Lab 3

Big Data, February 6, 2017

Labs 1 and 2: Common Issues

- Confusion about command-line utilities (`cd`, `ls`, `ssh`, etc.)
  - See HPC@NYU tutorial, linked in Lab 1 Part 2.
  - Tons of online resources (e.g., Google “command line tutorial”)
  - RTFM (“read the f***ing manual”): `man` *`command`* (e.g., `man ls`)
- Incorrectly configured aliases (e.g., `hfs` and `hjs` in Lab 1) and `.bashrc` file
  - *Do not* put `source .bashrc` in `.bashrc` (infinite recursion)
  - Add the following lines to your `.bash_profile` to invoke `.bashrc` (if not already present):

```
if [ -f ~/.bashrc ]; then

     source ~/.bashrc

fi
```

- Writing code in a Windows text editor (*recommendation: don’t*)
  - Different “newline” characters (Linux uses 0x0A, Windows: 0x0D0A)

Lab 2 Review: Skills Developed

- Writing a MapReduce program with Hadoop Streaming and Python
- Programming an algorithm from a high-level description
- Measuring the scalability of a parallel program (benchmarking)
- Optimizing a program by reducing data movement (blocked algorithm)

Lab 2 Review: Testing and Debugging Tips

Pro Tip: Search for error messages on Google!

1. Test with `cat` *`infiles`* `|` `python` *`mapper`* `|` `sort -n` `|` `python` *`reducer`*
   - *`infiles`* = list of input files
   - *`mapper`*`/`*`reducer`* = Python scripts implementing Map and Reduce

2. Test in Hadoop Streaming with 1 mapper and 1 reducer
   - Add `hadoop` flags `-D mapreduce.job.maps=1 -D mapreduce.job.reduces=1`

3. Study Hadoop error logs
   - `yarn logs -applicationId` *`application_ID`*
   - *`application_ID`* is a string like `application_1484865967044_2732`
   - Tons of output, usually the error is in the first few lines

Review: Apache Hadoop

(Apache) Hadoop is a software framework whose main modules are

- *Hadoop MapReduce*: Java library for writing MapReduce program
- *HDFS (Hadoop Distributed File System)*
- *Hadoop YARN (Yet Another Resource Negotiator)*: cluster manager/scheduler
- *Hadoop Common*: shared libraries

“Hadoop ecosystem”: collection of projects that extend Hadoop in various ways.

- Many are also Apache software projects
- This class: Apache Spark, Apache Hive, perhaps others.

Hadoop Streaming vs. Hadoop MapReduce

Hadoop Streaming is a ‘utility’ included in the Hadoop project:

- Exposes Hadoop MapReduce to other languages (e.g., shell scripts, Python).
- Uses standard streams to communicate with mapper/reducer.
- Technically, it is a (Java) Hadoop MapReduce application, whose `map()`/`reduce()` methods invoke user-provided executables.

Caveats:

- Exposes very limited subset of Hadoop MapReduce functionality.
- Potential performance hit due to I/O over standard streams.
- Reducer inputs a list of (*key*, *value*) pairs, vs. a single (*key*, *list-of-values*).

Other Python MapReduce Options

Apache Spark: biggest Hadoop MapReduce competitor (today)

Hadoop Streaming wrappers for Python:

- mrjob (by Yelp) and Luigi (by Spotify) include tools for task-chaining.

Other MapReduce implementations for Python

- Built-in `map()` and `reduce()` functions (v3+, `functools.reduce()`)
  - "Syntactic sugar", so don't expect any performance boost.
- Thread/process pools have `map()` and `reduce()` methods
  - Process pools: see class `Pool` in package `multiprocessing`
  - Thread pools: see class `Pool` in package `multiprocessing.dummy`
- Disco (by Nokia): Hadoop reimplemented in Erlang+Python. (Still alive?)

Lab 3: Amazon Web Services (AWS)

Follow the Lab 3 PDF on NYU Classes

Learning Objectives:

1. Create an S3 bucket
2. Launch an Elastic MapReduce (EMR) cluster
3. Run an EMR job
4. *Terminate your cluster!!*

Note: AWS provides a command-line interface, in addition to the graphical web interface:

https://aws.amazon.com/documentation/cli/