Hadoop / MapReduce

ICOS Big Data Summer Camp
May 16th, 2018
Patrick Park

Goals

- Brief Intro to Hadoop / MapReduce
- Understand Structure of MapReduce
- Learn Options for Processing at Scale
- Implement MapReduce in Python (MrJob)

What is Hadoop?

The name comes from a toy elephant of the main developer's son

What is Hadoop?

The name comes from a toy elephant of the main developer's son

A software framework for distributed storage and processing of big data using the MapReduce programming model.

Why Hadoop?

Big data problem:

- Storage: terabytes and beyond, hardware failure (annual failure rate ~ 2%)
- I/O: reading/writing files take long time
- Computation: Leveraging multiple processors
- Scalability: Challenging to add more machines

Also, if many people are running their scripts, how does the system efficiently distribute compute resources?

Hadoop Core Components

Problem 1

- How to store and access data

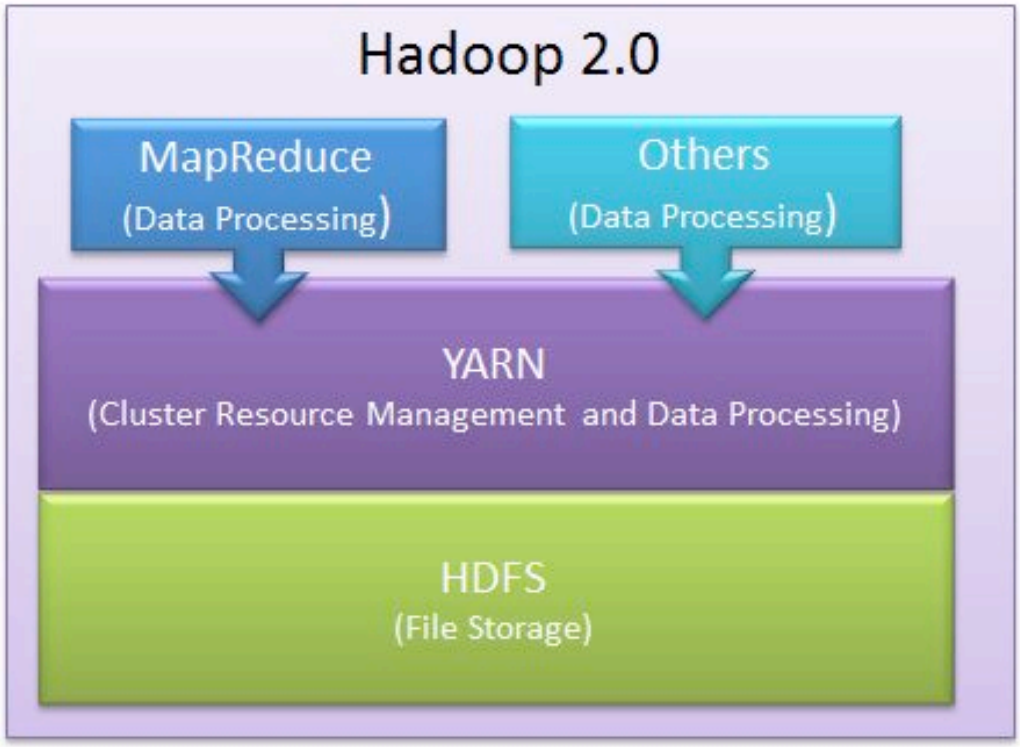
Problem 2

- How to manage/coordinate computing resources

Problem 3

- How to process data

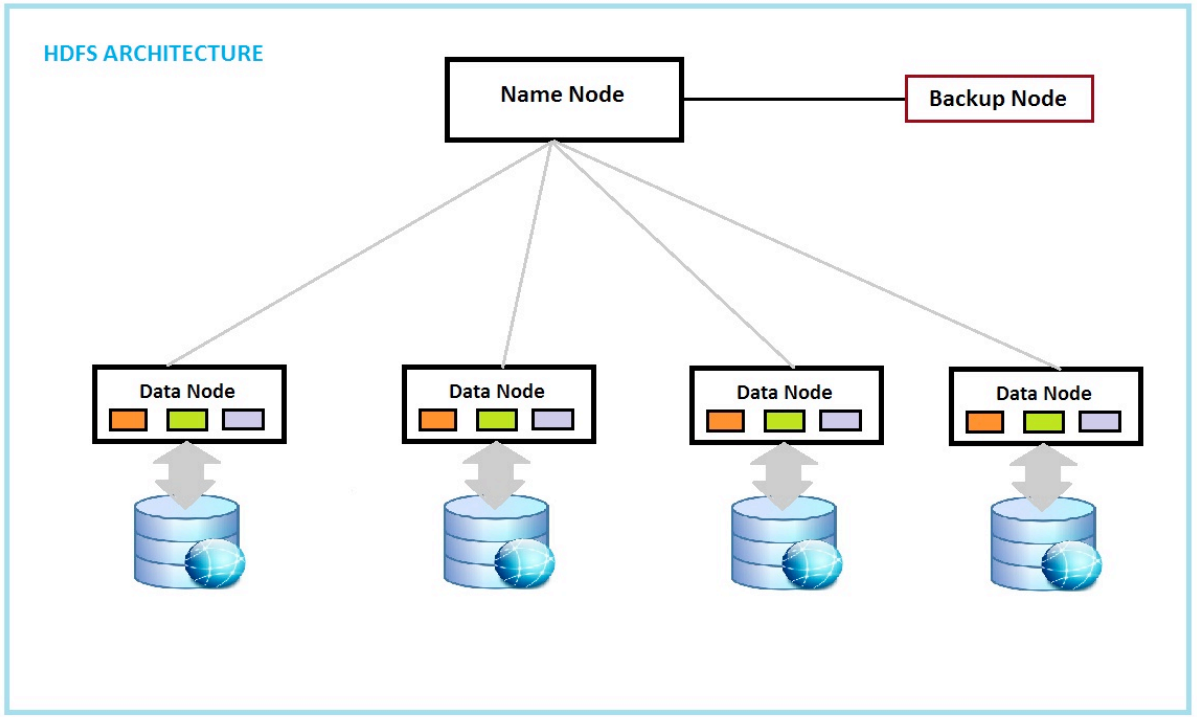
Hadoop Core Components



Data Storage / Access

Hadoop File System (HDFS):

A Java-based distributed file system. Saves multiple copies of the same data to tolerate disk failure



Annual disk failure rate: 2%

If you have same data on four different disks, what is the probability that you lose all four copies?

