MP1

How is it going?

- Due tomorrow at 11:59 pm.
- Latest Commit to the repo at the time will be graded.
- Last Office Hours today after the lecture until 7pm.
SSH Keys

Look for your SSH key in your Gitlab repository to access course cluster.

Please read the Cluster Page on the course website.

Course cluster address announced when MP2 is released
Wednesday

Wednesday Lecture Optional

- Office hours for MP2
- Troubleshooting access to the cluster
Outline

- Hadoop Overview
- Hadoop Distributed File System (HDFS) and YARN
- Hadoop Deployment Strategies
- Accessing the Course Cluster
Outline

- **Hadoop Overview**
- Hadoop Distributed File System (HDFS) and YARN
- Hadoop Deployment Strategies
- Accessing the Course Cluster
Hadoop

Open-source software for reliable, scalable, distributed computing.

Comes with:
- Hadoop Common
- Hadoop HDFS (Distributed Storage)
- Hadoop YARN (Resource Manager / Task Scheduler)
- Hadoop MapReduce (MapReduce Programming Model)
What Hadoop does for you

Hadoop abstracts:

- Parallelization
- Scheduling
- Resource Management
- Inter-machine communication
- Handling software/hardware failures
- Etc.

Reasoning: Extract common cloud computing primitives to build reliable cloud applications faster, and more reliably
Outline

- Hadoop Overview
- **Hadoop Distributed File System (HDFS) and YARN**
- Hadoop Deployment Strategies
- Accessing the Course Cluster
HDFS

What problem are we trying to solve?

- Distributing / replicating data to a multi-node cluster
- Replicating data intelligently, minimizing bandwidth
- Keeping replicas in multiple racks / geographic regions

HDFS is the Data Management Layer of Hadoop.

- Scalable, fault-tolerant, cost-efficient, data storage
Assumptions and Goals in Design

HDFS assumes there will be:
- Hardware Failure
- Large Datasets

HDFS aims for:
- Fault-Tolerance
- Portability Across Heterogeneous Hardware and Software Platforms
- Optimized sequential operations
- Streaming Data Access
HDFS - Data Replication

Datanodes

NAME NODE

1 3
4
6 1
2 5
8 5 3 7 8
7
HDFS - Data Replication

HDFS can reliably store very large files across machines. The blocks of files are replicated for fault tolerance.

Datanodes

1 3

4 6 1

8 5

3 7 8

2 5

7
HDFS - Data Replication

Datanodes
HDFS - Data Replication

Datanodes

New datanode is created by the namenode
HDFS - Data Replication

- How do we tell HDFS to replicate files?
  - Set replication factor for file/directory
    - "hdfs dfs -setrep <num_replicas> /path/to/file"
  - Hadoop will increase/decrease number of replicas accordingly
  - Maximum achievable replication is equal to number of datanodes in the cluster
YARN - Yet Another Resource Negotiator

A framework for job scheduling and cluster resource management. “How a job gets a container”

- Underneath Hadoop 2.x+
- Negotiates resources and schedules jobs
- Treats each server as a collection of containers.
  - Each container can run different types of tasks and might be different sizes.
- Users can submit any type of application supported by YARN.
YARN - Yet Another Resource Negotiator

A framework for **job scheduling** and **cluster resource management**. “How a job gets a container”

- Not just for MapReduce!
  - Used by Spark, Tez, Pig, and others
YARN

Three Main Components:

- **Global Resource Manager (RM)**
  - Keeps track of live NodeManagers and available resources.
  - Allocates available resources to appropriate applications.

- **Per-Server Node Manager (NM)**
  - Provides computational resources in forms of containers.
  - Manages processes running in containers.

- **Per-Application Application Master (AM)**
  - Coordinates the execution of all tasks within its application.
  - Asks for appropriate resource containers to run applications/tasks.
YARN inside the course cluster
YARN inside the course cluster

AM = Application Master
NM = Node Manager
YARN inside the course cluster

1. Need a container!

AM = Application Master
NM = Node Manager
YARN inside the course cluster

Resource Manager
Capacity Scheduler

AM 1
NM A
Worker A

2. Container Completed!

AM 2
NM B
Worker B

1. Need a container!

AM = Application Master
NM = Node Manager
YARN inside the course cluster

AM = Application Master
NM = Node Manager

1. Need a container!
2. Container Completed!
3. Worker A has a container - RM

AM 1
Worker A
NM A

AM 2
Worker B
NM B
YARN inside the course cluster

AM = Application Master
NM = Node Manager

1. Need a container!
2. Container Completed!
3. Worker A has a container - RM
4. Start the task!
YARN inside the course cluster

1. Need a container!
2. Container Completed!
3. Worker A has a container - RM
4. Start the task!

AM = Application Master
NM = Node Manager
YARN inside the course cluster

1. Need a container!
2. Container Completed!
3. Worker A has a container - RM
4. Start the task!

AM = Application Master
NM = Node Manager

AM = Application Master
NM = Node Manager
Why is it so important?

