CS 378 – Big Data Programming

Lecture 8

Custom Writable

Review

- Assignment 3 - InvertedIndex
- We'll look at implementation details of:
  - Mapper
  - Combiner (amount of shuffle data reduced slightly)
  - Reducer
  - Supporting classes
- Other questions/issues?

Assignment 4

- Recall the word statistics output:
  - Key: word
  - Value: document count, mean, variance

- Suppose that we collect this info every day
- Then aggregate these stats by week, month,…
  - But not reprocess the original emails for each aggregation

- Need to output counts with stats

Assignment 4

- Modified word statistics output:
  - Key: word
  - Value:
    - Longs: document count, total count, sum of squares
    - Doubles: mean, variance

- We have mixed types, so **`LongArrayWritable`** and **`DoubleArrayWritable`** aren't appropriate
- Solution: Write a custom class for our stats data

Review Writable

- Hadoop **`Writable`** interface
  - Inputs to and outputs from **`map()`**
  - Inputs to and outputs from **`reduce()`**
- Implements serialization for I/O
- Required methods:
  - **`readFields(DataInput in)`**
  - **`write(DataOutput out)`**

- Let's call our class: **`WordStatisticsWritable`**

Custom Writable

- Approach 1 for **`WordStatisticsWritable`**:
  - Include a **`LongArrayWritable`** and a **`DoubleArrayWritable`**

- Required methods:
  - **`write(DataOutput out)`**
  - Writes a **`LongArrayWritable`**, then a **`DoubleArrayWritable`**
  - **`readFields(DataInput in)`**
  - Reads a **`LongArrayWritable`**, then a **`DoubleArrayWritable`**

Custom Writable

- Approach 2 for **`WordStatisticsWritable`**:
  - Use primitive Java types (**`long`**, **`double`**)

- Required methods:
  - **`write(DataOutput out)`**
  - Write primitive values to **`DataOutput`** instance
    - **`writeLong(), writeDouble()`**
  - **`readFields(DataInput in)`**
  - Read primitive values from **`DataInput`** instance
    - **`readLong(), readDouble()`**

Custom Writable

- What other methods might we need for **`WordStatisticsWritable`**?

- For output to text file:
  - **`toString()`**

- For MRUnit tests:
  - **`equals()`**

Assignment 4 – Job 1

- For the daily run, input is files containing emails
- Output is text file (using **`TextOutputFormat`**):
  - Key: word
  - Tab
  - Values (comma separated):
  - document count, total count, sum of squares, mean, variance

- To aggregate multiple days of data, we need a job that reads multiple days of data in this format

Aggregator Job

- You'll write a second map-reduce job that:
- Reads the text files output by WordStatistics
  - From multiple runs of WordStatistics
- Aggregates the values:
  - Sum the counts
  - Compute new mean, variance
- Outputs the same format as WordStatistics
  - Might want to aggregate multi-day statistics as well

Aggregator

- Recall that Hadoop provides a file format class to read output generated with **`TextFileFormat`**:
  - **`KeyValueTextInputFormat`**

- Aggregator job should use this input file format
- Mapper converts input to **`WordStatisticsWritable`**
- Combiner - Can we use one?
- What does the reducer do?

Assignment 4

- Bonus: Write a single reduce class that works for:
  - WordStatistics combiner and reducer, and
  - WordStatisticsAggregator combiner and reducer.