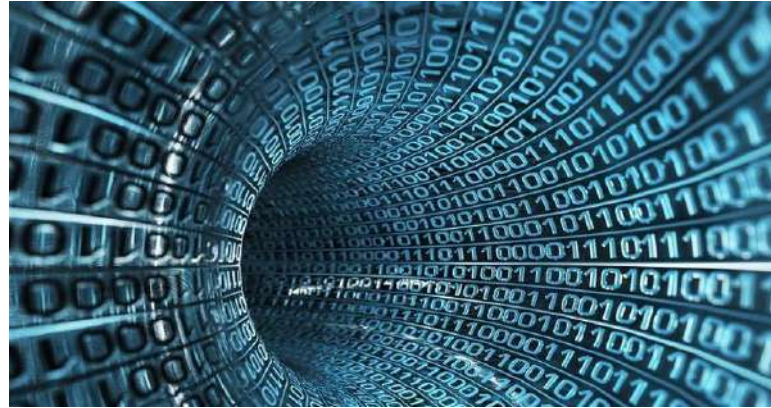


Sistemi Informativi Evoluti e Big Data



Map-Reduce – Deployment del codice

Università degli Studi di Brescia

Dipartimento di Ingegneria dell'Informazione



Un esempio in pratica: WordCount

```
map(String docid, String text):
```

```
  for each word w in text:  
    emit (w, 1);
```

```
reduce(String term, counts[]):
```

```
  int sum = 0;  
  for each c in counts:  
    sum += c;  
  emit (term, sum);
```



Implementazione: librerie

```
import org.apache.hadoop.conf.Configuration;  
import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.Mapper;  
import org.apache.hadoop.mapreduce.Reducer;  
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
```



Implementazione: map

Map(String docid, String text):
for each word w in text:
Emit(w, 1);

```
public static class WordCountMapper
    extends Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable ONE = new IntWritable(1);
    private final Text parola = new Text();
    public void map(LongWritable chiave, Text testo,
                    Context contesto) throws IOException,
                    InterruptedException {
        String linea = testo.toString();
        StringTokenizer tokenizer = new StringTokenizer(linea);
        while (tokenizer.hasMoreTokens()) {
            parola.set(tokenizer.nextToken());
            contesto.write(parola, ONE);
        }
    }
}
```

Implementazioni
dell'interfaccia
Writable (Text,
LongWritable,
FloatWritable,
BooleanWritable,
IntWritable,
ByteWritable,
ArrayWritable,
etc.)



Implementazione: reduce

```
public static class WordCountReducer extends Reducer<Text,
IntWritable, Text, IntWritable> {
    private IntWritable risultato = new IntWritable();
    public void reduce(Text chiave,
                        Iterable<IntWritable> valori,
                        Context contesto) throws IOException,
                        InterruptedException {
        int somma = 0;
        for(IntWritable val : valori) {
            somma += val.get();
        }
        risultato.set(somma);
        contesto.write(chiave, risultato);
    }
}
```

Map(String docid, String text):
for each word w in text:
Emit(w, 1);

Reduce(String term, counts[]):
int sum = 0;
for each c in counts:
sum += c;
Emit(term, sum);



Implementazione: Job

```
public class WordCount {  
    public static void main(String[] args) throws Exception {  
        Configuration conf = new Configuration();  
        Job job = Job.getInstance(conf, "Word Counter");  
        job.setJarByClass(WordCount.class);  
  
        job.setMapperClass(WordCountMapper.class);  
        job.setReducerClass(WordCountReducer.class);  
        FileInputFormat.addInputPath(job, new Path(args[0]));  
        FileOutputFormat.setOutputPath(job, new Path(args[1]));  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
        System.exit(job.waitForCompletion(true)?0:1);  
    }  
}
```

Hadoop – Documentazione utile

Un Wiki su Hadoop è disponibile al seguente link: <https://hadoop.apache.org/docs/stable/>

General

Overview

Single Node Setup

Cluster Setup

Commands Reference

FileSystem Shell

Compatibility

Specification

Downstream Developer's

Guide

Admin Compatibility

Guide

Interface Classification

FileSystem Specification

Common

CLI Mini Cluster

Native Libraries

Proxy User

Rack Awareness

Secure Mode

Service Level

Authorization

HTTP Authentication

Credential Provider API

Hadoop KMS

Tracing

Unix Shell Guide

HDFS

Architecture

User Guide

Commands Reference

NameNode HA With QJM

NameNode HA With NFS

Federation

ViewFs

Snapshots

Apache Hadoop 3.2.1

Apache Hadoop 3.2.1 incorporates a number of significant enhancements over the previous major release line (hadoop-3.2).

