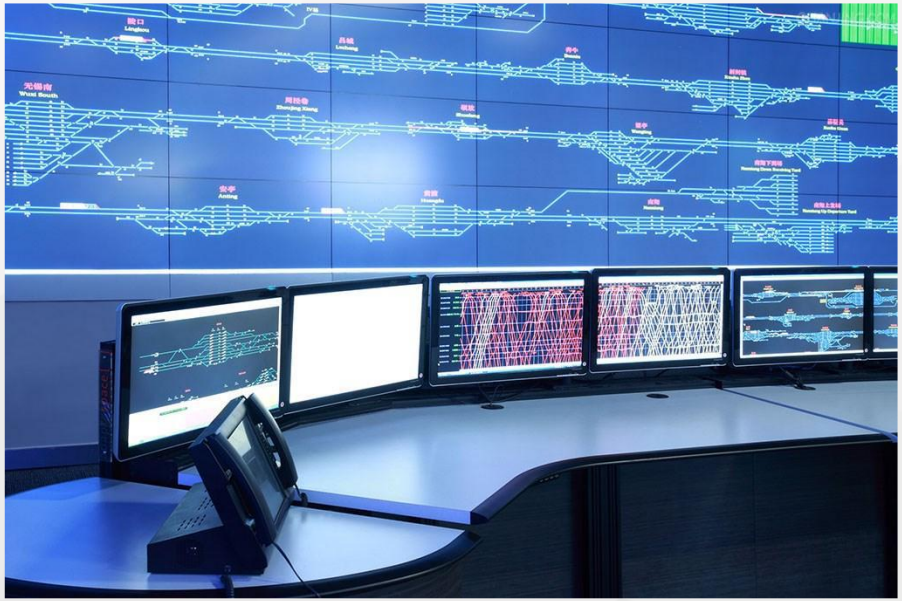
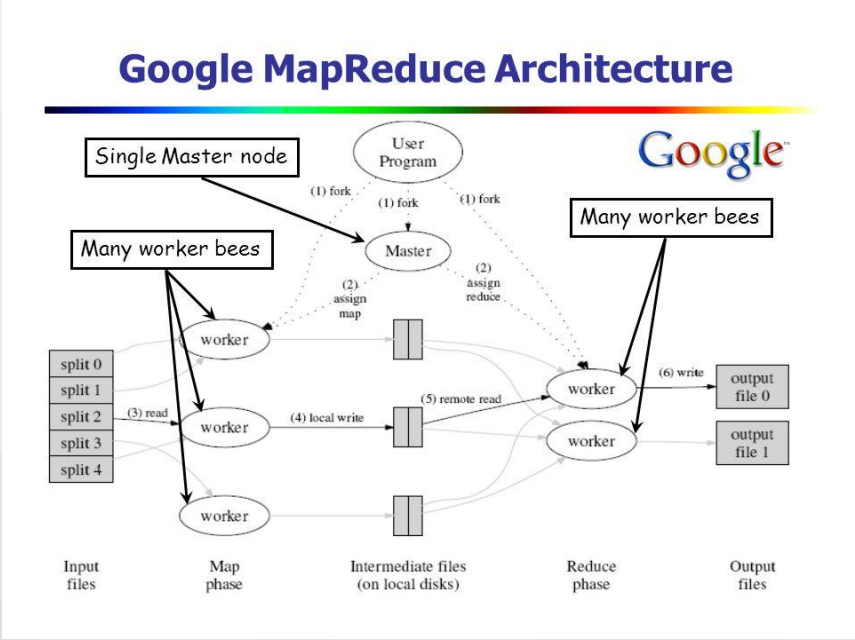


Hadoop Map-Reduce

2017.5 XenRon

CONTENTS



Review

Review

VirtualBox

ubuntu

docker

Review

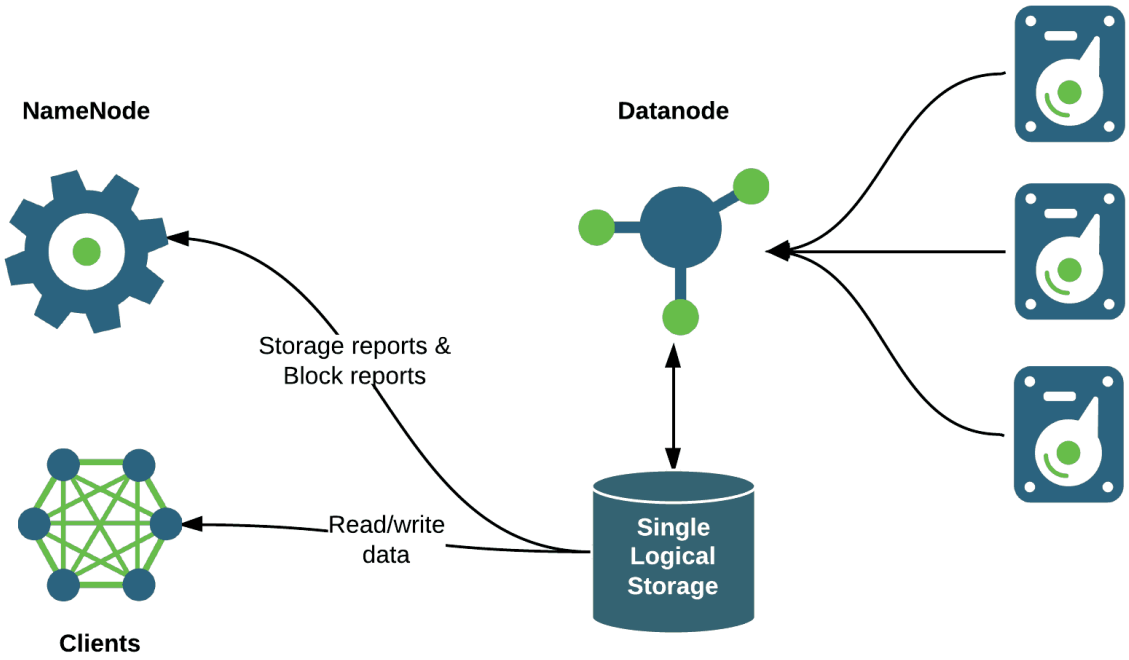
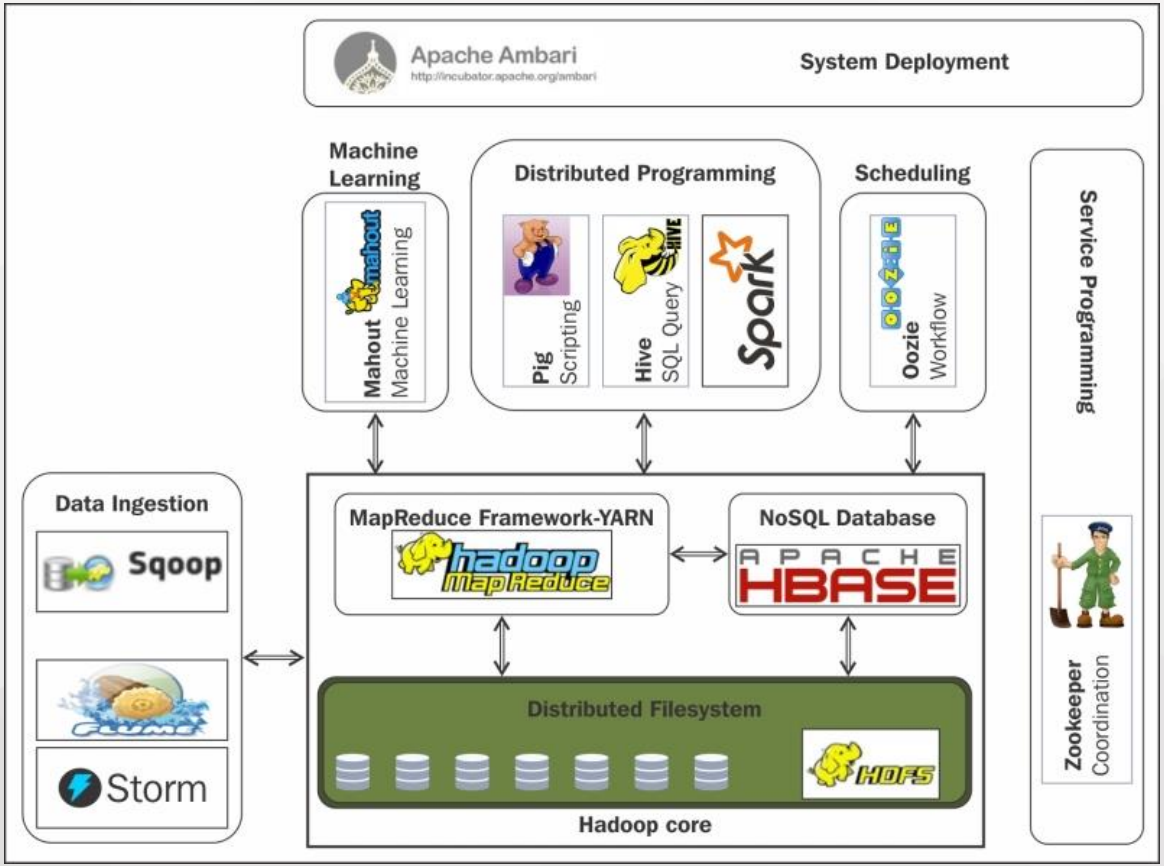


