

# Big Data Engineering

# Other Tools and Libraries

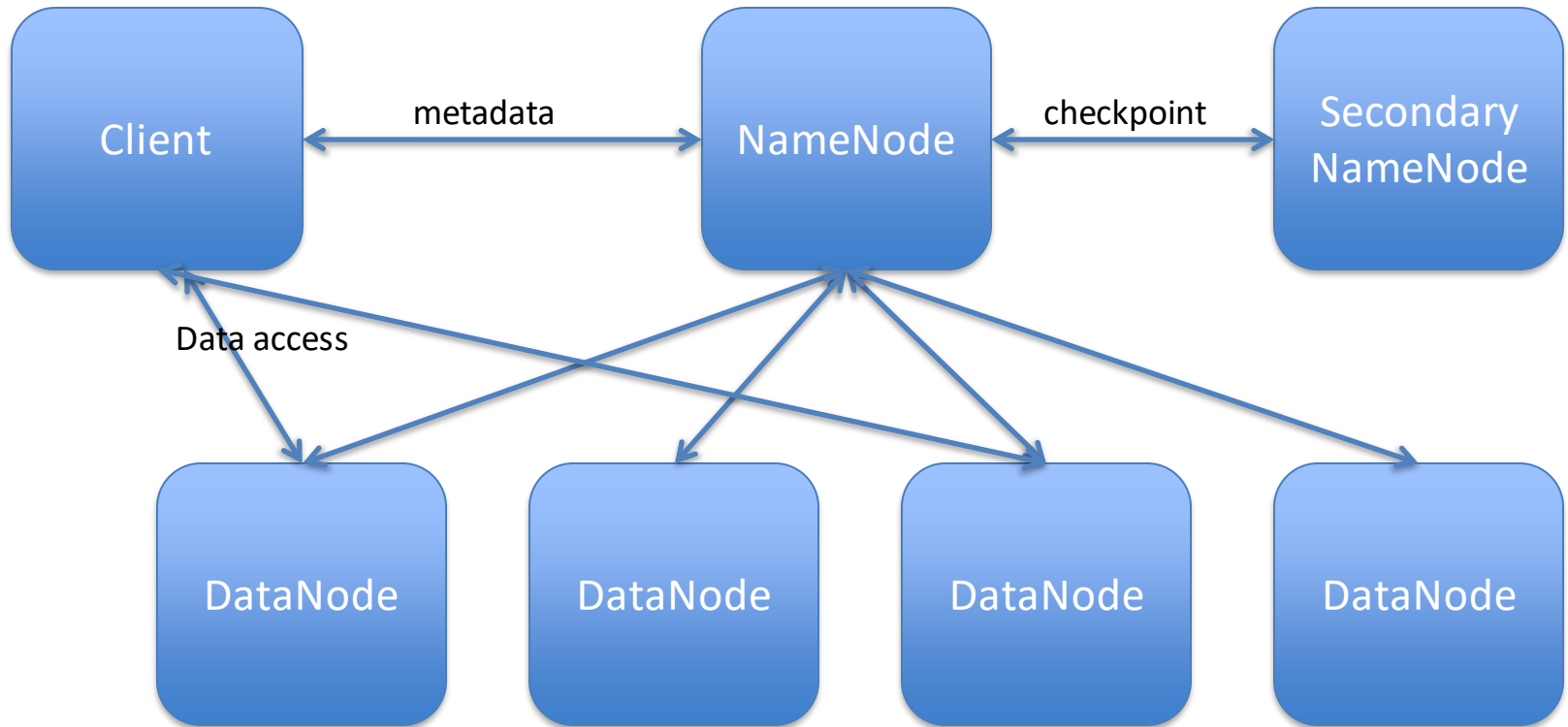
Adam Hill

April 2023



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# HDFS in a nutshell



# HDFS inspiration

- Google File System

- Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. 2003. The Google file system. In Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP '03). ACM, New York, NY, USA, 29-43.

