Hadoop Map-Reduce

2017.2　　XenRon

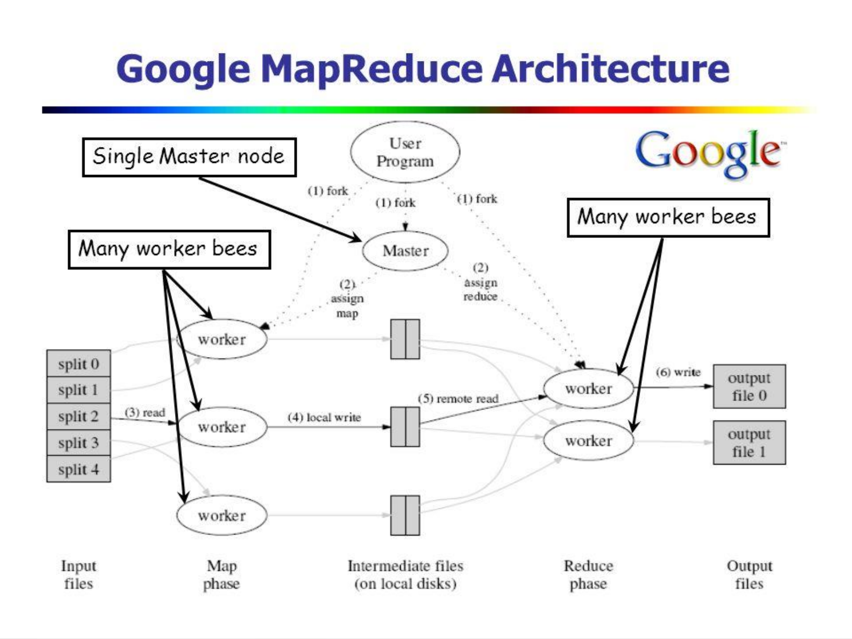
CONTENTS

Computation Modal 01

Map-Reduce 02

Word Count 03

Use Case 04

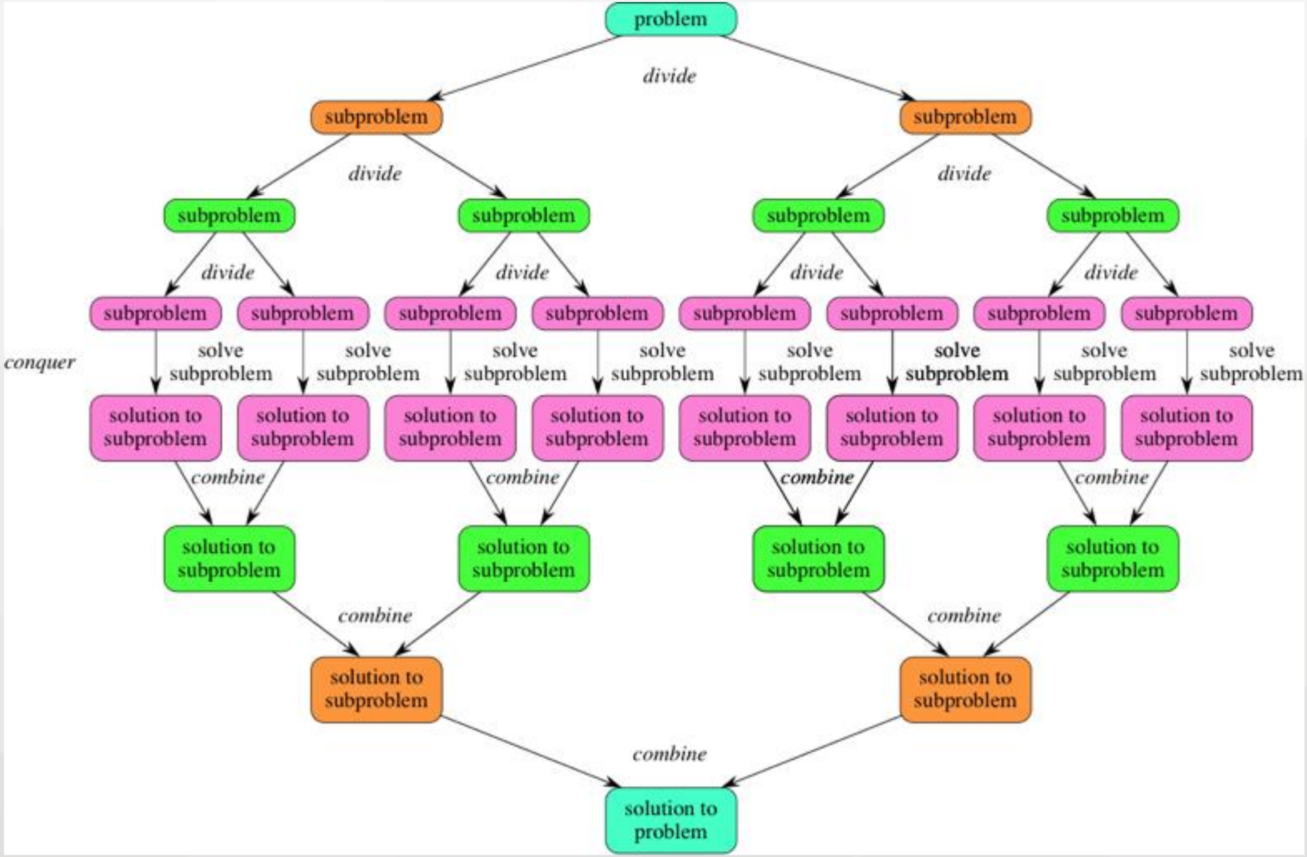


Google MapReduce Architecture

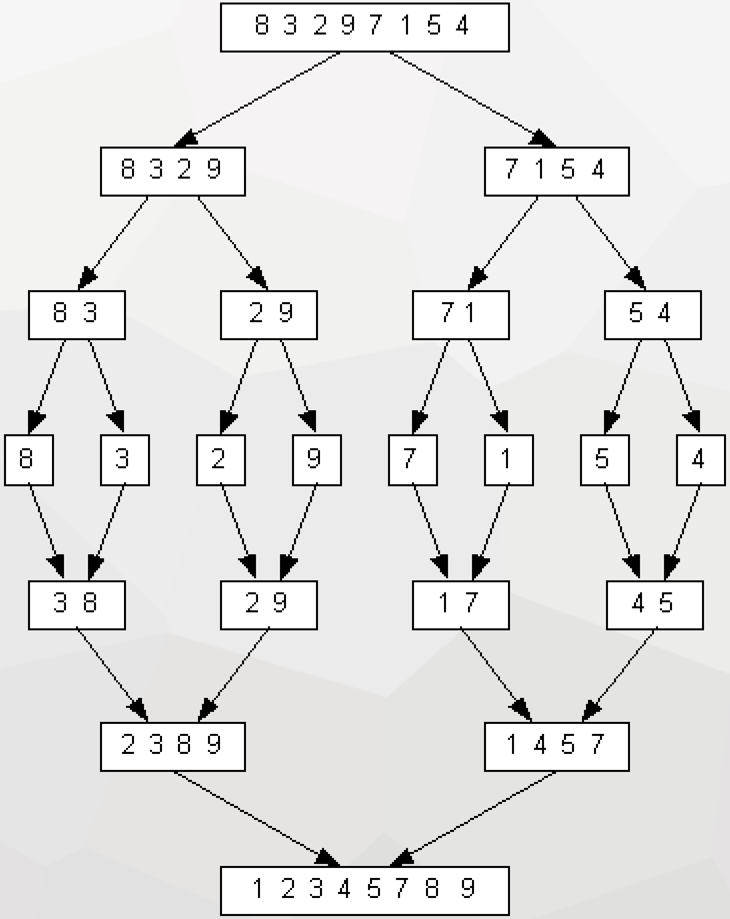
PART1

Computation Modal

Divide & Conquer



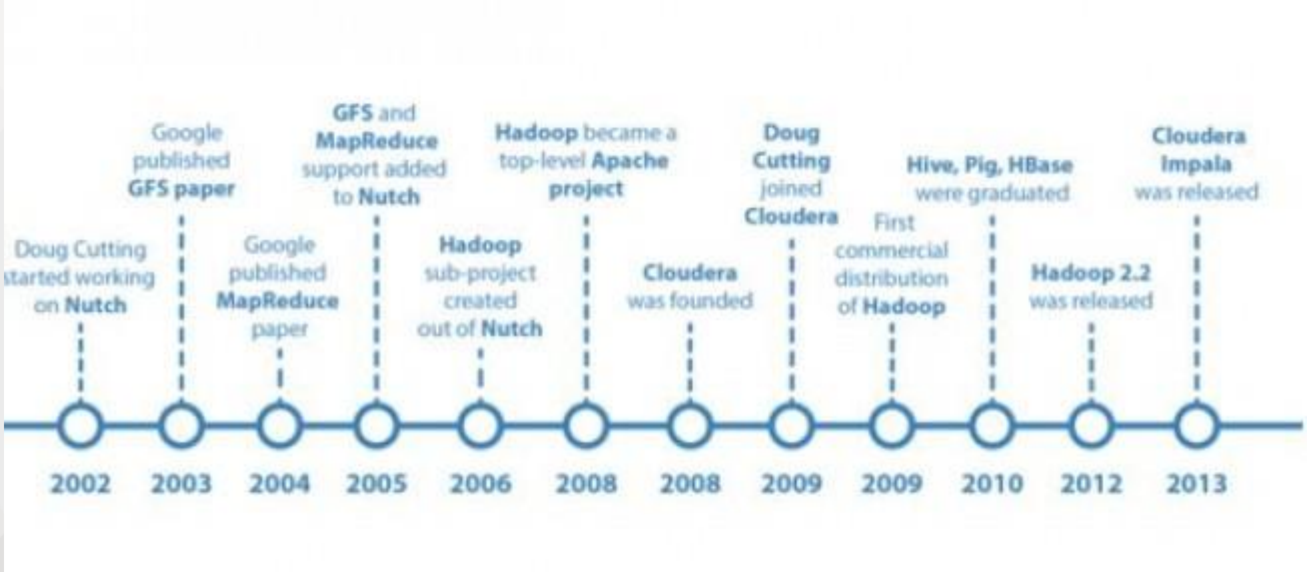
Divide & Conquer



PART2

Map-Reduce

Google



2002 — Doug Cutting started working on **Nutch**