Figure 1: A DataNode presented itself as a single logical storage

Hadoop Eco System



PART0

Preliminary Topics
事前準備

Java

Java SE Public Updates

Major Release	GA Date	End of Public Updates Notification	End of Public Updates
5.0	May 2004	Apr 2008	Oct 2009
6	Dec 2006	Feb 2011	Feb 2013
7	Jul 2011	Mar 2014	Apr 2015
8	Mar 2014	TBD	Sep 2017*

* or later, depending on factors described above.

http://www.oracle.com/technetwork/java/eol-135779.html

IDE

eclipse

Java EE - jspx2/src/eg/java/net/web/jspx/ui/pages/Page.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access | Java EE | SVN Repository Exploring

Page.java

jspx2 ▸ src ▸ eg.java.net.web.jspx.ui.pages ▸ Page ▸

```
1  package eg.java.net.web.jspx.ui.pages;
2
3  import java.io.IOException;
71
72 /**
73  * Main controller for All JSPX files. <br />
74  * used as default for pages without controller. <br />
75  * the content and master pages inherits from.
76  *
77  * @author amr.eladawy
78  *
79  */
80 public class Page
81 {
82     private static final Logger logger = LoggerFactory.getLogger(Page.class);
83
84     public static final String DispatchedPage = "DispatchedPage";
85
86     protected ServletContext context;
87     protected String charSet;
88
89     private String pageURL;
```

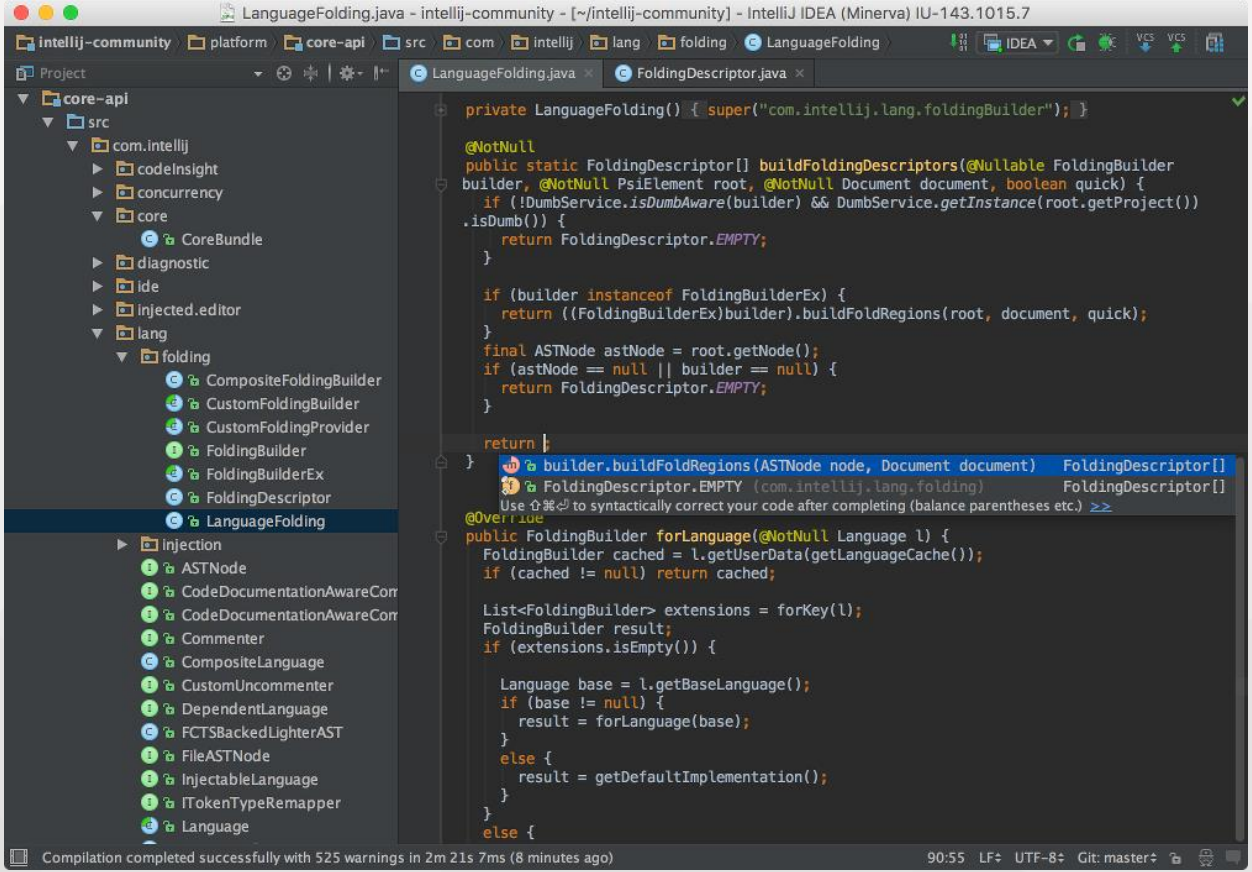
- jspx-demo2
- jspx-demo3
- jspx-maven
- jspx-web3
- >jspx2 [jspx-bay.cvs.sourceforge.net]
  - >src
    - eg.java.net.web.jspx
      - engine
      - ui
        - controls
        - >pages
          - ContentPage.java 1.1
          - MasterPage.java 1.1
          - >Page.java 1.1
            - >Page 1.1
          - PortletPage.java 1.1
    - org.kxml
    - resources
  - commons-fileupload-1.2.1.jar 1.1

Markers | Properties | Servers | Data Source Explorer | Snippets | Problems | Console | Progress | Search | Debug | Synchronize | Call Hierarchy | Search | Palette | Breakpoints

- NullPointerException: caught and uncaught
- SAXParseException: caught and uncaught
- StringIndexOutOfBoundsException: caught and uncaught
- AssetViewController [line: 35] - pageLoaded()