This release is generally available (GA), meaning that it represents a point of API stability and quality that we consider production-ready.

Overview

Users are encouraged to read the full set of release notes. This page provides an overview of the major changes.

Node Attributes Support in YARN

Node Attributes helps to tag multiple labels on the nodes based on its attributes and supports placing the containers based on expression of these labels.

More details are available in the [Node Attributes](#) documentation.

Hadoop Submarine on YARN

Hadoop Submarine enables data engineers to easily develop, train and deploy deep learning models (in TensorFlow) on very same Hadoop YARN cluster where data resides.

More details are available in the [Hadoop Submarine](#) documentation.

Storage Policy Satisfier

Supports HDFS (Hadoop Distributed File System) applications to move the blocks between storage types as they set the storage policies on files/directories.

More details are available in the [Storage Policy Satisfier](#) documentation.



I comandi HDFS su Hadoop

```
$:~hadoop-*/bin/hdfs dfs <command> <parameters>
```

- create a directory in hdfs



```
$:~hadoop-*/bin/hdfs dfs -mkdir input
```

- copy a local file in hdfs



```
$:~hadoop-*/bin/hdfs dfs -put /tmp/example.txt input
```

- copy result files from hdfs to local file system

```
$:~hadoop-*/bin/hdfs dfs -get output/result localoutput
```

- delete a directory in hdfs



```
$:~hadoop-*/bin/hdfs dfs -rm -r input
```


Navigare nel file system distribuito

Hadoop Overview Datanodes Snapshot Startup Progress Utilities ▾

[Browse the file system](#)
[Logs](#)

Overview 'localhost:9000' (active)

Started:	Wed Mar 25 16:49:08 CET 2015
Version:	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
Compiled:	2014-11-13T21:10Z by jenkins from (detached from e349649)
Cluster ID:	CID-1b20ec3f-9e75-4160-830c-3d6452564225
Block Pool ID:	

Browse Directory

user@mac:input

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	mac	supergroup	4.33 KB	1	128 MB	capacity-scheduler.xml
-rw-r--r--	mac	supergroup	1.3 KB	1	128 MB	configuration.xml
-rw-r--r--	mac	supergroup	318 B	1	128 MB	container-executor.clg
-rw-r--r--	mac	supergroup	884 B	1	128 MB	core-site.xml
-rw-r--r--	mac	supergroup	3.58 KB	1	128 MB	hadoop-env.cmd
-rw-r--r--	mac	supergroup	4.21 KB	1	128 MB	hadoop-env.sh
-rw-r--r--	mac	supergroup	2.43 KB	1	128 MB	hadoop-metrics.properties

http://host:50070



Navigare nel file system distribuito (Ambari)

http://host:8080

The screenshot displays the Ambari web interface. On the left is a dark sidebar with navigation links: Ambari, Dashboard, Services (HDFS, YARN, MapReduce2, Tez, Hive, HBase, Pig, Sqoop, Oozie, ZooKeeper, Storm), and a status icon. The main content area is titled 'Dashboard / Metrics' and includes tabs for METRICS, HEATMAPS, and CONFIG HISTORY. A red arrow points to the 'Files View' tab, which is highlighted. The 'Files View' window shows a directory listing for the path '/ > user > amy_ds'. A yellow banner indicates 'Total: 10 files or folders'. Below this is a table with columns: Name, Size, Last Modified, Owner, Group, Permission, Erasure Coding, and Encrypted. The table lists various files and directories, including .Trash, .sparkStaging, .staging, WordCount\$IntSumReducer.class, WordCount\$TokenizerMapper.class, WordCount.java, files-view, input, output, and wc.jar. On the right side of the interface, there are panels for 'Views' (Files View, Workflow Manager, DataNodes Live), 'CPU Usage' (No Data Available), and 'DataNodes Live' (1/1).

Ambari

Dashboard

Services

- HDFS
- YARN
- MapReduce2
- Tez
- Hive
- HBase
- Pig
- Sqoop
- Oozie
- ZooKeeper
- Storm

Dashboard / Metrics

METRICS HEATMAPS CONFIG HISTORY

Files View

Sandbox

Total: 10 files or folders

Select All New Folder Upload

Search in current directory...