2003 — Google published **GFS paper**

2004 — Google published **MapReduce** paper

2005 — **GFS** and **MapReduce** support added to **Nutch**

2006 — **Hadoop** sub-project created out of **Nutch**

2008 — **Hadoop** became a top-level **Apache project**

2008 — **Cloudera** was founded

2009 — **Doug Cutting** joined **Cloudera**

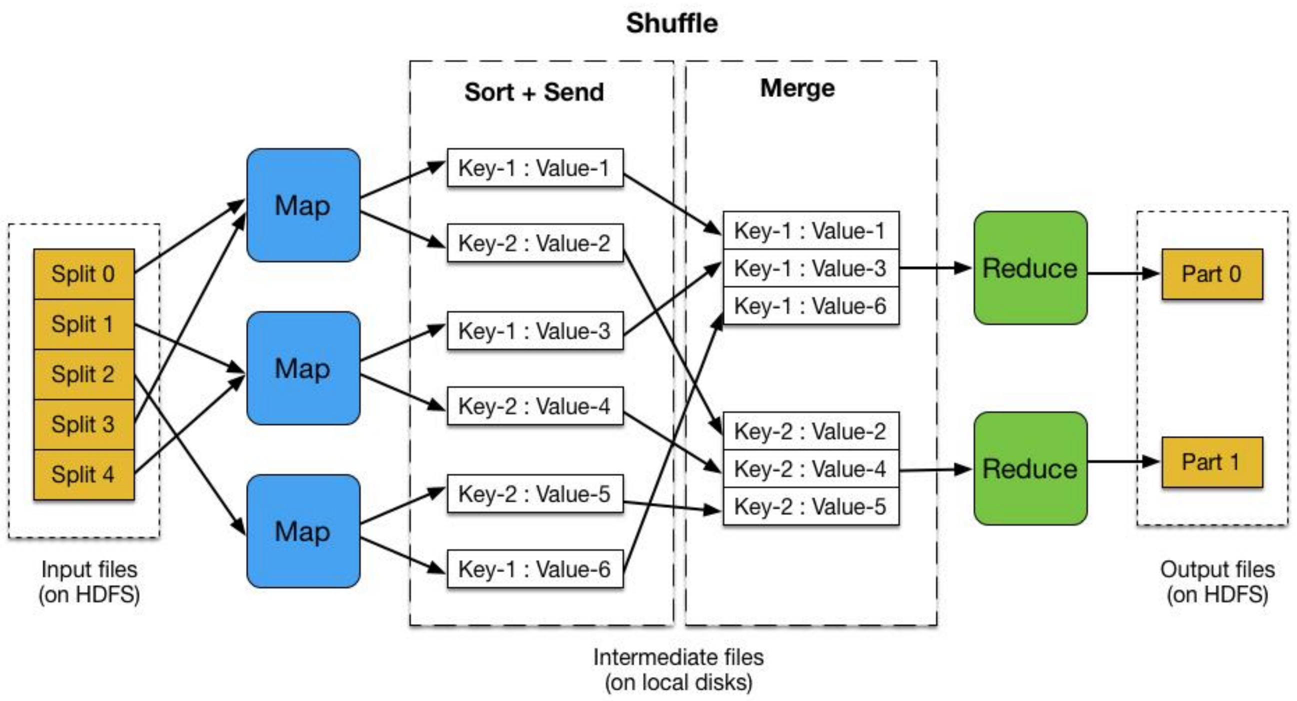
2009 — First commercial distribution of **Hadoop**

