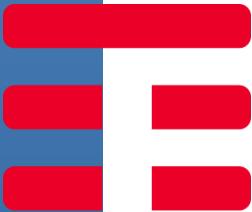




GRUPPO TELECOM ITALIA
Big Data Transformation

Big Data Transformation

TIM Academy



GRUPPO TIM

Big Data Transformation

Apache Spark Fundamentals & Query Fundamentals

 **TIM** Academy

GRUPPO TIM

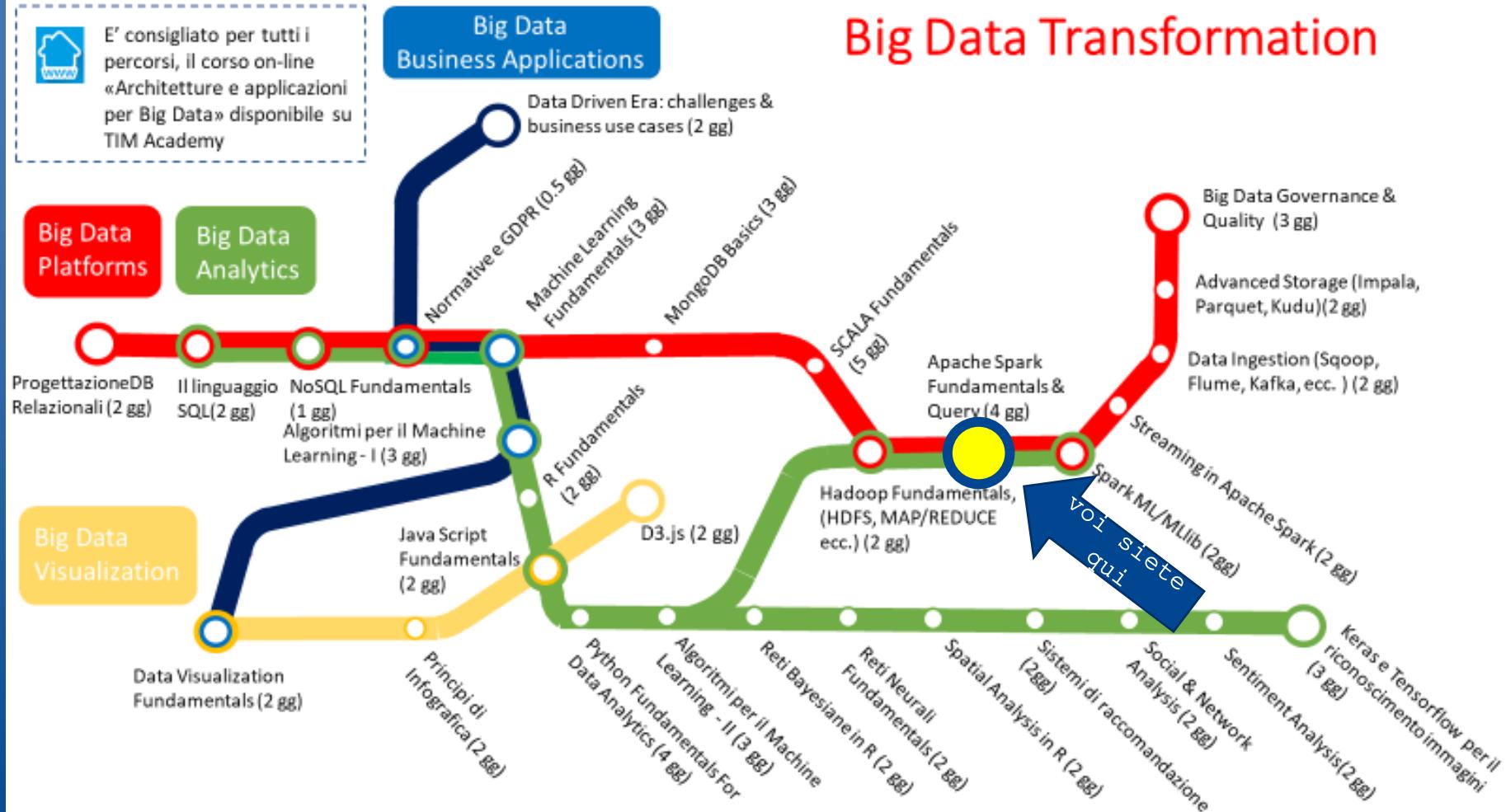
Big Data Transformation

Apache Spark Fundamentals & Query

 **TIM** Academy



E' consigliato per tutti i percorsi, il corso on-line «Architetture e applicazioni per Big Data» disponibile su TIM Academy



