CS60021: Scalable Data Mining

Sourangshu Bhattacharya

In this Lecture:

- Outline:
  - HDFS – Motivation
  - HDFS – User commands
  - HDFS – System architecture
  - HDFS – Implementation details

Sourangshu Bhattacharya
Computer Science and Engg.

Hadoop Map Reduce

- Provides:
  - Automatic parallelization and Distribution
  - Fault Tolerance
  - Methods for interfacing with HDFS for colocation of computation and storage of output.
  - Status and Monitoring tools
  - API in Java
  - Ability to define the mapper and reducer in many languages through Hadoop streaming.

What is Hadoop ?

- A scalable fault-tolerant distributed system for data storage and processing.

- Core Hadoop:
  - Hadoop Distributed File System (HDFS)
  - Hadoop YARN: Job Scheduling and Cluster Resource Management
  - Hadoop Map Reduce: Framework for distributed data processing.

- Open Source system with large community support.

  https://hadoop.apache.org/

HDFS

What's HDFS

- HDFS is a distributed file system that is fault tolerant, scalable and extremely easy to expand.
- HDFS is the primary distributed storage for Hadoop applications.
- HDFS provides interfaces for applications to move themselves closer to data.
- HDFS is designed to 'just work', however a working knowledge helps in diagnostics and improvements.

HDFS

- Design Assumptions
  - Hardware failure is the norm.
  - Streaming data access.
  - Write once, read many times.
  - High throughput, not low latency.
  - Large datasets.
- Characteristics:
  - Performs best with modest number of large files
  - Optimized for streaming reads
  - Layer on top of native file system.

HDFS

- Data is organized into file and directories.
- Files are divided into blocks and distributed to nodes.
- Block placement is known at the time of read
  - Computation moved to same node.
- Replication is used for:
  - Speed
  - Fault tolerance
  - Self healing.

