Hadoop Streaming

Originals of slides and source code for examples: http://www.coreservlets.com/hadoop-tutorial/
Also see the customized Hadoop training courses (onsite or at public venues) – http://courses.coreservlets.com/hadoop-training.html

Customized Java EE Training: http://courses.coreservlets.com/
Hadoop, Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at your location.

For live customized Hadoop training (including prep for the Cloudera certification exam), please email info@coreservlets.com

Taught by recognized Hadoop expert who spoke on Hadoop several times at JavaOne, and who uses Hadoop daily in real-world apps. Available at public venues, or customized versions can be held on-site at your organization.

• Courses developed and taught by Marty Hall
  – JSF 2.2, PrimeFaces, servlets/JSP, Ajax, jQuery, Android development, Java 7 or 8 programming, custom mix of topics
  – Courses available in any state or country. Maryland/DC area companies can also choose afternoon/evening courses.
• Courses developed and taught by coreservlets.com experts (edited by Marty)
  – Spring, Hibernate/JPA, GWT, Hadoop, HTML5, RESTful Web Services
Contact info@coreservlets.com for details
Agenda

• Implement a Streaming Job
• Contrast with Java Code
• Create counts in Streaming application

Hadoop Streaming

• Develop MapReduce jobs in practically any language
• Uses Unix Streams as communication mechanism between Hadoop and your code
  – Any language that can read standard input and write standard output will work
• Few good use-cases:
  – Text processing
    • scripting languages do well in text analysis
  – Utilities and/or expertise in languages other than Java
Streaming and MapReduce

- Map input passed over standard input
- Map processes input line-by-line
- Map writes output to standard output
  - Key-value pairs separate by tab (‘\t’)
- Reduce input passed over standard input
  - Same as mapper output – key-value pairs separated by tab
  - Input is sorted by key
- Reduce writes output to standard output

Implementing Streaming Job

1. **Choose a language**
   - Examples are in Python
2. **Implement Map function**
   - Read from standard input
   - Write to standard output - keys-value pairs separated by tab
3. **Implement Reduce function**
   - Read key-value from standard input
   - Write out to standard output
4. **Run via Streaming Framework**
   - Use $yarn command
1: Choose a Language

- Any language that is capable of
  - Reading from standard input
  - Writing to standard output
- The following example is in Python
- Let’s re-implement StartsWithCountJob in Python

```
#!/usr/bin/python
import sys

for line in sys.stdin:
    for token in line.strip().split(" "):  
        if token: print token[0] + '\t1'
```

2: Implement Map Code - countMap.py

1. Read one line at a time from standard input
2. tokenize
3. Emit first letter, tab, then a count of 1
3: Implement Reduce Code

- **Reduce is a little different from Java MapReduce framework**
  - Each line is a key-value pair
  - Differs from Java API
    - Values are already grouped by key
    - Iterator is provided for each key
  - You have to figure out group boundaries yourself
- **MapReduce Streaming will still sort by key**

```python
#!/usr/bin/python
import sys
(lastKey, sum)=(None, 0)
for line in sys.stdin:
    (key, value) = line.strip().split("\t")
    if lastKey and lastKey != key:
        print lastKey + '\t' + str(sum)
        (lastKey, sum) = (key, int(value))
    else:
        (lastKey, sum) = (key, sum + int(value))
if lastKey:
    print lastKey + '\t' + str(sum)
```
4: Run via Streaming Framework

- Before running on a cluster it’s very easy to express MapReduce Job via unit pipes

```bash
$ cat testText.txt | countMap.py | sort | countReduce.py
a 1
h 1
i 4
s 1
t 5
v 1
```

- Excellent option to test and develop

```bash
yarn jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.*.jar \
-D mapred.job.name="Count Job via Streaming" \  
-files $HADOOP_SAMPLES_SRC/scripts/countMap.py,\ 
   $HADOOP_SAMPLES_SRC/scripts/countReduce.py \ 
-input /training/data/hamlet.txt \ 
-output /training/playArea/wordCount/ \ 
-mapper countMap.py \ 
-combiner countReduce.py \ 
-reducer countReduce.py
```

-files option makes scripts available on the cluster for MapReduce
## Python vs. Java Map Implementation

### Python
```
#!/usr/bin/python
import sys
for line in sys.stdin:
    for token in line.strip().split(" "): 
        if token: print token[0] + '1'
```

### Java
```
package mr.wordcount;
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.
import org.apache.hadoop.mapreduce.Mapper;
public class StartsWithCountMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable countOne = new IntWritable(1);
    private final Text reusableText = new Text();
    @Override
    protected void map(LongWritable key, Text value, Context context)
    throws IOException, InterruptedException {
        StringTokenizer tokenizer = new StringTokenizer(value.toString());
        while (tokenizer.hasMoreTokens()) {
            reusableText.set(tokenizer.nextToken().substring(0, 1));
            context.write(reusableText, countOne);
        }
    }
}
```

## Python vs. Java Reduce Implementation

### Python
```
#!/usr/bin/python
import sys
(lastKey, sum)=(None, 0)
for line in sys.stdin:
    (key, value) = line.strip().split(" ")
    if lastKey and lastKey != key:
        print lastKey + ' ' + str(sum)
        (lastKey, sum) = (key, int(value))
    else:
        (lastKey, sum) = (key, sum + int(value))
if lastKey:
    print lastKey + ' ' + str(sum)
```

### Java
```
package mr.wordcount;
import java.io.IOException;
import org.apache.hadoop.io.
import org.apache.hadoop.mapreduce.Reducer;
public class StartsWithCountReducer extends 
    Reducer<Text, IntWritable, Text, IntWritable> {
    @Override
    protected void reduce(Text token, Iterable<IntWritable> counts,
            Context context) throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable count : counts) {
            sum+= count.get();
        }
        context.write(token, new IntWritable(sum));
    }
}
```
Reporting in Streaming

- Streaming code can increment counters and update statuses
- Write string to **standard error** in “streaming reporter” format
- To increment a counter:

  ```sh
  reporter:counter:<counter_group>,<counter>,<increment_by>
  ```

```python
#!/usr/bin/python
import sys

for line in sys.stdin:
    for token in line.strip().split(" "):
        if token:
            sys.stderr.write("reporter:counter:Tokens,Total,1\n")
    print token[0] + "\t1"
```

Print counter information in “reporter protocol” to standard error
Wrap-Up

Customized Java EE Training: http://courses.coreservlets.com/
Hadoop, Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at your location.

Summary

• Learned how to
  – Implement a Streaming Job
  – Create counts in Streaming application
  – Contrast with Java Code
Questions?

More info:
http://www.coreservlets.com/hadoop-tutorial - Hadoop programming tutorial
http://courses.coreservlets.com/hadoop-training.html - Customized Hadoop training courses, at public venues or onsite at your organization
http://courses.coreservlets.com/Course-Materials/java.html - General Java programming tutorial
http://www.coreservlets.com/java-8-tutorial - Java 8 tutorial
http://www.coreservlets.com/jsf-2-tutorial - JSF 2.1 tutorial
http://www.coreservlets.com/JSF-Tutorial/primefaces/ - PrimeFaces tutorial
http://www.coreservlets.com/JSF-Tutorial/jsf2/ - JSF 2.2 tutorial
http://coreservlets.com/ - JSF 2, PrimeFaces, Java 7 or 8, Ajax, jQuery, Hadoop, RESTful Web Services, Android, HTML5, Spring, Hibernate, Servlets, JSP, GWT, and other Java EE training

Customized Java EE Training: http://courses.coreservlets.com/
Hadoop, Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at your location.