## The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung  
Google\*

### ABSTRACT

We have designed and implemented the Google File System, a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients.

While sharing many of the same goals as previous distributed file systems, our design has been driven by observations of our application workloads and technological environment, both current and anticipated, that reflect a marked departure from some earlier file system assumptions. This has led us to reexamine traditional choices and explore radically different design points.

The file system has successfully met our storage needs. It is widely deployed within Google as the storage platform

### 1. INTRODUCTION

We have designed and implemented the Google File System (GFS) to meet the rapidly growing demands of Google's data processing needs. GFS shares many of the same goals as previous distributed file systems such as performance, scalability, reliability, and availability. However, its design has been driven by key observations of our application workloads and technological environment, both current and anticipated, that reflect a marked departure from some earlier file system design assumptions. We have reexamined traditional choices and explored radically different points in the design space.

First, component failures are the norm rather than the exception. The file system consists of hundreds or even thousands of storage machines built from inexpensive com-



# HDFS overview

- Good for streaming access to large files, reliability, scale
- Not good for random access, small files
- Blocks of data 64Mb in size (configurable)
- Each block can be replicated across multiple data nodes for High Availability (HA)



# HDFS Usage

- Spotify has 1600+ nodes, storing 60+ petabytes of data
  - <https://www.usenix.org/system/files/conference/fast17/fast17-niazi.pdf>
- One of Facebook's largest clusters (based on HDFS) holds more than 100 PB of data, processing more than 60,000 Hive queries a day
  - <https://www.facebook.com/notes/facebook-engineering/under-the-hood-scheduling-mapreduce-jobs-more-efficiently-with-corona/>



# HopFS

- HopFS is a drop-in replacement for HDFS, based on HDFS v2.0.4.

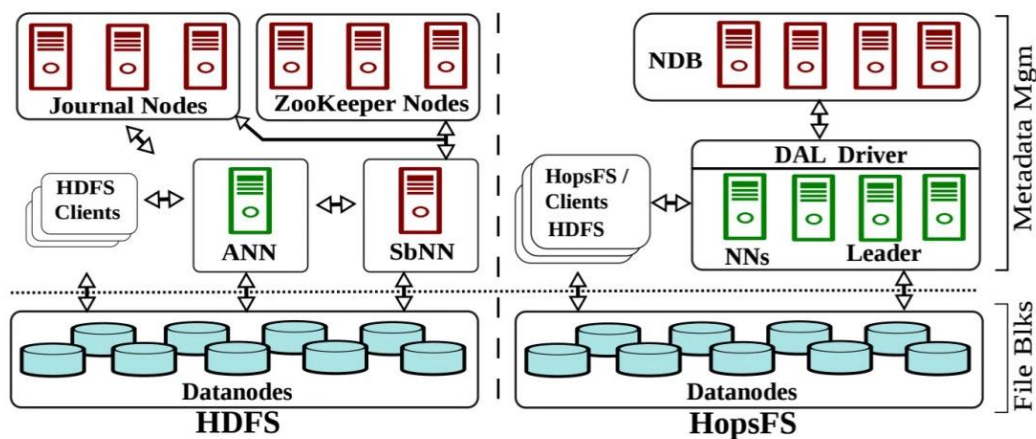


Figure 1: System architecture for HDFS and HopFS. For high availability, HDFS requires an Active NameNode (ANN), at least one Standby NameNode (SbNN), at least three Journal Nodes for quorum-based replication of the write ahead log of metadata changes, and at least three ZooKeeper instances for quorum based coordination. HopFS supports multiple stateless namenodes that access the meta-data stored in NDB database nodes.

