Ex1: MapReduce

4 points
Weeks 2 - 4

WordCount Example

http://wiki.apache.org/hadoop/WordCount

- reads text files and counts how often words occur
- input is text files
- output is text files
  - each line of which contains a word and the count of how often it occurred, separated by a tab.
- each mapper takes a line as input and breaks it into words then emits a key/value pair of the word and 1
- each reducer sums the counts for each word and emits a single key/value with the word and sum.

WordCount Example

```
 1 package org.myorg;
 2
 3 import java.io.IOException;
 4 import java.util.*;
 5
 6 import org.apache.hadoop.fs.Path;
 7 import org.apache.hadoop.conf.*;
 8 import org.apache.hadoop.io.*;
 9 import org.apache.hadoop.mapreduce.*;
10 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
11 import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
12 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
13 import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
14
```

WordCount Example

```
 15 public class WordCount {
 16
 17 public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {
 18    private final static IntWritable one = new IntWritable(1);
 19    private Text word = new Text();
 20
 21    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
 22        String line = value.toString();
 23        StringTokenizer tokenizer = new StringTokenizer(line);
 24        while (tokenizer.hasMoreTokens()) {
 25            word.set(tokenizer.nextToken());
 26            context.write(word, one);
 27        }
 28    }
 29 }
```

WordCount Example

```
30
31 public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable> {
32
33   public void reduce(Text key, Iterable<IntWritable> values, Context context)
34     throws IOException, InterruptedException {
35       int sum = 0;
36       for (IntWritable val : values) {
37           sum += val.get();
38       }
39       context.write(key, new IntWritable(sum));
40   }
41 }
42
```

WordCount Example

```
43 public static void main(String[] args) throws Exception {
44   Configuration conf = new Configuration();
45
46   Job job = new Job(conf, "wordcount");
47
48   job.setOutputKeyClass(Text.class);
49   job.setOutputValueClass(IntWritable.class);
50
51   job.setMapperClass(Map.class);
52   job.setReducerClass(Reduce.class);
53
54   job.setInputFormatClass(TextInputFormat.class);
55   job.setOutputFormatClass(TextOutputFormat.class);
56
57   FileInputFormat.addInputPath(job, new Path(args[0]));
58   FileOutputFormat.setOutputPath(job, new Path(args[1]));
59
60   job.waitForCompletion(true);
61 }
62 }
```

Compile and Run WordCount.java

Bring up a terminal (ctrl+alt+t) and check if all hadoop processes are running by executing jps

```
> jps
2352 SecondaryNameNode
2195 DataNode
9415 Jps
2072 NameNode
2652 JobHistoryServer
2493 ResourceManager
2591 NodeManager
```

If not, execute the following command lines

```
> start-dfs.sh
> start-yarn.sh
> mr-jobhistory-daemon.sh start historyserver
```

cd into the first exercise directory

```
> cd Exercises/Ex1
> make WordCount
```

Wait....

```
> hdfs dfs -cat wc_out/*
```

Your Tasks

- Week 1: WordCount.java
  - [E](10%) Introduce a combiner to the word count program
  - Answer the following questions:
    - [E](5%) Do we need to create a new method for the combiner? Explain why/why not.
    - [E](5%) How would you modify this program so that it counts the word-length frequencies instead?

MaxWordCount Example

- Stage 1: WordCount
- Stage 2: Find the most frequent word
  - Input: WordCount's output `<Word>\t<Count>`
  - Output: `<DummyKey>\t<Word>\t<Count>` (single line)

Your Tasks

- Week 2-3: MaxWordCount.java
  - [E](15%) Complete the 2nd Mapper
  - [E](15%) Complete the 2nd Reducer
  - Answer the following question:
    - [E](5%) How do we ensure that all entries in the 2nd stage go to the same reducer?
    - [E](5%) Do we need to create a separate combiner for the 2nd stage? Explain why/why not.

TopKWordCount Example

- Stage 1: WordCount
- Stage 2: Find the top-k most frequent words
  - Input: WordCount's output `<Word>\t<Count>`
  - Output: `<DummyKey>\t<Word>\t<Count>` (k lines)
- The k value is specified as a command line argument

Your Tasks

- Week 2-4: TopKWordCount.java
  - [E](5%) Create a 2nd Mapper
  - [D](10%) Create a 2nd Reducer
  - Answer the following question:
    - [E](5%) How do Mappers and Reducers accept user-defined parameters?

Your Tasks

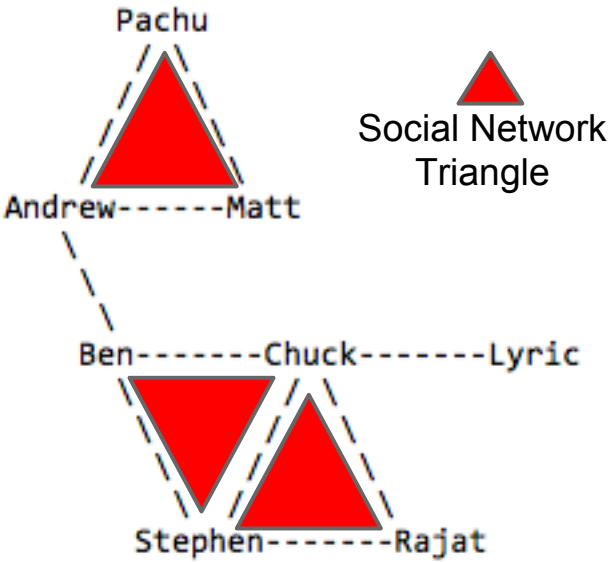
- Week 3-4: WordCount2.java
  - [D](10%) Modify the original WordCount mapper to perform local data reductions using a HashMap.
- Week 3-4: NGram.java
  - [D](10%) Modify the original WorldCount program to compute n-gram frequencies where the n value is specified as a command line argument

Additional Exercises

- Triangle Finding (http://www.vertica.com/2011/09/21/counting-triangles/)

Input:

```
Ben       Chuck
Ben       Stephen
Chuck     Stephen
Chuck     Rajat
Rajat     Stephen
Andrew    Ben
Andrew    Matt
Matt      Pachu
Chuck     Lyric
```



Additional Exercises

- Nearest Airport Search
  - Given a set P of airports and a query point q, find the airport p in P that minimizes the distance from p.
  - Data: List of airports with (lat,lon) coordinates (https://raw.githubusercontent.com/jpatokal/openflights/master/data/airports.dat)
  - Distance function: Great-circle distance (https://en.wikipedia.org/wiki/Great-circle_distance)