Apache Spark

Uno dei tool più usati nel campo dei *BIG DATA*



<https://www.whizlabs.com/blog/big-data-tools/>

- 1 • 
The Apache Hadoop logo features a yellow elephant icon next to the text "APACHE hadoop™". The word "hadoop" is written in a bold, lowercase, sans-serif font with a colorful, feather-like swoosh underneath it.
- 2 • 
The Apache Spark logo features a large orange star icon next to the text "APACHE Spark™". The word "Spark" is in a bold, lowercase, sans-serif font with a smaller orange star icon integrated into the letter "a". The background of the logo is a white sparkler effect.

Prerequisiti

- Conoscenza del linguaggio di programmazione Scala



- Aver seguito il corso Hadoop Fundamentals



Agenda del corso

	1° giorno	2° giorno	3° giorno	4° giorno	
mattina	9:30	9:00	9:00	9:00	lezione
pomeriggio					laboratorio
					lezione
	17:30	17:00	17:00	17:00	laboratorio

Agenda 1/8

— Introduzione

Presentazione del corso

Spark

Cos'è? A cosa serve? Chi lo usa? Punti di forza e
debolezze



Contesto e storia

Come è nato? Chi l'ha fatto? Quale problema affronta?

Com'è fatto?

Una prima presentazione dell'architettura

E in pratica?

Come si usa? Come lo useremo noi in laboratorio?



Agenda 2/8

— Le fondamenta

Strutture dati

RDDs, DataSets e DataFrames

Operazioni sui dati

Transformazioni (*narrow* e *wide*) e Azioni; Valutazioni «pigre»; persistenza (cache)

Esecuzione

Tasks e Jobs; Spark UI; resilienza (lineage); DAG

Un pizzico di teoria

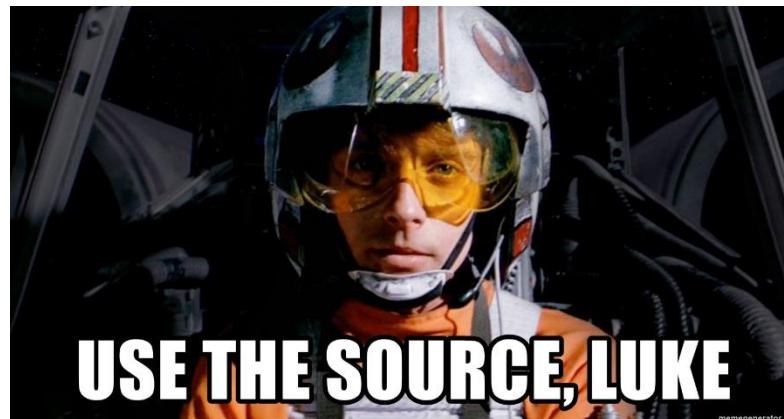
Programmazione funzionale per il calcolo parallelo

Coordinate

Documentazione; le varie distribuzioni: Cloudera, Hortonworks, ecc.; sorgente su github

Esercizi di laboratorio

Filtrri e aggregazioni; i widget di Spark Notebook



Agenda 3/8

— API

DataFrames e DataSet

Definizione e differenze

Gli schemi

Nome e tipo delle colonne; definiti manualmente o letti da un *data source*.

Catalyst

Spark SQL, piani logici e fisici.

I/O

Lettura e scrittura di file; HDFS, Hive, Local FS, S3, SequenceFile,...

