Week 4 : Big Data Architecture and Processing

JAMES COOK UNIVERSITY AUSTRALIA

1. The Problem of Processing Big Data

2. A new framework Distributed Computing with Hadoop

3. Options for data access and output in Hadoop

1. The Problem of Processing Big Data

JAMES COOK UNIVERSITY AUSTRALIA

In the past, all your data and computation had to fit on a single machine.
Working on larger datasets required a bigger machine (scaling up).

Traditional Solution : Faster processor, more memory

But there are limits to this type of scaling up.

- Expensive to purchase
- Expensive to support
- Higher risk of system failure from component faults

So what is the solution when scaling up no longer works ?

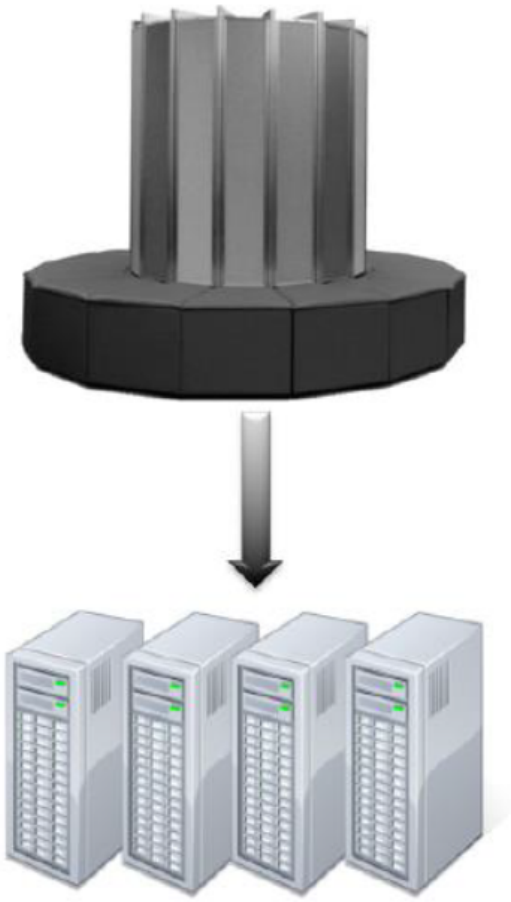
A better Solution : Distributed Systems

JAMES COOK UNIVERSITY AUSTRALIA

- **The better solution: more computers**
  - Distributed systems evolved
  - Use multiple machines for a single job
  - MPI, (Message Passing Interface), for example

"In pioneer days they used oxen for heavy pulling, and when one ox couldn't budge a log, we didn't try to grow a larger ox. We shouldn't be trying for bigger computers, but for *more systems* of computers."

– Grace Hopper

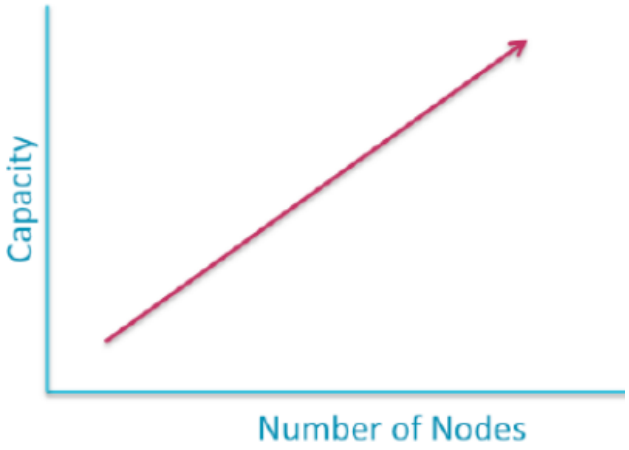


The new approach to Big Data Volume, Variety and Velocity

JAMES COOK UNIVERSITY AUSTRALIA

A cluster of machine but you have to write new programs to support distributed data processing.

Distributed Systems : Scaling out



Add more computers not make bigger ones

This is sometimes known as horizontal scalability or scaling out

Commodity hardware networked together that can be linearly scaled

The Data Bottleneck

JAMES COOK UNIVERSITY AUSTRALIA

Moore's Law has held for 40 years

- Processing Power doubles every 2 years
- Processing Speed is no longer the problem

Getting the data to the processor becomes the bottleneck

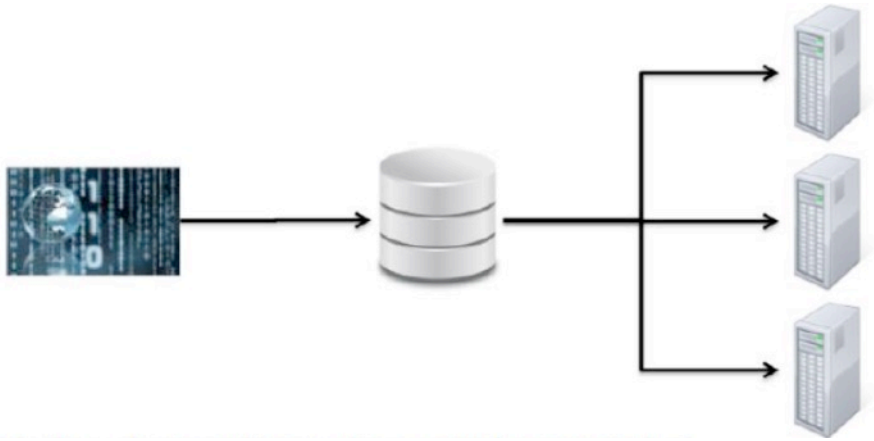
Quick Calculation

- Typical disk data transfer rate is 75MB/sec
- Time taken to transfer a 100GB of data to the processor, approximately 22 minutes
  - Assuming sustained reads
  - Actual time will be worse since most servers will have less than 100GB of RAM available

