Google File System
Assumptions/Goals

• Any component could fail
• Some large files instead of many small files
  – Impact
• Append-heavy write; sequential accesses
  – Impact
• ...

⇒ Different designs from traditional file systems
Overall architecture
Overall architecture

Coda

client
client
client
client

AFS server

AFS server

GFS

client
client
client

client

master

chunk server

chunk server

chunk server
Why does GFS have a master?

• Why didn’t Coda use it? (disadvantage of master)
  – Scalability
  – Availability

• Why does GFS use it?
  – Easy to manage
What are the chunkservers?

- Data replication across chunkservers
Normal file system access (single machine)

• What if I want to read/write “/a/b/c”, 5Kth byte
  – Read the i-node of root “/” (from disk)
  – Search i-node of “/”: find the data block
  – Read the data block of “/”: find #i-node of a
  – Read the i-node of a: find the data block
  – Read the data block of “a”: find #i-node of b ...
  – ...
  – Read i-node of c
Normal file system metadata

• What are meta-datas?
  – i-node

• Where are meta-datas?
  – disk

• What is the data block size? Why?
  – 4 K
Google file system read

• What if I want to read “/a/b/c”, 5Kth byte
  – Ask master
    • File-name + # chunk ➔ chunk handle ➔ list of chunkserver
  – Contact (closest) chunkserver
    • Compare version number
    • Get the data
Google file system meta-data

• What are the meta-data?
  – Mapping (filename, chunk handle, chunkserver)

• Where is the meta-data?
  – In memory

• What is the block size?
  – 64 M
Write in GFS

• Step 1: contact the master; find the chunk handle; find the chunkservers, primary server
• Step 2: propagate the data to all replicas
• Step 3: send the write request to primary
• Step 4: primary decides the order; sends command to all replicas
  – Write to 1 or write to all replicas?
    • all
  – Who decides the order among concurrent writes?
    • Primary chunkserver (i.e., the one has the lease)
Failures in GFS writes

• What if a chunkserver is down?
Concurrent updates in GFS

• Concurrent write
  ➔ consistent & undefined

• Atomic append
  – Step 1: (optional) padding
  – Step 2: write at primary specified location
  – Step 3: success, return to
  ➔ inconsistent & defined
Write in Google File System

• Does GFS provide strong consistency?
  – Why?

• Is user aware of the inconsistency in Coda/GFS?

• Are there “partitioned” writes in GFS?

• Does GFS rely on users to solve inconsistency?
  – Is it the same as in Coda?
Other comparison with Coda

• Is there local disk cache?
Failure tolerance

• Is the master the bottleneck?
Other topics

• Snapshot

• File deletion & garbage collection

• Replica placement, re-replication, balancing
Summary

• Workload affects design

• Master – chunkserver architecture