

CS 378 – Big Data Programming

Lecture 5

Summarization Patterns

Review

- Assignment 2 – Questions?
- mrunit – How do you test `map()` or `reduce()` calls that produce multiple outputs?
- Issues with calculating variance

Summarization

- Other summarizations of interest
 - Min, max, median
- Suppose we are interested in these metrics for paragraph length (Assignment 2 data)
 - If paragraph lengths are normally distributed, then the median will be very near the mean
 - If the distribution of paragraph lengths is skewed, then the mean and median will be very different

Summarization

- Min and max are straightforward
- For each paragraph, output two values
 - Min length (the length of the current paragraph)
 - Max length (the length of the current paragraph)
 - Key?
- Combiner will get a key and list of values pair
 - Select the min, max from the list, output the values
 - Key?
- Reducer does the same

Summarization

- Median
 - Get all the values, sort them, then find the middle
- Since our computation is distributed, no mapper sees all the values
- Should we send them all to one reducer?
 - Not utilizing map-reduce (computation not distributed)
 - Data sizes likely too large to keep in memory

Summarization

- Median
 - Keep the unique paragraph lengths, and
 - The frequency of each length
- Map output:
 - `<paragraph length, 1>`
- Combiner gets a list of these pairs and updates the count for recurring lengths
- Reducer does the same, then computes the median

Summarization

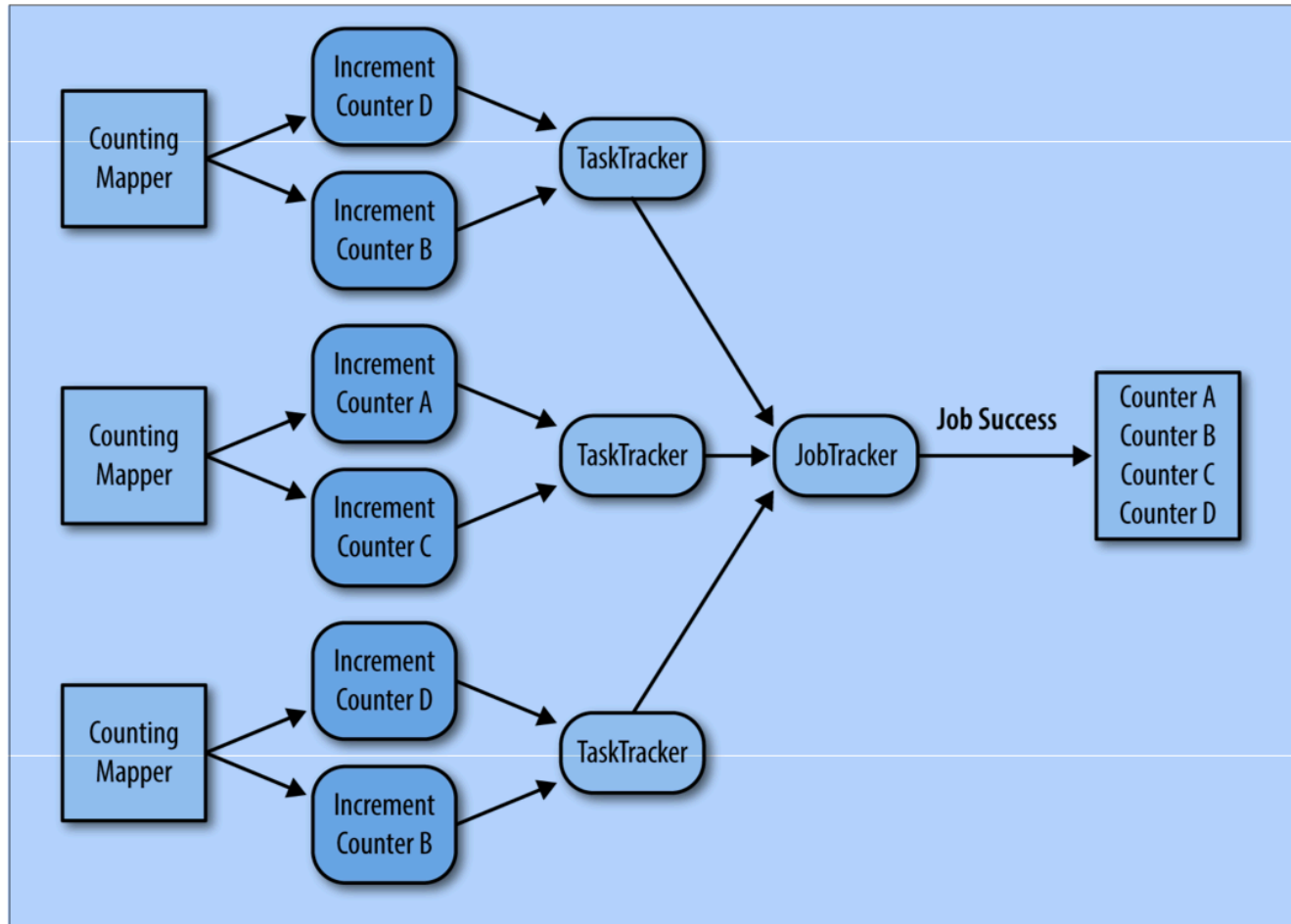
- Median
 - Hadoop provides the **SortedMapWritable** class
 - Can associate a frequency count with a paragraph length
 - Keeps the lengths in sorted order
- See the example in Chapter 2 of *Map-Reduce Design Patterns*
- How could we compute all in one pass over the data?
 - min, max, median