The Data Bottleneck - continued

JAMES COOK UNIVERSITY AUSTRALIA

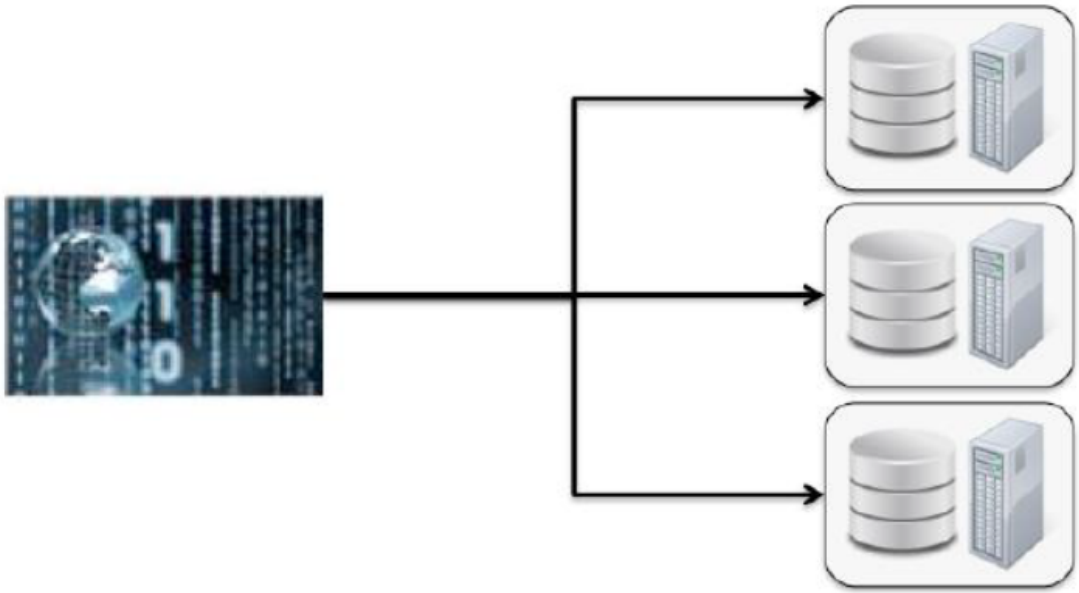
Traditionally, data is stored in a central location
Data is copied to processors at runtime
Fine for limited amounts of data



