JOHNS HOPKINS
CAREY BUSINESS SCHOOL

HDFS and YARN

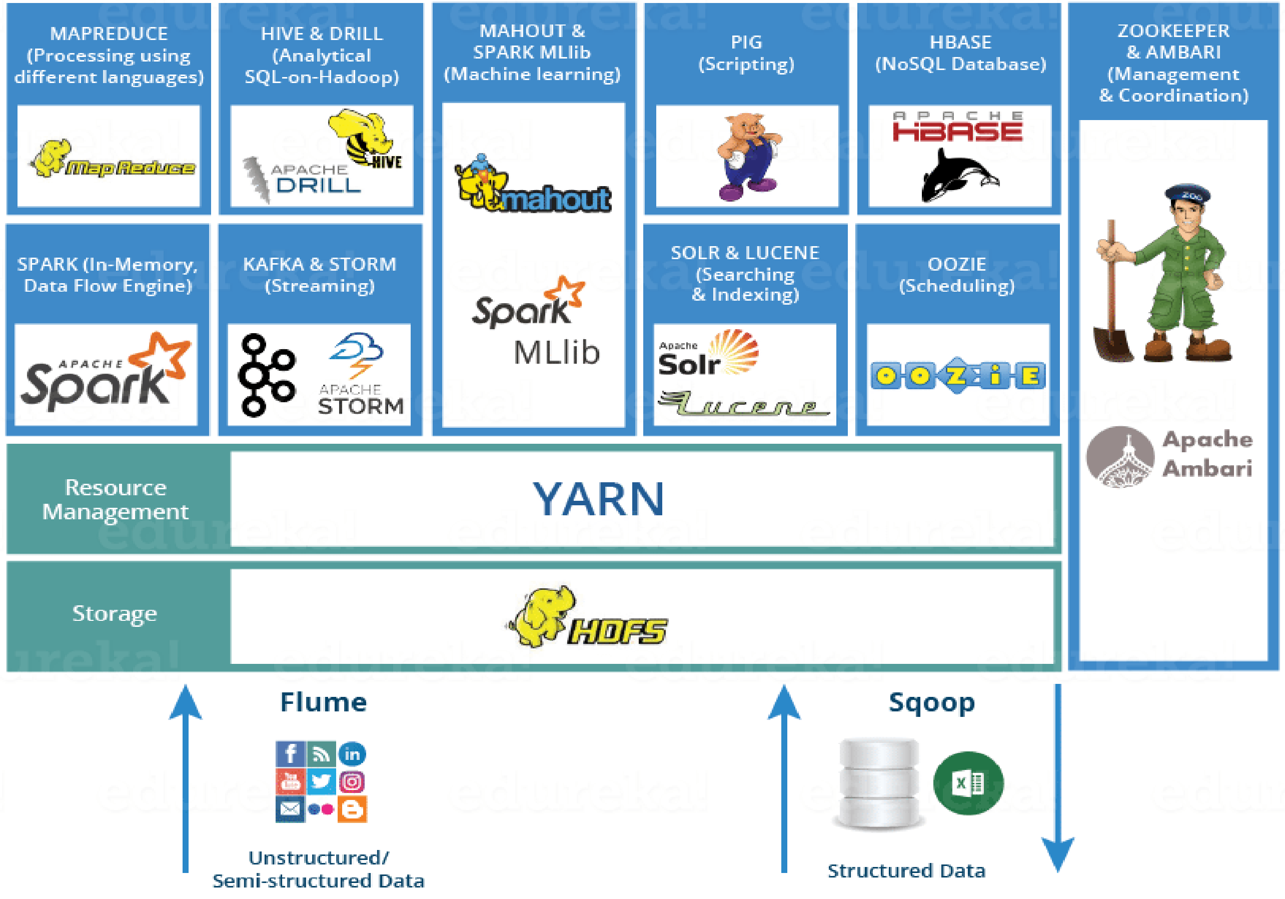
BU.330.740 Large Scale Computing on the Cloud