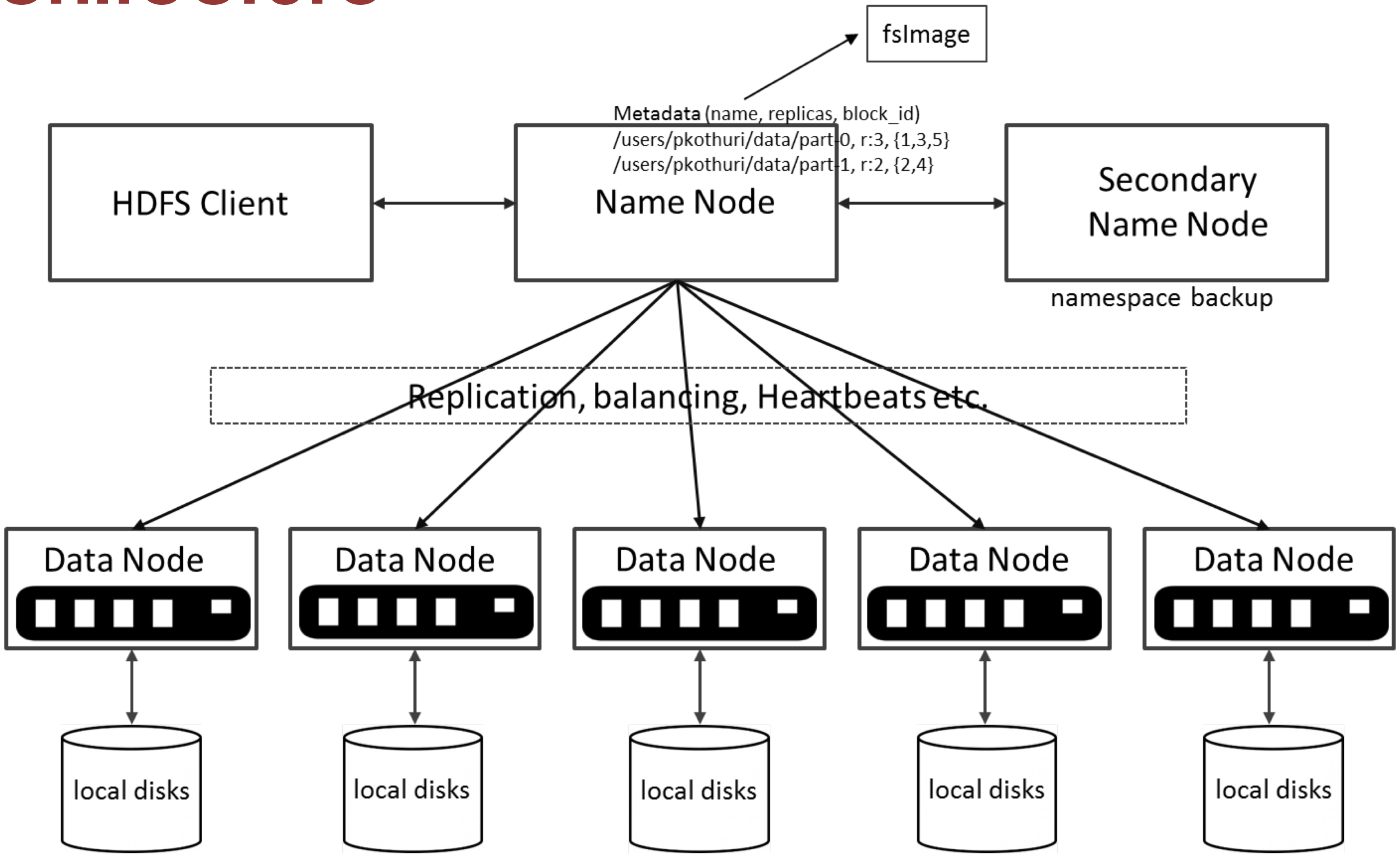
Components of HDFS

There are two (*and a half*) types of machines in a HDFS cluster

- NameNode :– is the heart of an HDFS filesystem, it maintains and manages the file system metadata. E.g; what blocks make up a file, and on which datanodes those blocks are stored.

- DataNode :- where HDFS stores the actual data, there are usually quite a few of these.

HDFS Architecture



HDFS – User Commands (dfs)

List directory contents

```
hdfs dfs –ls
hdfs dfs -ls /
hdfs dfs -ls -R /var
```

Display the disk space used by files

```
hdfs dfs -du /hbase/data/hbase/namespace/
hdfs dfs -du -h /hbase/data/hbase/namespace/
hdfs dfs -du -s /hbase/data/hbase/namespace/
```

HDFS – User Commands (dfs)

Copy data to HDFS

```
hdfs dfs -mkdir tdata
hdfs dfs -ls
hdfs dfs -copyFromLocal tutorials/data/geneva.csv tdata
hdfs dfs -ls –R
```

Copy the file back to local filesystem

```
cd tutorials/data/
hdfs dfs –copyToLocal tdata/geneva.csv geneva.csv.hdfs
md5sum geneva.csv geneva.csv.hdfs
```

HDFS – User Commands (acls)

List acl for a file

```
hdfs dfs -getfacl tdata/geneva.csv
```

List the file statistics – (%r – replication factor)

```
hdfs dfs -stat "%r" tdata/geneva.csv
```

Write to hdfs reading from stdin

```
echo "blah blah blah" | hdfs dfs -put - tdataset/tfile.txt
hdfs dfs -ls -R
hdfs dfs -cat tdataset/tfile.txt
```

Goals of HDFS

- **Very Large Distributed File System**
  – 10K nodes, 100 million files, 10 PB
- **Assumes Commodity Hardware**
  – Files are replicated to handle hardware failure
  – Detect failures and recovers from them
- **Optimized for Batch Processing**
  – Data locations exposed so that computations can move to where data resides
  – Provides very high aggregate bandwidth
- **User Space, runs on heterogeneous OS**

Distributed File System

- **Single Namespace for entire cluster**
- **Data Coherency**
  – Write-once-read-many access model
  – Client can only append to existing files
- **Files are broken up into blocks**
  – Typically 128 MB block size
  – Each block replicated on multiple DataNodes
- **Intelligent Client**
  – Client can find location of blocks
  – Client accesses data directly from DataNode

