CSCI 5408

Dr. Saurabh Dey

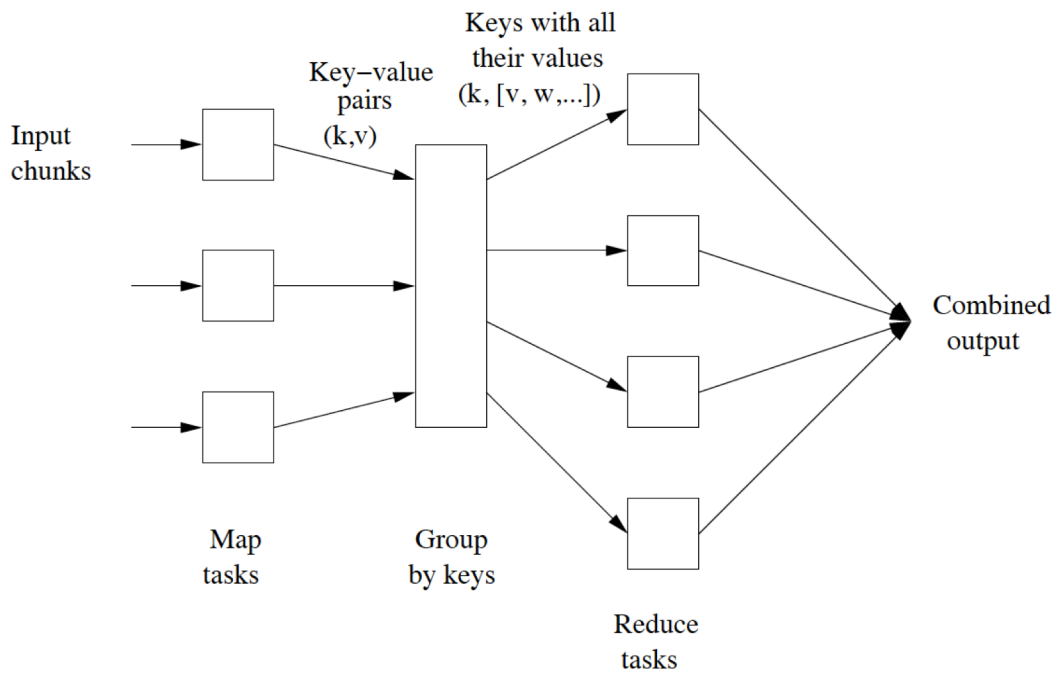
saurabh.dey@dal.ca

Outline

- MapReduce Programming Framework
- Hadoop Ecosystem
- YARN

MapReduce Programming Model

A MapReduce computation executes as follows:



Some number of Map tasks each are given one or more chunks from a distributed file system. These Map tasks turn the chunk into a sequence of key-value pairs. The way key-value pairs are produced from the input data is determined by the code written by the user for the Map function.

The key-value pairs from each Map task are collected by a master controller and sorted by key. The keys are divided among all the Reduce tasks, so all key-value pairs with the same key wind up at the same Reduce task.

The Reduce tasks work on one key at a time, and combine all the values associated with that key in some way. The manner of combination of values is determined by the code written by the user for the Reduce function.

J Leskovec, A Rajaraman, J Ullman, “Mining of Massive Datasets”, 2014

Source: https://hadoop.apache.org/

MapReduce Programming Framework

Input and Output types of a MapReduce job:

(input) `<k1, v1>` -> **map** -> `<k2, v2>` -> **combine** -> `<k2, v2>` -> **reduce** -> `<k3, v3>` (output)