Minghong Xu, PhD.
Associate Professor

Hadoop Overview

- An open source framework for writing and running distributed applications that process large amount of data

- Google: first to publicize MapReduce for scaled data processing
- Doug Cutting: develop the first version of Hadoop



MAPREDUCE (Processing using different languages)
Map Reduce
HIVE & DRILL (Analytical SQL-on-Hadoop)
APACHE DRILL
HIVE
MAHOUT & SPARK MLlib (Machine learning)
mahout
Spark MLlib
PIG (Scripting)
HBASE (NoSQL Database)
APACHE HBASE
ZOOKEEPER & AMBARI (Management & Coordination)
Apache Ambari
SPARK (In-Memory, Data Flow Engine)
APACHE Spark
KAFKA & STORM (Streaming)
APACHE STORM
SOLR & LUCENE (Searching & Indexing)
Apache Solr
Lucene
OOZIE (Scheduling)
OOZIE
Resource Management
YARN
Storage
HDFS
Flume
Unstructured/ Semi-structured Data
Sqoop
Structured Data

Key Components

- Distributed File System: HDFS
  - Most low-level knowledge of this course
- Operating System: YARN
- Original Distributed Processing Engine: MapReduce
- Improved MapReduce: Spark
- Distributed Query Language: Hive
- Distributed Scripting Language: Apache Pig

JOHNS HOPKINS
CAREY BUSINESS SCHOOL

Distributed File System

Hadoop Distributed File System

- ***Problem 1:** Data is too big to store on one machine.*
- **Solution:** Store the data on multiple machines!

- ***Problem 2:** Very high end machines are too expensive*
- **Solution:** Run on commodity hardware!

- ***Problem 3:** Commodity hardware can fail*
- **Solution:** Software is intelligent enough to handle hardware failure!

- ***Problem 4:** What happens to the data if the machine storing the data fails?*
- **Solution:** Replicate the data!