→ Very low

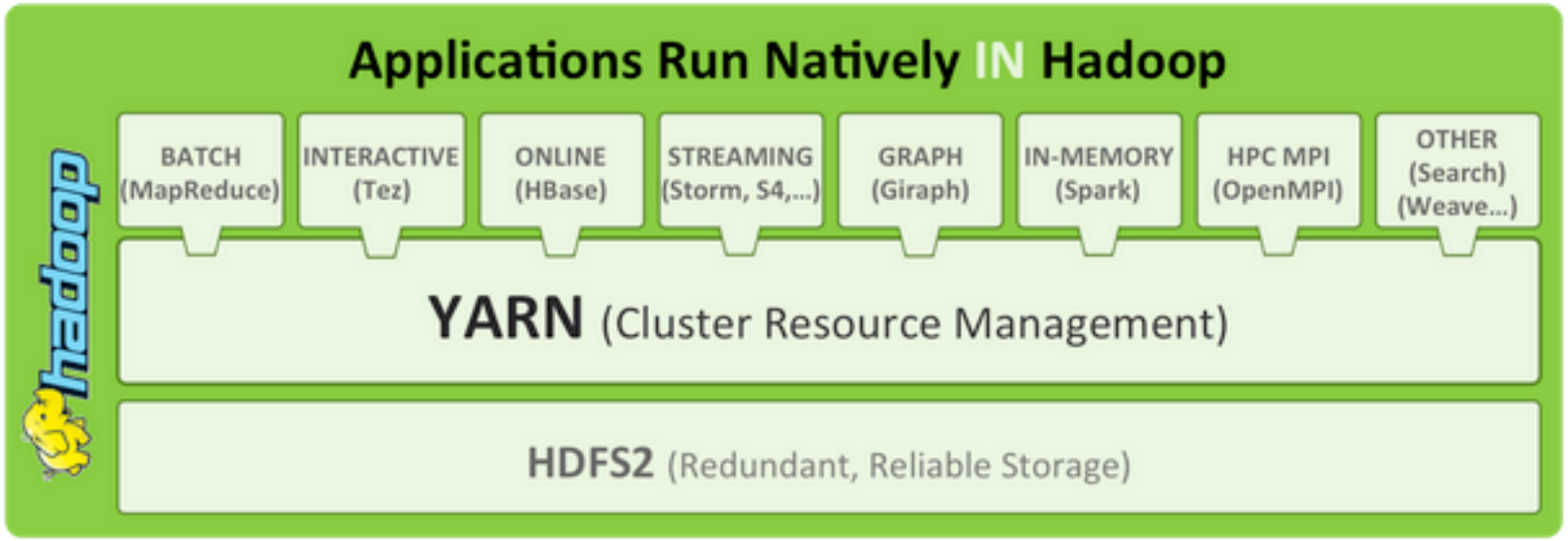
Resource Management

YARN (**Y**et **A**nother **R**esource **N**egotiator):

Manages cluster resources such as memory and CPU for multiple applications

- Beyond the scope of this lecture -

How to Process Data



Many Approaches
We focus on MapReduce

MapReduce

The big idea:

1. Split big data into smaller chunks
2. Process chunks simultaneously using multiple machines

Faster than processing big data from one machine

MapReduce

Map:

- Read data from HDFS
- Format them into key-value pairs

Sort by Key:

- Group the values with the same key

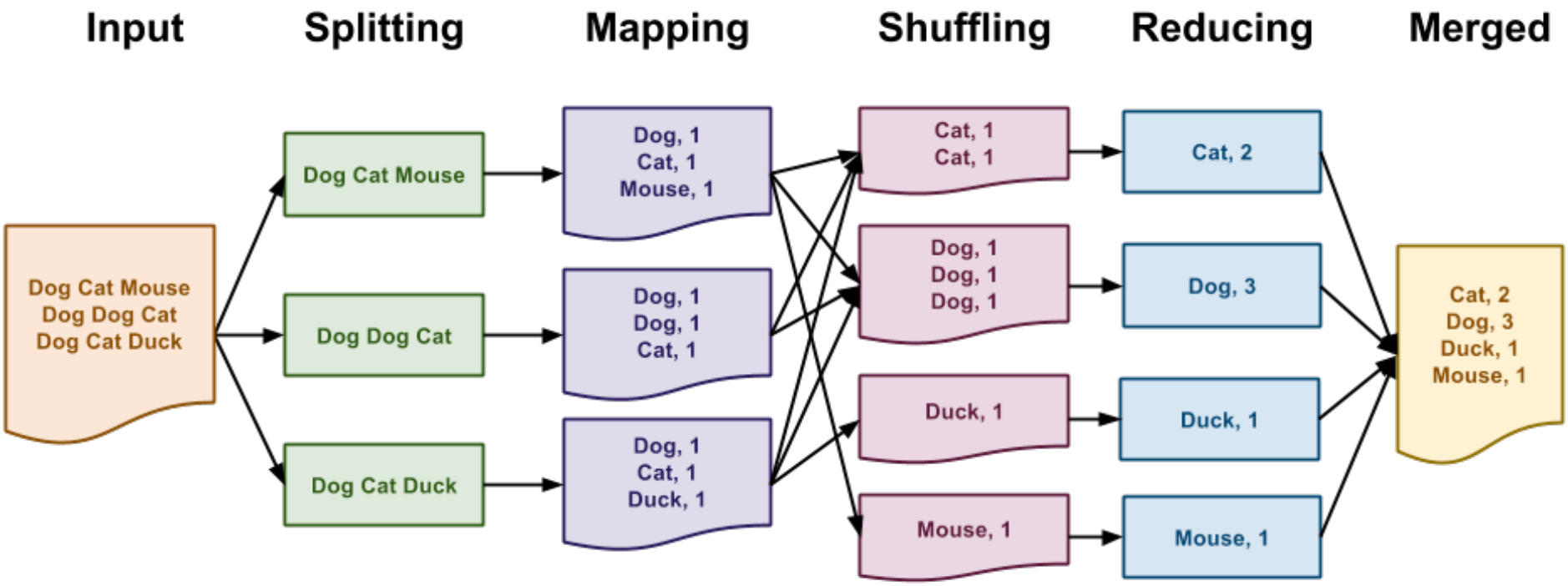
Reduce:

