Exercises

Unix Skills

Throughout the module, we will use our virtual machine. If you don't feel comfortable to work with Unix, make sure to read this tutorial (Tutorial seven is not needed).

You can try out the commands on your virtual machine. Just SSH into your machine with `ssh student@bdlc-XX.el.eee.intern`, where `XX` is your personal virtual machine number.

Remember, you have root access, so don't blindly trust internet tutorials. Especially if commands need `sudo`.

The Installation Guide should be successfully finished

To verify, run

```
su - hadoop
~/hadoop/bin/hadoop
```

which should not produce an error.

Run Some Hadoop Examples

Try to run some provided examples. An example program must be given as the first argument.

You will get the list of examples with

```
~/hadoop/bin/hadoop jar ~/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.1.jar
```

Valid program names are:

```
An example program must be given as the first argument.
Valid program names are:
  aggregatewordcount: An Aggregate based map/reduce program that counts the words in the input files.
  aggregatewordhist: An Aggregate based map/reduce program that computes the histogram of the words in the input files.
  bbp: A map/reduce program that uses Bailey-Borwein-Plouffe to compute exact digits of Pi.
  dbcount: An example job that count the pageview counts from a database.
  distbbp: A map/reduce program that uses a BBP-type formula to compute exact bits of Pi.
  grep: A map/reduce program that counts the matches of a regex in the
```

```
input.
  join: A job that effects a join over sorted, equally partitioned
datasets
  multifilewc: A job that counts words from several files.
  pentomino: A map/reduce tile laying program to find solutions to
pentomino problems.
  pi: A map/reduce program that estimates Pi using a quasi-Monte Carlo
method.
  randomtextwriter: A map/reduce program that writes 10GB of random
textual data per node.
  randomwriter: A map/reduce program that writes 10GB of random data per
node.
  secondarysort: An example defining a secondary sort to the reduce.
  sort: A map/reduce program that sorts the data written by the random
writer.
  sudoku: A sudoku solver.
  teragen: Generate data for the terasort
  terasort: Run the terasort
  teravalidate: Checking results of terasort
  wordcount: A map/reduce program that counts the words in the input
files.
  wordmean: A map/reduce program that counts the average length of the
words in the input files.
  wordmedian: A map/reduce program that counts the median length of the
words in the input files.
  wordstandarddeviation: A map/reduce program that counts the standard
deviation of the length of the words in the input files.
```

If a valid program name is provided as the first parameter, one sees the usage. E.g for Pi estimation with Monte Carlo:

```
~/hadoop/bin/hadoop jar ~/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-
examples-3.3.1.jar pi
```

We see the desired usage:

```
Usage: org.apache.hadoop.examples.QuasiMonteCarlo <nMaps> <nSamples>
```

Pi Estimation

Run the Pi estimator with `1` mapper and `1` sample.

```
~/hadoop/bin/hadoop jar ~/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-
examples-3.3.1.jar pi 1 1
```

Play around with the `<nMaps>` and `<nSamples>` and get a feeling for accuracy vs runtime.

Sudoku Solver

Let us try to solve a Sudoku. After "googling" for `hardest sudoku ever`, write a new file, called `puzzle.dat`, to your home directory with the hard Sudoku puzzle as the content.

```
cat puzzle.dat
8 ? ? ? ? ? ? ? ?
? ? 3 6 ? ? ? ? ?
? 7 ? ? 9 ? 2 ? ?
? 5 ? ? ? 7 ? ? ?
? ? ? ? 4 5 7 ? ?
? ? ? 1 ? ? ? 3 ?
? ? 1 ? ? ? ? 6 8
? ? 8 5 ? ? ? 1 ?
? 9 ? ? ? ? 4 ? ?
```

```
~/hadoop/bin/hadoop jar ~/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-
examples-3.3.1.jar sudoku ~/puzzle.dat
```