Writable | Smart Insert | 82 : 1 | support.jspx@gmail.com

IDE


LanguageFolding.java - intellij-community - [~/intellij-community] - IntelliJ IDEA (Minerva) IU-143.1015.7
intellij-community › platform › core-api › src › com › intellij › lang › folding › LanguageFolding
Project
LanguageFolding.java
FoldingDescriptor.java
core-api
src
com.intellij
codeInsight
concurrency
core
CoreBundle
diagnostic
ide
injected.editor
lang
folding
CompositeFoldingBuilder
CustomFoldingBuilder
CustomFoldingProvider
FoldingBuilder
FoldingBuilderEx
FoldingDescriptor
LanguageFolding
injection
ASTNode
Commenter
CompositeLanguage
CustomUncommenter
DependentLanguage
FCTSBackedLighterAST
FileASTNode
InjectableLanguage
ITokenTypeRemapper
Language
private LanguageFolding() { super("com.intellij.lang.foldingBuilder"); }
@NotNull
public static FoldingDescriptor[] buildFoldingDescriptors(@Nullable FoldingBuilder builder, @NotNull PsiElement root, @NotNull Document document, boolean quick) {
if (!DumbService.isDumbAware(builder) && DumbService.getInstance(root.getProject()).isDumb()) {
return FoldingDescriptor.EMPTY;
}
if (builder instanceof FoldingBuilderEx) {
return ((FoldingBuilderEx)builder).buildFoldRegions(root, document, quick);
}
final ASTNode astNode = root.getNode();
if (astNode == null || builder == null) {
return FoldingDescriptor.EMPTY;
}
return
}
builder.buildFoldRegions(ASTNode node, Document document) FoldingDescriptor[]
FoldingDescriptor.EMPTY (com.intellij.lang.folding) FoldingDescriptor[]
Use ⇧⌘⏎ to syntactically correct your code after completing (balance parentheses etc.) >>
@Override
public FoldingBuilder forLanguage(@NotNull Language l) {
FoldingBuilder cached = l.getUserData(getLanguageCache());
if (cached != null) return cached;
List<FoldingBuilder> extensions = forKey(l);
FoldingBuilder result;
if (extensions.isEmpty()) {
Language base = l.getBaseLanguage();
if (base != null) {
result = forLanguage(base);
}
else {
result = getDefaultImplementation();
}
}
else {
Compilation completed successfully with 525 warnings in 2m 21s 7ms (8 minutes ago)
90:55 LF UTF-8 Git: master

Dependency Management

Quick Start

Download | 4.3.7 CURRENT GA | MAVEN GRADLE

The recommended way to get started using `spring-framework` in your project is with a dependency management system – the snippet below can be copied and pasted into your build. Need help? See our getting started guides on building with Maven and Gradle.

```
<dependencies>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>4.3.7.RELEASE</version>
    </dependency>
</dependencies>
```

Spring Framework includes a number of different modules. Here we are showing `spring-context` which provides core functionality. Refer to the getting started guides on the right for other options.

Dependency Management


```
01. <project xmlns="http://maven.apache.org/POM/4.0.0"
02.          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
03.          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
04.
05. http://maven.apache.org/maven-v4_0_0.xsd">
06.
07.     <modelVersion>4.0.0</modelVersion>
08.     <groupId>com.technologyconversations</groupId>
09.     <artifactId>java-build-tools</artifactId>
10.     <packaging>jar</packaging>
11.     <version>1.0</version>
12.
13.     <dependencies>
14.         <dependency>
15.             <groupId>junit</groupId>
16.             <artifactId>junit</artifactId>
17.             <version>4.11</version>
18.         </dependency>
19.         <dependency>
20.             <groupId>org.hamcrest</groupId>
21.             <artifactId>hamcrest-all</artifactId>
22.             <version>1.3</version>
23.         </dependency>
24.     </dependencies>
25.
26.     <build>
27.         <plugins>
28.             <plugin>
29.                 <groupId>org.apache.maven.plugins</groupId>
30.                 <artifactId>maven-compiler-plugin</artifactId>
31.                 <version>2.3.2</version>
32.             </plugin>
33.         </plugins>
34.     </build>
35.
36. </project>
```

```
01. <plugin>
02.     <groupId>org.apache.maven.plugins</groupId>
03.     <artifactId>maven-checkstyle-plugin</artifactId>
04.     <version>2.12.1</version>
05.     <executions>
06.         <execution>
07.             <configuration>
08.                 <configLocation>config/checkstyle/checkstyle.xml</configLocation>
09.                 <consoleOutput>true</consoleOutput>
10.                 <failsOnError>true</failsOnError>
11.             </configuration>
12.             <goals>
13.                 <goal>check</goal>
14.             </goals>
15.         </execution>
16.     </executions>
17. </plugin>
18. <plugin>
19.     <groupId>org.codehaus.mojo</groupId>
20.     <artifactId>findbugs-maven-plugin</artifactId>
21.     <version>2.5.4</version>
22.     <executions>
23.         <execution>
24.             <goals>
25.                 <goal>check</goal>
26.             </goals>
27.         </execution>
28.     </executions>
29. </plugin>
30. <plugin>
31.     <groupId>org.apache.maven.plugins</groupId>
32.     <artifactId>maven-pmd-plugin</artifactId>
33.     <version>3.1</version>
34.     <executions>
35.         <execution>
36.             <goals>
37.                 <goal>check</goal>
38.             </goals>
39.         </execution>
40.     </executions>
41. </plugin>
```

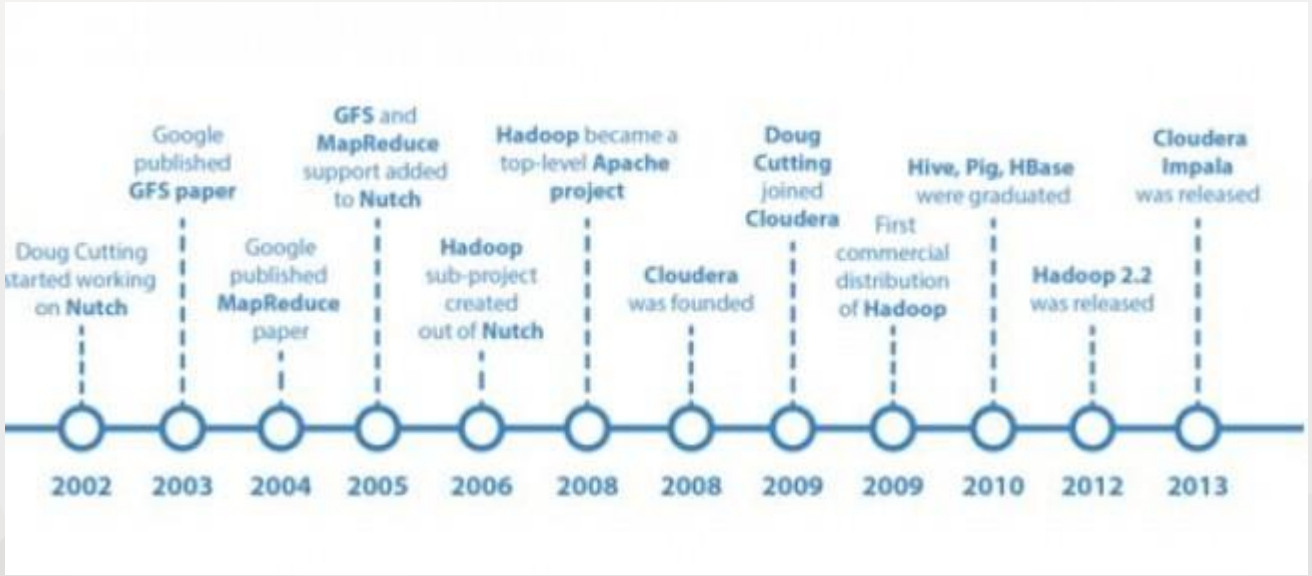
Dependency Management


```
apply plugin: 'java'
apply plugin: 'checkstyle'
apply plugin: 'findbugs'
apply plugin: 'pmd'

version = '1.0'

repositories {
    mavenCentral()
}

dependencies {
    testCompile group: 'junit', name: 'junit', version: '4.11'
    testCompile group: 'org.hamcrest', name: 'hamcrest-all', version: '1.3'
}
```

