CS W186 - Spring 2024
Exam Prep Section 11
Distributed Transactions

1 Conceptual Distributed Transactions

1. For each of the following statements, mark whether it's true or false and explain your reasoning.

   (a) Resolving a deadlock at a local node will always resolve distributed deadlock.

   (b) Suppose we make the changes needed so only a majority of participants need to vote yes for a transaction to commit. Participants can write an ABORT record in phase 1 of 2PC.

   (c) Presumed abort as presented in lecture no longer works in the scenario described in 1b.

   (d) If any machine sees COMMIT record in its log upon recovery, the transaction will commit. Additionally, describe how to distinguish whether a machine with a COMMIT record is a coordinator or participant.

   Answer:

   a is false. Distributed deadlock occurs whenever the union of waits-for graphs across different nodes contains cycles. Resolving one deadlock may not solve deadlock in the union of the nodes' waits-for graphs.

   b is false. Participants can't unilaterally abort anymore in this new scenario. The aborting participant can't determine the result of the vote by itself, so the participant must receive the result of the vote from the leader.

   c is true. If a participant crashes after voting no in phase 1, with presumed abort, no ABORT record will be flushed to disk. Coming back online, the participant would ordinarily presume abort, but in the scenario described, this may not be the case if the majority still votes to commit.

   d is true. Whether participant or coordinator, a node seeing a COMMIT record in its log on recovery must take the actions. If the commit record holds the participantIDs, then we know the machine must be a coordinator. Additionally, if this coordinator doesn't have an END record and only a COMMIT, it must send the result of the vote to all the participants. If there is a PREPARE record alongside the COMMIT record, then we know the node must be a participant.

2. For the following scenarios, describe what will happen without presumed abort, then describe what will happen with presumed abort.

   (a) Participant recovers and sees just a phase 1 ABORT record

(b) Participant recovers and sees a PREPARE record

(c) Participant recovers and sees a PREPARE and phase 2 ABORT record

(d) Coordinator recovers and sees an ABORT record

Answer:
a. Without presumed abort: Send ”no” to the coordinator. With presumed abort: Nothing! The coordinator will presume abort.

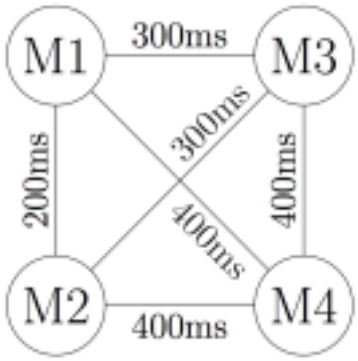
b. The case is the same with or without presumed abort. The participant will ask the coordinator the result of the vote and continue the process.

c. Without presumed abort, we'll send ACK back to the coordinator. With presumed abort, participants don't need to do anything after an abort record that was the result of the vote.

d. Without presumed abort: Inform the participants about the outcome of the vote. With presumed abort: Nothing! (Participants who don't know the decision will just ask later.)

2 Two Phase Commit Practice (Fall 2017 Final Question 3)

Our database runs on 4 machines and uses Two-Phase Commit. Machine 1 is the Coordinator, while Machines 2, 3, and 4 are Participants. Suppose our machines are connected such that the time it takes to send a message from Machine i to Machine j is 100·max(i,j) milliseconds (see graph below). Assume these communication latencies are symmetric: it takes the same amount of time to send from i to j as it takes to send from j to i. For example, sending a message between Machine 2 and Machine 4 takes 400 milliseconds in either direction. Assume that the transaction will commit (i.e. all subordinates vote yes), and that everything is instantaneous except for the time spent sending messages between two machines.



1. What is the first message Machine 1 sends?

   (a) VOTE YES

   (b) PREPARE

   (c) COMMIT

   (d) None of the above

   Answer: (b) PREPARE. The coordinator (i.e. Machine 1) begins the first round of two-phase commit by sending a PREPARE message to all of the subordinates.

2. What is the second message Machine 1 sends?

   (a) VOTE YES

   (b) PREPARE

   (c) COMMIT

(d) None of the above

Answer: (c) COMMIT. The coordinator (i.e. Machine 1) begins the second phase of two-phase commit by sending a COMMIT message to all of the subordinates. Note that the problem states that all subordinates vote yes in the first round which is why the coordinator sends a COMMIT message (instead of an ABORT message) to start the second round.

3. How much time passes from when Machine 1 sends its first message to when Machine 1 sends its second message?

   Answer: This is just the maximum time for a round-trip between Machine 1 and Machines 2–4, since Machine 1 sends its second message (COMMIT) once its first message (PREPARE) reaches each of the other machines and each of the machines respond back with a VOTE YES message. The time for a round-trip between Machine 1 and Machine i is $2 \cdot 100 \cdot i$, and the largest of these is with Machine 4: $2 \cdot 100 \cdot 4 = 800$ms.

4. What is the first message Machine 2 sends?

   (a) VOTE YES

   (b) PREPARE

   (c) COMMIT

   (d) None of the above

   Answer: (a) VOTE YES. A subordinate (e.g. Machine 2) responds to a PREPARE message from the coordinator with a VOTE YES or VOTE NO. The problem states that all subordinates vote yes, so Machine 2 sends a VOTE YES message.

5. What is the second message Machine 2 sends?

   (a) VOTE YES

   (b) PREPARE

   (c) COMMIT

   (d) None of the above

   Answer: (d) None of the above. The second message Machine 2 (a participant) sends is an ACK.

6. How much time passes from when Machine 2 sends its first message to when Machine 2 sends its second message?

   Answer: Let's have Machine 1 send the PREPARE message at time 0. This PREPARE message will reach Machine 2 at time 200ms, so Machine 2 sends its first message (VOTE YES) at time 200ms. Machine 1 sends the COMMIT message at time 800ms (when it hears back from Machine 4), and this message takes 200ms to reach Machine 2. Machine 2 therefore receives the COMMIT message at time 1000ms, and sends its second message (ACK). The time passed is therefore 1000ms - 200ms = 800ms.

7. True or False. A transaction is considered committed even if over half of the participants do not acknowledge the commit.

   Answer: True. A transaction must commit once the COMMIT message is sent out, even if all the participants promptly crash repeatedly and do not respond with ACKs as a result.

Now suppose that our implementation of 2-Phase Commit has an off-by-one bug where the Coordinator receives, but does not use, Machine 4's vote. That is, Machine 4's vote does not affect whether or not the transaction commits or aborts. Answer True or False for the following questions:

8. A transaction that should normally commit may be aborted instead.

9. A transaction that should normally abort may be committed instead.

10. A transaction that should normally commit may be committed properly.

11. A transaction that should normally abort may be aborted properly.

Answers:
8. False. If the transaction would normally commit, then Machines 2 and 3 must have been functional and voted yes, so the transaction would still have to commit with this bug.
9. True. If Machine 4 votes no and other participants vote yes, then the transaction should be aborted but commits anyways.
10. True. If all machines vote yes, then the transaction commits even if we ignore Machine 4's vote. 11. True. If Machine 2 or 3 votes no, then the transaction aborts in both cases.