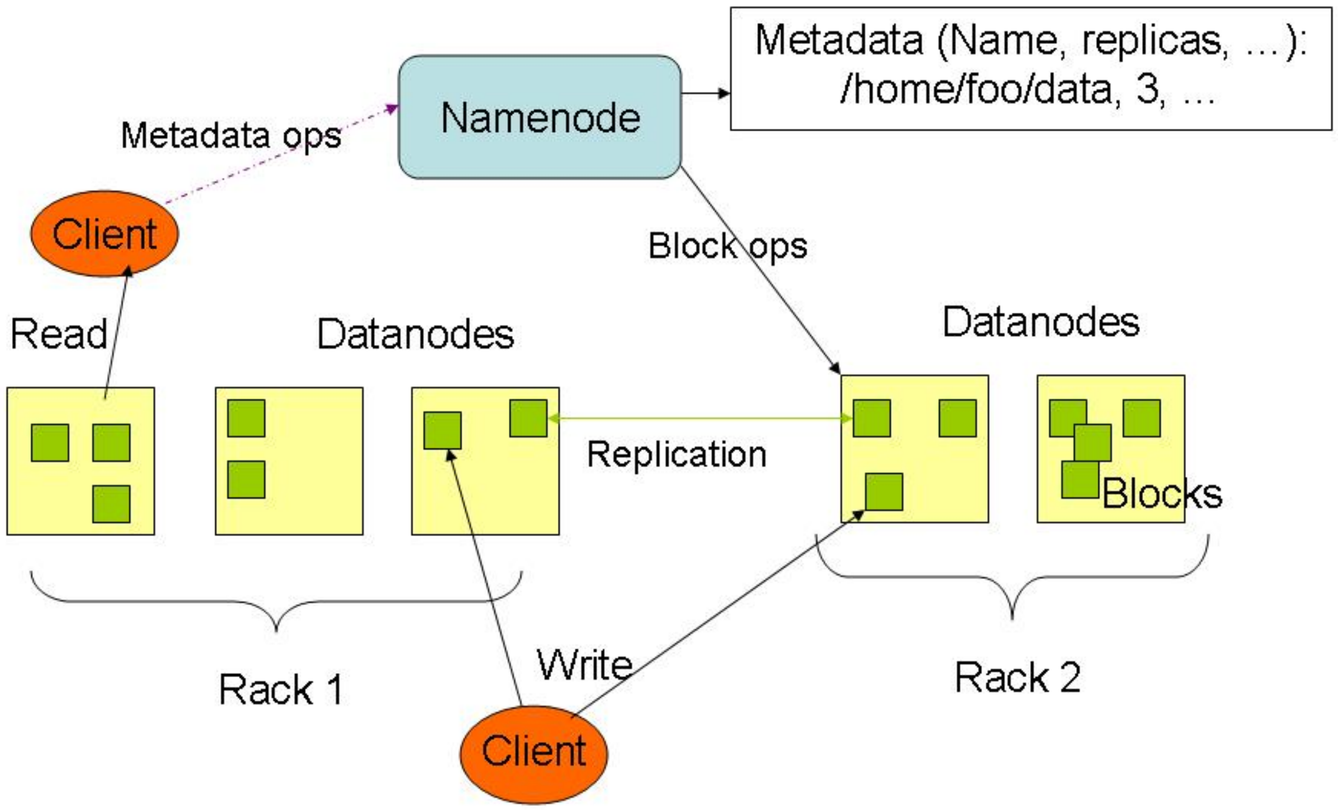
NameNode Metadata

- **Meta-data in Memory**
  – The entire metadata is in main memory
  – No demand paging of meta-data
- **Types of Metadata**
  – List of files
  – List of Blocks for each file
  – List of DataNodes for each block
  – File attributes, e.g creation time, replication factor
- **A Transaction Log**
  – Records file creations, file deletions. etc