2010 — **Hive, Pig, HBase** were graduated

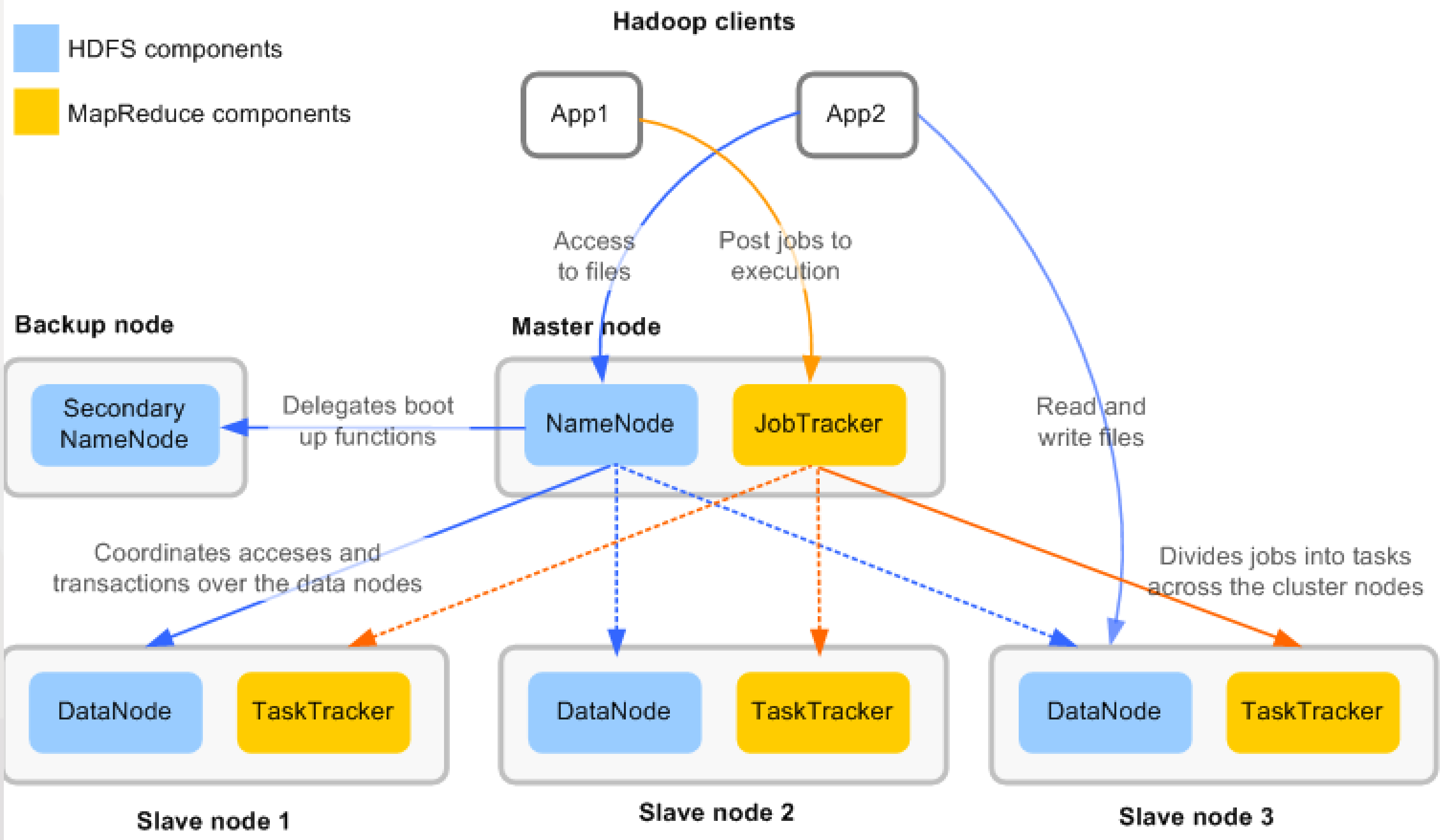
2012 — **Hadoop 2.2** was released

2013 — **Cloudera Impala** was released

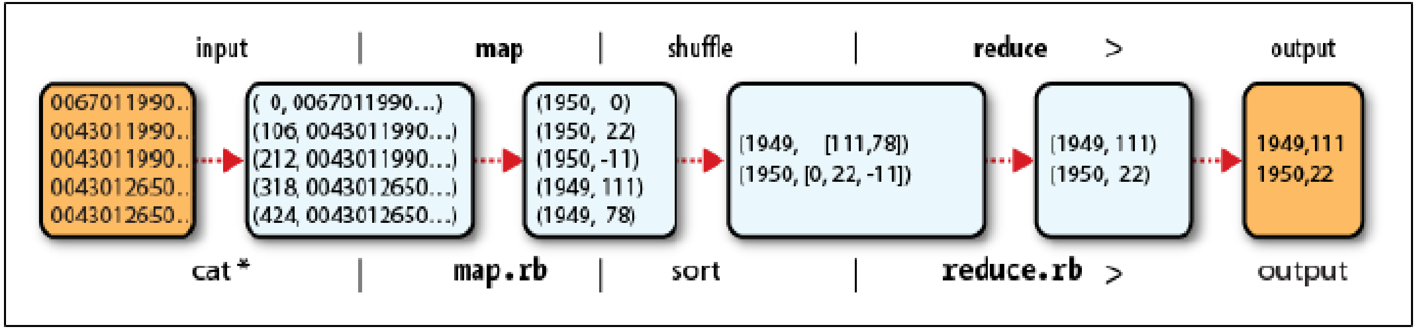
Map Reduce


Shuffle
Sort + Send
Merge
Split 0
Split 1
Split 2
Split 3
Split 4
Input files
(on HDFS)
Map
Map
Map
Key-1 : Value-1
Key-2 : Value-2
Key-1 : Value-3
Key-2 : Value-4
Key-2 : Value-5
Key-1 : Value-6
Key-1 : Value-1
Key-1 : Value-3
Key-1 : Value-6
Key-2 : Value-2
Key-2 : Value-4
Key-2 : Value-5
Reduce
Reduce
Part 0
Part 1
Output files
(on HDFS)
Intermediate files
(on local disks)

Map Reduce


Hadoop clients
HDFS components
MapReduce components
App1
App2
Access to files
Post jobs to execution
Backup node
Master node
Secondary NameNode
Delegates boot up functions
NameNode
JobTracker
Read and write files
Coordinates acceses and transactions over the data nodes
Divides jobs into tasks across the cluster nodes
DataNode
TaskTracker
DataNode
TaskTracker
DataNode
TaskTracker
Slave node 1
Slave node 2
Slave node 3