Colonne ed espressioni

Vari tipi di trasformazioni su DataFrame e DataSet; selezioni, filtri, or

Gestire le stringhe

Trimming, Split (Explode), Espressioni regolari, Date e TimeStamp, Json

Funzioni definite dall'utente

Le UDF

RDD (forse va spostato)

```
root
|-- InvoiceNo: string (nullable = true)
|-- StockCode: string (nullable = true)
|-- Description: string (nullable = true)
|-- Quantity: integer (nullable = true)
|-- InvoiceDate: timestamp (nullable = true)
|-- UnitPrice: double (nullable = true)
|-- CustomerID: double (nullable = true)
|-- Country: string (nullable = true)
```

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice
536365	85123A	WHITE HANGING HEA...	6	2010-12-01 08:26:00	...
536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	...
...					
536367	21755	LOVE BUILDING BLO...	3	2010-12-01 08:34:00	...
536367	21777	RECIPE BOX WITH M...	4	2010-12-01 08:34:00	...

SQL (forse va spostato)

Esercizi di laboratorio

Esercitazioni sugli argomenti trattati.

Agenda 4/8

— Aggregazioni

Tipi di aggregazione

Inter DataFrame, group by, window, grouping set, rollup

Statistiche

Conteggi, somme, media, varianza, ecc.

Aggregazione su tipi complessi

Insiemi e liste

Window

Funzioni su finestre di dati: ranking, funzioni analitiche e di aggregazione

Funzioni di aggregazione definite dall'utente

Joins

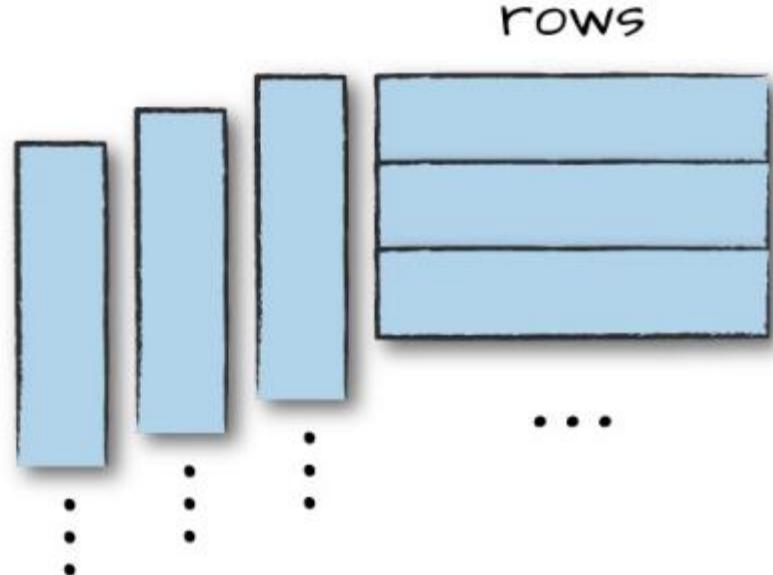
Inner Joins, Outer Joins, etc.

Esercizi di laboratorio

Esercitazioni sugli argomenti trattati.

$$\left(\sum_{i=1}^n p_i x_i^2 \right) - \mu^2$$

Window
frames



Agenda 5/8

Il cluster

Architettura di un'applicazione Spark
Spark Driver, Spark executors,
cluster manager (Spark standalone, YARN o Mesos)

