YARN, the Apache Hadoop Platform for Streaming, Realtime and Batch Processing

Eric Charles [http://echarles.net] @echarles
Datalayer [http://datalayer.io] @datalayerio

FOSDEM 02 Feb 2014 – NoSQL DevRoom
Eric Charles (@echarles)
Java Developer
Apache Member
Apache James Committer
Apache Onami Committer
Apache HBase Contributor
Worked in London with Hadoop, Hive, Cascading, HBase, Cassandra, Elasticsearch, Kafka and Storm
Just founded Datalayer
- **Map Reduce V1 Limits**
  - **Scalability**
    - Maximum Cluster size – 4,000 nodes
    - Maximum concurrent tasks – 40,000
    - Coarse synchronization in JobTracker
  - **Availability**
    - Job Tracker failure kills all queued and running jobs
    - No alternate paradigms and services
    - Iterative applications implemented using MapReduce are slow (HDFS read/write)

- **Map Reduce V2 (= “NextGen”) based on YARN**
  - (not 'mapred' vs 'mapreduce' package)
YARN as a Layer

All problems in computer science can be solved by another level of indirection
– David Wheeler

YARN a.k.a. Hadoop 2.0 separates the cluster and resource management from the processing components.
Components

- A global Resource Manager
- A per-node slave Node Manager
- A per-application Application Master running on a Node Manager
- A per-application Container running on a Node Manager
Yahoo! has been running 35000 nodes of YARN in production for over 8 months now since begin 2013

[http://strata.oreilly.com/2013/06/moving-from-batch-to-continuous-computing-at-yahoo.html ]
Our Federated / HA / Yarn clusters (K's of nodes) completed ~2M jobs; We can now truly say we have #Apache #Hadoop 2 in production.
Get It!

- Download
  - http://www.apache.org/dyn/closer.cgi/hadoop/common/
- Unzip and configure
  - mapred-site.xml
    - mapreduce.framework.name = yarn
  - yarn-site.xml
    - yarn.nodemanager.aux-services = mapreduce_shuffle
    - yarn.nodemanager.aux-services.mapreduce_shuffle.class = org.apache.hadoop.mapred.ShuffleHandler
<table>
<thead>
<tr>
<th>Service</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namenode</td>
<td><a href="http://namenode:50070">http://namenode:50070</a></td>
</tr>
<tr>
<td>Namenode Browser</td>
<td><a href="http://namenode:50075/logs">http://namenode:50075/logs</a></td>
</tr>
<tr>
<td>Secondary Namenode</td>
<td><a href="http://snamenode:50090">http://snamenode:50090</a></td>
</tr>
<tr>
<td>Resource Manager</td>
<td><a href="http://manager:8088/cluster">http://manager:8088/cluster</a></td>
</tr>
<tr>
<td>Application Status</td>
<td><a href="http://manager:8089/proxy/">http://manager:8089/proxy/</a>&lt;app-id&gt;</td>
</tr>
<tr>
<td>Resource Node Manager</td>
<td><a href="http://manager:8042/node">http://manager:8042/node</a></td>
</tr>
<tr>
<td>Mapreduce JobHistory Server</td>
<td><a href="http://manager:19888">http://manager:19888</a></td>
</tr>
</tbody>
</table>
YARNed

- Batch
  - Map Reduce
  - Hive / Pig / Cascading / ...
- Graph
  - Giraph
  - Hama
  - OpenMPI
- Streaming
  - Storm
  - Spark
  - Kafka
- Realtime
  - HBase
  - Memcached
Batch

Apache Tez: Fast response times and extreme throughput to execute complex DAG of tasks

“The future of #Hadoop runs on #Tez”
Streaming

- Storm [https://github.com/yahoo/storm-yarn]
  - Storm-YARN enables Storm applications to utilize the computational resources in a Hadoop cluster along with accessing Hadoop storage resources such as HBase and HDFS

- Spark
  - Need to build a YARN-Enabled Assembly JAR
  - Goal is more to integrate Map Reduce e.g. SIMR supports MRV1

- Kafka with Samza [http://samza.incubator.apache.org]
  - Implements StreamTask
    - Execution Engine: YARN
    - Storage Layer: Kafka, not HDFS
From “Storm and Hadoop: Convergence of Big-Data and Low-Latency Processing | YDN Blog - Yahoo.html”

@datalayerio | hacks@datalayer.io | https://github.com/datalayer
HBase

- Hoya [https://github.com/hortonworks/hoya.git]
  - Allows users to create on-demand HBase clusters
  - Allow different users/applications to run different versions of HBase
  - Allow users to configure different HBase instances differently
  - Stop / Suspend / Resume clusters as needed
  - Expand / shrink clusters as needed
  - CLI based
Graph

- Giraph
  - Offline batch processing of semi-structured graph data on a massive scale
    - Compatible with Hadoop 2.x
    - "Pure YARN" build profile
  - Manages Failure Scenarios
    - Worker/container failure during a job?
    - What happens if our App Master fails during a job?
  - Application Master allows natural bootstrapping of Giraph jobs
- Next Steps
  - Zookeeper in AM
  - Own Management WEB UI
  - ...

- Abstracting the Giraph framework logic away from MapReduce has made porting Giraph to other platforms like Mesos possible

*(from “Giraph on YARN - Qcon SF”)*
Options

- Apache Mesos
  - Cluster manager
  - Can run Hadoop, Jenkins, Spark, Aurora...

- Apache Helix
  - Generic cluster management framework
  - YARN automates service deployment, resource allocation, and code distribution. However, it leaves state management and fault-handling mostly to the application developer.
  - Helix focuses on service operation but relies on manual hardware provisioning and service deployment.
You Looser!

- More Devops and IO
- Tuning and Debugging the Application Master and Container is hard
- Both AM and RM based on an asynchronous event framework
  - No flow control
  - Deal with RPC Connection loose - Split Brain, AM Recovery... !!!
  - What happens if a worker/container or a App Master fails?
- New Application Master per MR Job - No JVM Reuse for MR
  - Tez-on-Yarn will fix these
- No Long living Application Master (see YARN-896)
- New application code development difficult
- Resource Manager SPOF (chuch... don't even ask this)
- No mixed V1/V2 Map Reduce (supported by some commecrial distribution)
You Rocker!

- Sort and Shuffle speed gain for Map Reduce
- Real-time processing with Batch Processing Collocation brings
  - Elasticity to share resource (Memory/CPU/...)
  - Sharing data between realtime and batch - Reduce network transfers and total cost of acquiring the data
- High expectations from #Tez
  - Long Living Sessions
  - Avoid HDFS Read/Write
- High expectations from #Twill
  - Remote Procedure Calls between containers
  - Lifecycle Management
  - Logging
WHY porting your App on YARN?

- Benefit from existing -*yarn projects
- Reuse unused cluster resource
- Common Monitoring, Management and Security framework
- Avoid HDFS write on reduce (via Tez)
- Abstract and Port to other platforms
Summary

- YARN brings
  - One component, One responsibility!!!
    - Resource Management
    - Data Processing
  - Multiple applications and patterns in Hadoop

- Many organizations are already building and using applications on YARN

- Try YARN and Contribute!
Thank You!
(Special Thx to @acmurthy and @steveloughran for helping tweets)
Questions?
@echarles @datalayerio
http://datalayer.io/hacks
http://datalayer.io/jobs