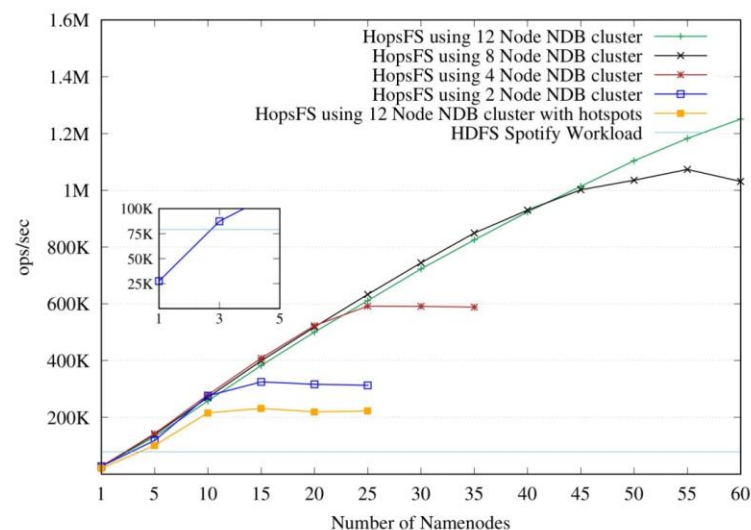
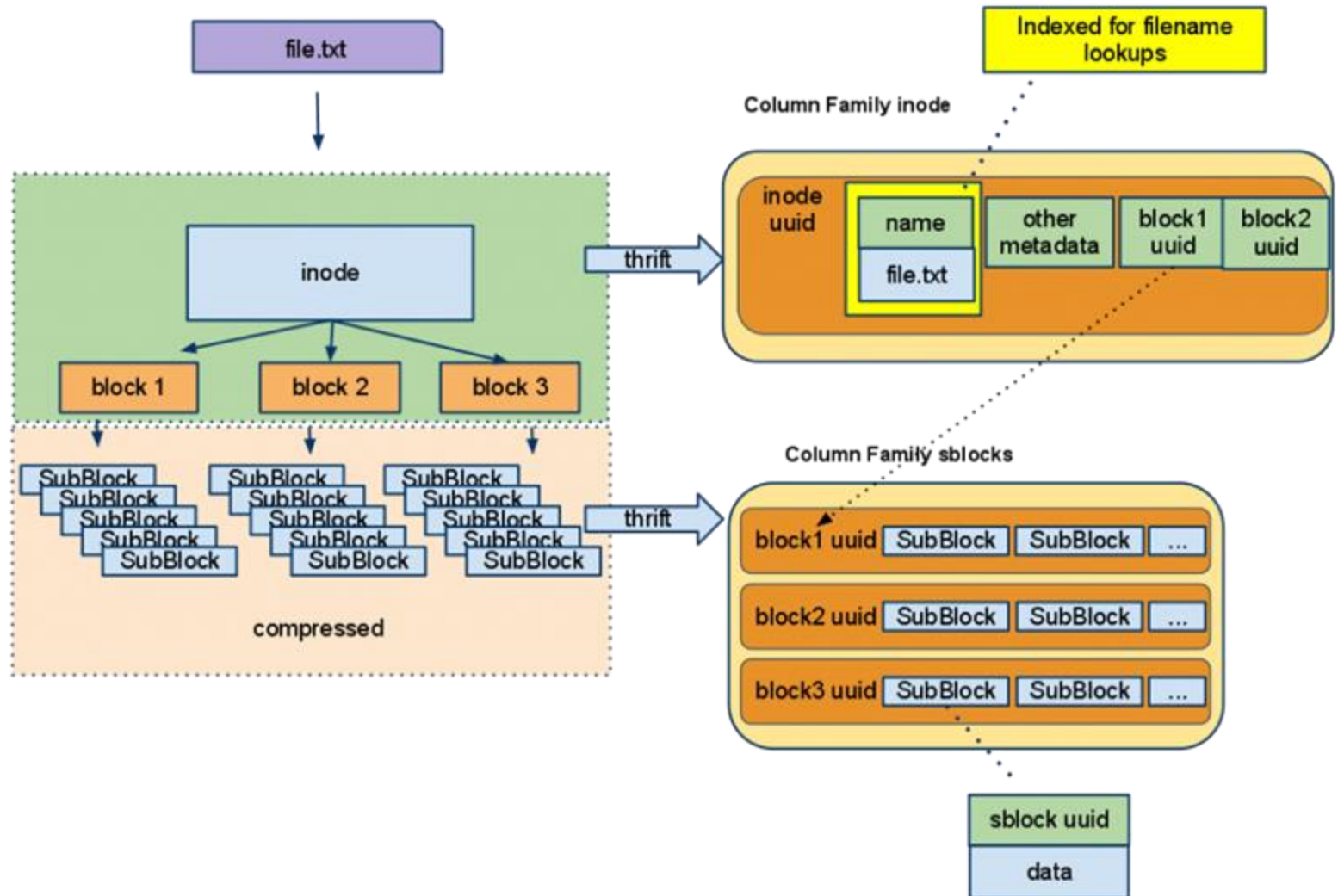