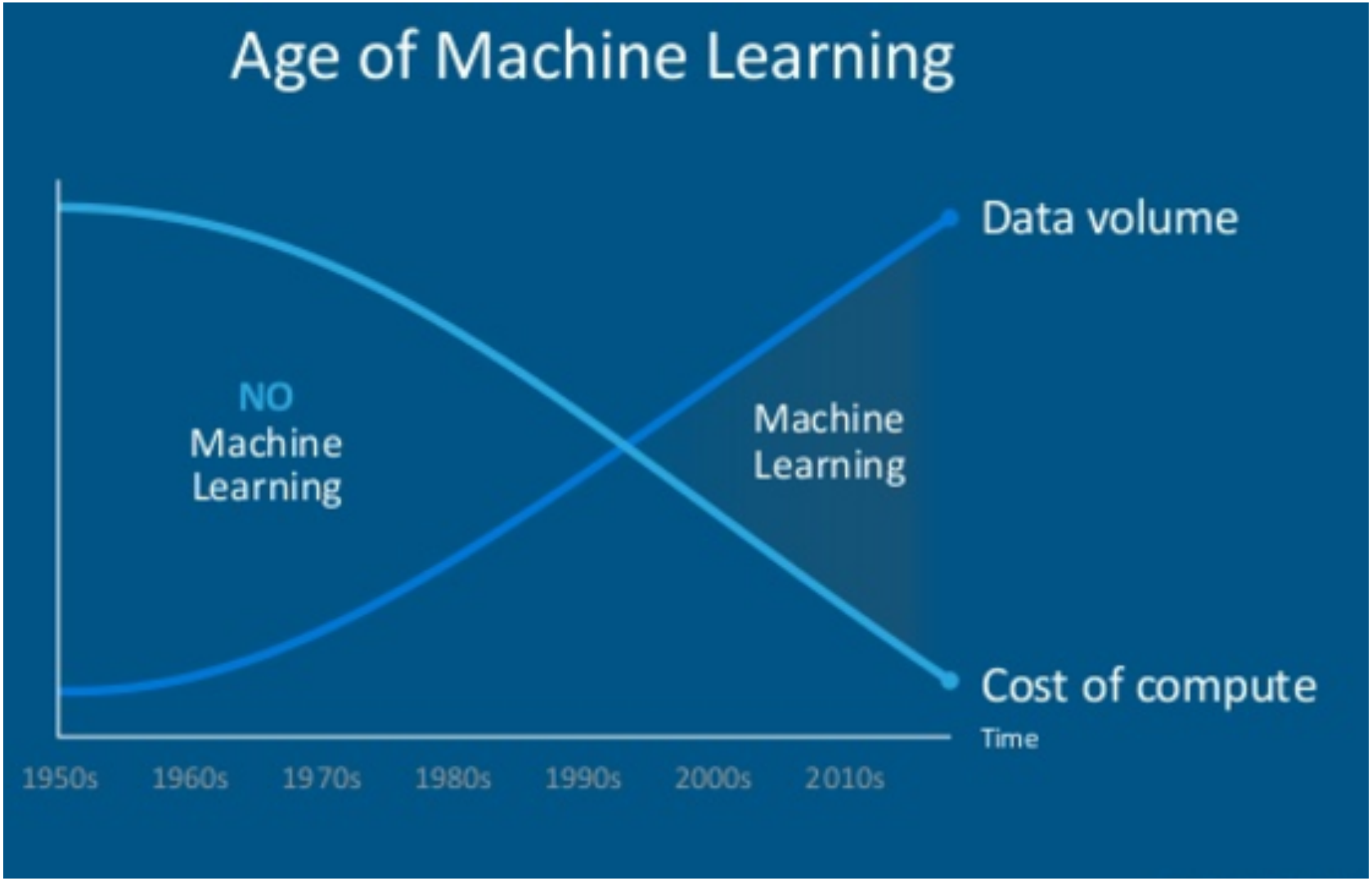
New Approach to Data Distribution

JAMES COOK UNIVERSITY AUSTRALIA

Distribute Data when it is being stored
Run computation where the data is stored



What is the value of distributed computing



2. A new framework
Distributed Computing with Hadoop

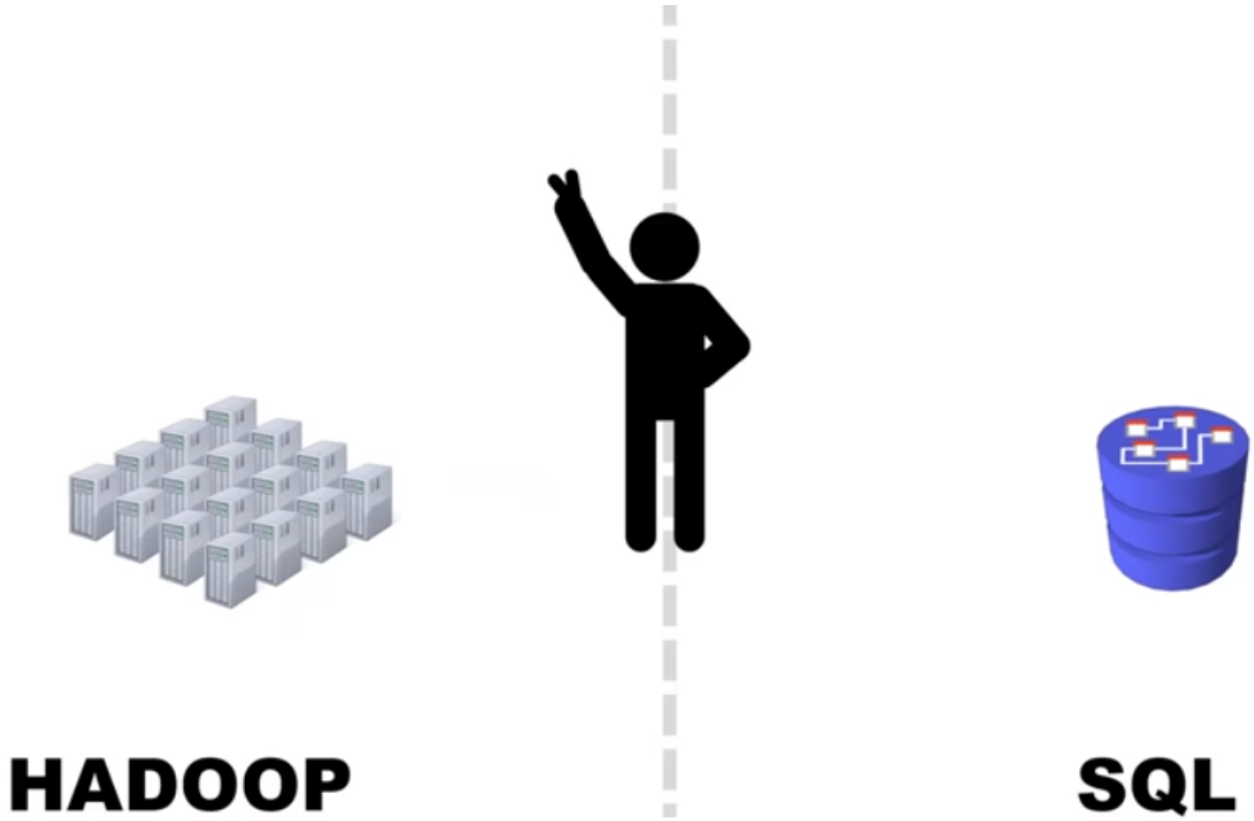
JAMES COOK UNIVERSITY
AUSTRALIA

A software framework for supporting distributed data on distributed computing clusters

- Fault tolerant
- Scalable
- Open Source

Hadoop Distributed File System (HDFS)

Video : Comparison



https://www.youtube.com/watch?v=MfF750YVDxM

Limitations/Assumptions of Distributed Computing

JAMES COOK UNIVERSITY AUSTRALIA

1. The network is reliable
2. Latency is zero
3. Bandwidth is infinite
4. The network is secure
5. Topology doesn’t change