Map Reduce



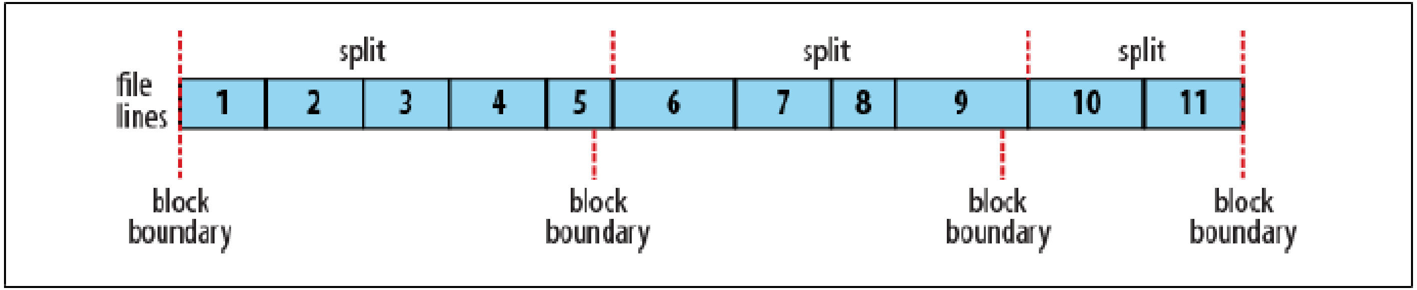
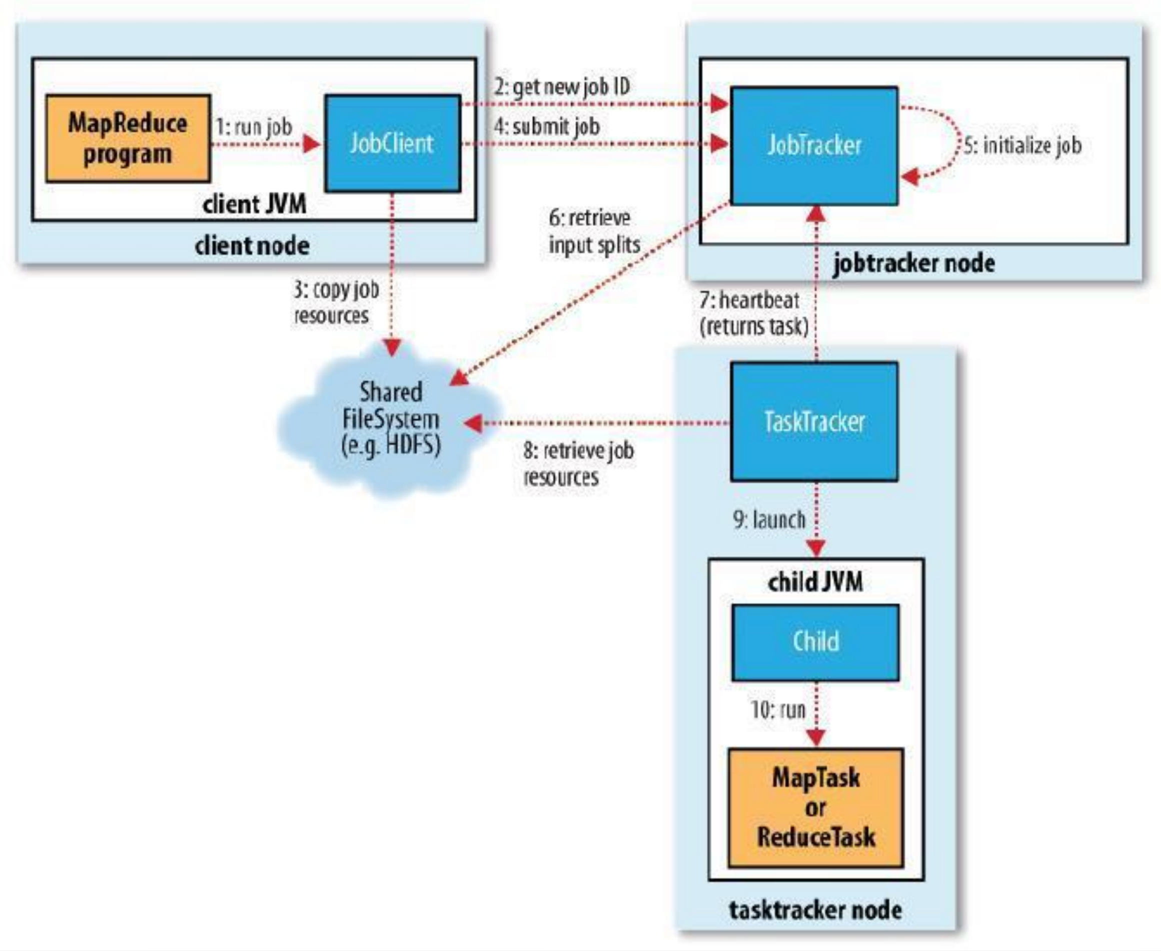
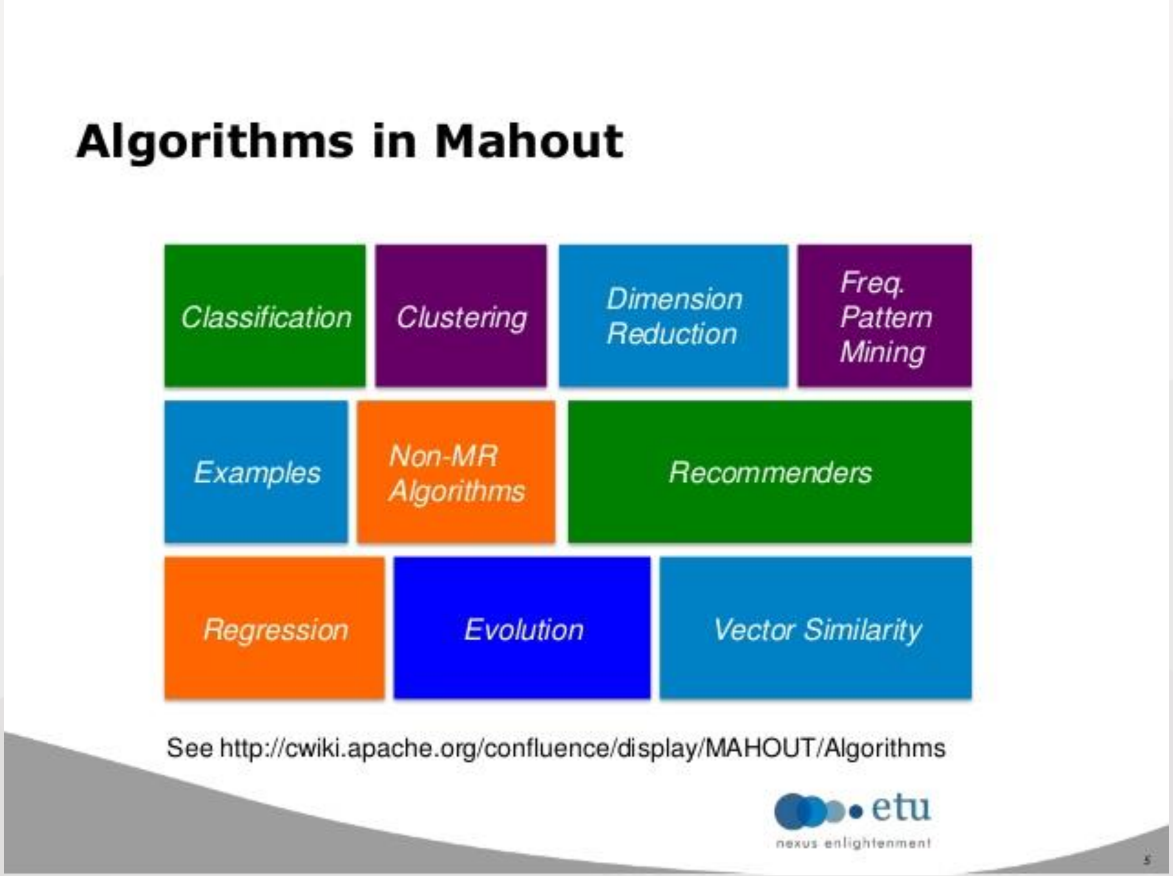


Figure 7-3. Logical records and HDFS blocks for TextInputFormat

Map Reduce


MapReduce program
1: run job
JobClient
client JVM
client node
2: get new job ID
4: submit job
JobTracker
5: initialize job
jobtracker node
6: retrieve input splits
3: copy job resources
Shared FileSystem (e.g. HDFS)
7: heartbeat (returns task)
TaskTracker
8: retrieve job resources
9: launch
child JVM
Child
10: run
MapTask or ReduceTask
tasktracker node