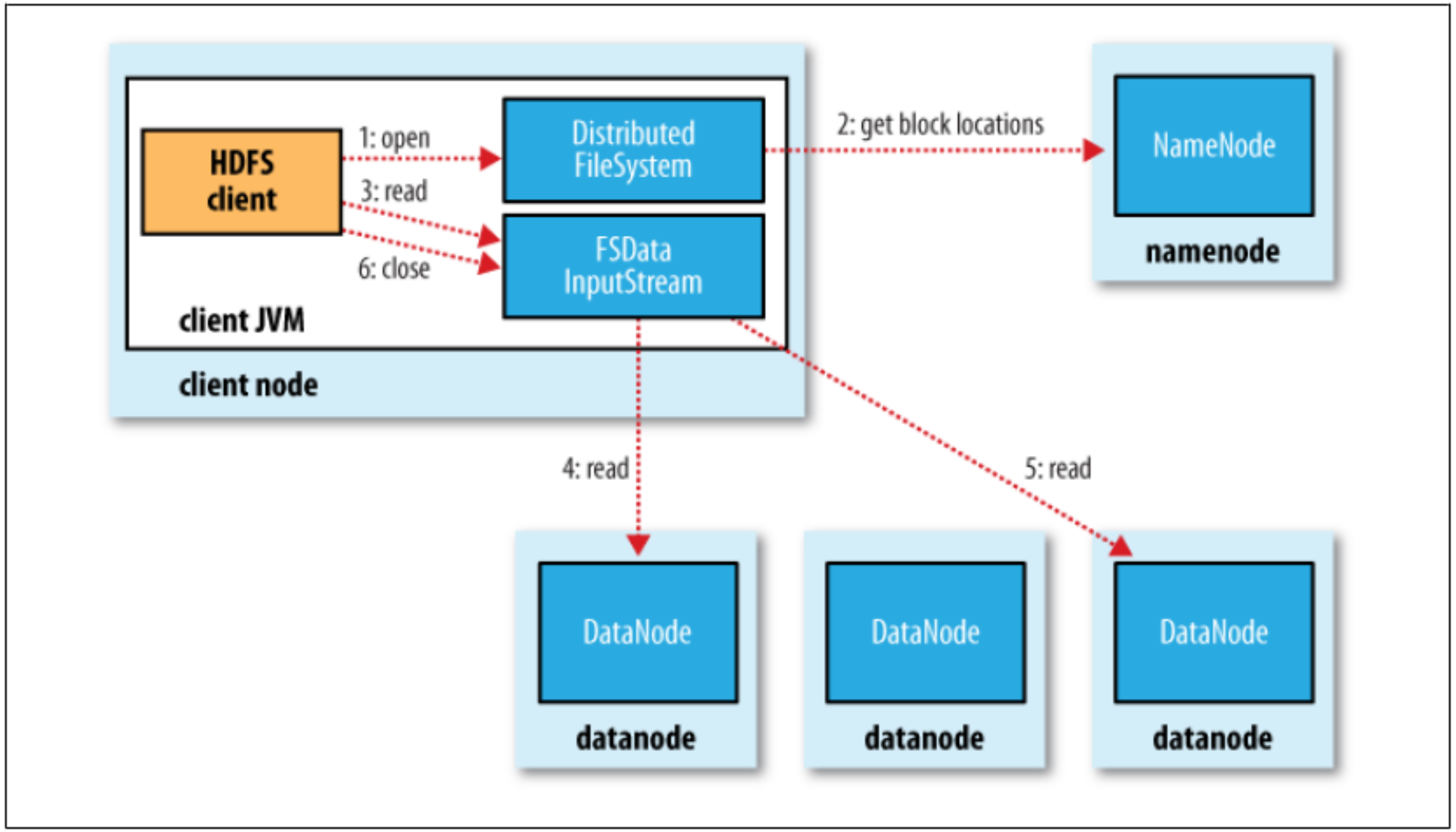
DataNode

- **A Block Server**
  - Stores data in the local file system (e.g. ext3)
  - Stores meta-data of a block (e.g. CRC)
  - Serves data and meta-data to Clients
- **Block Report**
  - Periodically sends a report of all existing blocks to the NameNode
- **Facilitates Pipelining of Data**
  - Forwards data to other specified DataNodes