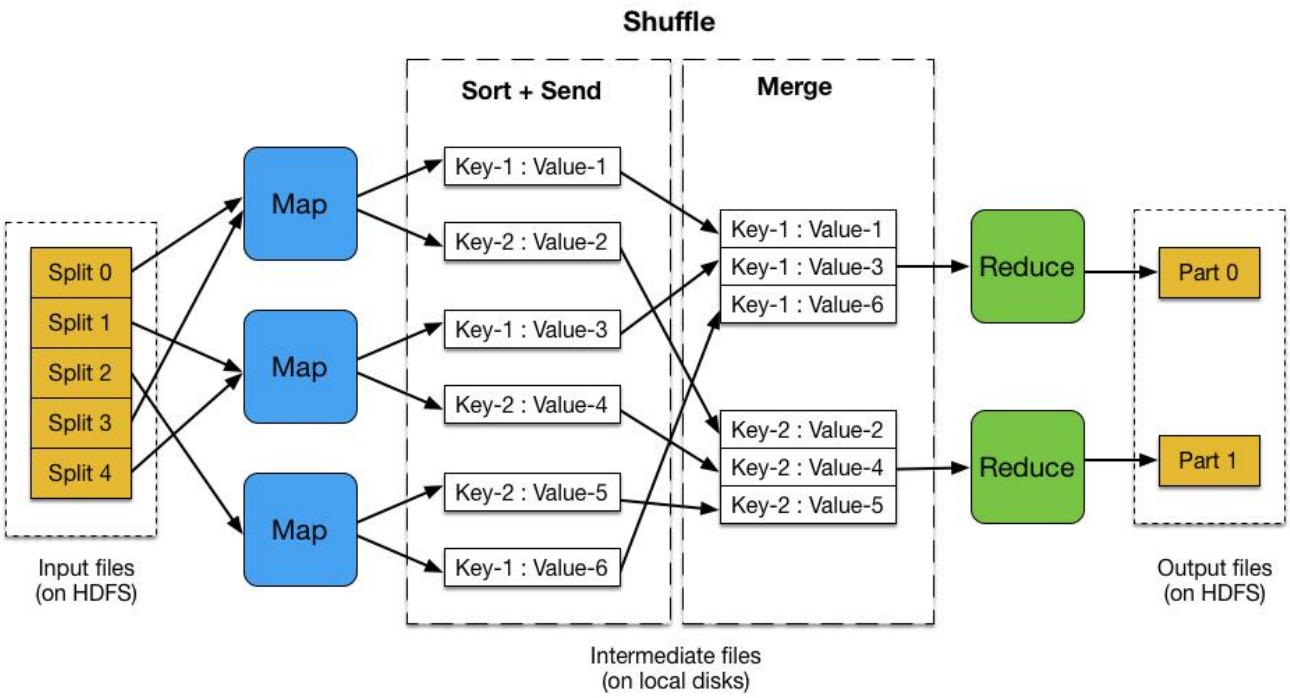
PART1

Map-Reduce

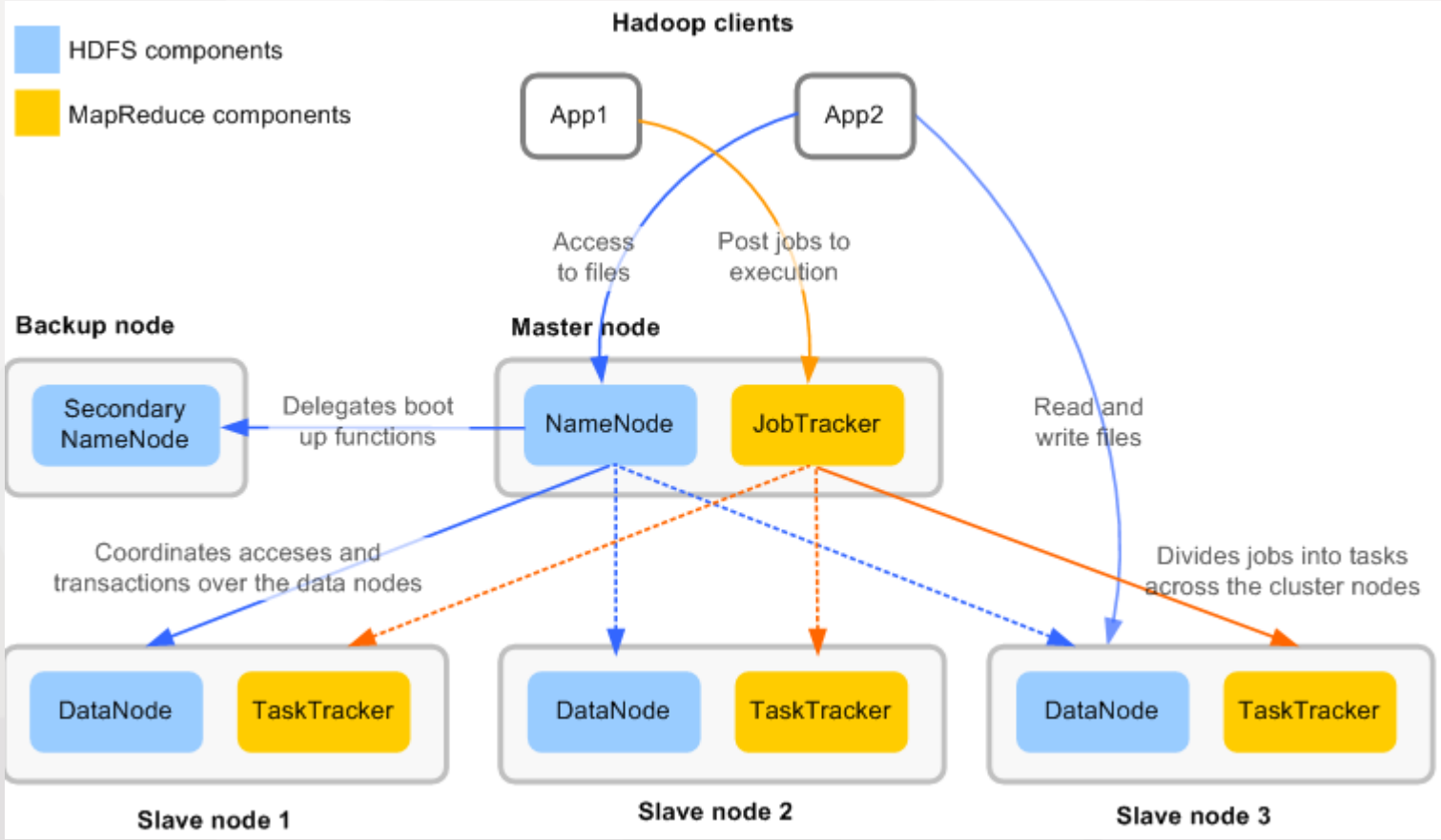
Google


Doug Cutting started working on Nutch
2002
Google published GFS paper
2003
Google published MapReduce paper
2004
GFS and MapReduce support added to Nutch
2005
Hadoop sub-project created out of Nutch
2006
Hadoop became a top-level Apache project
2008
Cloudera was founded
2008
Doug Cutting joined Cloudera
2009
First commercial distribution of Hadoop
2009
Hive, Pig, HBase were graduated
2010
Hadoop 2.2 was released
2012
Cloudera Impala was released
2013

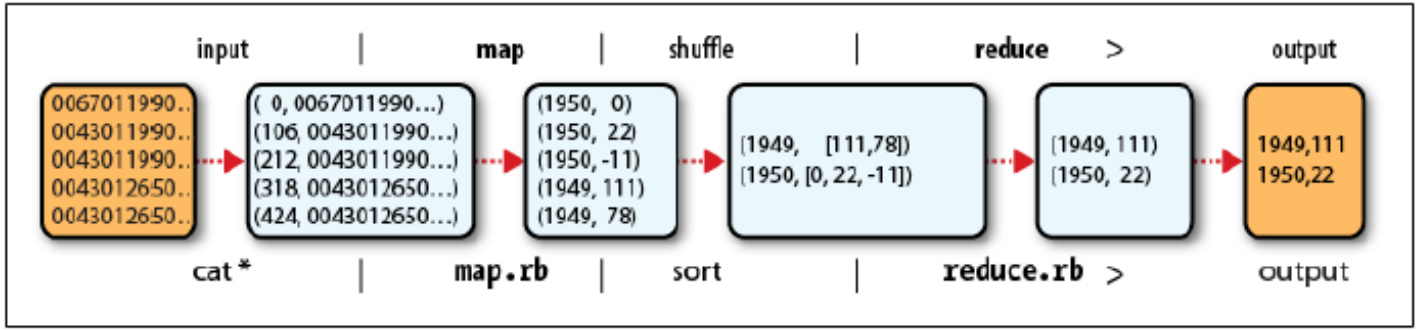
Map Reduce


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


HDFS components
MapReduce components
Hadoop clients
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



