File Systems: GFS vs. HDFS

Ciprian Lucaci

Daniel Straub

Distributed File Systems - Motivation

The problem

- Big Data
- Drive speed: 500 MB/s (SSD)
- 1 PB of data

- 1 super computer
- **138 hours** loading time

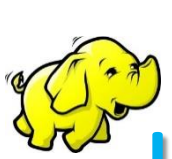
- 100 machines
- **2.8 hours** loading

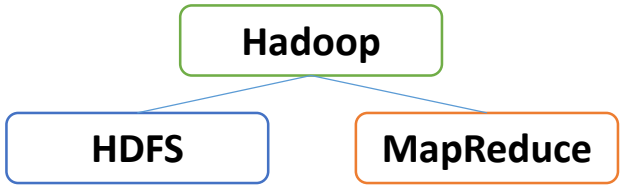
The solution

- Scalable
- Fault-tolerant
- Low cost
- Distributed computation
- Distributed storage

Moving computation is faster than moving data!
HDFS & GFS

HDFS - Architecture





- read, write, create, delete
- files, directories

Filename: block-ids (node#block#)
/user/dir1/file1: n1b1, n1b2, n3b1, b4b3
/user/dir2/file2: n3b7, n4b8, n1b6, b2b5

Client

Namenode

Secondary node

Metadata
- Edit log
- System image
- Block checksum
- Block location
- Namespace tree