Name	Size	Last Modified	Owner	Group	Permission	Erasure Coding	Encrypted
.Trash	--	2019-12-05 13:00	amy_ds	hdfs	drwx-----		No
.sparkStaging	--	2019-12-05 09:31	amy_ds	hdfs	drwxr-xr-x		No
.staging	--	2019-12-05 13:35	amy_ds	hdfs	drwx-----		No
WordCount\$IntSumReducer.class	1.7 kB	2019-12-02 13:48	amy_ds	hdfs	-rw-r--r--		No
WordCount\$TokenizerMapper.class	1.7 kB	2019-12-02 13:48	amy_ds	hdfs	-rw-r--r--		No
WordCount.java	2.0 kB	2019-12-02 13:48	amy_ds	hdfs	-rw-r--r--		No
files-view	--	2019-12-04 19:49	amy_ds	hdfs	drwxr-xr-x		No
input	--	2019-12-05 12:23	amy_ds	hdfs	drwxr-xr-x		No
output	--	2019-12-05 13:34	amy_ds	hdfs	drwxr-xr-x		No
wc.jar	3.0 kB	2019-12-02 10:48	amy_ds	hdfs	-rwxrwxr--		No

Views

- Files View
- Workflow Manager
- DataNodes Live

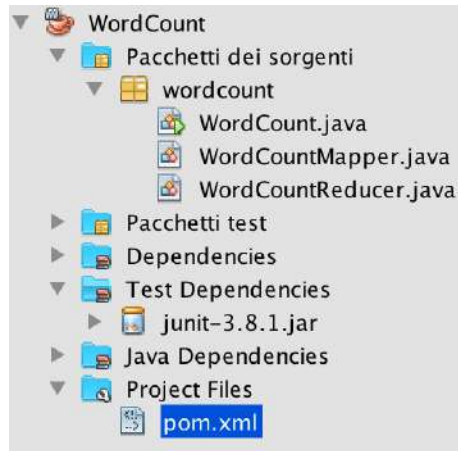
1/1

CPU Usage

No Data Available

Compilare un'applicazione MapReduce

Come progetto MAVEN attraverso l'IDE NetBeans (dalla versione 6.9 in poi):



```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany</groupId>
  <artifactId>WordCount</artifactId>
  <version>1.0</version>
  <packaging>jar</packaging>
  <dependencies>
    <dependency>
      <groupId>org.apache.hadoop</groupId>
      <artifactId>hadoop-core</artifactId>
      <version>1.2.0</version>
    </dependency>
    <dependency>
      <groupId>log4j</groupId>
      <artifactId>log4j</artifactId>
      <version>1.2.16</version>
    </dependency>
  </dependencies>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
</project>
```



Eseguire un'applicazione MapReduce

Esempio: **WordCount**

1. Generazione del file *jar* contenente le classi *Mapper*, *Reducer* e *Job* (per esempio, compilando il progetto MAVEN in NetBeans)
2. Spostamento del file jar sulla distribuzione Hadoop (file system locale)
3. Creazione delle cartelle che conterranno gli input e il risultato dell'elaborazione
4. Esecuzione dell'elaborazione parallela del codice
5. Visualizzazione dei risultati

Eseguire un'applicazione MapReduce

Esempio: **WordCount**

1. Generazione del file *jar* contenente le classi *Mapper*, *Reducer* e *Job* (per esempio, compilando il progetto MAVEN in NetBeans)
2. Spostamento del file *jar* sulla distribuzione Hadoop (file system locale)

```
scp -P 2222 <files-to-transfer> <hadoop_user>@<host>:/<destination-on-server>
```



Eseguire un'applicazione MapReduce

Esempio: **WordCount**

1. Generazione del file *jar* contenente le classi *Mapper*, *Reducer* e *Job* (per esempio, compilando il progetto MAVEN in NetBeans)
2. Spostamento del file jar sulla distribuzione Hadoop (file system locale)
3. Creazione delle cartelle che conterranno gli input e il risultato dell'elaborazione (da prompt dei comandi)