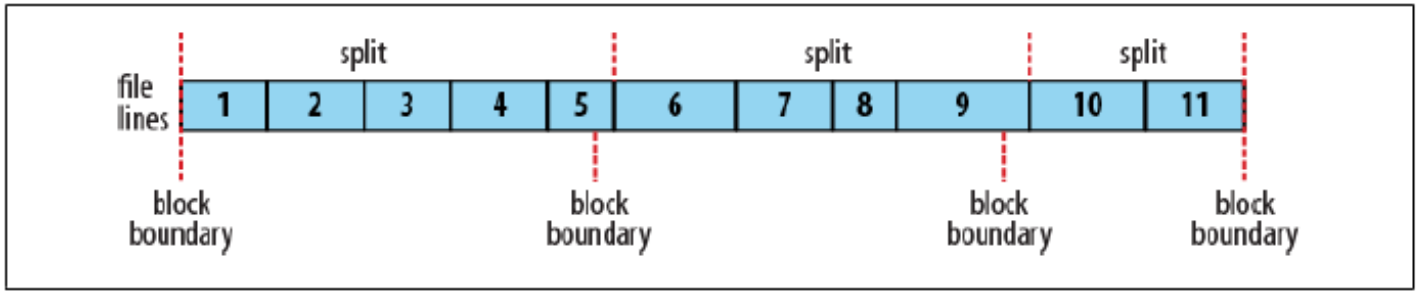
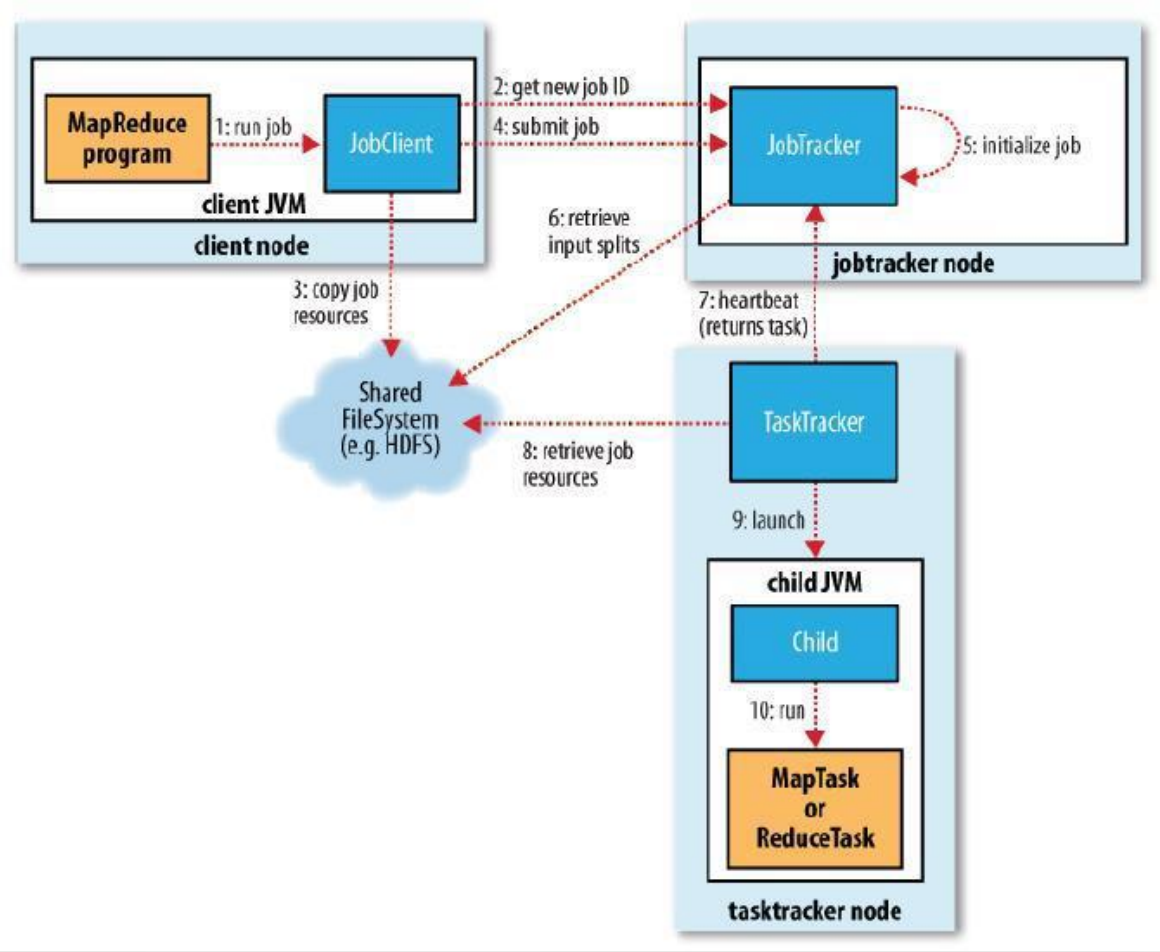


Figure 7-3. Logical records and HDFS blocks for TextInputFormat

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
7: heartbeat (returns task)
Shared FileSystem (e.g. HDFS)
8: retrieve job resources
TaskTracker
9: launch
child JVM
Child
10: run
MapTask or ReduceTask
tasktracker node

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

Map Reduce Model #1
Iterator Map-Reduce

A Project by Brad Lyon
Want an App for this? Tell me

Exploration of the Google PageRank Algorithm

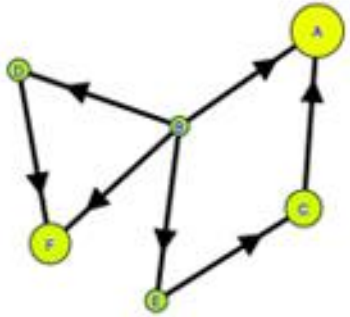
Circles correspond to web pages, links correspond to hyperlinks

Questions/Comments
A Few Notes About This

Damping Factor d:
0.85

Generate New

- *Drag nodes to rearrange*
- *Click on node, then click on another to create link*
- *Double-click to add node (max 14)*
- ***Del+mouseover** deletes node*
- *Click on link to delete*
- *Mouseover link to see location in matrix*



Sorted PageRank Vector
(8 Iterations)

Explore Convergence

Click here to explore/hide how the sequence converged to the solution

C	0.19
D	0.11
E	0.11
B	0.09

Hide Matrix Stuff

The PageRank vector **x** satisfies **x=Gx**, where

Google Matrix **G** = d * [(Hyperlink Matrix **H**) + (Dangling Nodes Matrix **A**)] + ((1-d)/N) * (NxN Matrix **U** of all 1's)

$$G = 0.85 * \left[\text{Hyperlink Matrix } H + \text{Dangling Nodes Matrix } A \right] + \frac{0.15}{6} * \text{Matrix } U \text{ with all 1's}$$

Hyperlink Matrix H

	A	B	C	D	E	F
A		1/4	1			
B						
C					1	
D		1/4				
E		1/4				
F		1/4		1		

Dangling Nodes Matrix A

	A	B	C	D	E	F
A	1/6					1/6
B	1/6					1/6
C	1/6					1/6
D	1/6					1/6
E	1/6					1/6
F	1/6					1/6

Matrix U with all 1's

	A	B	C	D	E	F
A	1	1	1	1	1	1
B	1	1	1	1	1	1
C	1	1	1	1	1	1
D	1	1	1	1	1	1
E	1	1	1	1	1	1
F	1	1	1	1	1	1

0.17	0.24	0.88	0.03	0.03	0.17

Map Reduce Model #2
Job Control

Naïve Bayes in MapReduce

- Map
  - Input data $\{x, y\}$ from a subgroup of data
  - Output: 3 types of keys

$$key = (x_j = a_{pj}^j, y = c_k), value = \sum_{subgroup} 1(x_j = a_{pj}^j \mid y = c_k)$$

$$key = (y = c_k), value = \sum_{subgroup} 1(y = c_k)$$

$$key = "samples", value = \sum_{subgroup} 1$$

- Reduce
  - Sum all the values of each key
  - Compute the conditional and marginal probabilities