- Edit log
- System image

- Heartbeat
- Report blocks

DN1

B2 ... B1 B3 Bn

Rack 1

Rack 2

Rack 3

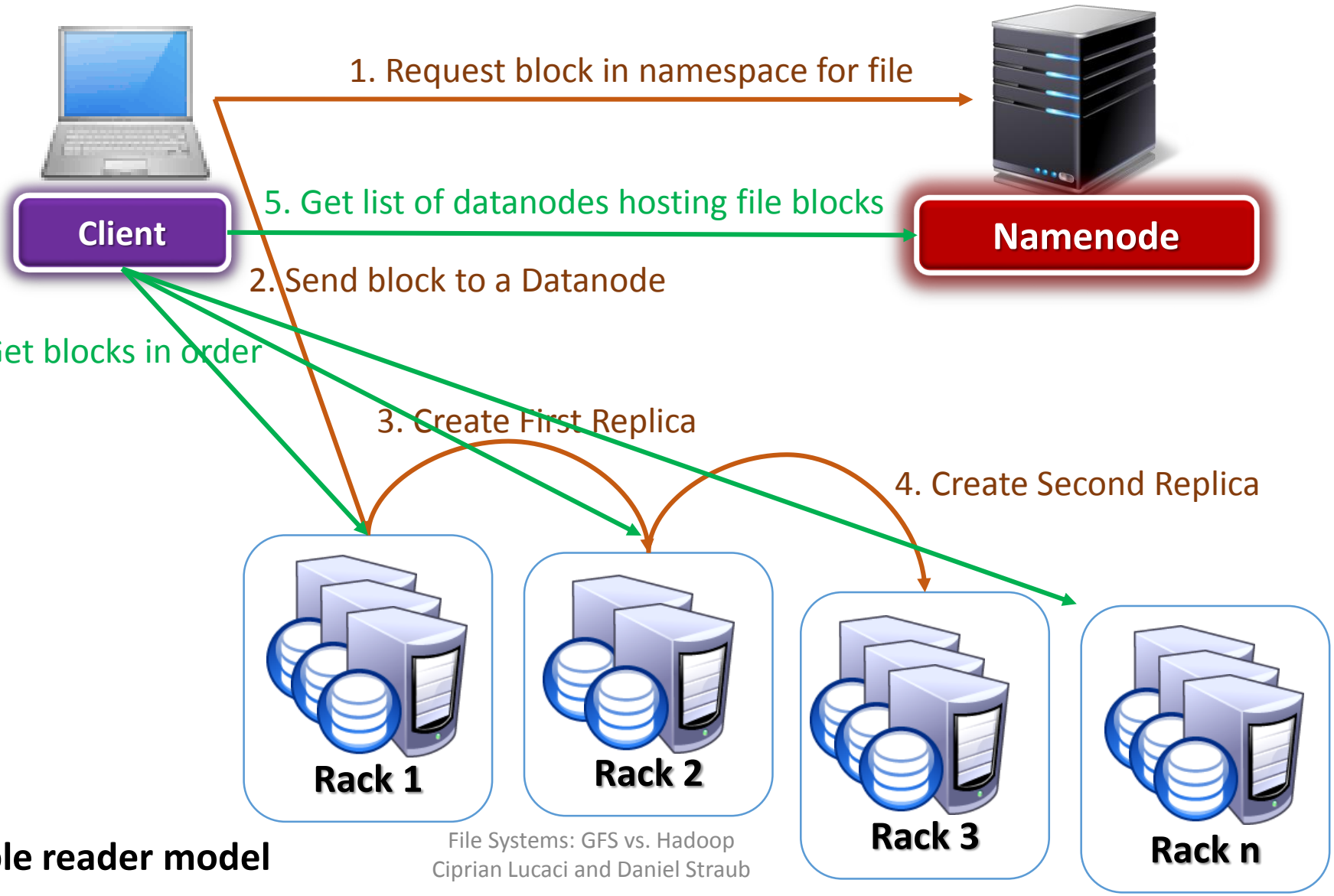
Datanodes

Rack n

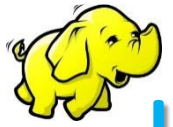
- 64 MB / block (default)
- Split large files into blocks

HDFS - Workflow

- **Write**
- **Read**



Single-writer, multiple reader model



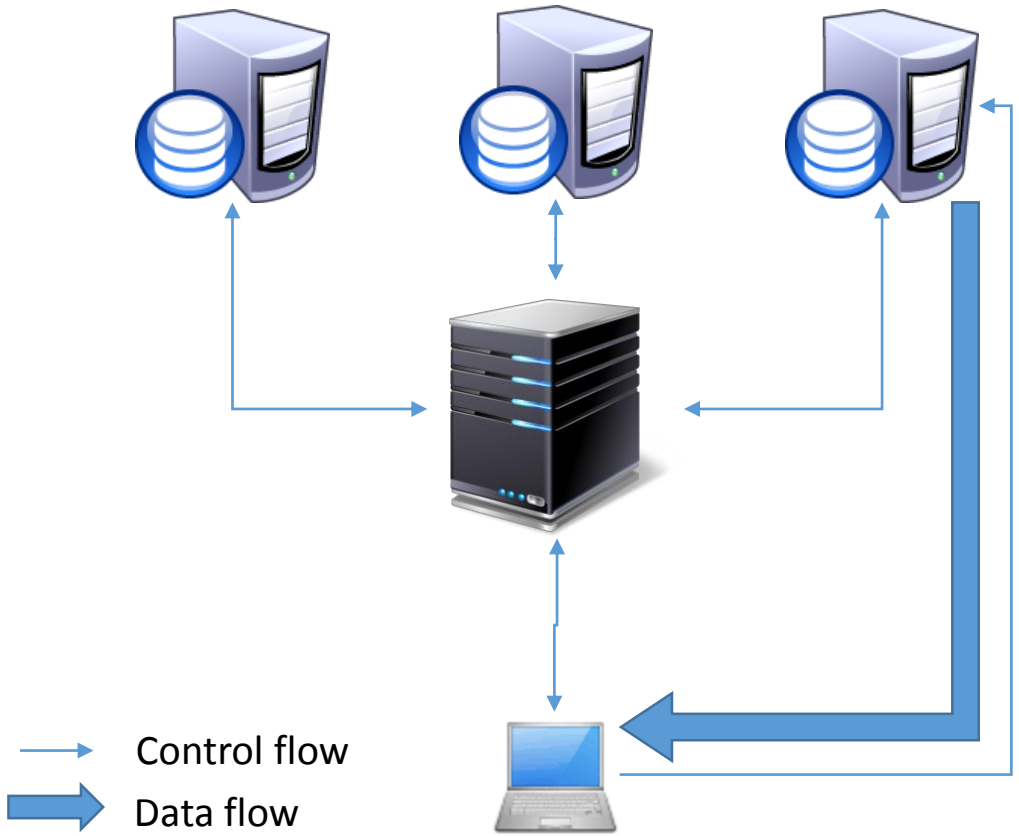
HDFS - Purpose

- Large files
- Commodity hardware
- Enable streaming
- Batch processing
- Big amount of small files
- Concurrent modification
- Arbitrary modification - appends only
- General purpose applications
- Multiple reads
- Cross platform: Java, Thrift, Rest