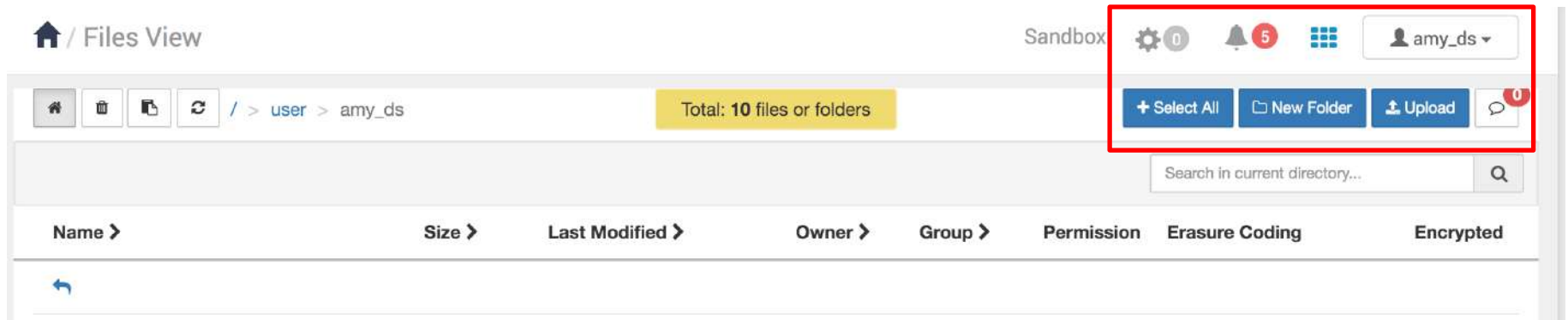
```
hdfs dfs -mkdir input
hdfs dfs -put divina_commedia.txt input
hdfs dfs -mkdir output
```



Eseguire un'applicazione MapReduce

Esempio: **WordCount**

1. Generazione del file *jar* contenente le classi *Mapper*, *Reducer* e *Job* (per esempio, compilando il progetto MAVEN in NetBeans)
2. Spostamento del file jar sulla distribuzione Hadoop (file system locale)
3. Creazione delle cartelle che conterranno gli input e il risultato dell'elaborazione (tramite Ambari)



Eseguire un'applicazione MapReduce

Esempio: **WordCount**

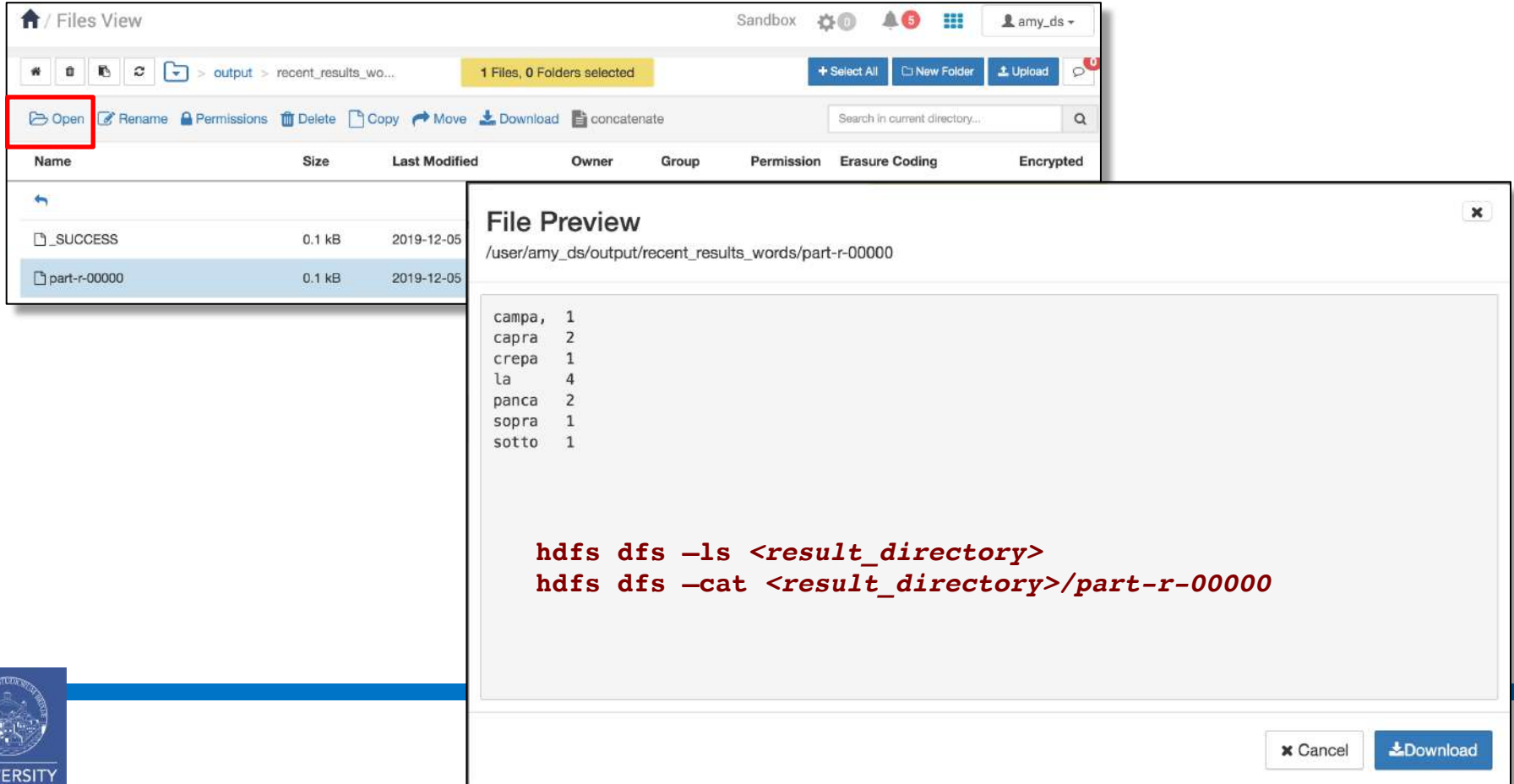
1. Generazione del file *jar* contenente le classi *Mapper*, *Reducer* e *Job* (per esempio, compilando il progetto MAVEN in NetBeans)
2. Spostamento del file jar sulla distribuzione Hadoop (file system locale)
3. Creazione delle cartelle che conterranno gli input e il risultato dell'elaborazione
4. Esecuzione dell'elaborazione parallela del codice

```
$:~hadoop-*/bin/hadoop jar <path-jar> <jar-MainClass> <jar-parameters>
```

```
hadoop jar <nome-file-jar> wordcount/WordCount <input> <output>
```