Modi di esecuzione

Cluster mode, client mode, local mode

Ciclo di vita di un'applicazione Spark

Fuori e dentro Spark; tasks, jobs, stages

La sessione

SparkSession, SparkContext, SQLContext, HiveContext

Il programma

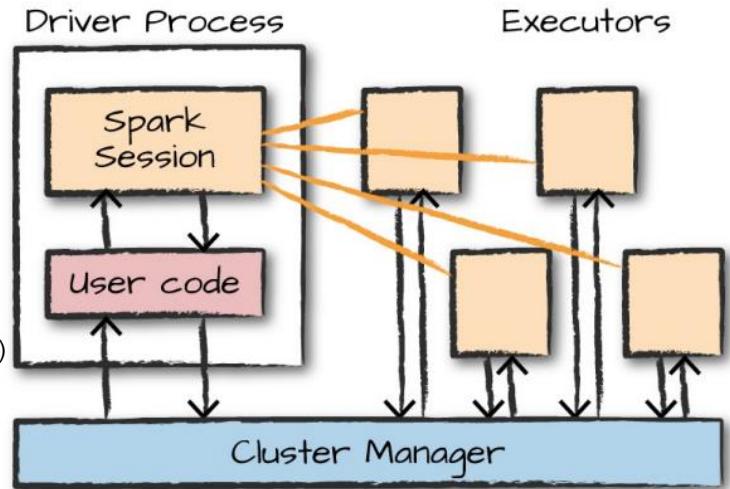
Struttura; compilazione (maven, sbt); spark-submit

Configurazione

SparkConf, argomenti di spark-submit;

Esercizi di laboratorio

Scrittura dei primi programmi in Spark; compilazione ed esecuzione.



Agenda 6/8

— DO e DON'T

Strumenti di monitoraggio e debugging
Spark UI; log; submit in modalità client

3 Ragioni per non usare i RDD
Sono obsoleti, più lenti e più complicati

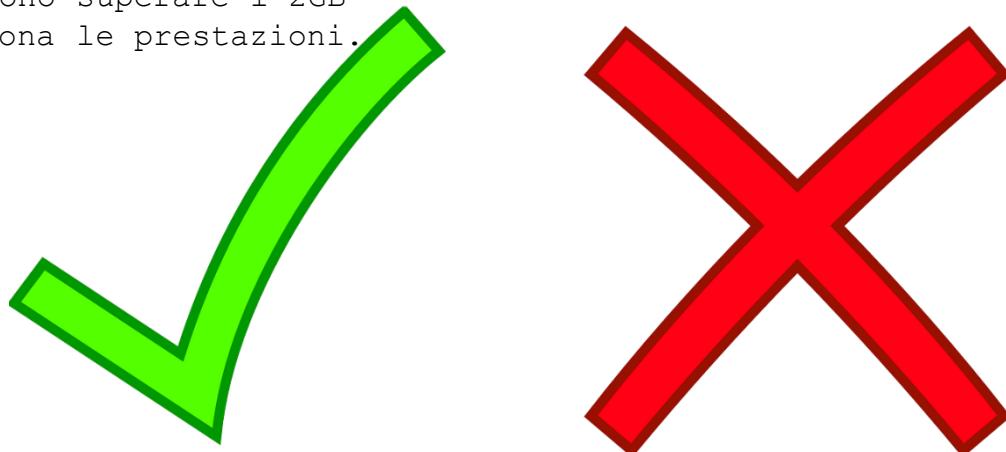
Decisioni, decisioni, decisioni!
Numero di esecutori? N. di core per esecutore? memoria per esecutore?

Numero di partizioni
I blocchi nello shuffle non devono superare i 2GB
Il numero di partizioni condiziona le prestazioni.

Controllare lo shuffling

Controllare il DAG
GroupByKey => ReduceByKey
Reduce => TreeReduce
Uso di tipi complessi

Esercizi di laboratorio
Esercizi di recapitolazione



Agenda 7/8

— Spark stack

Packages

Spark SQL, Spark Streaming, Spark ML, GraphFra

Spark Streaming

Breve introduzione: ci sarà un corso dedicato

Spark ML

Breve introduzione: ci sarà un corso dedicato

Nota: ML (basato su DataFrames) vs. MLLib (basato su RDD)

