Big Data Technology
Core Hadoop: HDFS-YARN Internals

Eshcar Hillel
Yahoo!

Ronny Lempel
Outbrain

*Based on slides by Edward Bortnikov & Ronny Lempel
Roadmap

- Previous class – Map-Reduce Motivation

- This class – HDFS and YARN overview
  - HDFS - Distributed Filesystem
  - YARN - Scheduling and Resource Management (Yet Another Resource Negotiator)
Percolator: Incremental Processing System

HDFS & YARN: The core of Hadoop

Source: http://hortonworks.com/hdp/
Hardware Layout

- Few dozens of nodes
- 1000s nodes
- Node
- Rack
- Rack Switch
- Base Switch

CS236620 Big Data Technology
Example: Yahoo Hadoop Cluster
Core Hadoop: The Big Picture

Source: [http://www.slideshare.net/Hadoop_Summit/hadoop-crash-course-workshop-at-hadoop-summit](http://www.slideshare.net/Hadoop_Summit/hadoop-crash-course-workshop-at-hadoop-summit)

CS236620 Big Data Technology
HDFS 101

- Highly available cost-efficient distributed storage system
- Highly scalable
  - ~5 thousand commodity servers clusters
  - Up to 200 PB data
  - Billions files and block
- Master/Worker architecture
HDFS Architecture

Data servers (DataNodes)

Metadata server (NameNode)
How HDFS Works?

- Files are write-once-read-many
  - Optimized for appends and sequential scans
- Data replication
  - Files split into large blocks (128MB) (why so large?)
  - Each block is replicated at multiple nodes
  - Typically, 3 replicas per block, might be more
- Placement policies
  - Typically, 2 replicas in a single rack, 1 replica in a remote rack (why?)
**NameNode (NN) DataNodes (DNs)**

- **NameNode** manages cluster metadata
  - Namespace tree
  - Blocks to datanodes mapping

- **DataNodes** store data
  - Blocks are stored on local filesystem
  - Send heartbeats to NameNode
  - Get instructions in response
    - Remove/replicate block, shutdown, block report
Data Read and Write Operations

Data servers (DataNodes)

Client

Read

Write

Metadata access

Metadata server (NameNode)

File

CS236620 Big Data Technology
Data Read and Write Operations

- Each replica can serve the data
  - Served from a replica that is closest to the reader

- Single-writer semantics
  - Obtain lease for writing
  - Per-block: get datanode list to host replication
  - Per-block: replication pipeline
NameNode Scalability

- Namenode limits the cluster scale
- Does not scale beyond the size of the NN heap
- The scale of data
  - Past (2007): 100s nodes, 100s TB, millions files
  - Present (2015): 1000s nodes (10x), 200PB (1000x), 100s millions files (100x)
  - Future: 10000s, xxxEB, Billions of files
- (Future) Extend scalability by decoupling namespace and block manager
  - Namespace scalability via key-value stores (HDFS-8286)
  - Block manager amenable to sharding/scale-out (HDFS-5477)
Highly Available NameNode

- Backup NameNode provides redundancy and supports high availability (HA)

Data service (datanode)

ZK

heartbeats

FailoverController

Journal Nodes

Fencing

Monitor health

Metadata (backup NN)

Metadata (primary NN)

Monitor health

CS236620 Big Data Technology
YARN 101

- A framework for job scheduling and resource management
  - Distributes computation across HDFS nodes
  - Enables a variety of concurrent data access applications
- Master/worker architecture
YARN Architecture

- **Central ResourceManager (RM)**
  - Constraints-based scheduler allocating resources
    - Capacity, queues, etc.
  - No monitoring or status tracking

- **Per-Node NodeManager (NM)**
  - Manages available resources in a single node
  - Launches ApplicationContainer monitors resources usage
    - cpu, memory, disk, network
  - Reports to RM

- **Per-application ApplicationMaster (AM)**
  - Negotiates resources from RM
  - Works with NMs to execute and monitor tasks
YARN Architecture

Client → ResourceManager → Node Manager

Job submission → AppMaster → Job status

Resource request → Node Manager → Resource request

Node status → Node Manager → Node status

MapReduce status → Node Manager → MapReduce status

CS236620 Big Data Technology
Highly Available Resource Manager

- Realized through a primary/backup architecture
  - Primary RM is Active
  - Backup RM is in Standby mode waiting to take over should anything happen to the Active

CS236620 Big Data Technology
Multi-Tenancy

- MR was not designed for multi-user systems
  - Hard to accommodate heterogeneous requirements
    - E.g., production batches vs research ad-hoc queries
  - No real resource protection
- Take 1: build a cluster per user
- Take 2: one platform to rule them all
  - Scales much better
  - Requires OS-style resource management
  - Either build support into MR, or use cloud virtualization
Summary

- Scalable computing stack by decoupling
  - Distributed filesystem
  - Job scheduling and resource management
- Master nodes (NN and RM) are highly available
- Multitenancy: supports multiple users and applications
Further Reading

- Google Filesystem (GFS) paper
- Apache Hadoop
- Hadoop YARN
Next...

- Implementing Map-Reduce atop YARN