Hadoop History

JAMES COOK UNIVERSITY AUSTRALIA

Hadoop is based on work done by Google in the late 1990s/early 2000s

- Google File System (GFS) published in 2003,
- MapReduce published in 2004

The work takes a radical new approach to the problem of distributed computing

- Meets all the requirements so far for reliability and scalability

Core concept: distribute the data as it is initially stored in the system

- Individual nodes can work on data local to those nodes
  - No data transfer over the network is required for initial processing

Hadoop High level Overview

JAMES COOK UNIVERSITY AUSTRALIA

When data is loaded into the system, it is split into blocks

- Typically 64mb or 128mb

Map Tasks (the first part of the Map reduce system) work on relatively small portions of data

- Typically a single block (default size is 128MB

A master program allocates work to nodes such that a Map task will work on a block of data stored locally on that node wherever possible

- Many nodes work in parallel, each on their own part of the overall dataset

Map and Reduce

JAMES COOK UNIVERSITY AUSTRALIA

MapReduce refers to two separate and distinct tasks; map and reduce.

Map always runs first, possibly followed by reduce.

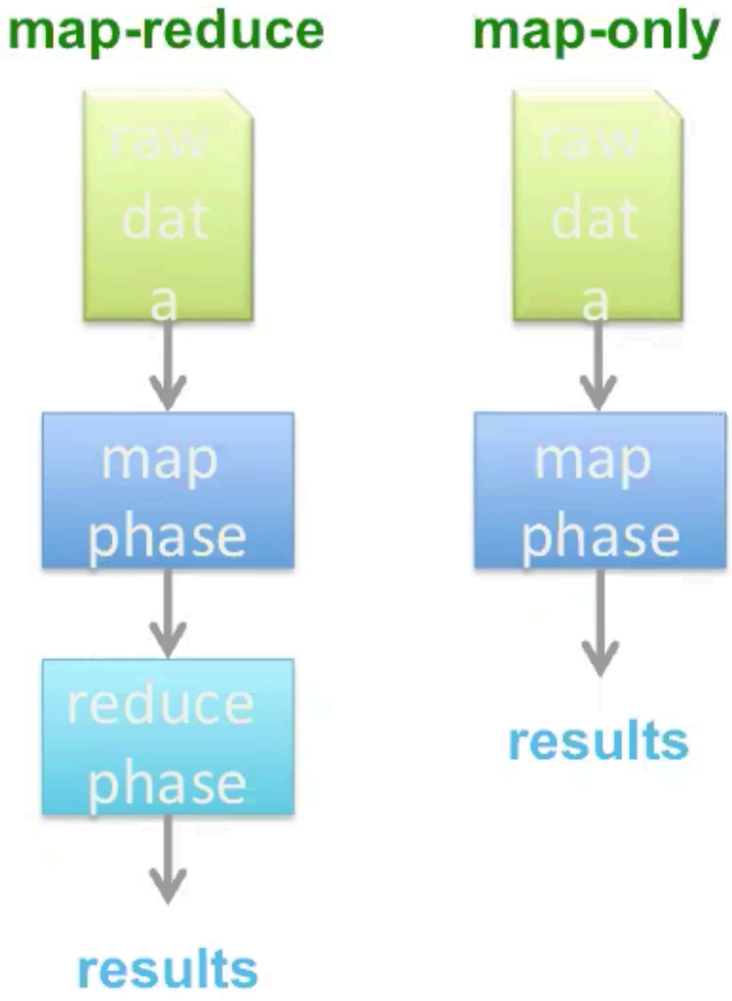
DEPENDING ON THE TASK, REDUCE MIGHT NOT BE NECESSARY.

MapReduce applications are written in a variety of programming and scripting languages.

Java, Python, Perl, and others

MapReduce processes the data on each slave node in parallel and then aggregates the results.

MapReduce has been the basis for Hadoop's data processing scalability historically



Core Hadoop Concepts

JAMES COOK UNIVERSITY AUSTRALIA

Applications are written in high-level code

- Developers need not worry about network programming, temporal dependencies or low level infrastructure

Nodes talk to each other as little as possible

- Developers should not write code which communicates between nodes
- “Shared Nothing” architecture….make is highly fault tolerant

Data is spread among machines in advance

- Computation happens where the data is stored, wherever possible
  - data is replicated multiple times for increase availability and reliability
  - Default replication factor is 3 giving greater availability and performance

The Hadoop Ecosystem : The Zoo

JAMES COOK UNIVERSITY AUSTRALIA

Apache Ambari
http://incubator.apache.org/ambari

APACHE FALCON

Spark

HIVE

OOZIE

Hadoop (HDFS)

