LACPP

Distributed Database

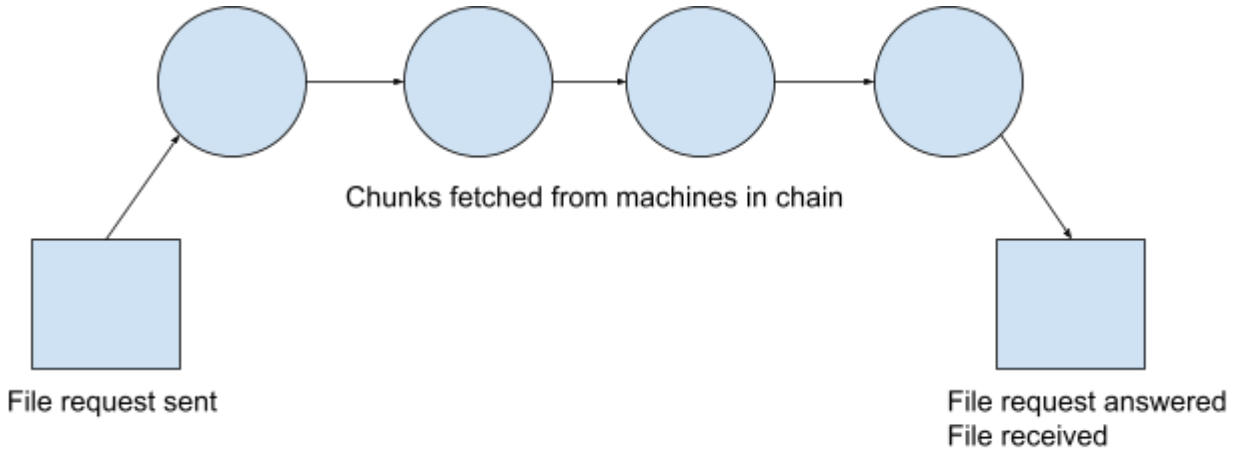
Linked Structure

Using linked refs to connect diff machines containing different files
All machines contain "Start Nodes" for all files

Downside: Slow

Upside: Redundant, Private, “Atomic” (as order exists in read/edit)

Has to impl TimeOut for slow/failed chunks



Direct Transactions (BROADCAST) (We should go with this)

Using a single key to call all machines, those who contain the key in their FileStore return their respective chunk, other machines ignore/ignored

Downside: No order, order data has to be implemented ***(KEY, (ORDER, CHUNK))***

Not private (everyone listens for any files requested)

Upside: Fast

Has to impl TimeOut for slow/failed chunks

OPERATIONS:

DELETE: If no response from machine, notify user that operation is unable to be performed (Ask if user wants to HARDDELETE?)

1. Delete sequence: First fetch the file and machines that returned the chunks,
    a. If FAIL notify user that operation is unable to be performed,
    b. ELSE try to delete from all machines, if delete has failed on one or more machines ADD the file back to machines available

alt: HARDDELETE: If no response from machine, drop machine, drop file from other machines since chunk missing

ADD: If no response from machine, drop machine, add file to other machine that is available

VIEW**:** if no response from machine, drop machine, return error fetching file

COUNT: get keys from all machines, remove duplicates, count elements. In case of no response, notify user of missing chunk(s)