Discussion Section 7

Distributed Systems
Spring 2014

Midterm - Q8 (a)

Sequential Consistency

P1		R(x)a		W(x)c	
P2	W(x)a		W(x)b		
P3				R(x)c	R(x)b
P4				R(x)c	R(x)b

- It is sequential consistent
- The writes should be read in the same order by all processes
- Both P3 and P4 read c and b in the same order

Midterm - Q8 (b)

Causal Consistency

P1		R(x)a		W(x)c		
P2	W(x)a		W(x)b			
P3				R(x)a	R(x)b	R(x)c
P4				R(x)a	R(x)c	R(x)b

- Causally consistent
- Causal Relationships :
  - (1) R(x)a -> W(x)c
  - (2) W(x)a -> W(x)b
- b & c have no causal relationship so they can be read in any read
- Its not necessary to read a.

2-Phase Commit Protocol

Basic idea

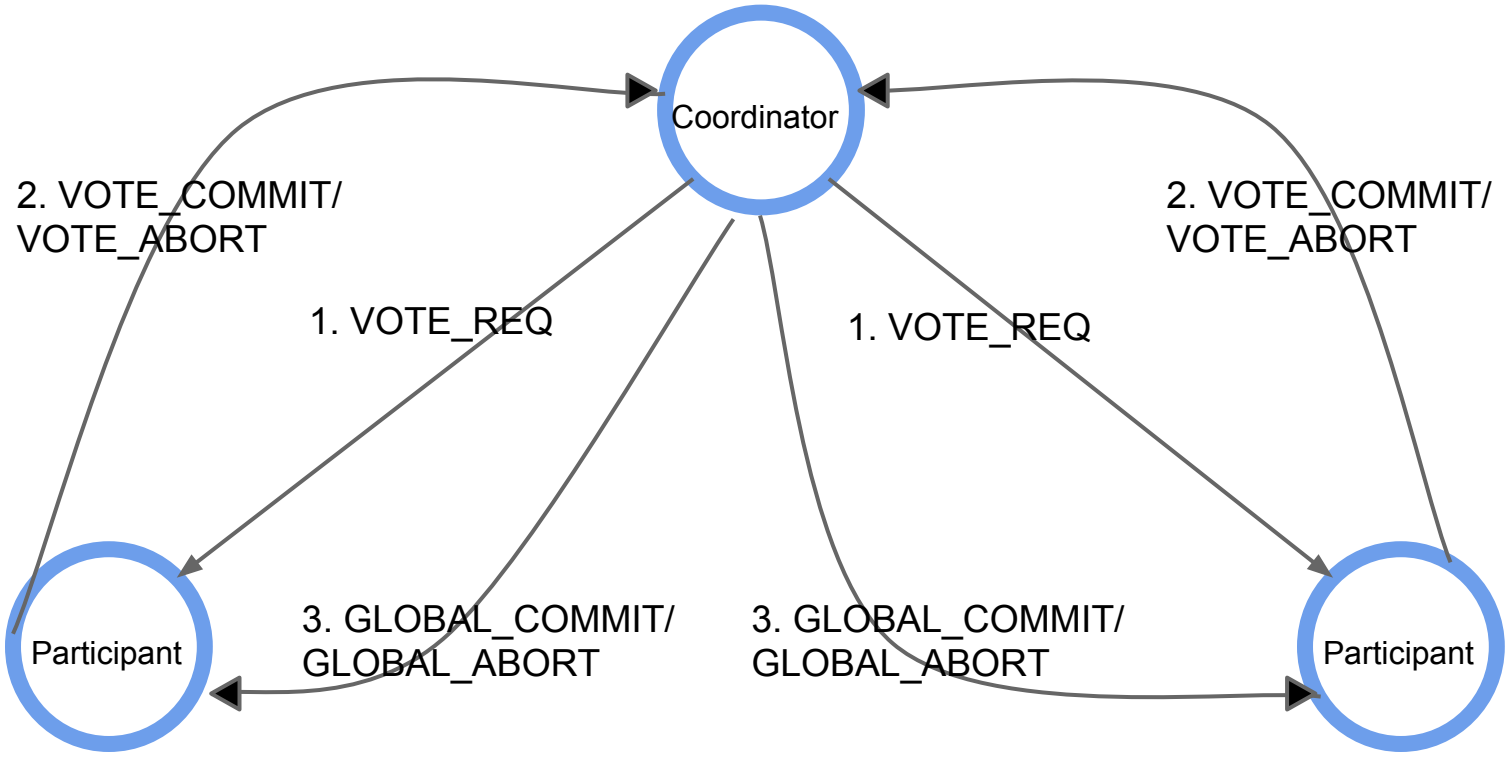
Single **coordinator** and several **participants**

There are two phases (divided in 4 steps):

1. Coordinator sends a “VOTE_REQUEST” to all the participants.
2. If a participant receives a “VOTE_REQUEST”, it either replies with a “VOTE_COMMIT” or “VOTE_ABORT”
3. Coordinator collects all votes and if **all** participants committed, it sends a “GLOBAL_COMMIT”, otherwise it sends “GLOBAL_ABORT”
4. After voting, each participant waits for the decision of the coordinator and either locally commit or locally aborts.

2-Phase Commit Protocol

Basic idea



Problems Chapter 8

Two-phase commit protocol

Q: In the two-phase commit protocol, why can blocking never be completely eliminated, even when the participants elect a new coordinator?

A: Because the new coordinator may also crash as well, so the remaining participants may not be able to reach a decision.