Apache Solr

APACHE HBASE

accumulo

mahout

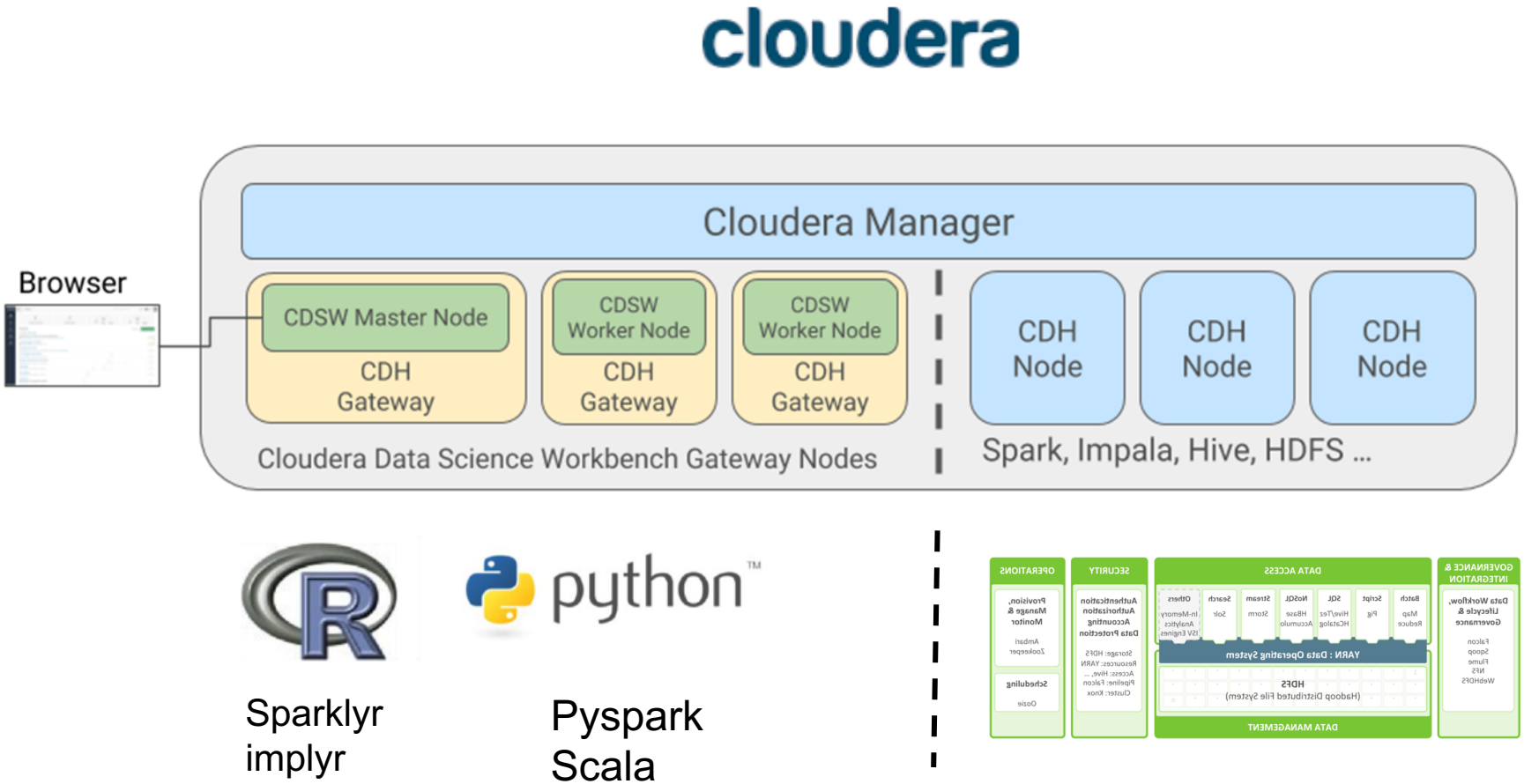
FLUME

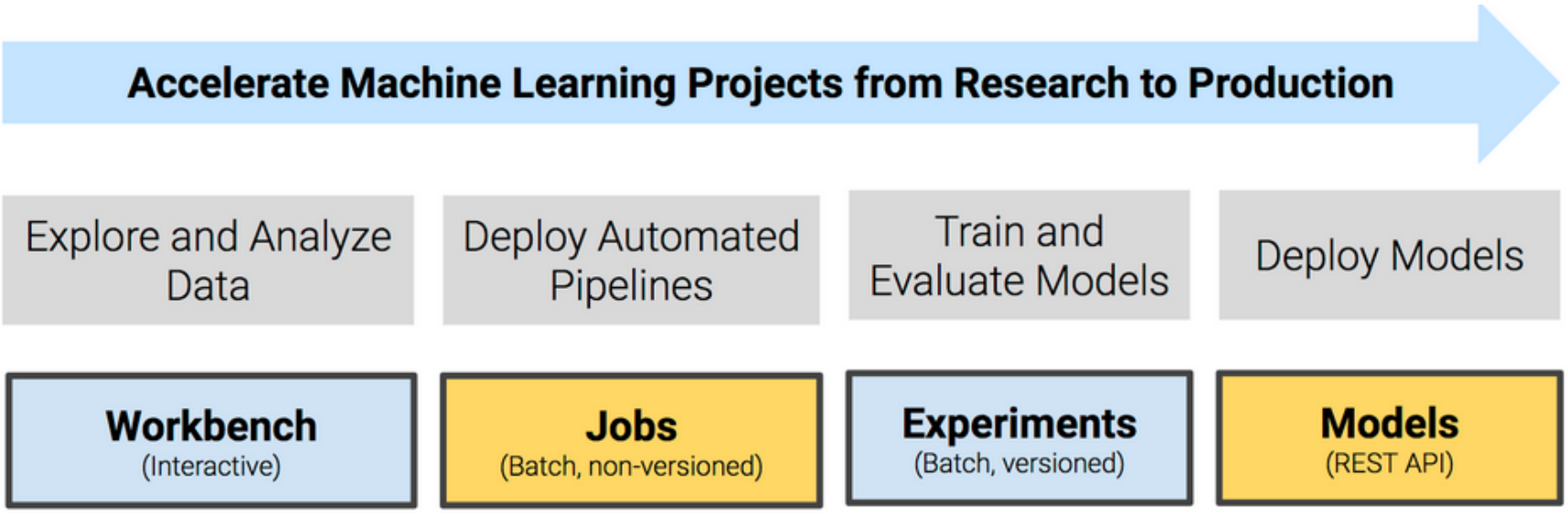
STORM

HORTONWORKS
POWERING THE FUTURE OF DATA

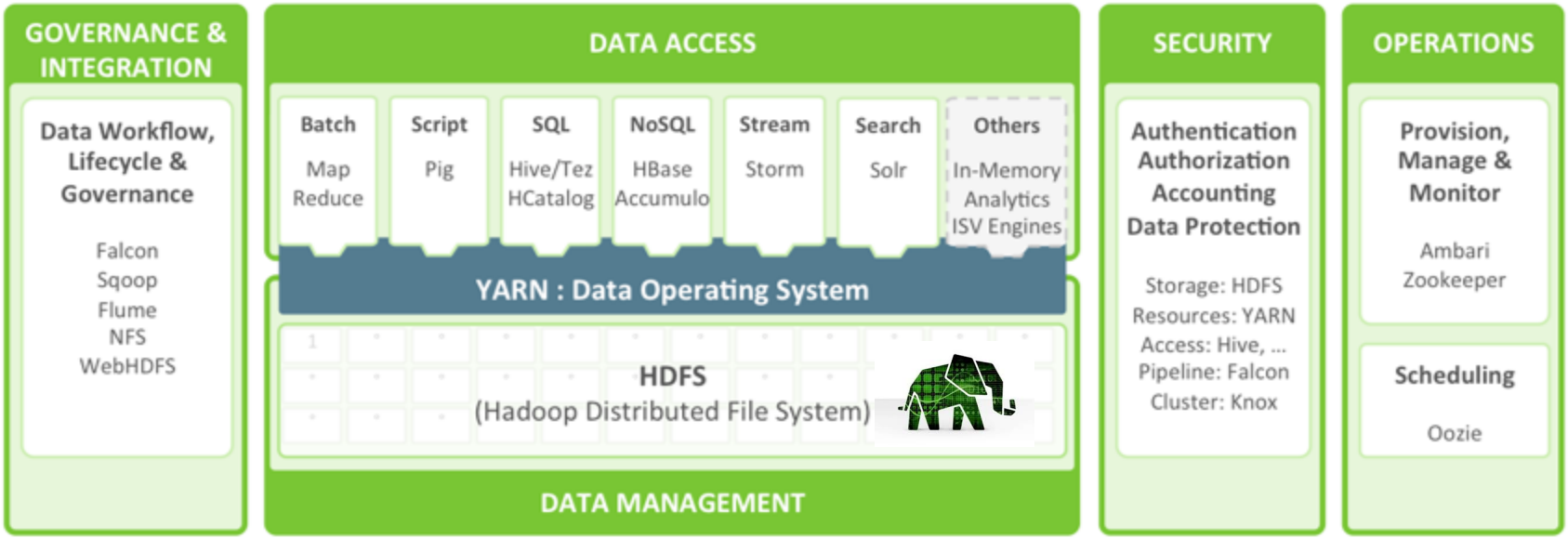
cloudera

Hadoop and Data Science ecosystem




Accelerate Machine Learning Projects from Research to Production
Explore and Analyze Data
Deploy Automated Pipelines
Train and Evaluate Models
Deploy Models
Workbench
(Interactive)
Jobs
(Batch, non-versioned)
Experiments
(Batch, versioned)
Models
(REST API)

Hadoop Architecture



cloudera

Hadoop (HDFS) vs Traditional Database

JAMES COOK UNIVERSITY AUSTRALIA

HDFS vs RDBMS

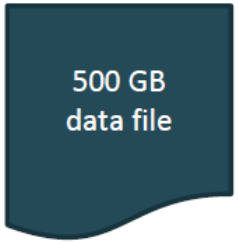
HDFS
Distributed File processing and storage

- Assumes a task will require reading a significant amount of data off of a disk
- Does not maintain any data structure
- Simply reads the entire file
- Scales well (increase the cluster size to decrease the read time of each task)