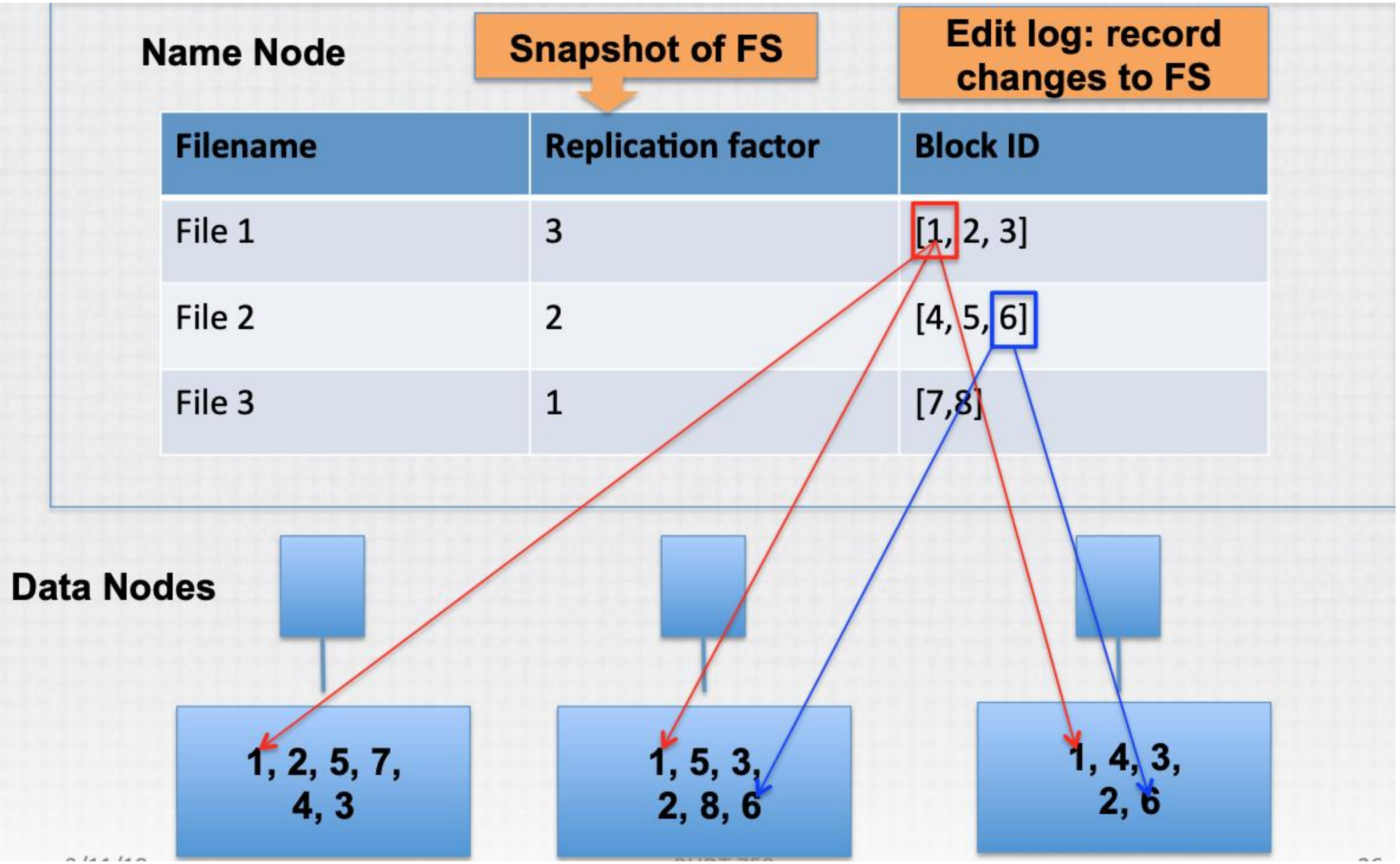
Solutions (Cont.)

- ***Problem 5:*** *How can distributed machines organize the data in a coordinated way?*
- Insights from your own team work experience…
  - *How to assign tasks to different workers in an efficient way?*
  - *What happens if tasks fail?*
  - *How do workers exchange results?*
  - *How to synchronize distributed tasks allocated to different workers?*
- **Solution**: primary-secondary structure

HDFS Primary-Secondary Architecture

- Single NameNode
  - Sometimes a backup: secondary NameNode
- Many (Thousands) DataNodes
- Files are split into fixed sized blocks and stored on data nodes
- Data blocks are replicated for fault tolerance and fast access
  - By default: 3

Illustration



Name Node

Snapshot of FS

Edit log: record changes to FS

Filename	Replication factor	Block ID
File 1	3	[1, 2, 3]
File 2	2	[4, 5, 6]
File 3	1	[7,8]

Adapt from K. Zhang's notes, Spring 2019

NameNode

- Manages file system namespace, and file metadata
- Mapping file to list of blocks
- Mapping of datanode to list of blocks
- Monitor datanode health
- Replicate missing blocks

