Transactions Wrapup & Copy on Write FS

Transactions

Many disk
writes \{ tx-begin \\
upates \} \rightarrow \text{concurrent disk writes can be reordered} \rightarrow \text{tx-end}

Concurrent requests r1 r2 r3 r4

\text{issued to disk}

\text{heard back from disk}

\text{resolve the incomplete txns}

\rightarrow \text{compute a checksum (write as part of txs-end).}

\rightarrow \text{on recovery, validate committed txns' checksum before applying them}
Logging isn't the only way to achieve crash safety!

Copy-on-write Filesys

- in-place update:
  - Can't overwrite multiple disk blocks atomically (trans & journaling)
  - Achieve crash consistency by writing changes elsewhere first

- Copy-on-write:
  - Write changed blocks to new locations
  - Once all new blocks are written, use a single disk write to make all the new blocks visible at once

- No fixed location for almost all structures & data blocks, data & metadata locations change after every update
Cow Filesys

⇒ recursive update problem:

- if directory entry stores actual location of the metadata

  - updated data
    - updated inode
      - [name/block#]
    - updated parent data
      - path till root
    - updated parent inode
  - updated root dir
    - root dir inode

  \[ \text{with logical pointer} \]
  \[ \text{same inode #, different block #} \]
  \[ \text{or track different versions of inodetable's inode in an array} \]
  \[ \text{inodetable inode loc} \]

\[ \text{superblock (when changes become visible)} \]
\[ \text{fixed location} \]
<table>
<thead>
<tr>
<th>Root Inode</th>
<th>Root Inode’s Indirect Blocks</th>
<th>Inode</th>
<th>File’s Indirect Blocks</th>
<th>File’s Data Blocks</th>
</tr>
</thead>
</table>

- Metadata for indelatable data
- Data pointers
- Data layout via indexed blocks
- New Data Block
wash has no effect on the filesystem
lots of overhead (many blocks to change for a single write) → batch updates!
*provides crash safety without logging*, but often times a log is still used to speed up sync.