Map Reduce



Algorithms in Mahout

Classification

Clustering

Dimension Reduction

Freq. Pattern Mining

Examples

Non-MR Algorithms

Recommenders

Regression

Evolution

Vector Similarity

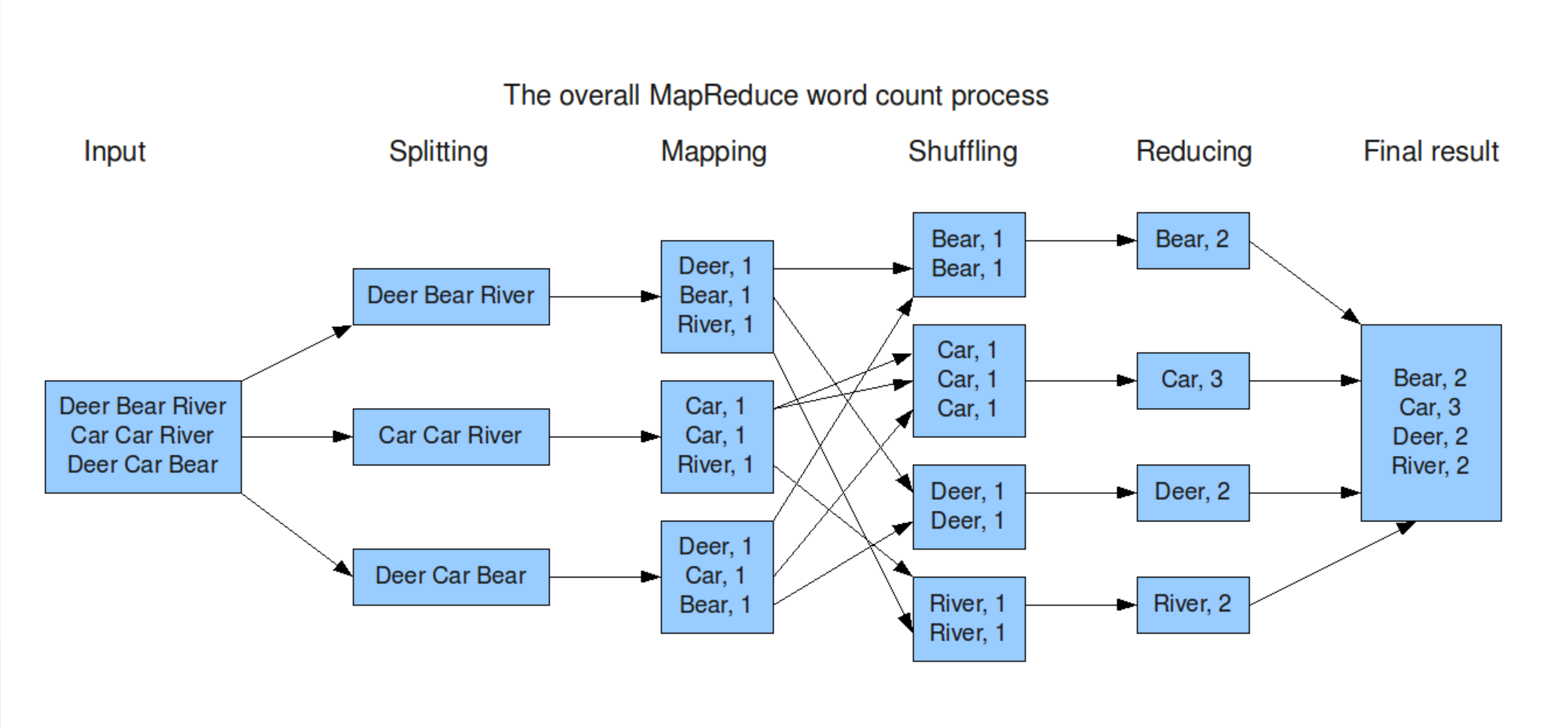
See http://cwiki.apache.org/confluence/display/MAHOUT/Algorithms