Second Programming Assignment

Commit Protocol Implementation

(Recalling from the lecture)

The **distributed commit** problem involves having an operation being performed by each member of a process group, or none at all.

Three protocols are presented:

- 1-phase commit protocol
- **2-phase commit protocol**
- 3-phase commit protocol

2-Phase Commit Protocol

Pseudo-code (Coordinator)

```
while START _2PC to local log;
multicast VOTE_REQUEST to all participants;
while not all votes have been collected {
    wait for any incoming vote;
    if timeout {
        write GLOBAL_ABORT to local log;
        multicast  GLOBAL_ABORT to all participants;
        exit;
    }
    record vote;
}
if all participants sent VOTE_COMMIT and coordinator votes COMMIT{
    write GLOBAL_COMMIT to local log;
    multicast GLOBAL_COMMIT to all participants;
} else {
    write GLOBAL_ABORT  to local log;
    multicast GLOBAL_ABORT to all participants;
}
```

2-Phase Commit Protocol

Pseudo-code (Participant)

```
write INIT to local log;
wait for VOTE_REQUEST from coordinator;
if timeout {
    write VOTE_ABORT to local log;
    exit;
}
if participant votes COMMIT {
    write VOTE_COMMIT to local log;
    send VOTE_COMMIT to coordinator;
    wait for DECISION from coordinator;
    if timeout {
        multicast DECISION_REQUEST to other participants;
        wait until DECISION is received; /* remain blocked */
        write DECISION to local log;
    }
    if DECISION == GLOBAL_COMMIT
        write GLOBAL_COMMIT to local log;
    else if DECISION == GLOBAL_ABORT
        write GLOBAL_ABORT to local log;
} else {
    write VOTE_ABORT to local log;
    send  VOTE ABORT to coordinator;
}
```

2-Phase Commit Protocol

Pseudo-code (Participant)

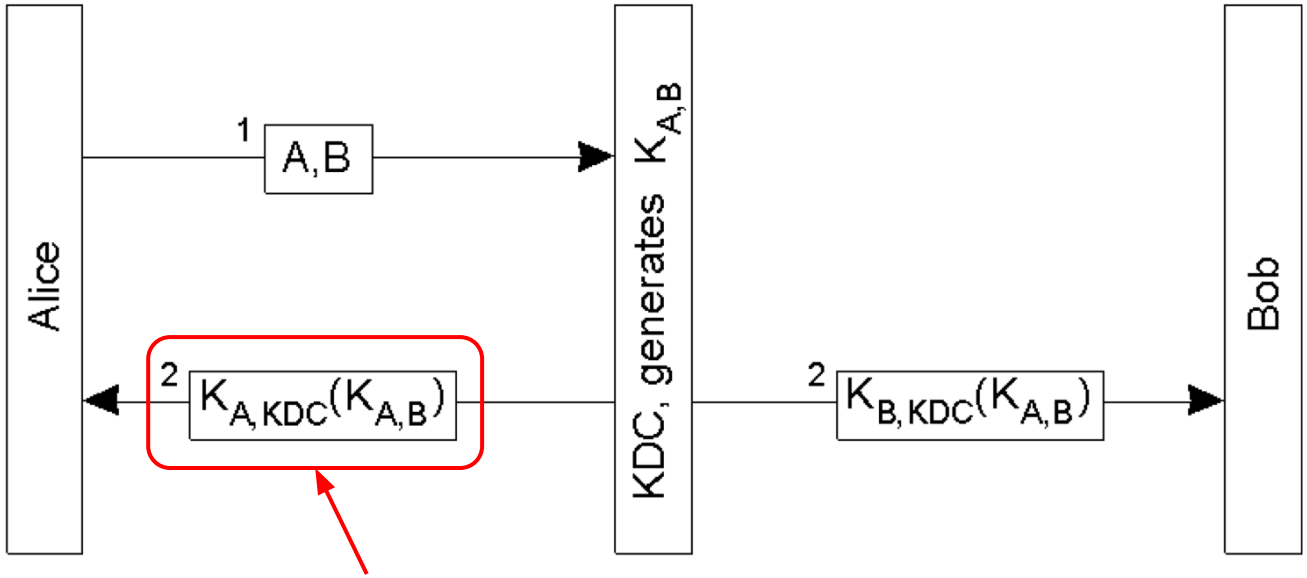
Executed by participants when they receive a request from another participant for the global decision

```
/* executed by a separate thread */

while true {
    wait until any incoming DECISION_REQUEST is received; /* remain blocked */
    read most recently recorded STATE from the local log;
    if STATE == GLOBAL_COMMIT
        send GLOBAL_COMMIT to requesting participant;
    else if STATE == INIT or STATE == GLOBAL_ABORT
        send GLOBAL_ABORT to requesting participant;
    else
        skip;  /* participant remains blocked */

}
```

Problems Chapter 9

Key Distribution Center (KDC)

Q: Why is it not necessary in for the KDC to know for sure it was talking to Alice (or anyone else pretending to be Alice) when it receives a request for a secret key that Alice can share with Bob?



Only Alice can decrypt it!

Asymmetric Cryptosystem

RSA

Let's take two prime numbers : **p = 3, q = 7**
=> n = 3 x 7 = 21

z = (p-1)(q-1) = 2 x 6 = 12 (=2x2x3x4)

d = 5

(d x e) = 1 mod 12 => (5 x e) = 1 mod 12 => e = 5

Encrypted Value, C = P^e mod n = 3^5 mod 21 = 12

Decrypted Value, P = C^d mod n = 12^5 mod 21
= 248832 mod 21 = 3