- Performance
- Latency
- Cost Efficiency
Other Scheduler Considerations

- Data Locality
  - Many worker nodes are also Data nodes
  - Try to schedule jobs on workers that already have a copy of the input data
  - Why? Then we don’t have to transfer data to the worker on startup
Fault Tolerance

- **Node Failure**
  - **No heartbeat from NM to RM**
    - RM let all AMs know
  - **NM fails to keep track of each task in the node**
    - If task fails while in-progress, mark the task as idle and restart them
  - **No heartbeat from AM to RM**
    - RM restarts AM, syncs up with its running tasks

- **Resource Manage Failure (on Master)**
  - Secondary RM gets spun up.
  - Use last checkpoint and sync up.
How does it look as a whole?

MP2

MapReduce

YARN

HDFS

Worker A

AM 1

Data Block 1

NM A

Worker B

AM 2

Data Block 2

NM B
How does it look as a whole?
How does it look as a whole?
Outline

- Hadoop Overview
- Hadoop Distributed File System (HDFS)
- **Hadoop Deployment Strategies**
- Accessing the Course Cluster
Submitting Hadoop jobs

- Input/Output **must** reside in HDFS
- Program code can live in the normal filesystem
  - Programs can be copied to all nodes upon job startup, because they're small

Submitting a streaming job to Hadoop:

```bash
$ hadoop jar $STREAMING_JAR \
  -files mapper.py, reducer.py \
  -mapper mapper.py -reducer reducer.py \
  -input /shared/my_cool_dataset -output my_job_out
```
Hadoop Jobs on course cluster

- We MRJob made things a bit easier for you:

  $ python my_mr_job.py -r hadoop
  
  --output-dir my_job_out  # HDFS directory
  --no-output             # Suppress output via STDOUT
  hdfs:///data/my_cool_dataset  # HDFS path for input
Hadoop Jobs

- Where is your output stored? In HDFS
  - Check your output directory for files named part-XXXXX

- Hadoop Job Counters
  - Displayed after job completes
  - Can be viewed during job in the Hadoop web interface
  - Information about: HDFS usage, MapReduce records and I/O
Common Pitfalls

● Make sure HDFS output directory doesn’t already exist
  ○ Remove directories with “hdfs dfs -rm -r /path/to/directory”

● “My job isn’t starting” / “My job is stalled”
  ○ There may be others in front of you in the queue
  ○ Check the Hadoop web interface (instructions in the MP2 doc)
  ○ Or, run “yarn application -list” to see if other jobs are running

● “My job crashed with a ‘subprocess failed’ error”
  ○ This is almost always an error in your code
  ○ If your program crashes for any line of input, this can crash the whole job
  ○ Solution: Test locally where you can get better logging
Outline

- Hadoop Overview
- Hadoop Distributed File System (HDFS)
- Hadoop Deployment Strategies
- Accessing the Course Cluster
Course Cluster Warnings

- Do not share your AWS credentials with anyone.
- Do not do other course work on the course cluster.
- Do not run any other side projects with the course cluster.
- Do not use excessive resources or break the cluster.

We monitor the cluster usages closely, and the instructors reserve the right to revoke your cluster privilege if deemed necessary.
Hadoop Web Interfaces

- Check MP2 documentation
- Very useful/informative for viewing real-time progress of your jobs
Hadoop Web Interfaces
### Hadoop Web Interfaces

<table>
<thead>
<tr>
<th>Task</th>
<th>Progress</th>
<th>Status</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>SUCCEEDED</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>SUCCEEDED</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
<tr>
<td>task_1517153180021_0011</td>
<td></td>
<td>RUNNING</td>
<td></td>
</tr>
</tbody>
</table>

Showing 1 to 20 of 20 entries
MP 2

Read the MP Documentation Carefully

We won’t answer Piazza questions if the answers are in the doc.
MP 2

Due in next Tuesday (2/6) at 11:59pm

“Hadoop MR with real DataSets on a Hadoop cluster “

Please start early!

Cluster can get crowded near the deadline.

> Check Piazza for Q&A and Announcements