Support Vector Machine in MapReduce

- Map
  - Input: $\{(x, y)\}$
  - Output:

$$key = GGW, value = 2w + 2C \sum_{subgroup} (w.x_i - y_i) x_i$$

- Reduce
  - Aggregate the values of gradient from all mappers
  - Update

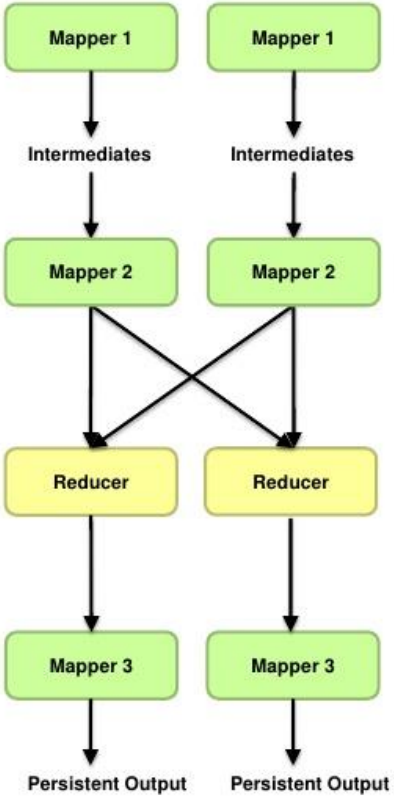
$$w = w - \eta * \nabla G_w$$

- Driver program that sets up the iterations and checks for convergence

Map Reduce Model #3
Chain-Mapper Chain-Reduce

Chaining in Hadoop

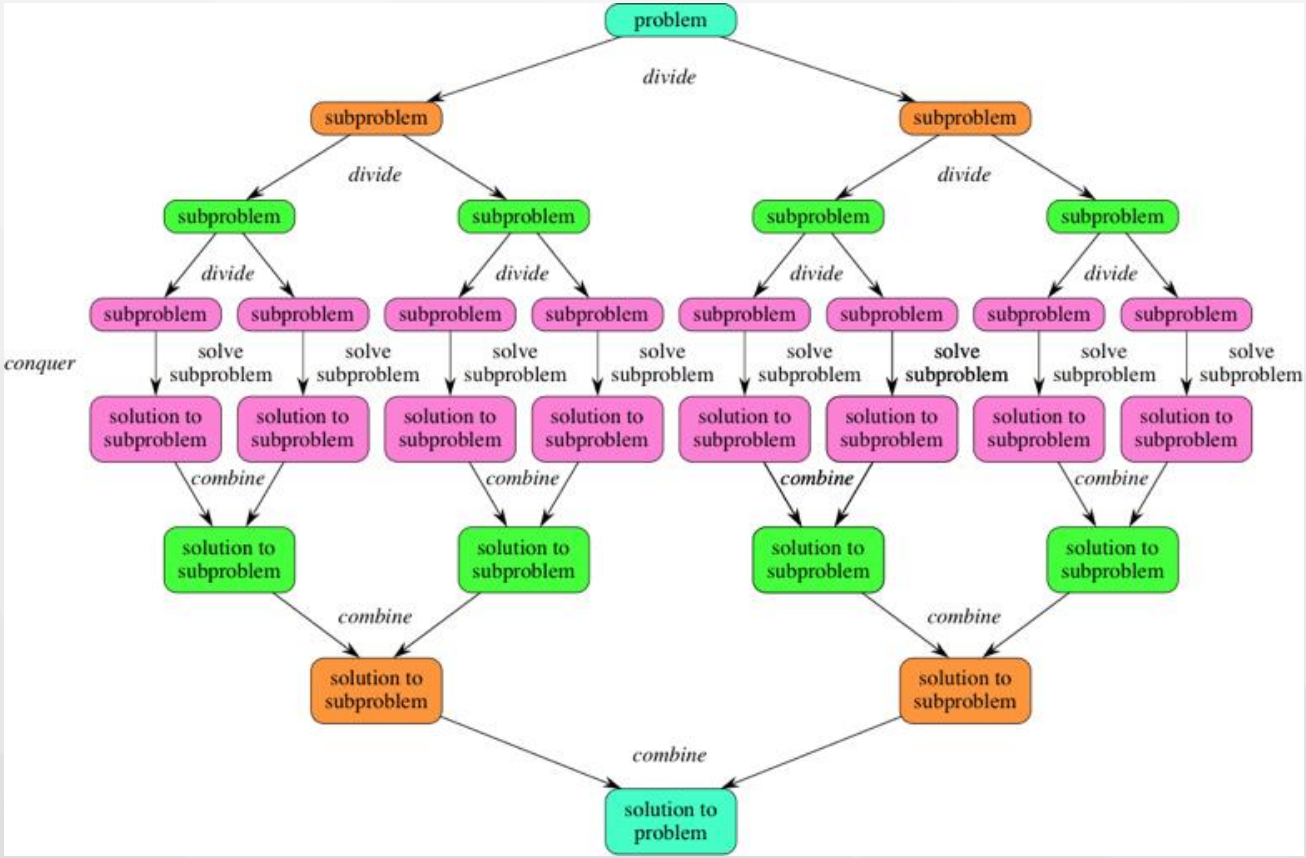
- Map+ Reduce Map*
  - 1 or more Mappers
    - Can use IdentityMapper
  - 1 reducer
    - No reducers: conf.setNumReduceTasks(0)?
  - 0 or more Mappers
- Usual *combiners* and *partitioners*
- By default, data passed between Mappers by usual writing of intermediate data to disk
  - Can always use side-effects…
  - There is a better, built-in way to bypass this and pass (Key,Value) pairs *by reference* instead
    - Requires different Mapper semantics!