Counters

- Hadoop map-reduce infrastructure provides counters
 - Accessed by group name
 - Cannot have a large number of counters
 - For example, can't use counters to solve WordCount
 - A few tens of counters can be used
- Counters are stored in memory on **JobTracker**

Counters

Figure 2-6, MapReduce Design Patterns

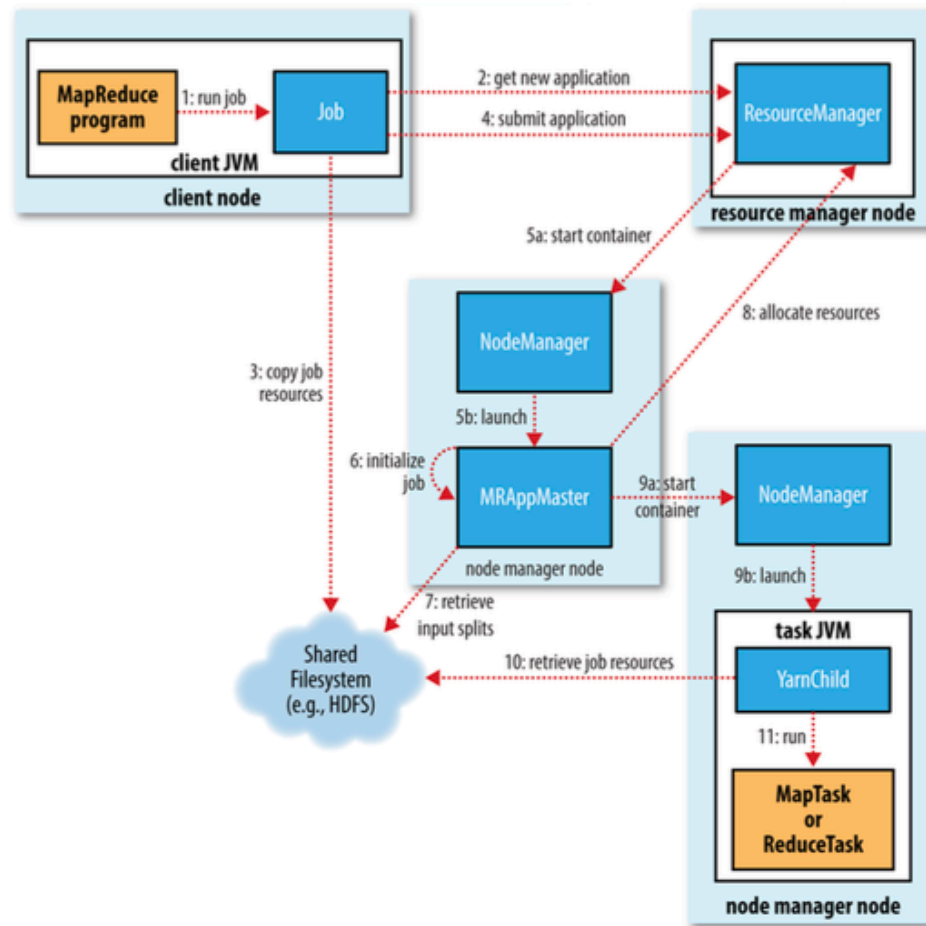


How Hadoop MapReduce Works

- We've seen some terms like:
 - Job, JobTracker, TaskTracker (MapReduce 1)
 - Job, ResourceManager, NodeManager (YARN, MapReduce 2)
- Let's look at what they do
- Details from Chapter 7, *Hadoop: The Definitive Guide 4th Edition*

How Hadoop MapReduce Works

Figure 7-1, Hadoop: The Definitive Guide 4th Edition



Job Submission

- Job submission
 - Input files exist? Can splits be computed?
 - Output directory exist?
 - If yes, it fails. Hadoop expects to create this directory
 - Copy resources to HDFS
 - JAR files
 - Configuration file
 - Computed file splits

Resource Manager

- Creates tasks (work to be done)
 - Map task for each input split
 - Requested number of reducer tasks
 - Job setup, job cleanup tasks
- Map tasks are assigned to task trackers that are “close” to the input split location
 - Data local preferred
 - Rack local next
- Reduce task can go anywhere. Why?
- Scheduling algorithm orders the tasks