WordCount: visualizzazione dei risultati



The screenshot displays a file explorer window titled "Files View" with a "Sandbox" environment. The breadcrumb path is "> output > recent_results_wo...". A toolbar at the top includes icons for file operations and buttons for "Select All", "New Folder", and "Upload". A red box highlights the "Open" button in the toolbar. Below the toolbar is a table of files:

Name	Size	Last Modified	Owner	Group	Permission	Erasure Coding	Encrypted
_SUCCESS	0.1 kB	2019-12-05					
part-r-00000	0.1 kB	2019-12-05					

A "File Preview" window is open, showing the content of the selected file: `/user/amy_ds/output/recent_results_words/part-r-00000`. The preview displays a list of words and their counts:

```
campa, 1
capra 2
crepa 1
la 4
panca 2
sopra 1
sotto 1
```

Below the preview, two terminal commands are shown in red text:

```
hdfs dfs -ls <result_directory>
hdfs dfs -cat <result_directory>/part-r-00000
```

At the bottom right of the preview window are "Cancel" and "Download" buttons.



WordCount: utilizzo di un Combiner

Nella configurazione del *Job*:

```
job.setMapperClass(WordCountMapper.class);  
job.setCombinerClass(WordCountReducer.class);  
job.setReducerClass(WordCountReducer.class);
```



Testare un'applicazione MapReduce

- Librerie in Python che permettono il deployment e l'esecuzione in locale di un programma Map-Reduce
- Utili per testare programmi in assenza di un cluster o in attesa di fare il caricamento su un cluster
- Esempi di librerie (Python): **mrjob**, **pymr**

```
python code.py  
    input_file > output_file
```

```
from mrjob.job import MRJob  
import re  
  
WORD_RE = re.compile(r"[\w']+")  
  
class MRWordFreqCount(MRJob):  
  
    def mapper(self, _, line):  
        for word in WORD_RE.findall(line):  
            yield (word.lower(), 1)  
  
    def combiner(self, word, counts):  
        yield (word, sum(counts))  
  
    def reducer(self, word, counts):  
        yield (word, sum(counts))  
  
if __name__ == '__main__':  
    MRWordFreqCount.run()
```