HDFS Architecture

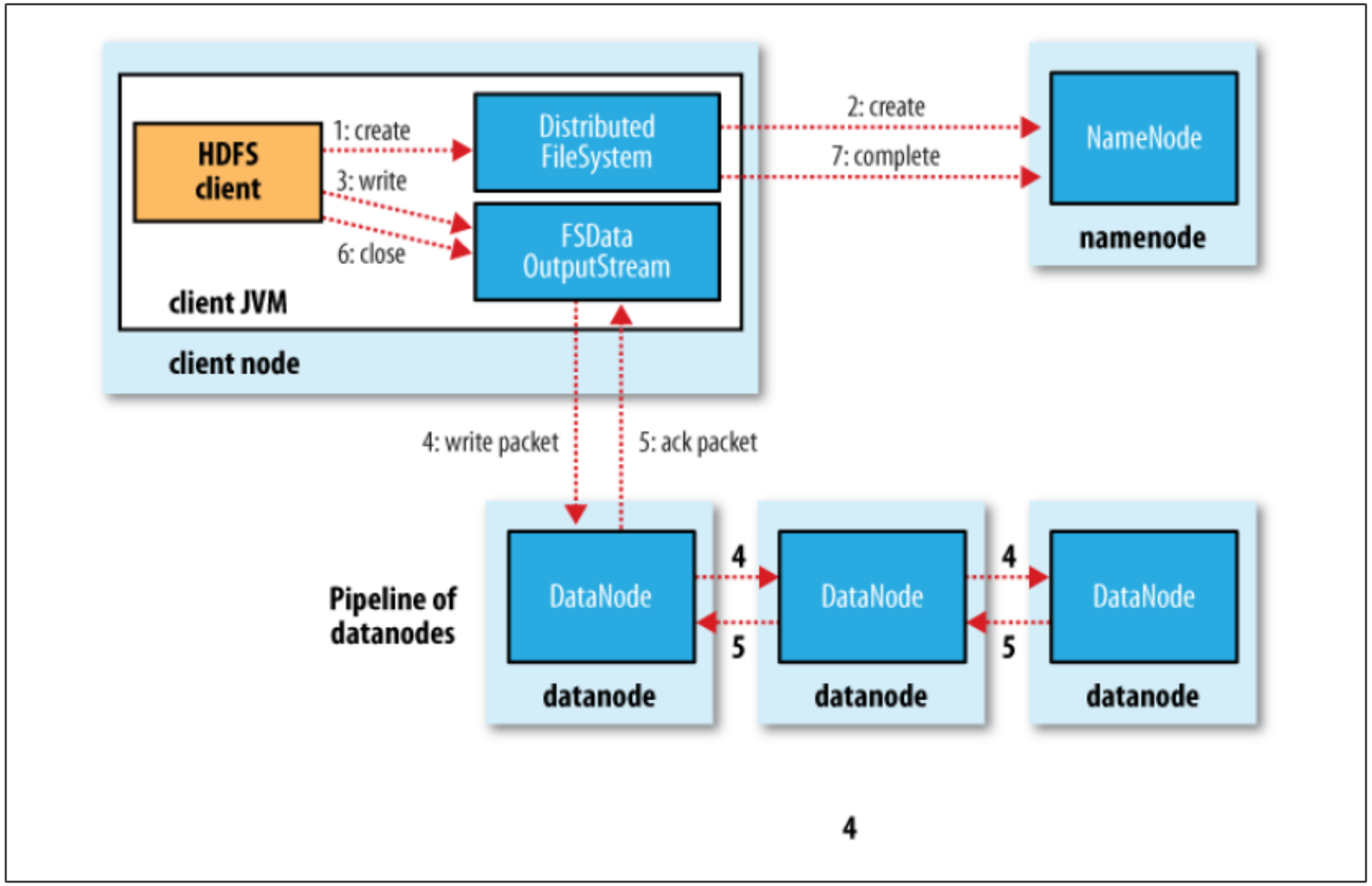


HDFS read client



Source: Hadoop: The Definitive Guide

HDFS write Client



Source: Hadoop: The Definitive Guide

Block Placement

- **Current Strategy**

  -- One replica on local node

  -- Second replica on a remote rack

  -- Third replica on same remote rack

  -- Additional replicas are randomly placed

- **Clients read from nearest replica**

- **Would like to make this policy pluggable**

NameNode Failure

- **A single point of failure**
- **Transaction Log stored in multiple directories**
  - A directory on the local file system
  - A directory on a remote file system (NFS/CIFS)
- **Need to develop a real HA solution**

Data Pipelining

- Client retrieves a list of DataNodes on which to place replicas of a block
- Client writes block to the first DataNode
- The first DataNode forwards the data to the next DataNode in the Pipeline
- When all replicas are written, the Client moves on to write the next block in file

Conclusion:

- We have seen:
  - The structure of HDFS.
  - The shell commands.
  - The architecture of HDFS system.
  - Internal functioning of HDFS.

References:

- Jure Leskovec, Anand Rajaraman, Jeff Ullman. **Mining of Massive Datasets.** *2nd edition. - Cambridge University Press.* *http://www.mmds.org/*

- Tom White. **Hadoop: The definitive Guide.** Oreilly Press.