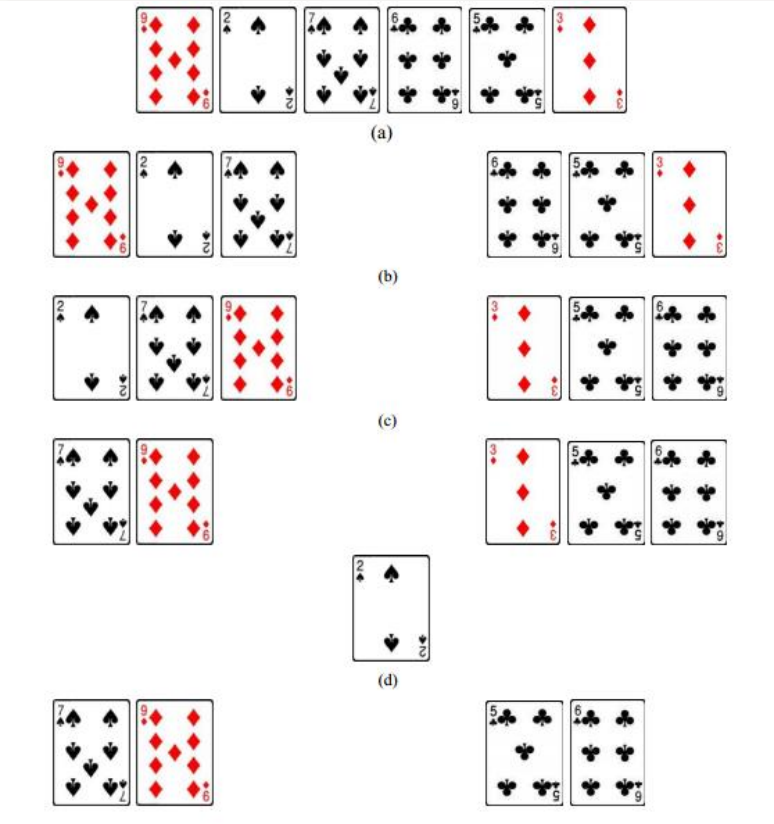
PART2

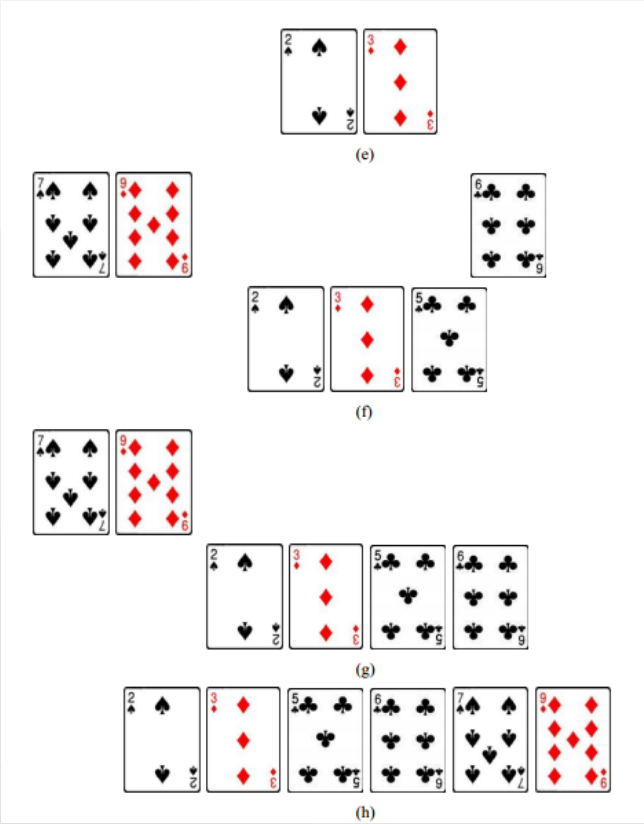
Computation Modal
計算模型

Divide & Conquer

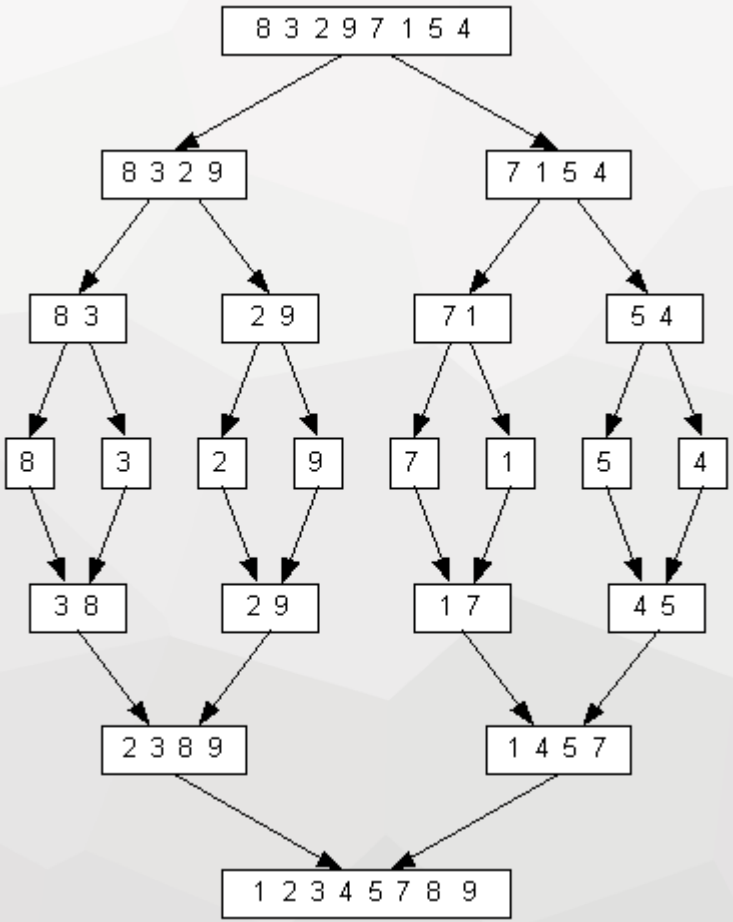


Divide & Conquer





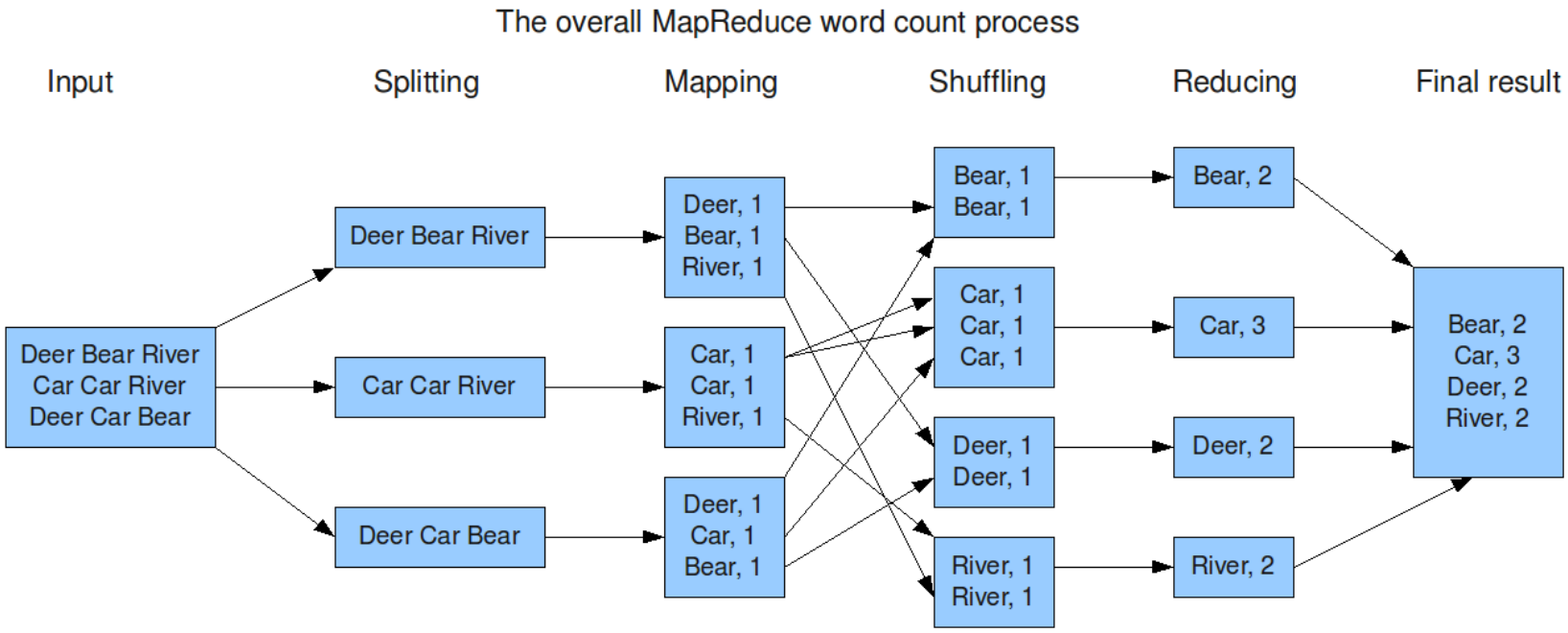
Divide & Conquer



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

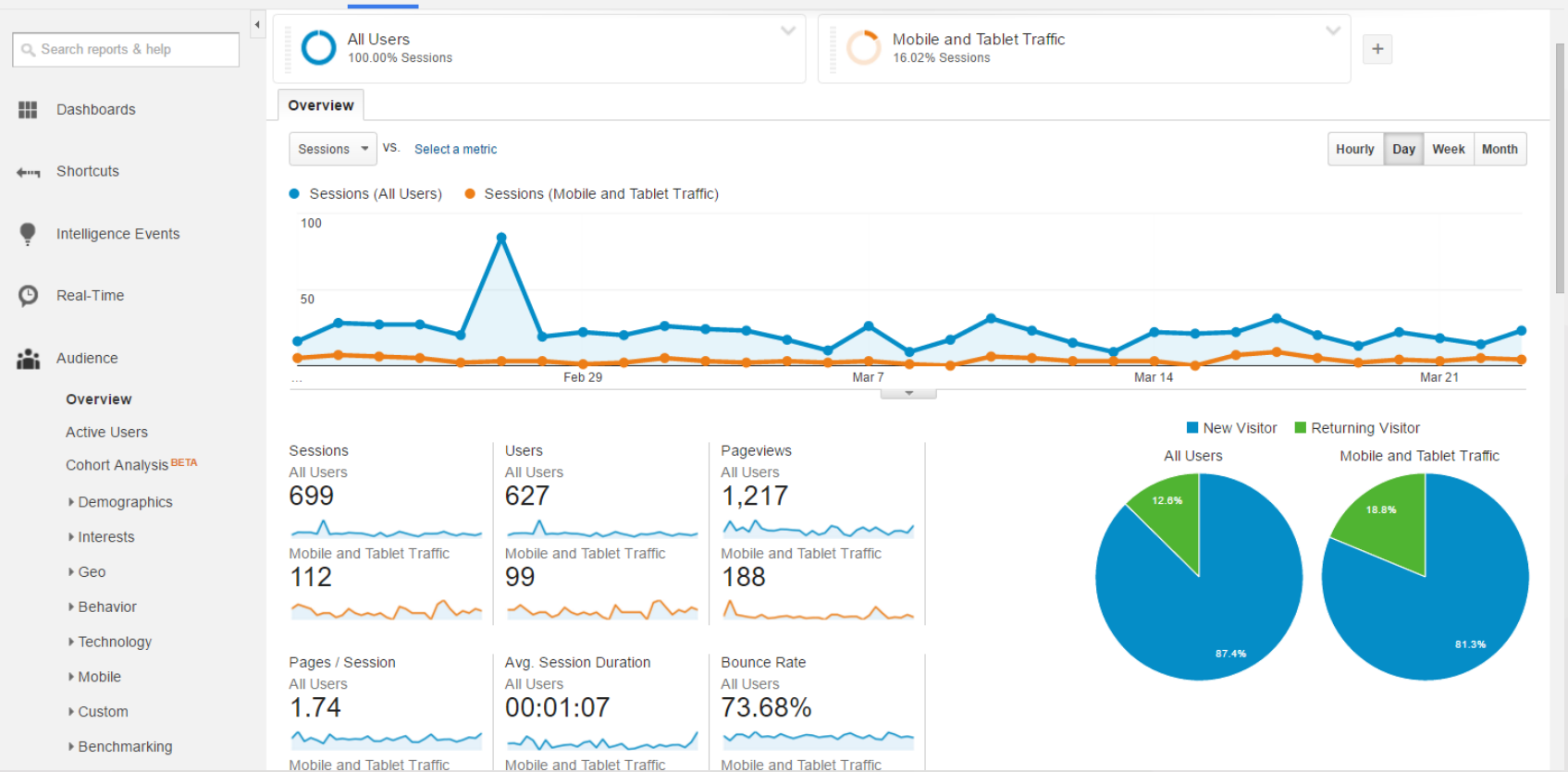
Property	Job setter method	Input types		Intermediate types		Output types	
		K1	V1	K2	V2	K3	V3
Properties for configuring types:							
mapreduce.job.inputformat.class	setInputFormatClass()	•	•				
mapreduce.map.output.key.class	setMapOutputKeyClass()			•			
mapreduce.map.output.value.class	setMapOutputValueClass()				•		
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