DataNode

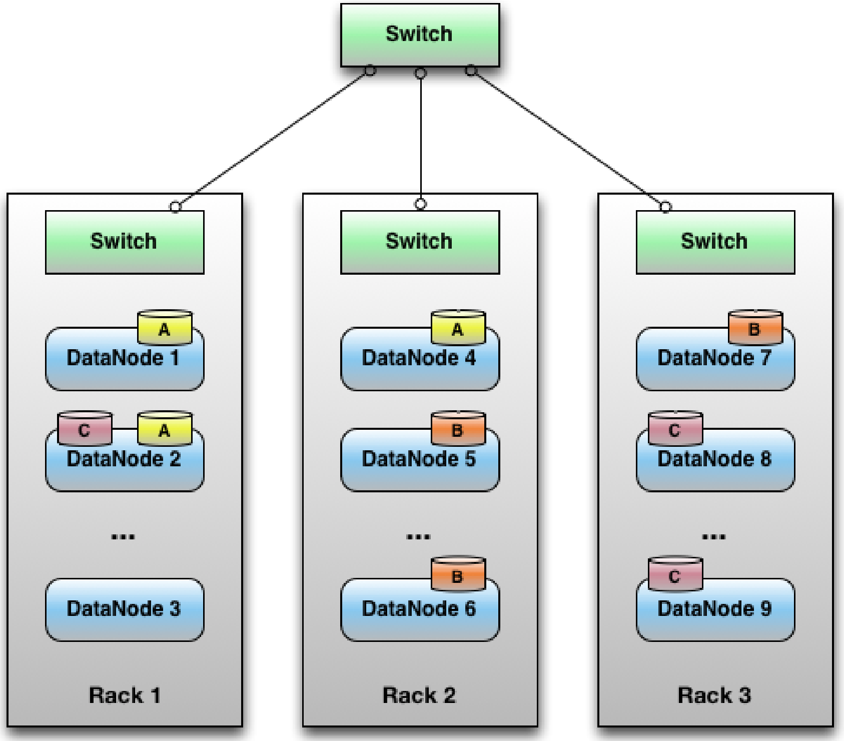
- Handle block storage & block integrity
- Periodically send reports to NameNode
- Clients access the blocks directly from data nodes

- *Q: Why not access blocks through NN?*
- Reasons:
  - Prevent NN from being the bottleneck of the cluster
  - Allow HDFS to scale to large number of concurrent clients
  - Spread the data traffic across the cluster

Data Replicate

- **Frist replica** is put on one node in the local rack
- **Second one** is put on a node in a different (remote) rack
- **Third one** is on a different node in the same remote rack
- **Additional** replicas are randomly placed

- Objectives: load balancing, fast access, fault tolerance.

