process 9
Acceptor 4: message <learn> received from process 8
Acceptor 2: message <learn> received from process 8
Acceptor 3: message <prepare> received from process 7
Proposer 6: message <acknowledge> received from process 2
Acceptor 2: message <accept> received from process 6
Acceptor 2: accepted value: second
Acceptor 1: message <accept> received from process 6
Acceptor 1: accepted value: second
Proposer 7: message <acknowledge> received from process 3
Learner 9: message <value> received from process 2
Acceptor 0: message <prepare> received from process 5
Acceptor 4: message <accept> received from process 7
Acceptor 4: accepted value: third
Learner 8: message <value> received from process 1
Acceptor 2: message <accept> received from process 7
Acceptor 2: accepted value: third
Proposer 7: message <accept> received from process 4
Proposer 7: accepted, stopping
Acceptor 1: message <prepare> received from process 5
Acceptor 3: message <accept> received from process 7
Acceptor 3: accepted value: third
Proposer 5: message <reject_prepare> received from process 1
Learner 8: message <value> received from process 3
Learner 9: message <value> received from process 4
Proposer 6: message <accept> received from process 1
Proposer 6: accepted, stopping
Acceptor 2: message <prepare> received from process 7
Learner 9: message <value> received from process 3
Proposer 5: message <acknowledge> received from process 0
Acceptor 1: message <accept> received from process 5
Acceptor 1: accepted value: second
Learner 8: message <value> received from process 4
Acceptor 3: message <accept> received from process 6
Acceptor 3: rejected value: second
Acceptor 0: message <accept> received from process 5
Acceptor 0: accepted value: second
Acceptor 1: message <learn> received from process 9
```

```
Acceptor 2: message <accept> received from process 5
Acceptor 2: rejected value: second
Acceptor 3: message <prepare> received from process 6
Proposer 5: message <accept> received from process 0
Proposer 5: accepted, stopping
Learner 9: message <value> received from process 1
Learner 8: message <value> received from process 0
Learner 8: message <value> received from process 2
Learner 8: consensus value: second
Learner 9: message <value> received from process 0
Learner 9: consensus value: second
```

# F Proposer Partial Acceptor Awareness Failure

# G Unreliable Message Passing

```
Preparing Test
Test Unreliable Message Passing
threads set up
Acceptor 2: message <learn> received from process 6
Acceptor 1: message <learn> received from process 6
Acceptor 0: message <prepare> received from process 4
Acceptor 2: message <prepare> received from process 5
Proposer 4: message <acknowledge> received from process 0
Acceptor 0: message <learn> received from process 7
Acceptor 1: message <learn> received from process 7
Proposer 5: message <acknowledge> received from process 2
Acceptor 0: message <prepare> received from process 5
Acceptor 2: message <prepare> received from process 3
Acceptor 1: message <prepare> received from process 4
Proposer 3: message <reject_prepare> received from process 2
Proposer 4: message <acknowledge> received from process 1
Acceptor 2: message <learn> received from process 7
Proposer 5: message <acknowledge> received from process 0
Acceptor 1: message <accept> received from process 4
Acceptor 1: accepted value: second
Acceptor 0: message <accept> received from process 5
Learner 7: message <value> received from process 1
Acceptor 0: accepted value: third
Learner 6: message <value> received from process 1
Acceptor 2: message <prepare> received from process 4
Proposer 4: message <accept> received from process 1
Proposer 4: accepted, stopping
Learner 7: message <value> received from process 0
Proposer 5: message <accept> received from process 0
Proposer 5: accepted, stopping
Acceptor 1: message <prepare> received from process 3
Proposer 3: message <reject_prepare> received from process 1
Acceptor 2: message <prepare> received from process 3
Acceptor 0: message <learn> received from process 6
Acceptor 1: message <prepare> received from process 3
Proposer 3: message <acknowledge> received from process 2
```

```
Learner 6: message <value> received from process 0
Acceptor 0: message <prepare> received from process 3
Acceptor 2: message <accept> received from process 5
Acceptor 2: rejected value: third
Acceptor 1: message <prepare> received from process 5
Proposer 3: message <reject_prepare> received from process 0
Proposer 3: message <acknowledge> received from process 1
Acceptor 2: message <accept> received from process 3
Acceptor 2: accepted value: second
Learner 7: message <value> received from process 2
Learner 7: consensus value: second
Learner 6: message <value> received from process 2
Learner 6: consensus value: second
Proposer 3: message <accept> received from process 2
Proposer 3: accepted, stopping
```

---