Figure 6: HopsFS and HDFS throughput for Spotify workload.

# CassandraFS (not open source)



# Amazon S3

- Simple Storage Service
- Unlimited storage of files
  - Up to 5 terabytes each
  - Stored in named “buckets”
  - Accessible via AWS APIs or HTTP
  - Authenticated or Public





# Spark packages

- A wide set of plugins
  - Currently 148 community donated plugins
- Data connectors
  - Cassandra, Couchbase, Mongo, CSV, etc
- Machine Learning, Neural networks
- Streaming
- etc



# Using Spark Packages

Automatic download from the web:

```
bin/spark-shell
```

```
--packages com.databricks:spark-csv_2.11:1.2.0
```



# Locality

- Spark understands the locality of data:
  - Already in memory
  - HDFS location
  - Cassandra location



# Spark Extras

Spark  
SQL

Spark  
Streaming

Spark  
MLlib

SparkR

GraphX

Spark Core

# Spark Extras

- Spark Streaming
  - Realtime analysis in Spark
- Spark MLlib
  - Like Mahout – Machine learning in Spark
- GraphX
  - Graph processing in Spark
- SparkR
  - R statistical analysis on Spark



# Spark MLlib

- Simple stats and correlation testing
- Classification and regression
- Collaborative Filtering
  - Alternating Least Squares
- Clustering
  - k-means, etc
- Frequent Pattern Mining
- Plus more



# MLlib example

```
from pyspark.mllib.fpm import FPGrowth

data = sc.textFile("data/mllib/sample_fpgrowth.txt")

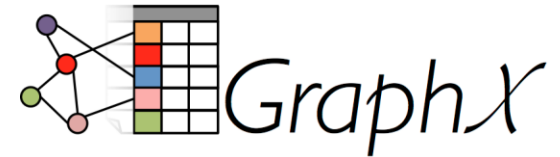
transactions = data.map(lambda line: line.strip().split(' '))

model = FPGrowth.train(transactions, minSupport=0.2,
                        numPartitions=10)

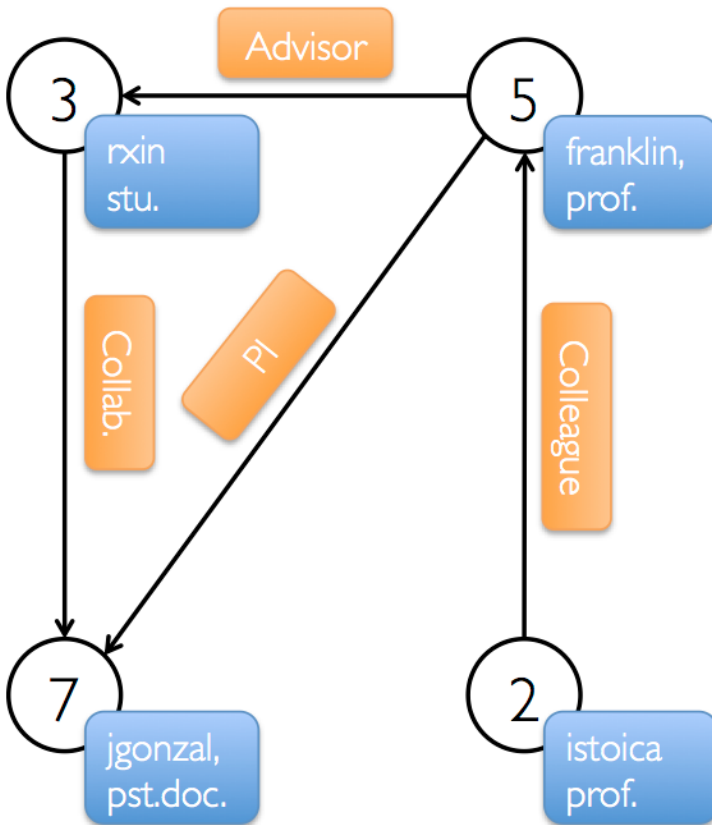
result = model.freqItemsets().collect()
for fi in result:
    print(fi)
```