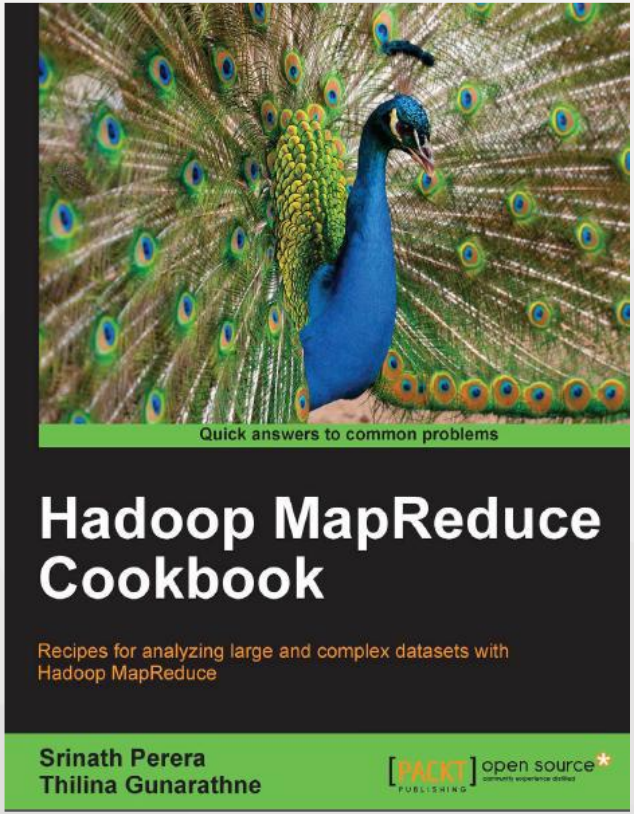

Search reports & help
Dashboards
Shortcuts
Intelligence Events
Real-Time
Audience
Overview
Active Users
Cohort Analysis BETA
Demographics
Interests
Geo
Behavior
Technology
Mobile
Custom
Benchmarking
All Users
100.00% Sessions
Mobile and Tablet Traffic
16.02% Sessions
Overview
Sessions
VS.
Select a metric
Hourly
Day
Week
Month
Sessions (All Users)
Sessions (Mobile and Tablet Traffic)
100
50
Feb 29
Mar 7
Mar 14
Mar 21
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
New Visitor
Returning Visitor
All Users
12.6%
87.4%
Mobile and Tablet Traffic
18.8%
81.3%

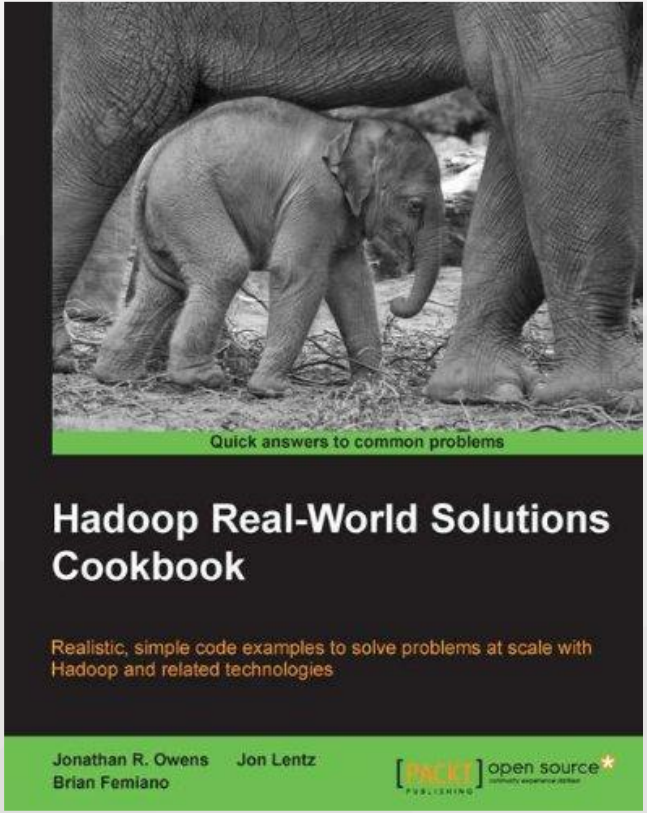
LBS

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

HomeWork

The End