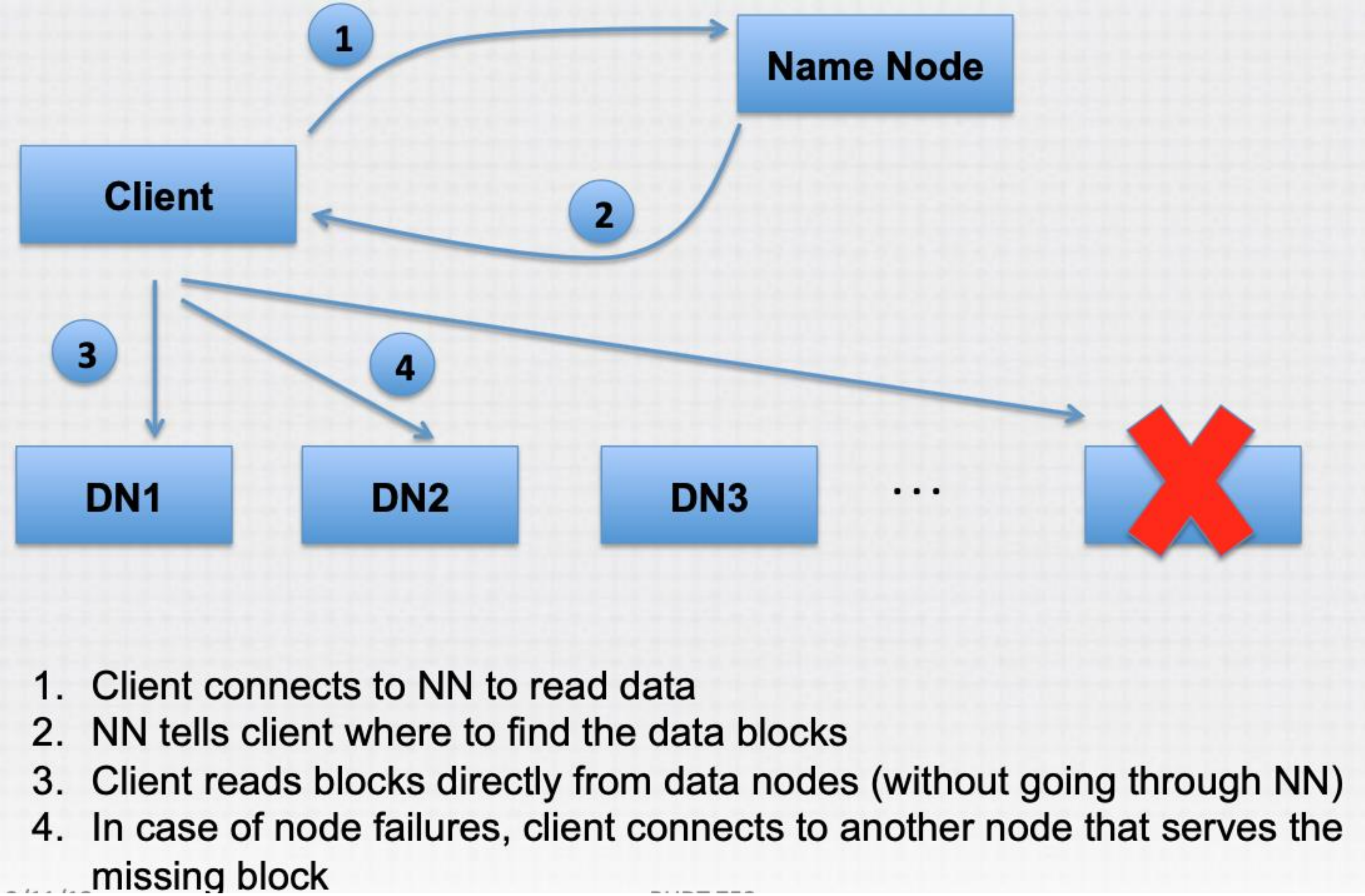


HDFS Network Topology

- The critical resource in HDFS is **bandwidth**, distance is defined based on that
- Measuring bandwidths between any pair of nodes is too complex
- **Basic Idea:**
  - Processes on the same node
  - Different nodes on the same rack
  - Nodes on different racks in the same data center (cluster)
  - Nodes in different data centers