- Perform computation on the grouped values
- Write output to disk

Uses of MapReduce

- Grouping and Aggregation (word count)
- Sorting
- Merging, joining
- Filtering
- Set operations (union, intersection, difference)
- Graph Processing (Iterative Message Passing)
- Machine learning (k-means, logistic regression)

Example: Word Count



Running MapReduce “Locally”

Typically, one develops MapReduce code with a small sample dataset

Code is tested locally on a laptop

Running MapReduce in Distributed Mode

When code is tested and production-ready, run large-scale MapReduce jobs:

On-campus **Flux Hadoop** cluster

Cloud platforms (**Amazon, Microsoft, Google**)

Python's MrJob Library can be used on all of them

Flux Hadoop

ARC-TS ADVANCED RESEARCH COMPUTING TECHNOLOGY SERVICES UNIVERSITY OF MICHIGAN

ADVANCED RESEARCH COMPUTING

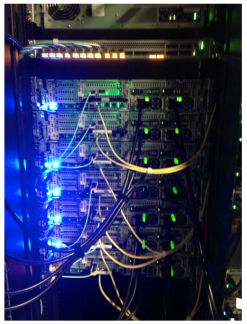
COMPUTATIONAL SCIENCE | DATA SCIENCE | TECHNOLOGY SERVICES | CONSULTING SERVICES

ABOUT | SYSTEMS AND SERVICES | TRAINING AND WORKSHOPS | NEWS | EVENTS | CONTACT US

Flux Hadoop Cluster

The Flux Hadoop Cluster is an upgraded Hadoop cluster currently available as a technology preview with no associated charges to U-M researchers. The cluster is an on-campus resource that provides a different service level than most cloud-based Hadoop offerings, including:

- high-bandwidth data transfer to and from other campus data storage locations with no data transfer costs
- very high-speed inter-node connections using 40Gb/s Ethernet.



The cluster provides 112TB of total usable disk space, 40GbE inter-node networking, Hadoop version 2.6.0, and several additional data science tools.

Aside from Hadoop and its Distributed File System, the ARC-TS data science service includes:

- Pig, a high-level language that enables substantial parallelization, allowing the analysis of very large data sets.
- Hive, data warehouse software that facilitates querying and managing large datasets residing in distributed storage using a SQL-like language called HiveQL.
- Sqoop, a tool for transferring data between SQL databases and the Hadoop Distributed File System.

USER GUIDE

BACK TO SYSTEMS & SERVICES

ORDER SERVICE

Using the Flux Hadoop environment requires a Flux user account (available at no cost), but currently does not require a Flux allocation.