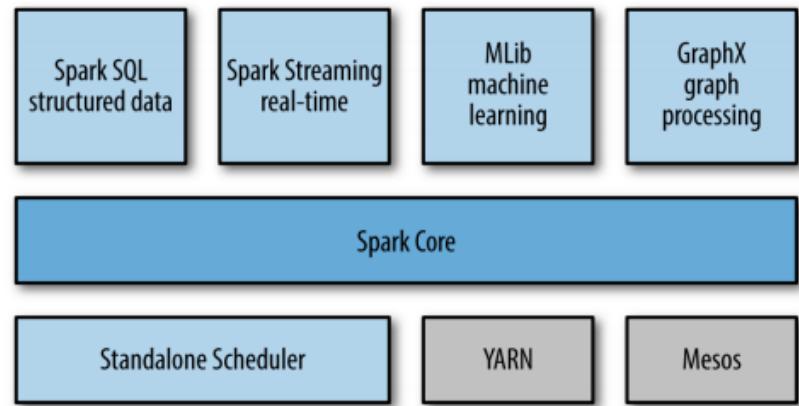
GraphFrames

Breve introduzione: approfondimento nel pomeriggio

Nota: GraphFrames (basato su DataFrames; 2016) vs. GraphX (basato su RDD; 2014)

Esercizi di laboratorio

Esercizi di recapitolazione



Agenda 8/8

— GraphFrames

Cos'è un grafo?

Vertici e archi; concetti e algoritmi base.

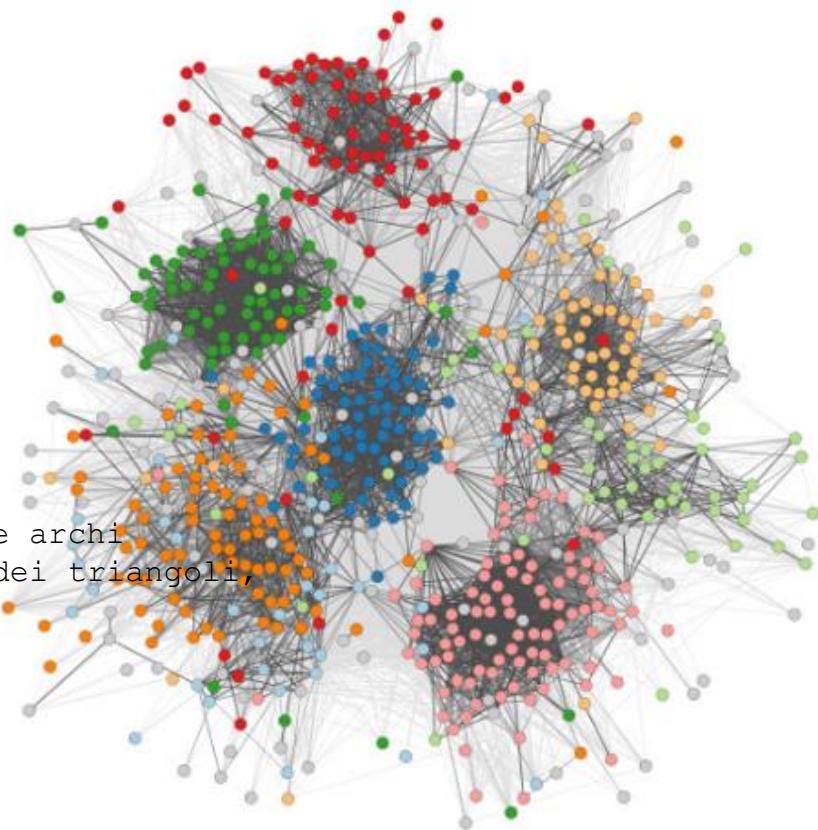
GraphX vs GraphFrames

La classe GraphFrame

- Come creare un grafo a partire da vertici e archi
- Cercare le componenti connesse, conteggio dei triangoli, pageRank, ecc.
- Funzioni di ricerca del motif usando il Domain Specific Language (DSL)