RDBMS
Traditional Database

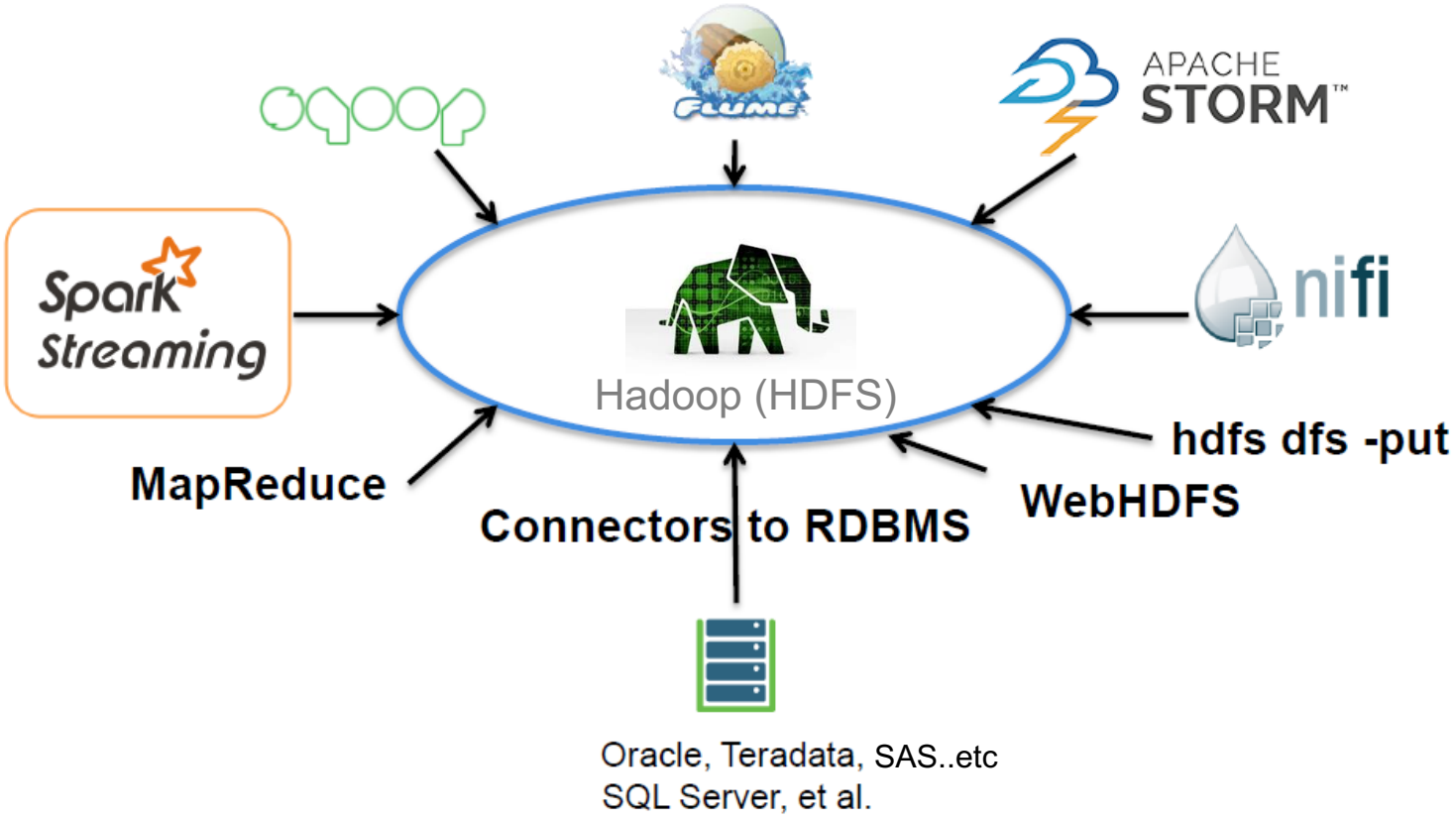
- Uses indexes to avoid reading an entire file (very fast lookups)
- Maintains a data structure in order to provide a fast execution layer
- Works well as long as the index fits in RAM

- 2,000 blocks of size 256MB
- 1.9 seconds of disk read for each block
- On a 40 node cluster with eight disks on each node, it would take about 14 seconds to read the entire 500 GB