GFS - Purpose

- Large files (100MB and more)
- Commodity hardware (failures are the norm)
- (Concurrently) appending files
- Files are mostly read sequentially
- High data throughput
- Small files
- Modifying (not appending) writes
- Small random reads
- High latency

GFS - Architecture



Chunkserver 1
Chunk a
Chunk d
Chunk k
…

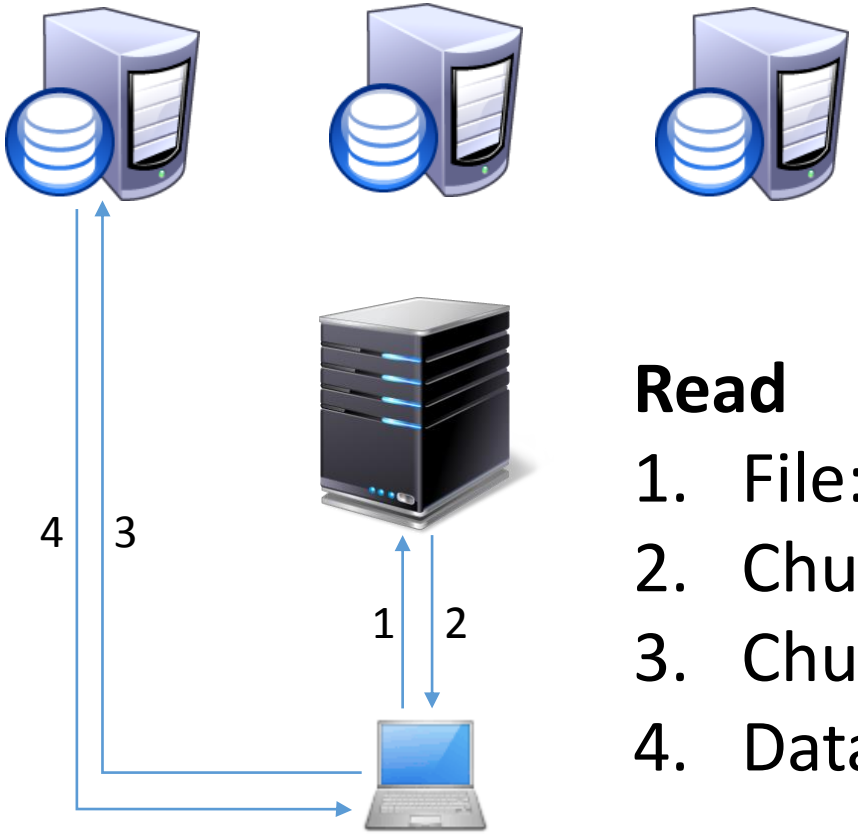
Chunkserver 2
Chunk b
Chunk d
Chunk e
…

Chunk: 64MB of data

Master
File1: chunk a, chunk c, chunk d, …
File2: chunk b, chunk x
…

Client

GFS - Workflow



CS 1	CS 2
Chunk a	Chunk b
Chunk d	**Chunk d**
Chunk k	Chunk e
…	…

Master

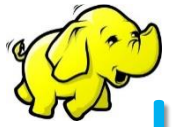
File1: chunk a, chunk c, chunk d, …

File2: chunk b, chunk x

…

Read

1. File: File1, chunk index: 3
2. Chunk d, location: CS 1, CS 2
3. Chunk d, byte range: 1-1000
4. Data: byte 1-1000



HDFS vs. GFS

Similarities

- Large files
- Write once, read multiple times
- Appending of files
- Streaming of files

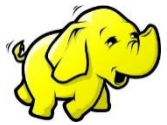
Differences

- Block id vs. Chunk index
- Concurrently appending to files
- Open source vs. Closed source
- Namenode vs. Master fail

Conclusion

- Scalable
- Fault-tolerant
- Low cost
- Distributed computation
- Distributed storage
- Reliable streaming

References



- HDFS
  - K. Shvachko, H. Kuang, S. Radia, R. Chansler. The Hadoop Distributed File System. 2010

- GFS
  - S. Ghemawat, H. Gobioff, S. Leung. The Google File System. 2003