Task Execution

- Configured for several map and reduce tasks
- Each task has status info (state, progress, counters)
- Periodically sends info to **MRAppMaster**
 - Running, successful completion, failed
 - Progress (% complete)
- For a new task
 - Copy files to local file system (JAR, configuration)
 - Launch a new JVM (**YarnChild** drives execution)
 - Load the mapper/reducer class and run the task

Task Progress

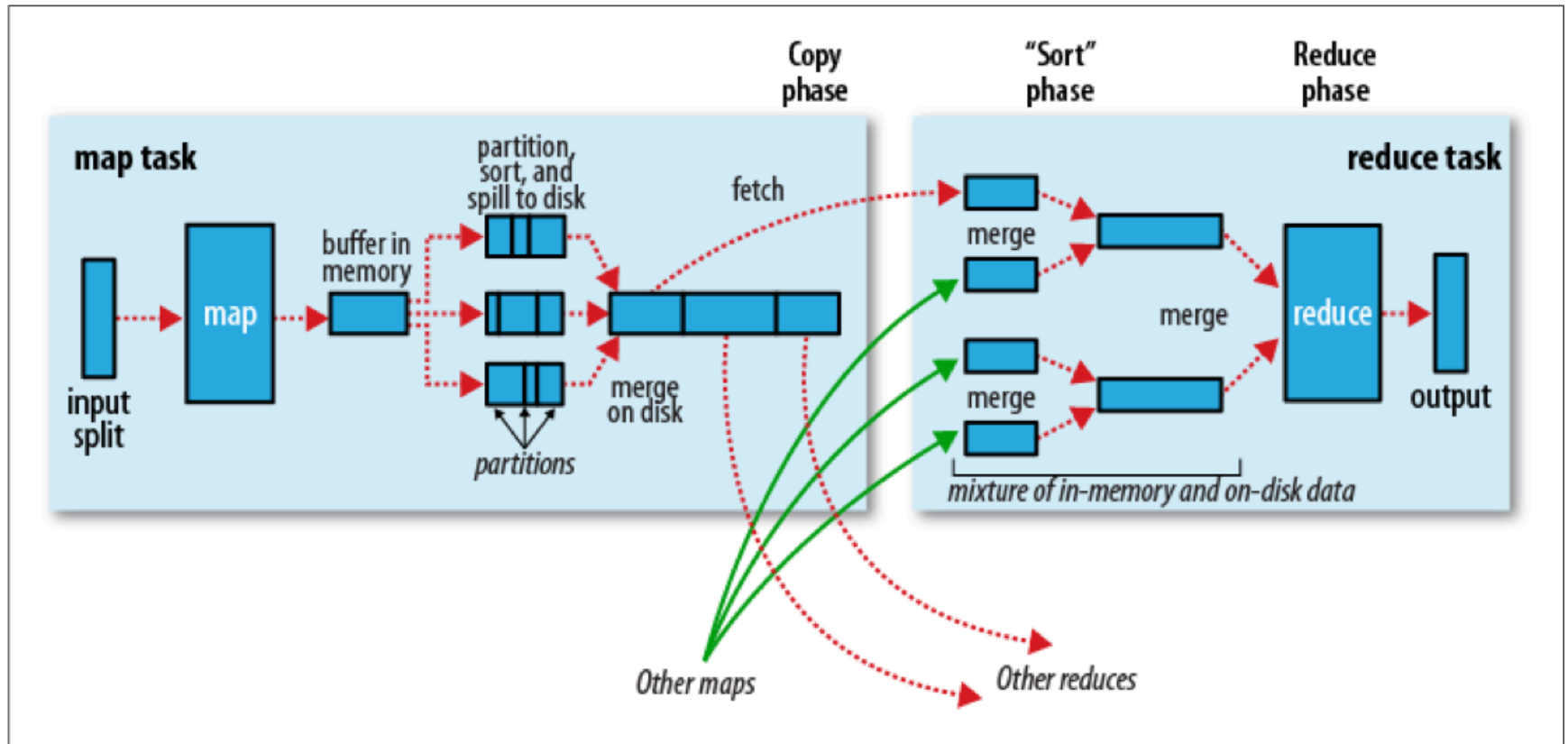
- Read in input pair (mapper or reducer)
- Write an output pair (mapper or reducer)
- Set the status description
- Increment a counter
- Reporting progress

Task Progress

- Mapper – straightforward
 - How much of the input has been processed
- Reducer – more complicated
 - Sort, shuffle and reduce are considered here
 - Progress is an estimate of how much of the total work has been done
 - One-third allocated to each

Shuffle

Figure 7-4, Hadoop: The Definitive Guide 4th Edition



MapReduce in Hadoop

Figure 2.4, Hadoop - The Definitive Guide