# GraphX



## Property Graph



## Vertex Table

Id	Property (V)
3	(rxin, student)
7	(jgonzal, postdoc)
5	(franklin, professor)
2	(istoica, professor)

## Edge Table

SrcId	DstId	Property (E)
3	7	Collaborator
5	3	Advisor
2	5	Colleague
5	7	PI





# R

- R is an open source system for statistics and graphics
  - Based on the S language from AT&T Bell Labs
- Supports a wide variety of statistical techniques and graphing tools
- An extensible set of packages that provide extra functions via CRAN
  - The Comprehensive R Archive Network



# SparkR

- A lightweight approach to use Spark from within R
- Also works with MLlib for machine learning
- Allows complex statistical analysis to be done on a Spark cluster



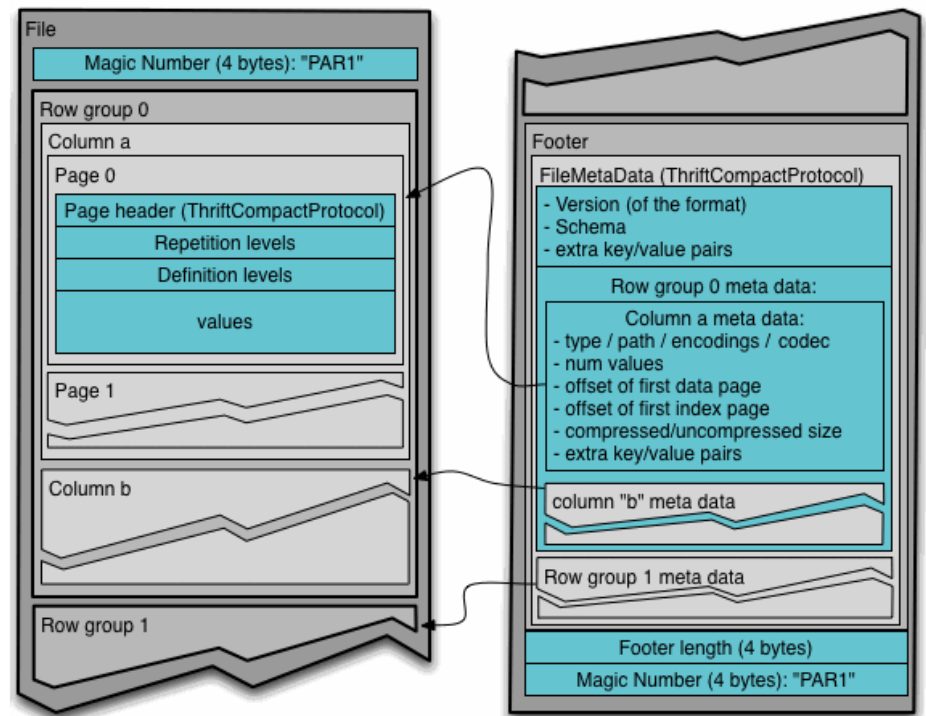
# Apache Avro

- A compact data storage and transmission system
  - Uses schemas of data to ensure it can be read by the receiver
  - Supports dynamic typing
- Used by RPC or data collection systems
  - Fast binary protocols
- Also supports storage
  - Hence used by many Big Data apps including Hadoop and Spark



