CSCI 1330: Day 18 (Dist Transaction)
Today

1. Distributed Transaction
   a) ACID
   b) Locks
   c) 2PC (two phase commit)
   d) Ordering Guarantees
goal: transfer $100 from theo to alakshoth

check_balance (theo) update (theo, -$100) update (alakshoth, +$100)

Dist transaction: trying to make changes across the different shards (each shard is managed by a group of AMs)

Passive/active/cozy is ordering within a group of AMs for a shard

you want all or nothing: all events are executed or none are

Replication techniques are within a shard

Dist transaction is across many shards
ACID

atomicity
* all events execute or none
* 2PL (two phase commit)

consistency
* you want each short to go from safe state to safe state

isolation
* each transaction has perception that its executing in isolation
* execute transactions in a serial order one after the other

* locks (pessimistic vs optimistic)

log safety

Durability
* changes from events last/persist after transaction(s)

Memory vs Disk
* reset when server restarts data on disk is persistent

* making sure things are stored to disk
Atomicity (1PC, 2PC, 3PC, 4PC)

1. Submit all events in transaction
2. Figure out the shards & leader for each shard
3. Gets the shards to execute events in the transaction

Transaction = lists of events to execute at different shards

Participants are the leaders of each shard involved in a transaction

Why would requests fail?
(why will a request carrying an event to a leader fail?)

* Leader failures → During election system will not be able to process events
* Network partition → If the coordination is in the minority partition then no events can be processed

Two Phase Commit:

transaction succeeds (commits) → if all participants commit their events
transaction fails (ABORTS) → if at least one participant fails (abort)
JPC (two phase commit)

FE coordination

Prepare (determine output of executing event but don't store it)

Commit/ABORT (voting)

RESULTS OF VOTING (COMMIT/ABORT)

Participants

W = \# of participants

If N commits

In this failure coordinator automatically aborts the transaction

Participants that is unable to get prepare will not participate & thus the events on that path are not going to happen

Participant fails before voting (failure doesn't retry after X retries)
Participant fails after voting

1. Try to do a rollback but eventually participant will come back up with "prepared" information still store

2. Instead of rollback, the coordination will wait/try until P1 is back up

Coordinator fails before vote coming (everyone waits)

coordinator received all votes & propose "commit"

vote ?

P1

vote

P2

vote

P3

all voted now are waiting for the response of vote

rollback change vote

JPC does not allow participants to change votes

Participants change to "abort"
Participants

Coordinators

fail — stalls the transaction
after completing prepare but before reading toward termination
out votes

Participants

failure before voting ->
then about everything

if participant fails after voting -> then just retry
until participant acknowledges receiving results

No progress can be made
Today

1) ACID semantics
2) differences btw dist transaction & replication strategies
3) Atomicity $\Rightarrow$ 2PC

Next Class

4) Isolation
5) Distributed file systems
When to use distributed Transactions?

1. Money transfer between users

2. World map is broken into tiles, each tile is a shard (each tile is merged by a different replication strategy)

3. Online network place closely interacting users in same shard so that short interactions are low delay