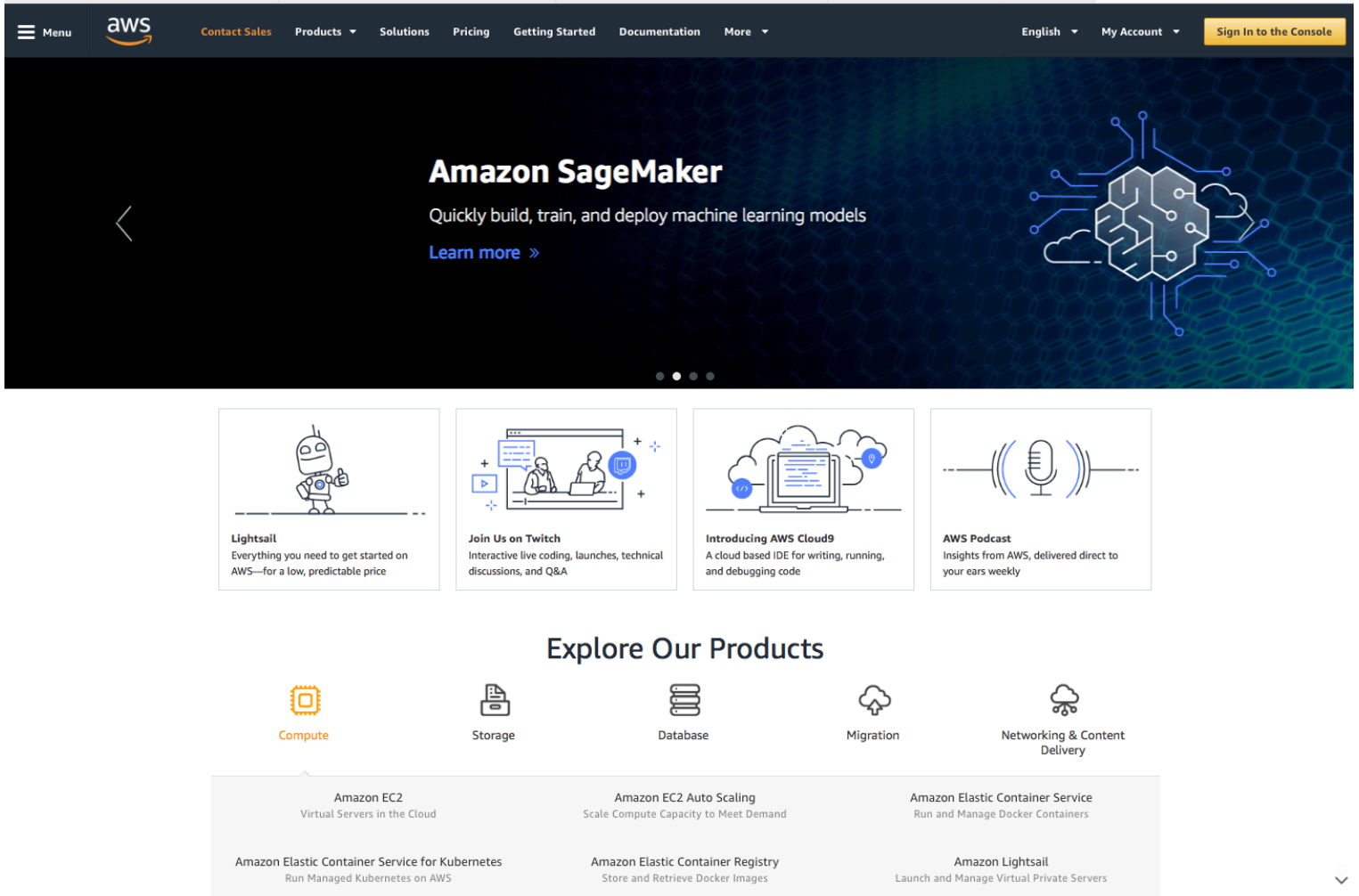
TO ORDER:

Email hpc-support@umich.edu.

For more information: data-science-support@umich.edu.

Currently free of charge, but in beta testing phase (More info on Thursday)

Cloud Platforms



MapReduce in the Cloud

Pay only the compute hours you consume

Frees you up from maintaining a physical Hadoop cluster

Easily scalable to hundred's of virtual servers

<WARNING>
Running buggy code can quickly burn $$

General references

Apache Hadoop: http://hadoop.apache.org/

MrJob: https://pythonhosted.org/mrjob/

Flux Hadoop: http://arc-ts.umich.edu/hadoop/

AWS: https://aws.amazon.com/

Azure: https://azure.microsoft.com/en-us/solutions/hadoop/

Google: https://cloud.google.com/hadoop/

Questions?

If none…

MrJob Exercise

Start jupyter notebook

and

Open mrjob_intro.ipynb

MrJob Configuration File

MrJob configuration file controls the specifics of the EMR cluster

- Number of instances
- Bidding price for compute time
- EMR version
- Where to save log files

Running a Job

```
python  your_script.py  s3://your-data-location  \
-r emr \
--no-output \
--output-dir=s3://output-location/ \
--conf-path=a_config.conf
```

Execution of this mrjob script
(a) spins up an EMR cluster on AWS based on the .conf file,
(b) runs your mrjob script and stores the output in AWS, and
(c) shuts down the EMR cluster when the job completes.

Word Count Demo

- Count words in a large text file (14GB) of English Wikipedia pages retrieved in 2017
- Cluster spin-up and run time: 1h 46min with 48 virtual cores (three m3.2xlarge instances)
- $0.254/hour * 2 hours * 3 instances = $1.52

Modes of Execution

Local mode:

- Run MapReduce code on a single machine
- Does not involve HDFS access
- Useful for code development and testing

Distributed mode:

- Run code on a networked cluster of machines
- Involves HDFS access
- Hadoop cluster vs. cloud