# Apache Parquet

- Apache Parquet is a columnar data storage model
  - Works with Hadoop, Spark and many others
  - Efficient storage of data
  - Based on another Google system called Dremel



# Cluster management systems for Big Data

- YARN
  - Part of Hadoop but significantly rebuilt since Hadoop 1
- Mesos
  - Popular Apache project
  - Built to be a resource manager for a complete datacenter
    - Supports many workloads (e.g. Docker as well as Spark)

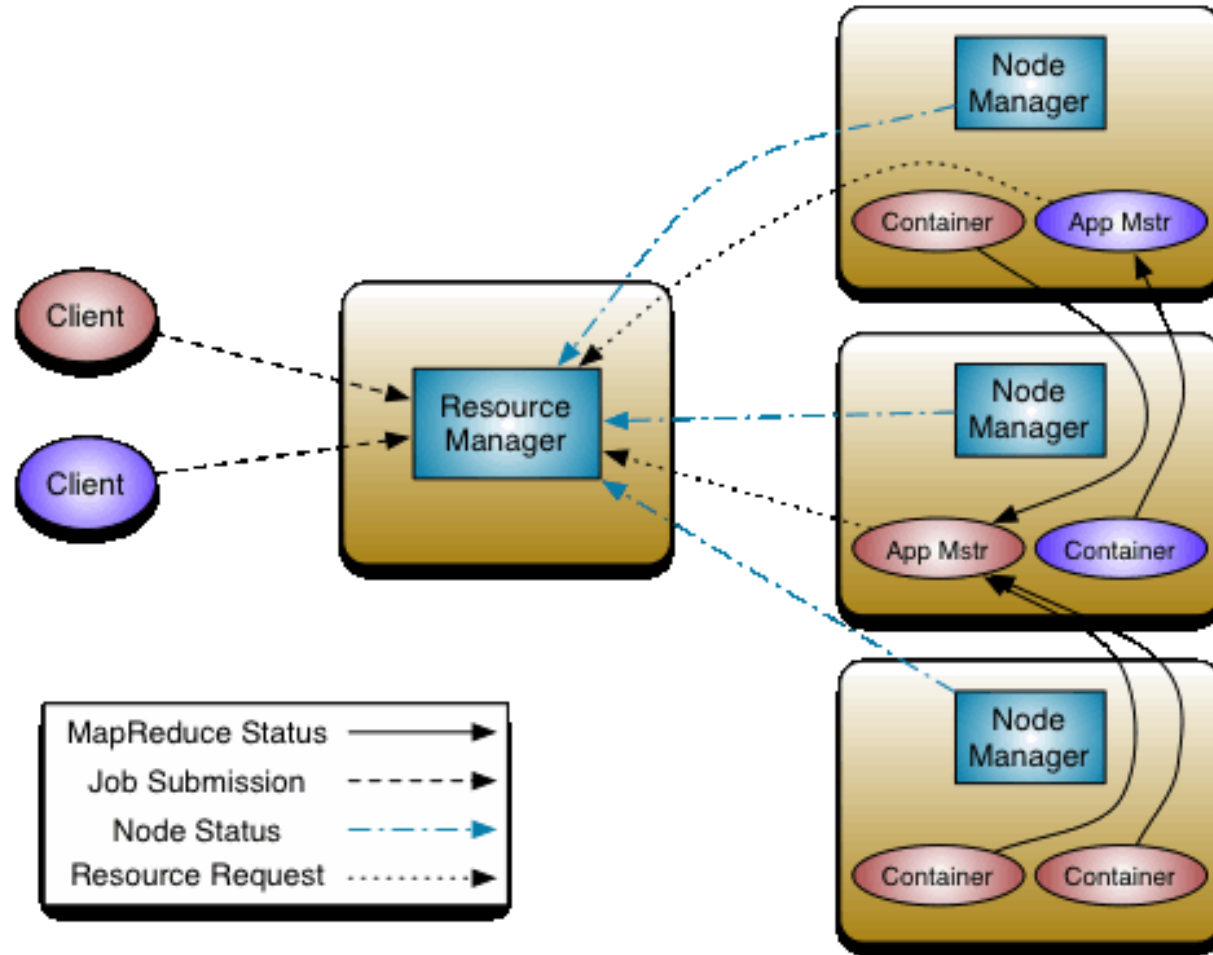


# What is YARN?

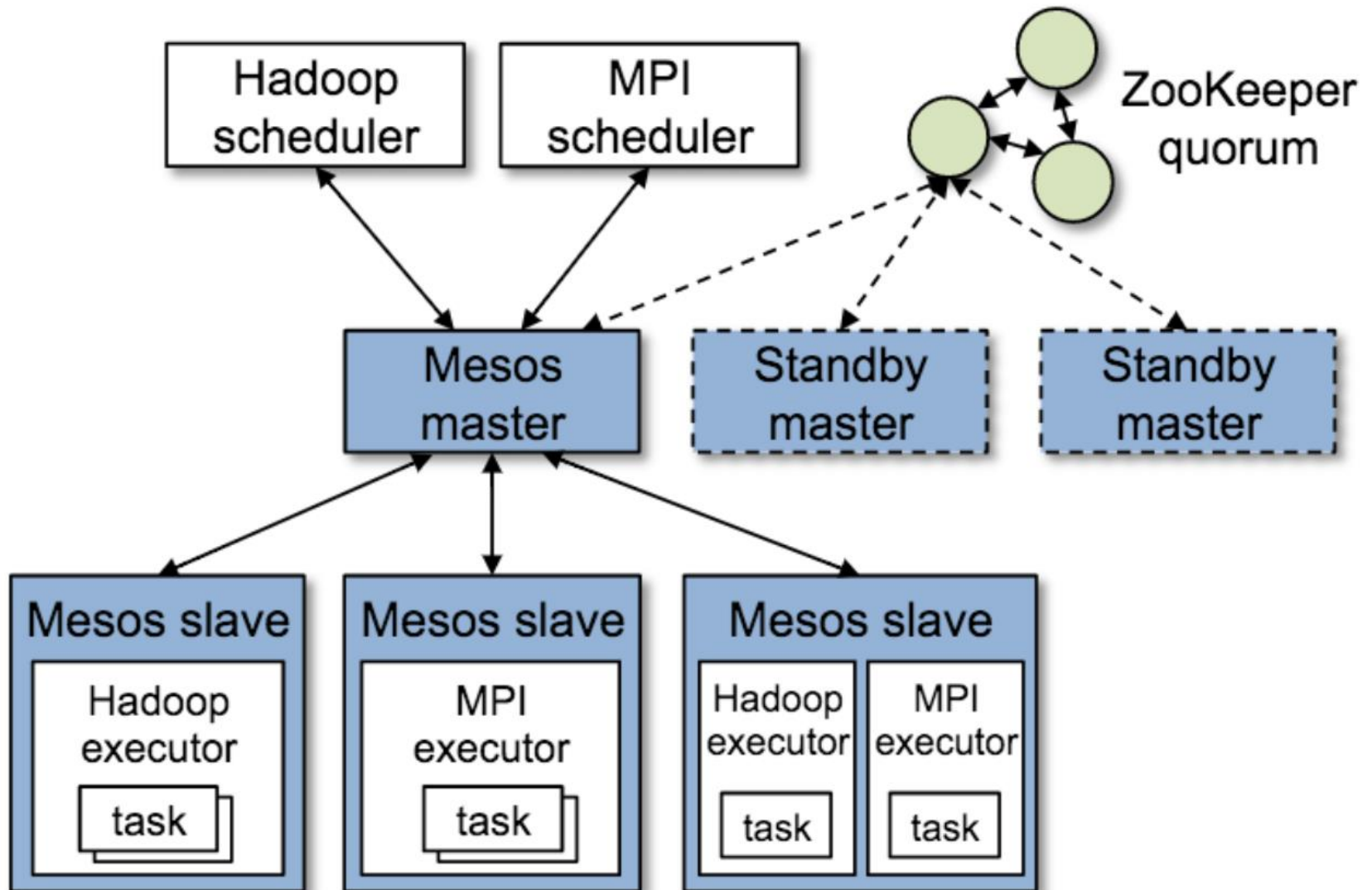
- **YARN is the system that runs your code on multiple nodes**
- Hadoop 2.0 replacement for the cluster manager
  - Basically a model to distribute and manage workloads
  - Not just MapReduce but supports other workloads



