Streaming

Igor Yakushin
ivy2@uchicago.edu

MapReduce: using streaming

- Streaming interface allows one to write MapReduce in any language although the functionality is limited and the performance is worse than what one can get by using native Java API.
- Mapper takes input from stdin, breaks it into records - lines in a text file - and is expected to print to stdout pairs of key and value separated by Tab; everything before the first Tab is considered a key, the rest of the line is considered a value
- Similarly reducer is expected to receive such pairs in stdin and prints its results to stdout
- When launching a job, one needs to use Hadoop's streaming jar and specify mapper, reducer, input and output files as options

MapReduce: using streaming

- In this example we implement WordCount.java functionality in python
- mapper.py:

```
import sys

for line in sys.stdin:
    line = line.strip()
    words = line.split()
    for word in words:
        print '%s\t%s' % (word, 1)
```

MapReduce: using streaming

- reducer.py:

```
from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

for line in sys.stdin:
    line = line.strip()
    word, count = line.split('\t', 1)
    try:
        count = int(count)
    except ValueError:
        continue

    if current_word == word:
        current_count += count
    else:
        if current_word:
            print '%s\t%s' % (current_word, current_count)
        current_count = count
        current_word = word

if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

MapReduce: using streaming

- To run a job:

```
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/hadoop-streaming.jar \
       -input /user/$USER/wordcount/README \
       -output /user/$USER/wordcount/streaming-out-py \
       -file mapper.py -mapper mapper.py -file reducer.py -reducer reducer.py
```