How to run Spark

Igor Yakushin
ivy2@uchicago.edu

December 27, 2019

Spark on midway cluster: sinteractive

- So far we have seen:
  - How to submit Spark job to a local node with spark-submit or spark-sql
  - How to run pyspark locally in python interpreter
  - How to run pyspark locally in jupyter notebook
- If you are doing some heavy computations on login nodes your jobs might get killed since login nodes are not meant for it
- Instead you might want to grab a compute node and do the same processing there. For example:

```
sinteractive -p broadwl --exclusive --time=3:00:00
```

- The above will give you all the cores (28) and memory (64G) on one compute node from broadwl partition for 3 hours and would allow you to work interactively.
- You can even run jupyter notebook on the compute nodes. For more details, see References section.

Spark on midway cluster: Lab 7: single node in batch

- If you want to submit your job to a queue and run it on a single node instead of working interactively, create a batch file, single.batch

```
#!/bin/bash

#SBATCH --job-name=single
#SBATCH --exclusive
#SBATCH --nodes=1
#SBATCH --time=00:10:00
#SBATCH --partition=broadwl
#SBATCH --output=single_%j.out
#SBATCH --error=single_%j.err
####SBATCH --account=rcc-guest

module load spark/2.3.0
export MASTER="local[*]"
spark-submit --master $MASTER perceptron.py
```

Spark on midway cluster: Lab 7: single node in batch

- Notice: we are reusing perceptron example from Lab 5.
- To submit a job to midway

  `sbatch single.batch`
- To monitor the job, use

  `squeue -j <jobid>`

  or

  `squeue -u <username>`
- To cancel the job

  `scancel <jobid>`

Spark on midway cluster: Lab 7: multiple nodes in batch

- If you want to take advantage of using multiple nodes, you have to start master Spark server on one node and slave Spark servers on the other nodes, possibly including the master node. Slaves should know the address of the master.
- The job is submitted to the master Spark server.
- To automate this procedure, start-spark-slurm.sh and stop-spark-slurm.sh scripts are used.

```
...
module load spark/2.3.0

start-spark-slurm.sh

export MASTER=spark://$HOSTNAME:7077
spark-submit --master $MASTER perceptron.py

stop-spark-slurm.sh
```

Spark on Hadoop cluster: batch

- To submit a job to a Hadoop cluster you simply need to use `spark2-submit --master yarn <your program>.py` on the login node of the Hadoop cluster
- Hadoop takes care of distributing the job across the nodes. You can overwrite default number of executors in command line arguments to `spark-submit`
- Under Hadoop, Spark expects its input in HDFS and puts the output into HDFS if you use I/O-related command on RDD or DataFrame. You can still get input from local file system by using the full path with `file:///` in front.
- To run `perceptron.py` from Lab 7 inside Hadoop:

```
hdfs dfs -put data
make hadoop
```

Spark on Hadoop cluster: jupyter

- Point your browser to

  `https://hadoop.rcc.uchicago.edu/`
- Login using your midway credentials.
- Browse into `Spark/labs/3` and open `lab3.ipynb`
- Change the kernel to `pySpark 2.2.0`:
  `Kernel -> Change Kernel -> pySpark 2.2.0`
- One can execute a cell in the notebook with Shift+Enter.
- The main thing to remember: unless you shut down the notebook, it continues running and using Hadoop resources even if you close the browser and turn off your computer!!! As a result, the next user might not be able to get access to Spark either from jupyter or even in batch. This happens especially often at the end of the semester when students are doing the final project. There is a script running killing pyspark jobs that have been running for more than 3 hours.

Spark on Hadoop cluster: jupyter

Two ways to shut down a jupyter notebook:

- When inside the notebook: `File -> Close and Halt`
- When in the file browser outside of the notebook: select "Running" tab and press the yellow "Shutdown" button near any running notebook.