Bandwidth becomes less

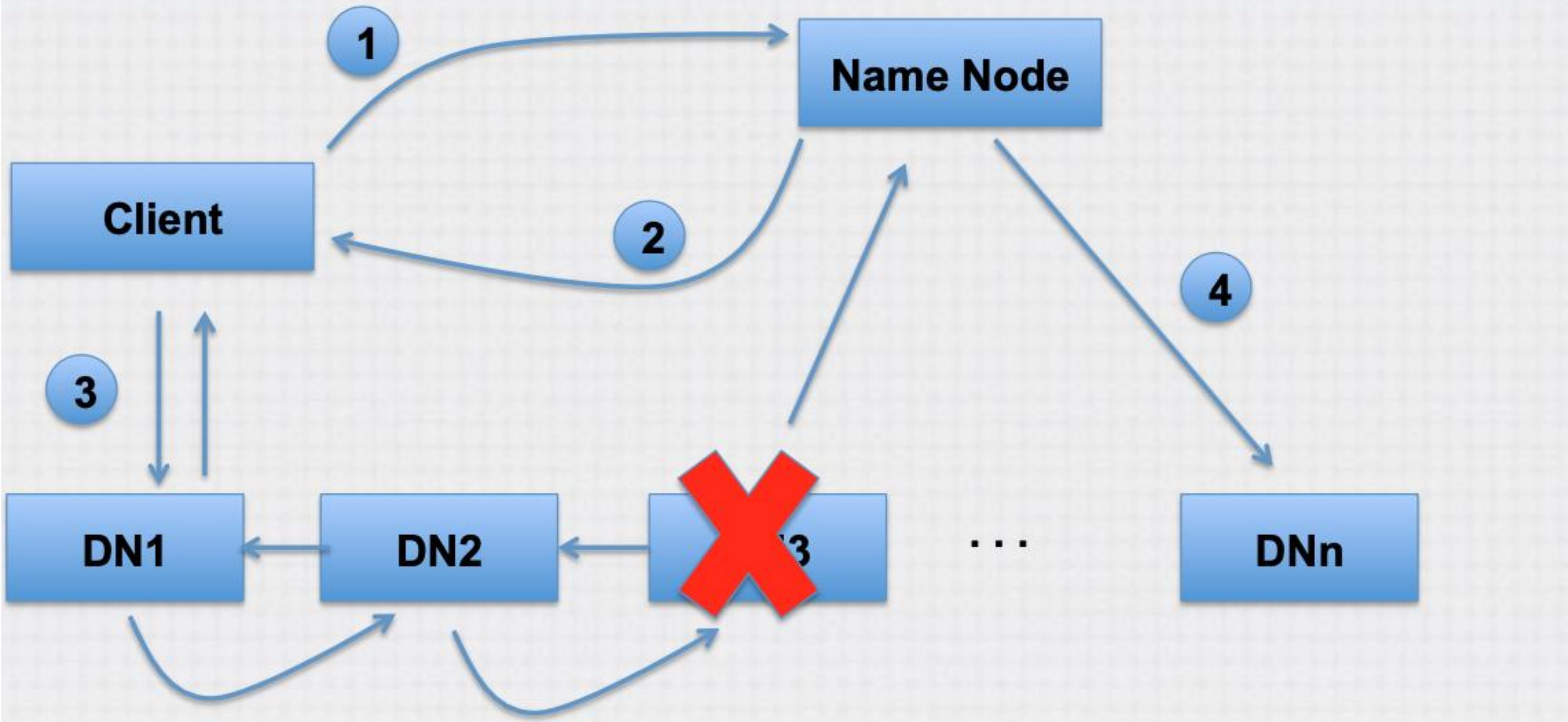
Data Read



1. Client connects to NN to read data
2. NN tells client where to find the data blocks
3. Client reads blocks directly from data nodes (without going through NN)
4. In case of node failures, client connects to another node that serves the missing block

Adapt from K. Zhang’s notes, Spring 2019

Data Write



1. Client connects to NN to write data
2. NN tells client write these data nodes
3. Client writes blocks directly to data nodes with desired replication factor
4. In case of node failures, NN will figure it out and replicate the missing blocks

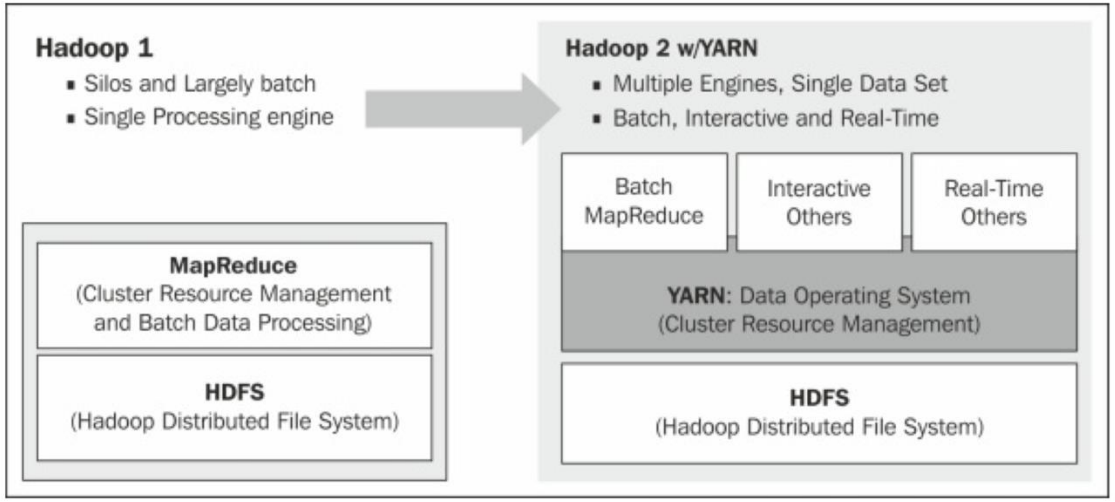
Adapt from K. Zhang’s notes, Spring 2019

JOHNS HOPKINS
CAREY BUSINESS SCHOOL

Yet Another Resource Negotiator

What is YARN?