etu
nexus enlightenment
5

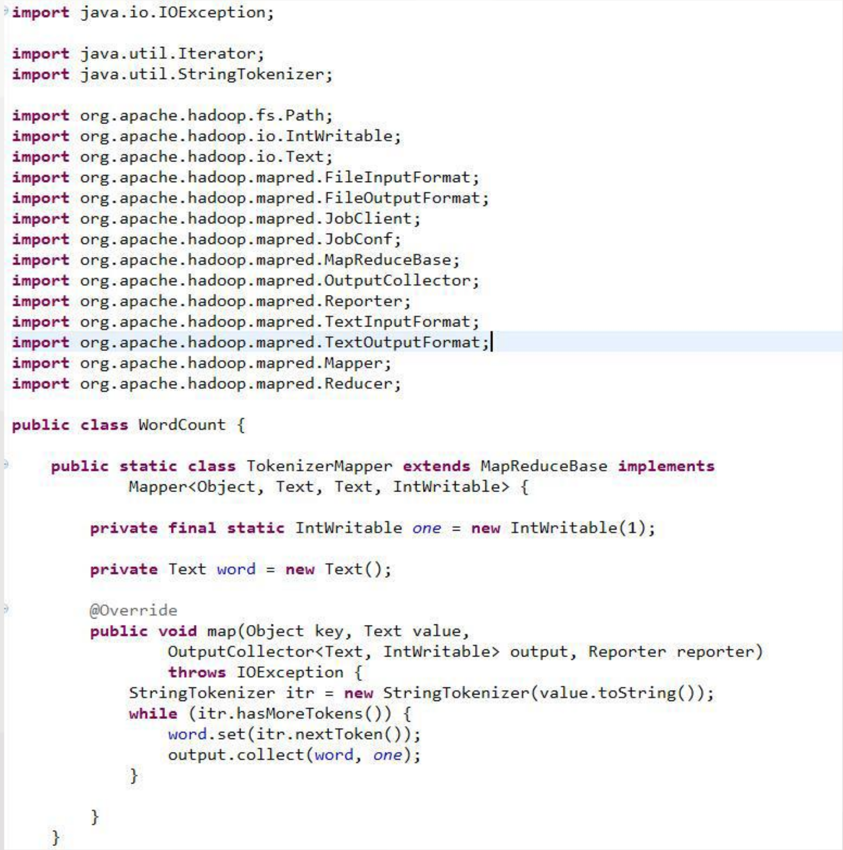
PART3

Word Count

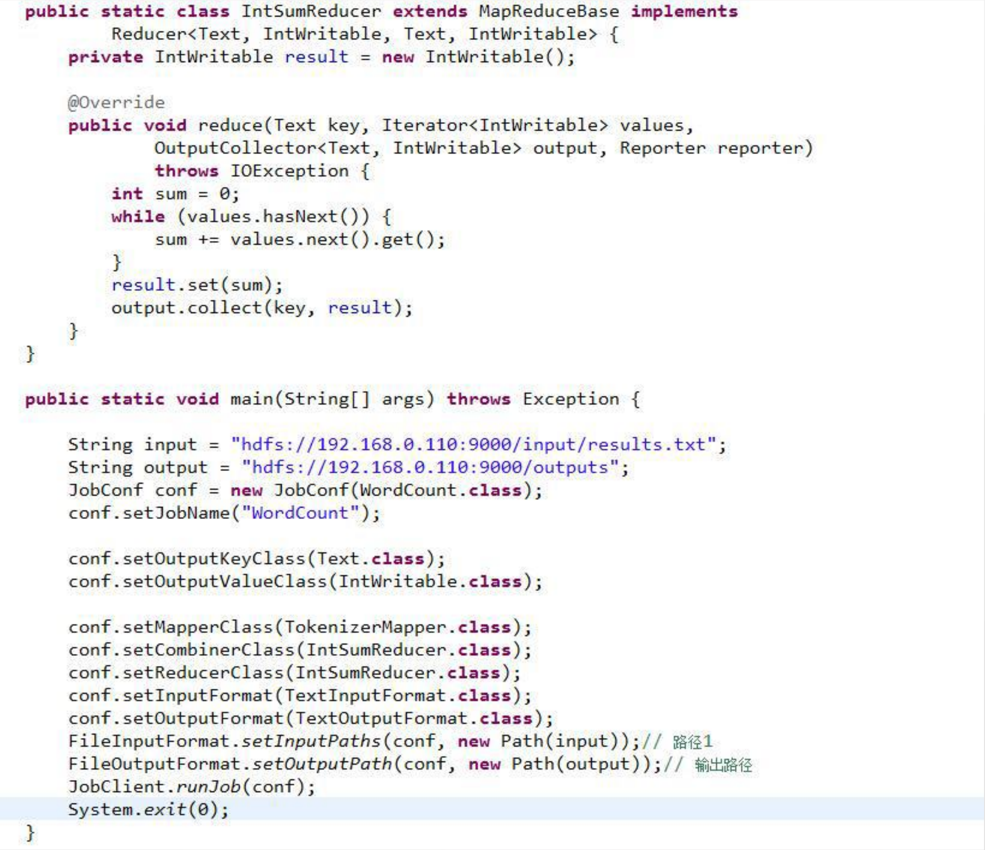
Word Count


The overall MapReduce word count process
Input
Splitting
Mapping
Shuffling
Reducing
Final result
Deer Bear River
Car Car River
Deer Car Bear
Deer Bear River
Car Car River
Deer Car Bear
Deer, 1
Bear, 1
River, 1
Car, 1
Car, 1
River, 1
Deer, 1
Car, 1
Bear, 1
Bear, 1
Bear, 1
Car, 1
Car, 1
Car, 1
Deer, 1
Deer, 1
River, 1
River, 1
Bear, 2
Car, 3
Deer, 2
River, 2
Bear, 2
Car, 3
Deer, 2
River, 2

Word Count



```java
import java.io.IOException;

import java.util.Iterator;
import java.util.StringTokenizer;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.Reducer;

public class WordCount {

    public static class TokenizerMapper extends MapReduceBase implements
            Mapper<Object, Text, Text, IntWritable> {

        private final static IntWritable one = new IntWritable(1);

        private Text word = new Text();

        @Override
        public void map(Object key, Text value,
                OutputCollector<Text, IntWritable> output, Reporter reporter)
                throws IOException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                output.collect(word, one);
            }

        }
    }
```



```java
public static class IntSumReducer extends MapReduceBase implements
        Reducer<Text, IntWritable, Text, IntWritable> {
    private IntWritable result = new IntWritable();

    @Override
    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();
        }
        result.set(sum);
        output.collect(key, result);
    }
}

public static void main(String[] args) throws Exception {

    String input = "hdfs://192.168.0.110:9000/input/results.txt";
    String output = "hdfs://192.168.0.110:9000/outputs";
    JobConf conf = new JobConf(WordCount.class);
    conf.setJobName("WordCount");

    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);

    conf.setMapperClass(TokenizerMapper.class);
    conf.setCombinerClass(IntSumReducer.class);
    conf.setReducerClass(IntSumReducer.class);
    conf.setInputFormat(TextInputFormat.class);
    conf.setOutputFormat(TextOutputFormat.class);
    FileInputFormat.setInputPaths(conf, new Path(input));// 路径1
    FileOutputFormat.setOutputPath(conf, new Path(output));// 输出路径
    JobClient.runJob(conf);
    System.exit(0);
}
```