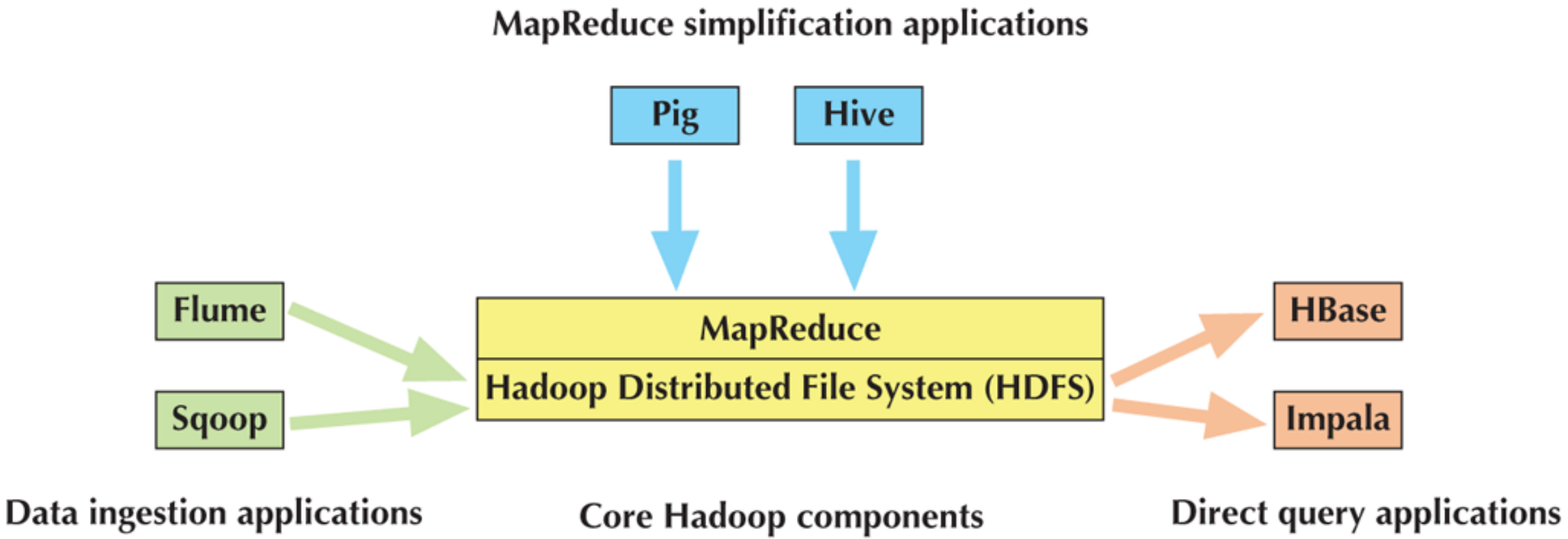
```
  public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws
IOException {
    String line = value.toString();
    StringTokenizer tokenizer = new StringTokenizer(line);
    while (tokenizer.hasMoreTokens()) {
     word.set(tokenizer.nextToken());
     output.collect(word, one);
    }
   }
  }
```

Mapper

Reducer

```
.   public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter)
throws IOException {
.    int sum = 0;
.    while (values.hasNext()) {
.     sum += values.next().get();
.    }
.    output.collect(key, new IntWritable(sum));
.   }
```

Hadoop Ecosystem



- Hive is a data warehousing system
- Works with non-relational data
- HiveQL is used.
- Batch processing is done

Pig

- Pig is a tool for compiling a high-level scripting language, named Pig Latin, into MapReduce jobs for executing in Hadoop.
- Pig Latin is scripting, therefore, procedural

- Flume is a component for ingesting data into Hadoop.
- Designed primarily for harvesting large sets of data from server log files, like clickstream data from web server logs.
- Possibility exists of performing some transformations on the data as it is being harvested.

- Sqoop is a tool for converting data back and forth between a relational database and the HDFS.
- Sqoop is similar to Flume – both bring data to HDFS

- HBase is a column-oriented NoSQL database.
- Highly distributed and designed to scale out easily. It does not support SQL or SQL-like languages, relying instead on lower-level languages such as Java for interaction.
- Avoids the delays caused by batch processing, making it more suitable for fast processing involving smaller subsets of the data
- Used by Facebook for its messaging system.