- Cluster resource management system for Hadoop
- Introduced in Hadoop 2 to improve MapReduce implementation
- Connect between high level applications (Spark, HBase) and low level Hadoop environment
- Large-scale, distributed <u>operating system</u> for big data applications


Hadoop 1
▪ Silos and Largely batch
▪ Single Processing engine
Hadoop 2 w/YARN
▪ Multiple Engines, Single Data Set
▪ Batch, Interactive and Real-Time
Batch MapReduce
Interactive Others
Real-Time Others
MapReduce (Cluster Resource Management and Batch Data Processing)
YARN: Data Operating System (Cluster Resource Management)
HDFS (Hadoop Distributed File System)
HDFS (Hadoop Distributed File System)

Components

Resource Manager

- 1 per cluster
- Track resource in a cluster, schedule applications
- Single point of failure, but can be restored in case of failures

Application Master

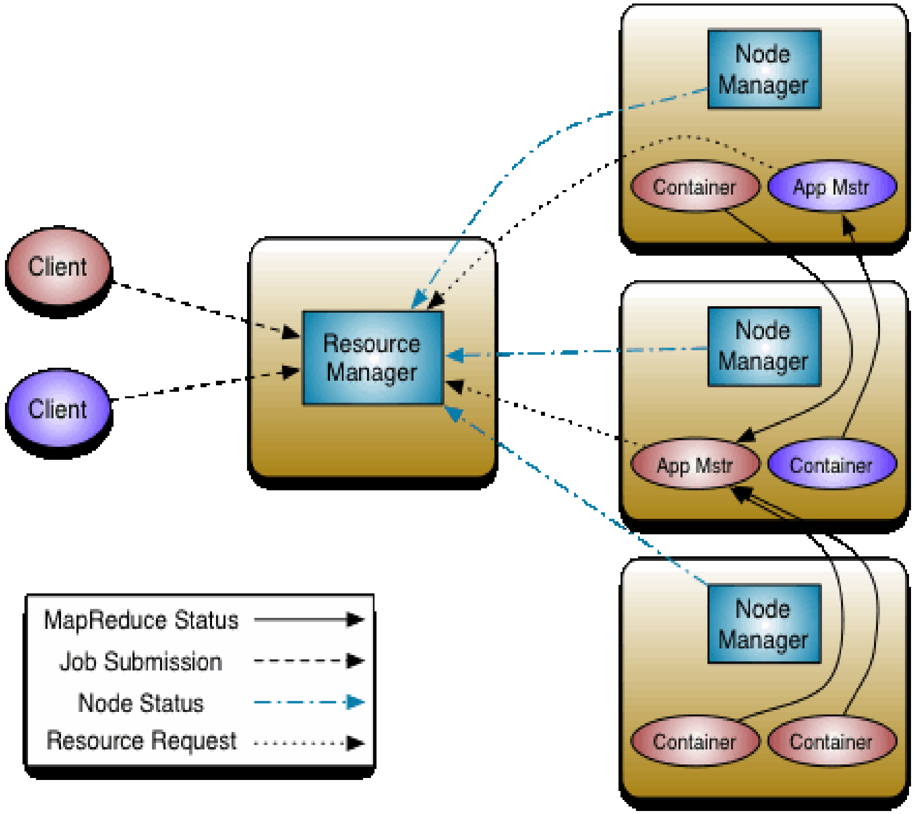
- Run in a separate process on a datanode
- One instance per application
- Send status and resource needs to RM
- Can run lightweight task on the same node