MapReduce API

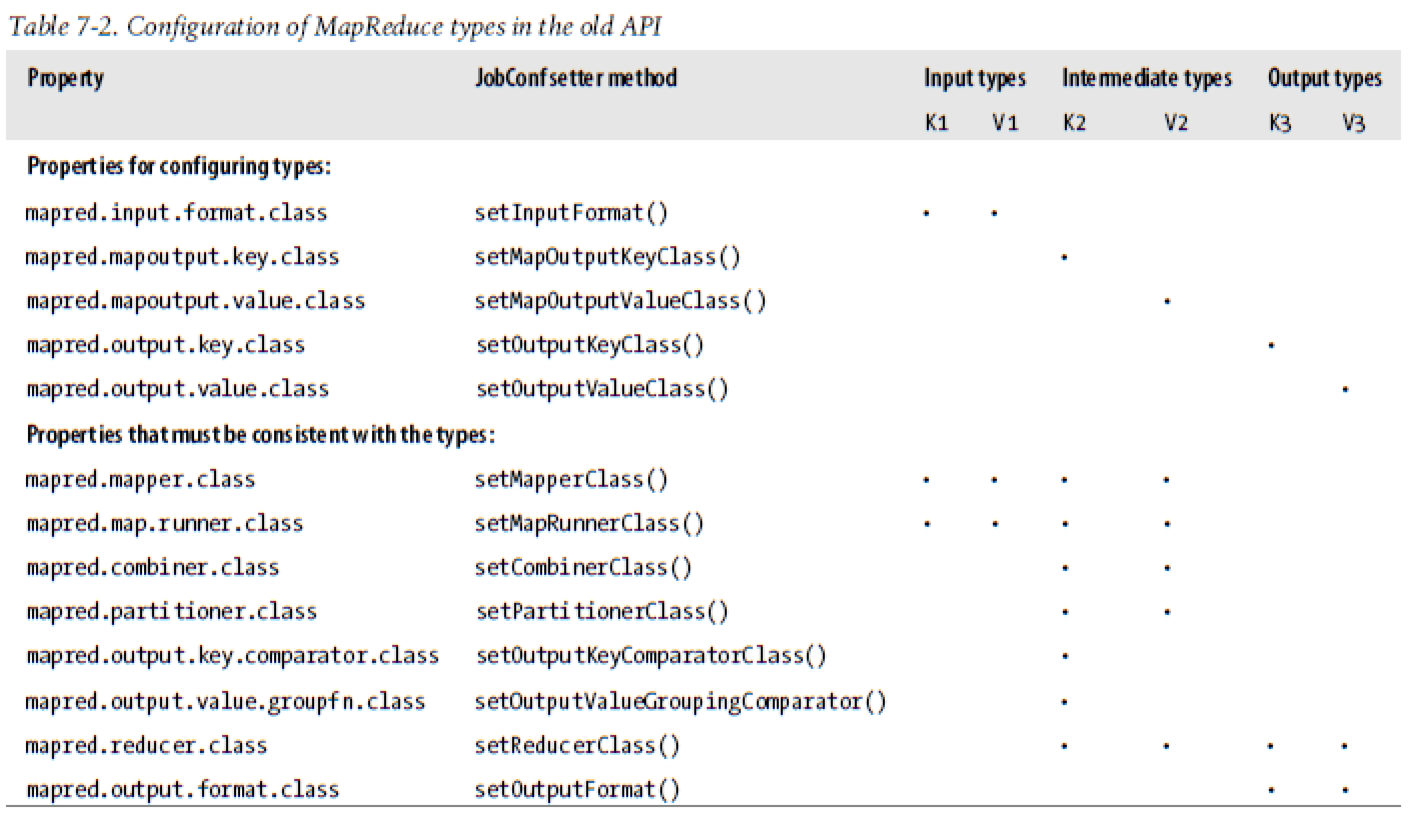


Table 7-2. Configuration of MapReduce types in the old API

Property	JobConf setter method	Input types		Intermediate types		Output types	
		K1	V1	K2	V2	K3	V3
Properties for configuring types:							
`mapred.input.format.class`	`setInputFormat()`	•	•				
`mapred.mapoutput.key.class`	`setMapOutputKeyClass()`			•			
`mapred.mapoutput.value.class`	`setMapOutputValueClass()`				•		
`mapred.output.key.class`	`setOutputKeyClass()`					•	
`mapred.output.value.class`	`setOutputValueClass()`						•
Properties that must be consistent with the types:							
`mapred.mapper.class`	`setMapperClass()`	•	•	•	•		
`mapred.map.runner.class`	`setMapRunnerClass()`	•	•	•	•		
`mapred.combiner.class`	`setCombinerClass()`			•	•		
`mapred.partitioner.class`	`setPartitionerClass()`			•	•		
`mapred.output.key.comparator.class`	`setOutputKeyComparatorClass()`			•			
`mapred.output.value.groupfn.class`	`setOutputValueGroupingComparator()`			•			
`mapred.reducer.class`	`setReducerClass()`			•	•	•	•
`mapred.output.format.class`	`setOutputFormat()`					•	•

