Running Spark on YARN*

Montasser AKERMI

Last Update: March, 2025

*The course material is hosted at https://akermi.org/.

Learning objectives

Spark can be run on different deployment modes: Standalone, YARN, Kubernetes, and Mesos (which is deprecated as of Spark 3.2.0). The objectives of this session are to run Spark on an existing YARN cluster, learn how to submit Spark jobs to the YARN cluster, and enable Spark jobs to log to HDFS.

Prerequisites

- Java version 1.8
- Hadoop cluster version 3.x

Setting up Spark to run on YARN

1. Download Spark only in one node (e.g., YARN master node):

   ```
   wget https://dlcdn.apache.org/spark/spark-3.5.5/spark-3.5.5-bin-hadoop3.tgz
   ```

2. Extract the downloaded archive:

   ```
   tar -xvf spark-3.5.5-bin-hadoop3.tgz
   ```

3. Rename Spark home folder

   ```
   mv spark-3.5.5-bin-hadoop3 spark-3.5.5
   ```

4. Set the necessary variable environments in `.bashrc`

   ```
   export SPARK_HOME=$HOME/spark-3.5.5
   export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
   # export LD_LIBRARY_PATH=$HADOOP_HOME/lib/native:$LD_LIBRARY_PATH
   ```

5. Update /.bashrc

   ```
   source ~/.bashrc
   ```

6. Create `spark-defaults.conf` file

   ```
   nano $SPARK_HOME/conf/spark-defaults.conf
   ```

7. Set spark.master to yarn, set the default memory allocated to Spark Driver in cluster mode, and configure Spark executors' memory allocation, all in `spark-defaults.conf`

   ```
   spark.master  yarn
   spark.driver.memory  2g
   spark.executor.memory  2g
   ```

8. Configure memory allocation in `yarn-site.xml` file

   ```
   <property>
     <name>yarn.scheduler.maximum-allocation-mb</name>
     <value>4096</value>
   </property>
   ```

Submitting Spark jobs to YARN cluster

1. To calculate Pi using SparkPi on YARN:

```
spark-submit --deploy-mode cluster --class org.apache.spark.examples.SparkPi \
  $SPARK_HOME/examples/jars/spark-examples_3.5.5.jar 10
```

2. To count words in text files in a given directory:

```
spark-submit --deploy-mode cluster --class tp.WordCount wordcount.jar input output
```

While the application is running in client mode, check different web user interfaces (UI):

- Spark web UI on `http://nn:4040`
- YARN web UI on `http://nn:8088`
- Namenode web UI on `http://nn:9870`
- Hadoop JobHistory web UI on `http://nn:19888`

Enabling Spark jobs to log to HDFS

1. Enable eventLog and set up the Spark log directory in `spark-defaults.conf`

```
spark.eventLog.enabled  true
spark.eventLog.dir  hdfs://nn:9000/spark-logs
```

2. Create spark-log directory

```
hadoop fs -mkdir /spark-logs
```

3. Configure history server in `spark-defaults.conf`

```
spark.history.provider  org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.logDirectory  hdfs://nn:9000/spark-logs
spark.history.fs.update.interval  10s
spark.history.ui.port  18080
```

4. Run the history server

```
start-history-server.sh
```

Running Spark shell

The Spark shell can be accessed by executing the following command:

```
spark-shell
```