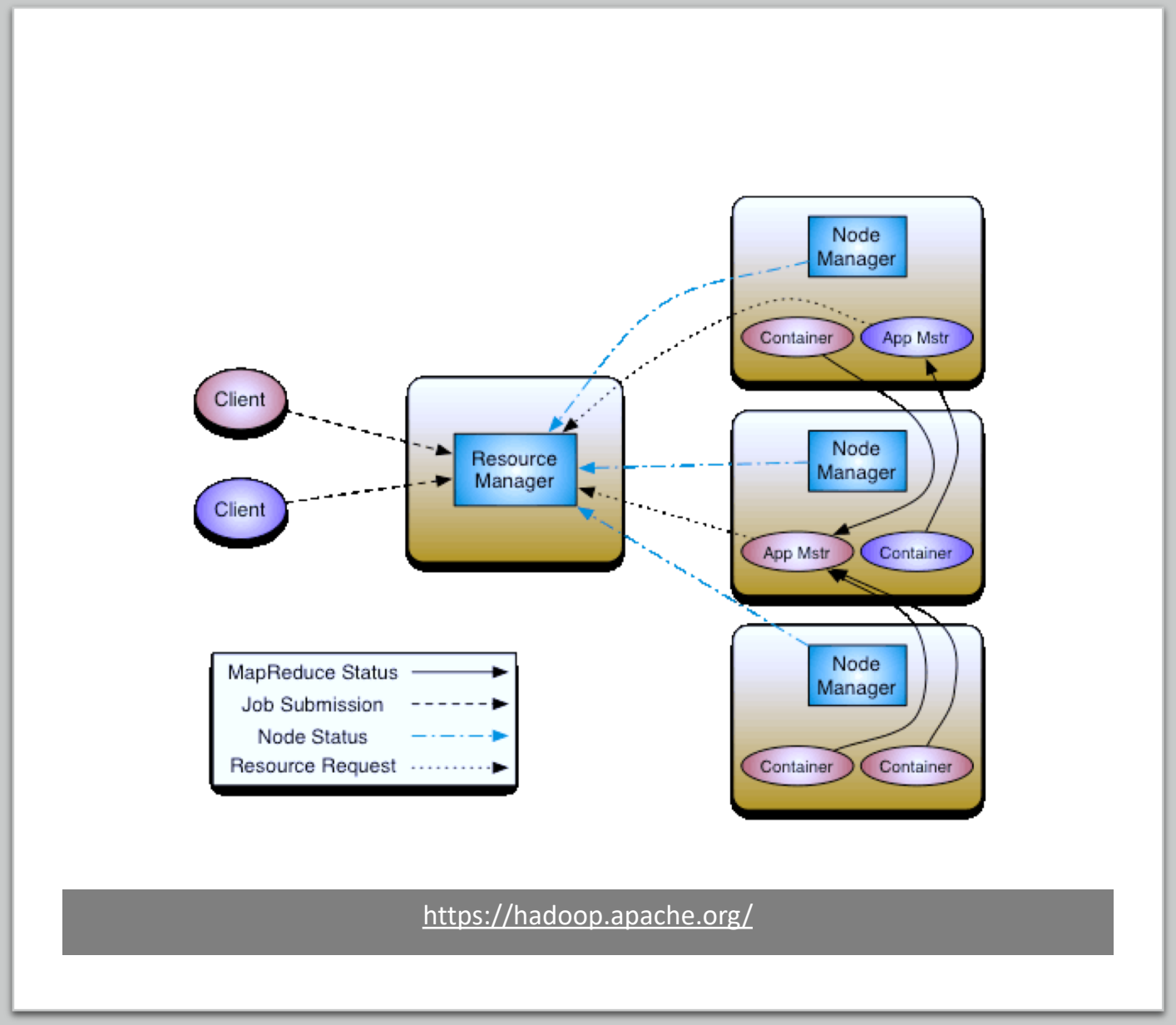
Impala

- With Impala, analysts can write SQL queries directly against the data while it is still in HDFS.

- Impala makes heavy use of in-memory caching on data nodes. It is generally considered an appropriate tool for processing large amounts of data into a relatively small result set.

YARN

- The Hadoop version 2 is known as YARN (Yet Another Resource Negotiator)
- The central idea of YARN is separation of cluster resource management from jobs management. This is suitable for multitenancy
- The **ResourceManager** and the per worker node **NodeManager** together form the platform on which any Application can be hosted on YARN.



https://hadoop.apache.org/

Questions to Consider

- What is the primary difference between Hadoop v1 and YARN?
- Can we use MapReduce other than count operation?
- What is the advantage of using a column oriented database?