MapReduce API

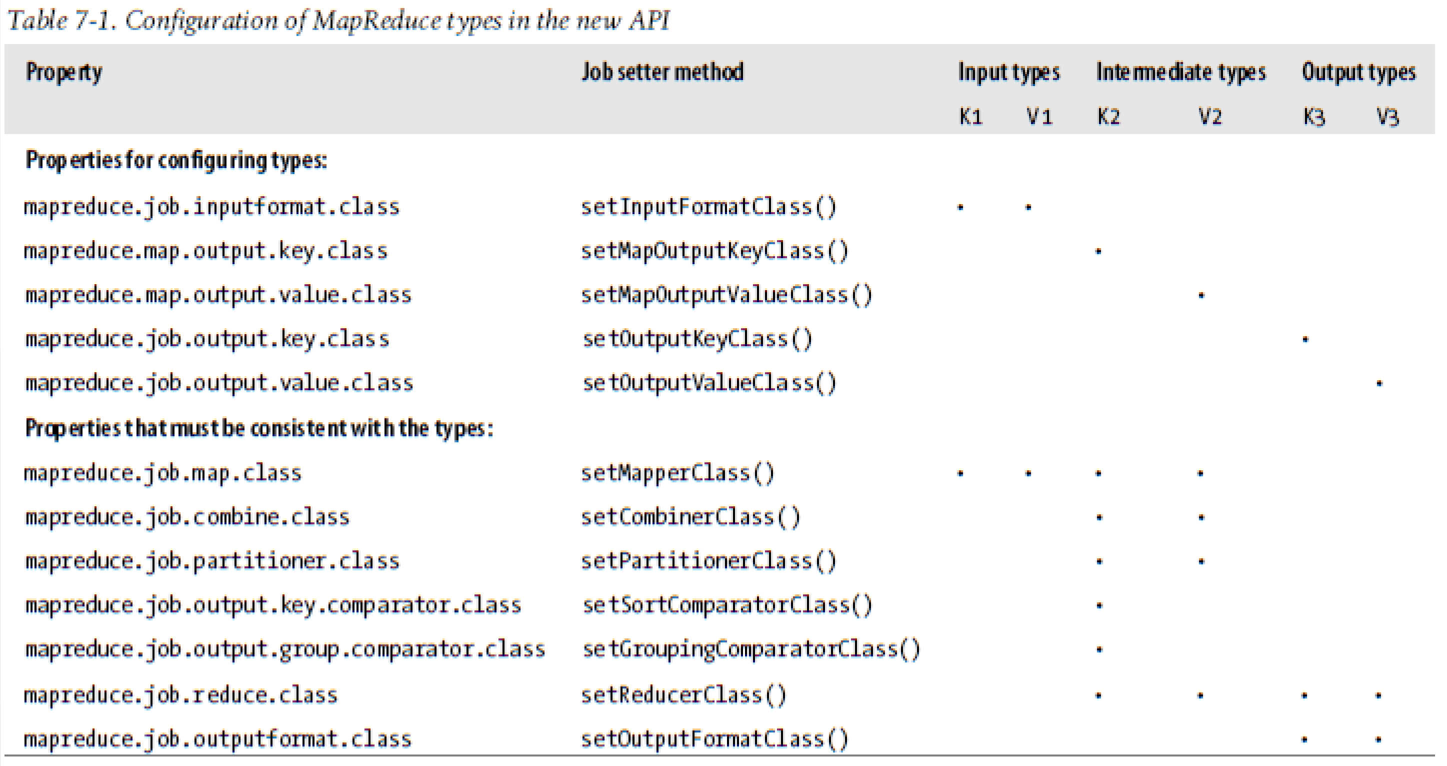


Table 7-1. Configuration of MapReduce types in the new API

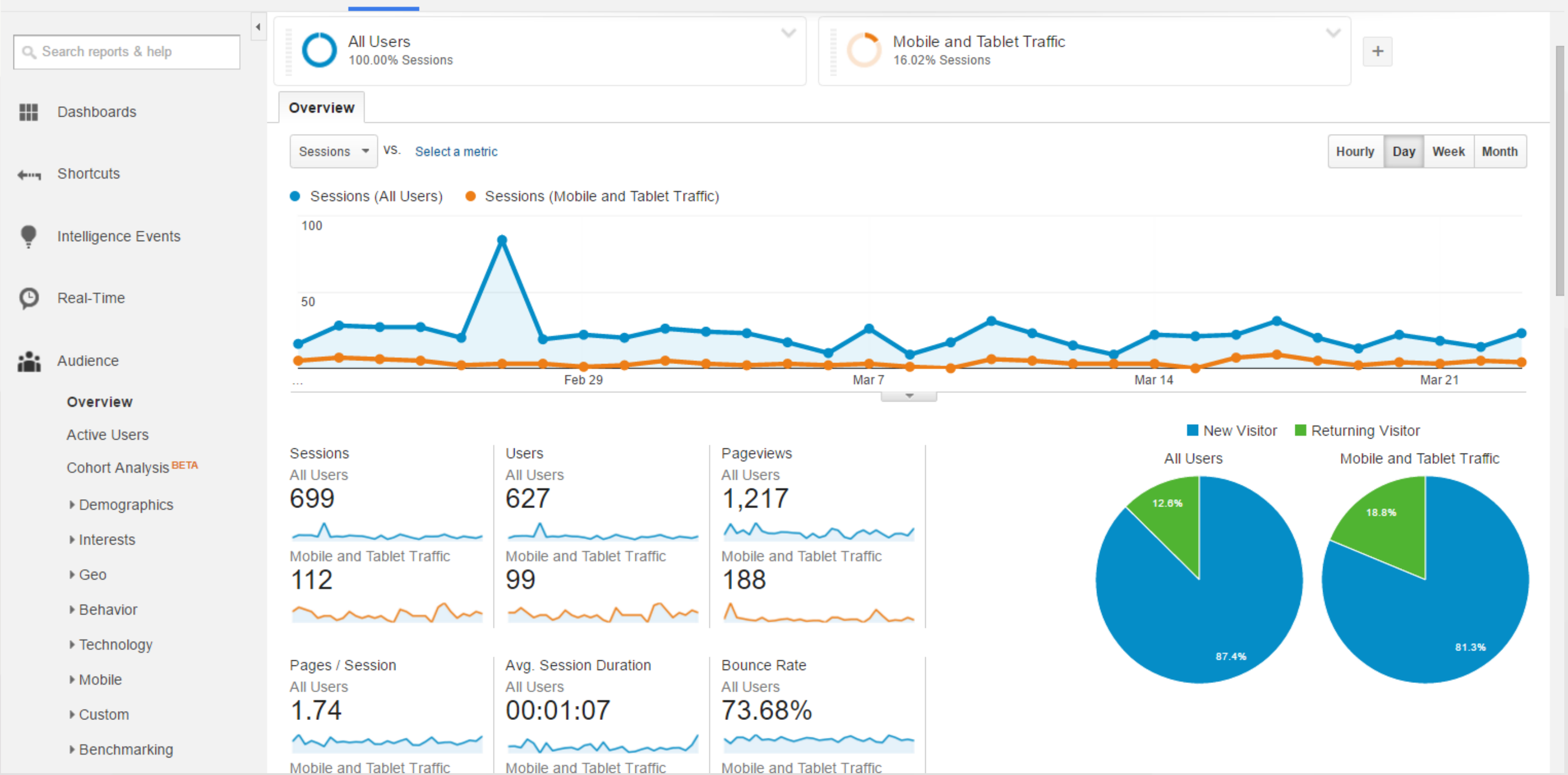
Property	Job setter method	Input types		Intermediate types		Output types	
		K1	V1	K2	V2	K3	V3
Properties for configuring types:							
`mapreduce.job.inputformat.class`	`setInputFormatClass()`	•	•				
`mapreduce.map.output.key.class`	`setMapOutputKeyClass()`			•			
`mapreduce.map.output.value.class`	`setMapOutputValueClass()`				•		
`mapreduce.job.output.key.class`	`setOutputKeyClass()`					•	
`mapreduce.job.output.value.class`	`setOutputValueClass()`						•
Properties that must be consistent with the types:							
`mapreduce.job.map.class`	`setMapperClass()`	•	•	•	•		
`mapreduce.job.combine.class`	`setCombinerClass()`			•	•		
`mapreduce.job.partitioner.class`	`setPartitionerClass()`			•	•		
`mapreduce.job.output.key.comparator.class`	`setSortComparatorClass()`			•			
`mapreduce.job.output.group.comparator.class`	`setGroupingComparatorClass()`			•			
`mapreduce.job.reduce.class`	`setReducerClass()`			•	•	•	•
`mapreduce.job.outputformat.class`	`setOutputFormatClass()`					•	•

PART4

Use Case

KPI

KPI



All Users
100.00% Sessions

Mobile and Tablet Traffic
16.02% Sessions

Sessions
All Users
699
Mobile and Tablet Traffic
112

Users
All Users
627
Mobile and Tablet Traffic
99

Pageviews
All Users
1,217
Mobile and Tablet Traffic
188

Pages / Session
All Users
1.74
Mobile and Tablet Traffic

Avg. Session Duration
All Users
00:01:07
Mobile and Tablet Traffic

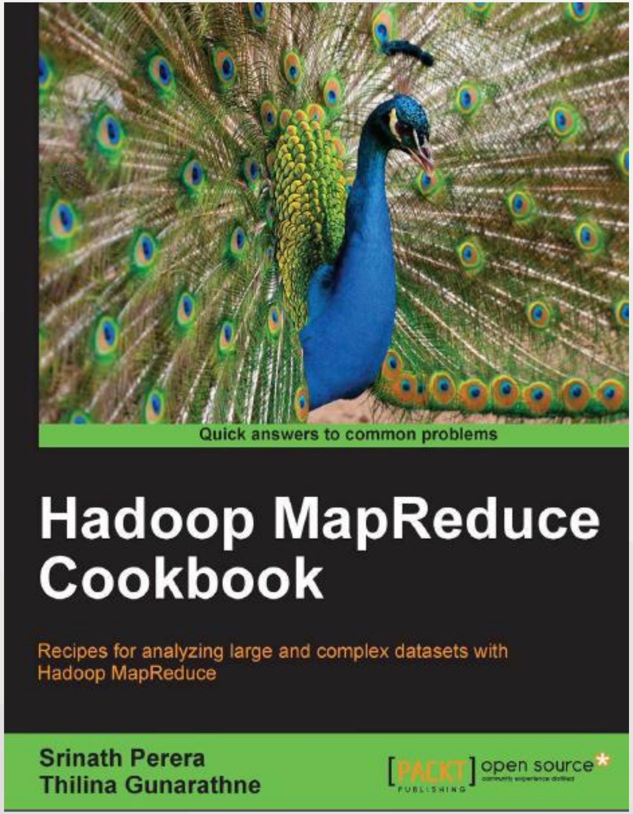
Bounce Rate
All Users
73.68%
Mobile and Tablet Traffic

LBS

PART5

Reference Books

Books



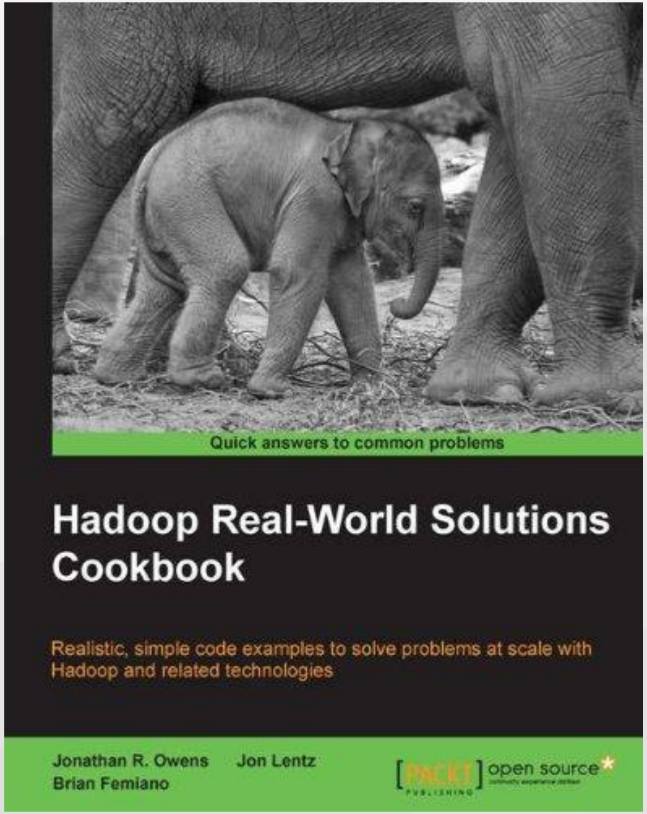
Quick answers to common problems

Hadoop MapReduce Cookbook

Recipes for analyzing large and complex datasets with Hadoop MapReduce

Srinath Perera
Thilina Gunarathne

[PACKT] open source



Quick answers to common problems

Hadoop Real-World Solutions Cookbook

Realistic, simple code examples to solve problems at scale with Hadoop and related technologies

Jonathan R. Owens
Jon Lentz
Brian Femiano

[PACKT] open source

The End