# Parallel and Distributed Computing
# DD2443 - Pardis24
# Exercises for Lecture 13

Name: Casper Kristiansson

October 10, 2024

## Exercise 1

### Question

Three-phase commit ("3PC") requires multiple processes to simultaneously commit or to simultaneously abort a transaction.

We assume that processes can crash at any time during a three-phase commit, and there are no Byzantine processes. We also assume the network is asynchronous, and that messages are never spontaneously lost or reordered during transmission. However, note that a process may crash in the middle of a broadcast, which may cause some broadcasted messages to be correctly delivered to their destination, and some broadcasted messages to be lost. A process broadcasting requests can receive replies while other processes have yet to receive the request.

Three-phase commit involves the following steps:

1. TM/coordinator sends ready to all participants
2. Cohorts answer yes or no
3. TM sends prepare or abort
4. Cohorts answer ack and abort, if applicable
5. TM aborts or sends commit
6. Cohorts commit and answer ackCommit

7. TM commits

Answer the following two questions about three-phase commit.

a) In all steps but one, processes wait for a message before that step can be executed. In which steps do which processes have to wait for what messages?

b) If a message does not arrive because the sender has crashed, the waiting process times out. For each of the steps where processes wait, how should processes react to a timeout? Note that if there is any uncertainty about the outcome, no process can decide to commit. Hint: Can the processes safely abort, or are they forced to commit? Must they elect a new coordinator?

## Answer

**(1)**

- Step 2: Cohorts wait for "ready" from the TM, if it doesn't receive it the transaction will not be continued.
- Step 3: TM waits "yes" or "no" responses from all of the different cohorts
- Step 4: Cohorts wait for "prepare" or "abort" from the TM. This is because it needs a response with an acknowledgment.
- Step 5: TM waits for "ack" responses from cohorts. In this step, the TM will abort the process if it so happens that it hasn't received all of them.
- Step 6: Cohorts wait for "commit". In this process, the cohorts will wait until it actually gets the commit to know then to commit the transaction and before sending the acknowledgment to the commit
- Step 7: In the last step the TM will wait for all of the acknowledgment sot that the transaction has been committed.

**(2)**

In general, the processes should be aborted for safety in case of a timeout, as no process can be committed when uncertain. In general, it is not safe for cohorts to decide on their own in Step 6; they may want to communicate with other cohorts, either in case they don't get the "commit" message to learn about the state of the transaction. A situation where this might happen is if the TM crashes in which the entire process can be defined if it has been committed or not and therefore the last steps need to be fully executed.

# Exercise 2

## Question

Assume that we run Paxos on a set of 3 nodes A, B, C, that act as acceptors. Assume further that there are two nodes Q and R that act as proposers. The implementation of the acceptors is as in the lecture slides. The two proposers use the implementation given in the figure below.

Draw a timeline containing all transmitted messages if a user invokes choose(A,B,1,22,1) on Q at time T0 and choose(B,C,2,33,2) on R at time T0 + 0.5s. Assume that node processing time is zero and that all messages arrive within 0.5s.

## Answer

A timeline of this Paxos process involving the two proposers Q and R with the three accepters A,B,C will look like the following:

**T0**

1. Q sends a prepare request to both the accepters A and B with the following proposal with the values (1,22,1).

**T0 + 0.5s**

1. At this time R sends a prepare request to the other chain of the two accepters B and C with a higher-numbered proposal for priority sake with the values (2,33,2).

With this structure the communication between T0 and T1 will result in A promising to go with Q's proposal, B will receive both the proposals from Q and R and C will promise to R. But because R had a higher proposal number with property it will lead to B promising to R.

**T0 + 1s to T0 + 2s**

1. During this window R sends out an acceptable request to both B and C after receiving their promises in the last step.

**T0 + 2.5**

1. At the last step B and C accept the Rs proposal which results in the outcome that the Rs proposal is accepted due to the majority and therefore, the Qs proposal will fail.

# Exercise 3

## Question

a) Assume in Paxos that the register nmax is faulty such that nmax may return a value less than the most recently assigned. Can this pose a problem to the Paxos algorithm? Explain what happens in a worst-case scenario.

b) Paxos assumes there are no Byzantine failures. That is, memory failures as in (a) are not supposed to happen. Under these original assumptions, can you explain what the purpose of the Paxos prepare step is?

## Answer

### (1)

In Paxos, if the nmax is faulty it will most likely return a value that is less than the more recent one. This will often create a big issue because this might lead to situations where older proposals might be accepted rather than newer ones which has been agreed with the other nodes. This means the worst case scenario happens if a proposal uses the wrong sequence number and an acceptor isn't aware of any more recent proposal and therefore accepts the proposal. This will lead to an inconsistent and unstable state.

### (2)

The preparation steps for the Paxos as stated are extremely important to make sure that the agreements are handled ina the correct order. This can be achieved by making sure that there are no conflicts by making sure that when a proposal number has been accepted no older proposal can be accepted. It can also find safe values, meaning if a proposal gets accepted and the proposers learn of any previously accepted values than we refuse these types of values. Using this combination with an increase sequence number in newer proposals we can make sure that the process is safe and consistent.