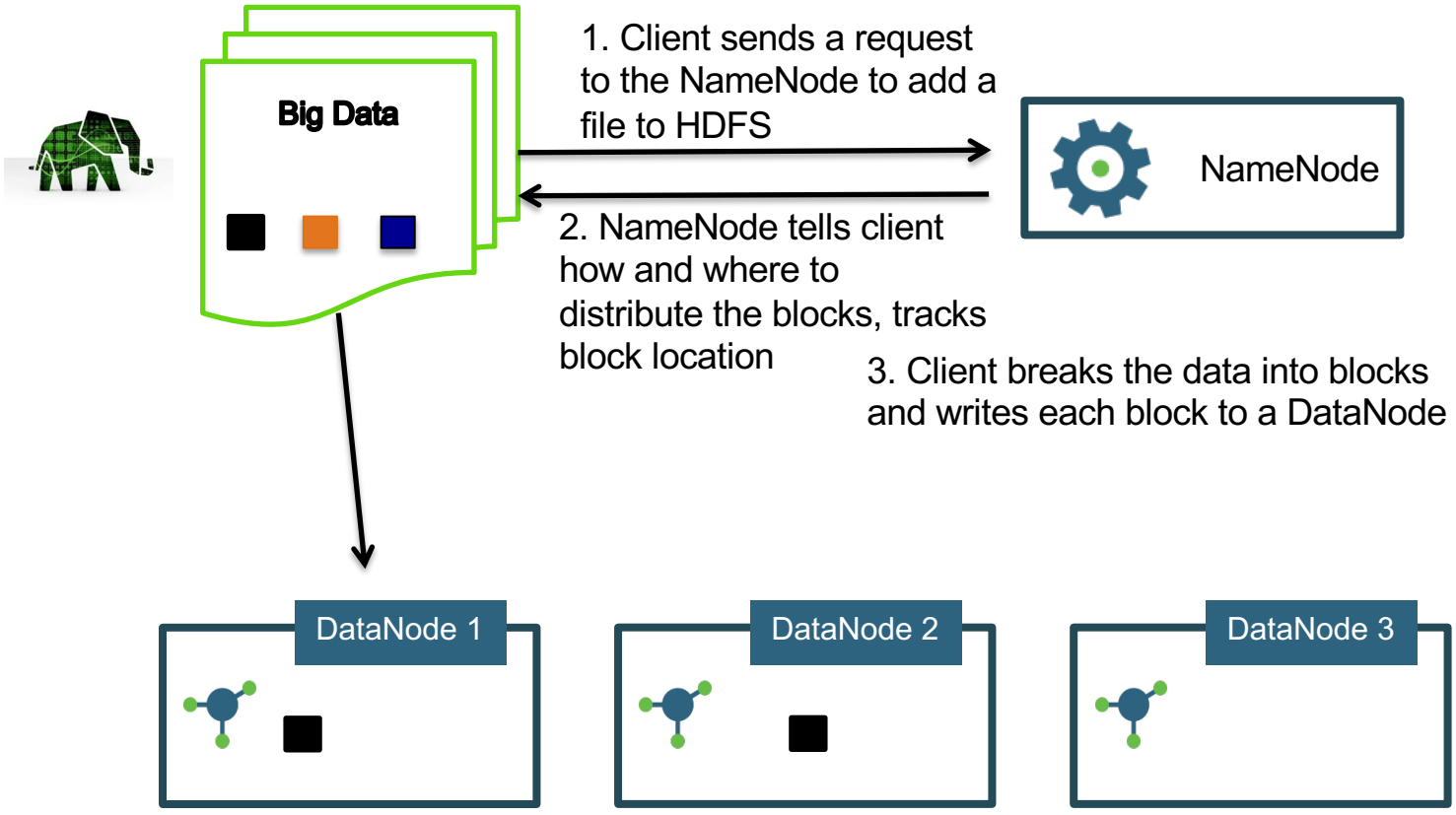


61 minutes to read this data off of a disk (assuming a transfer rate of 1,030 Mbps)

Options for Data Input into HDFS



Hadoop File System (HDFS) Block Storage

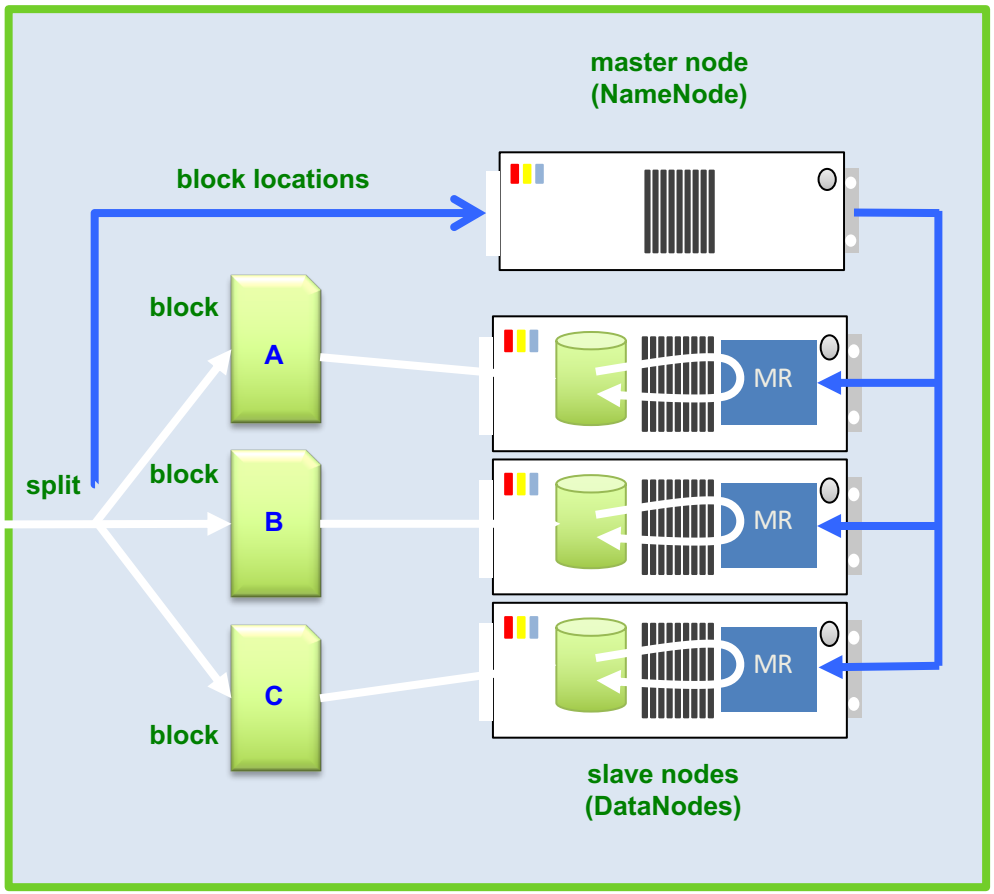


4. The DataNode replicates each block to two other DataNodes (as chosen by the NameNode)

HDFS Data Storage Overview

JAMES COOK UNIVERSITY AUSTRALIA

HDFS automatically:
-splits large files into blocks
-spreads blocks across cluster
-tracks block locations
-replicates blocks (not shown)



distributed applications like MapReduce get block information to access and analyze data

Simple Overview of Data Input and output into Hadoop (HDFS)