Node Manager

- 1 per node
- Monitor node resources such as CPU, Memory, Disk space, Network etc.
- Collect log data, report to Resource Manager

Steps to run Yarn application

- Client make request to Resource Manager to run application
- Resource Manager request Node Manager to allocate container for creating Application Master instance on available node
  - Container: basic unit of hardware allocation
- When Application Master instance already run, it sends request (heartbeat, resource needs) to Resource Manager



Scheduling in Yarn

- Important task since Hadoop cluster is shared between many users and tasks; which to run first?
- Hadoop 1: FIFO scheduler with fixed cpu, memory, disk count
- Hadoop 2: allow Capacity and Fair schedulers with dynamic cpu, memory and disk count
- Yarn supports all three

Illustrations

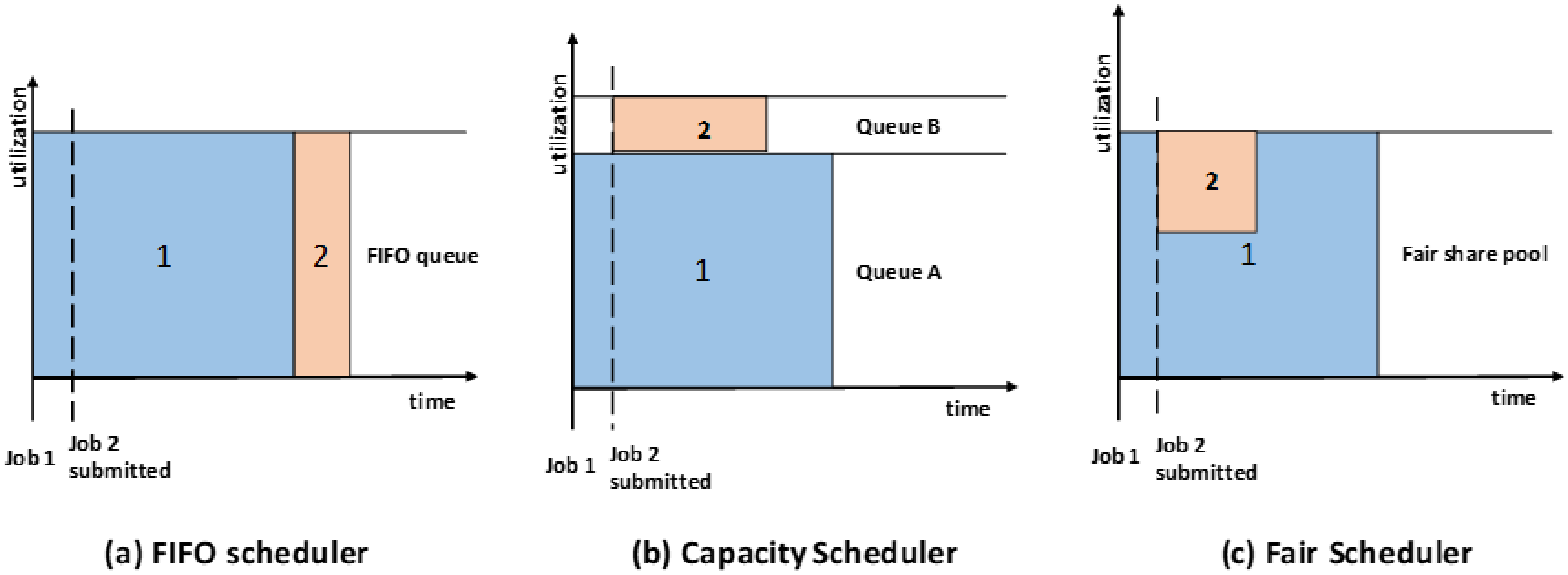


Figure 1: YARN Schedulers' cluster utilization vs. time