# YARN architecture



# Apache Mesos





# AWS: Infrastructure-as-a-Service

The screenshot displays the AWS Management Console's 'All services' page, organized into a grid of categories. Each category is represented by an icon and a title, followed by a list of specific services. Some services include an external link icon (a square with a diagonal line).

- ▼ All services**
  - Compute**
    - EC2
    - Lightsail
    - ECR
    - ECS
    - EKS
    - Lambda
    - Batch
    - Elastic Beanstalk
    - Serverless Application Repository
    - AWS Outposts
    - EC2 Image Builder
  - Storage**
    - S3
    - EFS
    - FSx
    - S3 Glacier
    - Storage Gateway
    - AWS Backup
  - Database**
    - RDS
    - DynamoDB
    - ElastiCache
    - Neptune
    - Amazon Redshift
  - Satellite**
    - Ground Station
  - Quantum Technologies**
    - Amazon Braket
  - Management & Governance**
    - AWS Organizations
    - CloudWatch
    - AWS Auto Scaling
    - CloudFormation
    - CloudTrail
    - Config
    - OpsWorks
    - Service Catalog
    - Systems Manager
    - AWS AppConfig
    - Trusted Advisor
    - Control Tower
    - [AWS License Manager](#)
    - AWS Well-Architected Tool
    - Personal Health Dashboard
    - AWS Chatbot
    - Launch Wizard
    - AWS Compute Optimizer
  - Media Services**
    - Elastic Transcoder
    - Video Streams
  - Security, Identity, & Compliance**
    - IAM
    - Resource Access Manager
    - Cognito
    - Secrets Manager
    - GuardDuty
    - Inspector
    - Amazon Macie
    - AWS Single Sign-On
    - Certificate Manager
    - Key Management Service
    - CloudHSM
    - Directory Service
    - WAF & Shield
    - AWS Firewall Manager
    - Artifact
    - Security Hub
    - Detective
  - AWS Cost Management**
    - AWS Cost Explorer
    - AWS Budgets
    - AWS Marketplace Subscriptions
  - Mobile**
    - AWS Amplify
    - Mobile Hub
    - AWS AppSync

sole.aws.amazon.com/license-manager/home?region=eu-west-2

# EC2 / AWS main functions

- EC2 (Elastic Compute Cloud)
  - Instances
    - Servers of various sizes
  - AMIs (Amazon Machine Images)
    - Server images
  - Elastic Block Storage (EBS)
    - Virtualized Hard drives
  - VPC (Virtual Private Cloud)
    - Secure network space
- S3 (Simple Storage Solution)
  - “Buckets” of data
  - Longer term storage of data



# AWS - EMR

- EMR (Elastic Map Reduce)
  - runs on top of EC2
  - provides straightforward interface for launching clusters and running Spark
  - lab will go through the process of starting a cluster with Spark and other python packages installed, running Jupyter notebooks on the cluster, and introduce SSH connection for more intensive python scripts



# Questions?



© Paul Fremantle 2015. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. See <http://creativecommons.org/licenses/by-nc-sa/4.0/>