Esercizi di laboratorio

Esercizi di recapitolazione



— Conclusioni

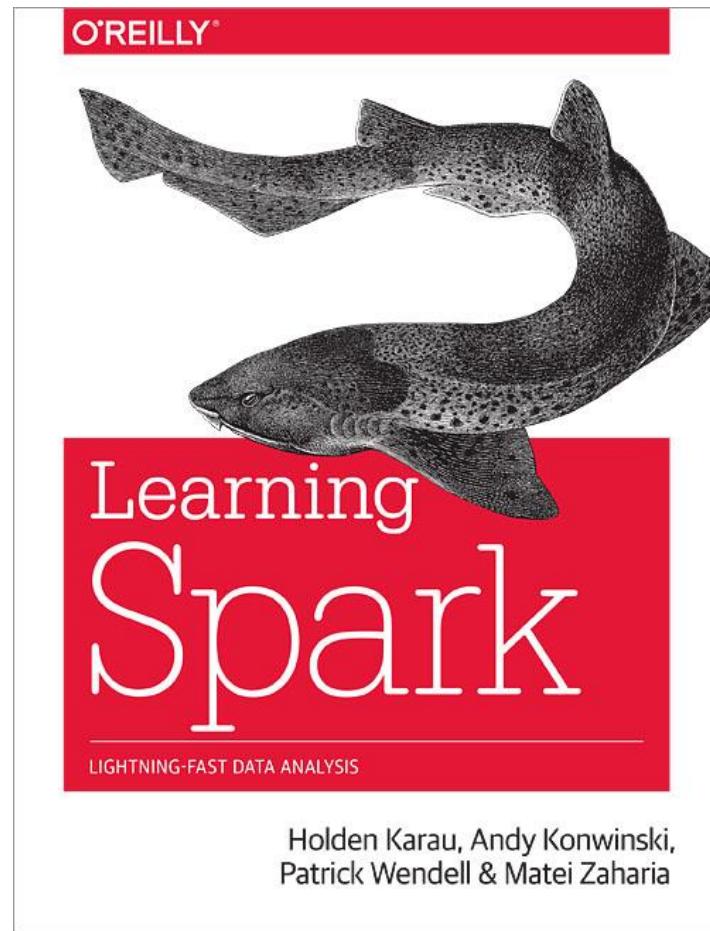
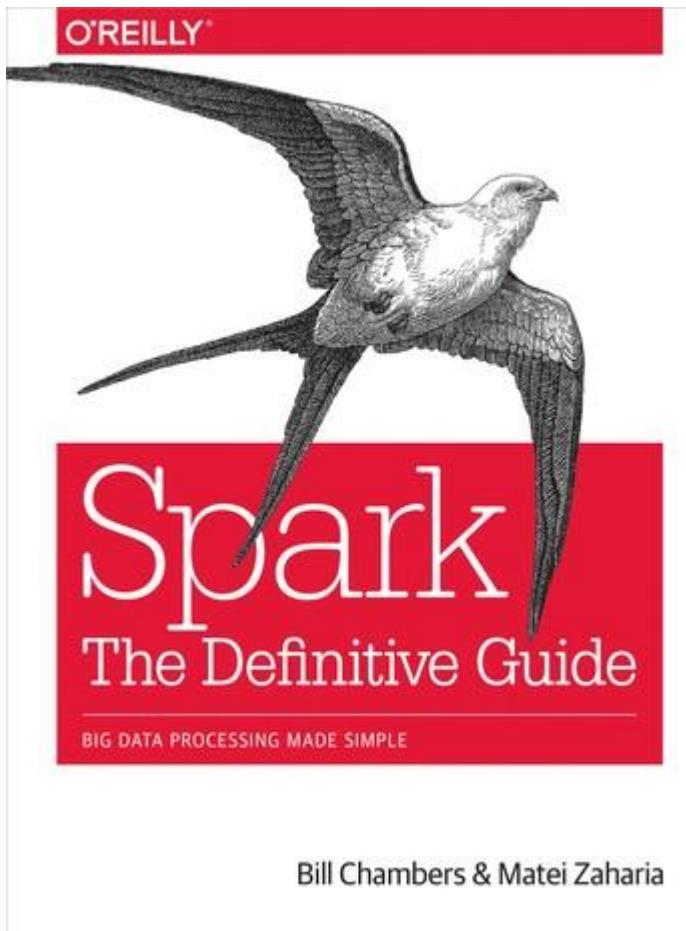
Recapitolazione e domande

Compilazione del quesito





Riferimenti bibliografici



Cos'è
Apache
Spark?

Cos'è Apache Spark?



WIKIPEDIA
The Free Encyclopedia

“Apache Spark is an **open-source distributed general-purpose cluster-computing framework**”.

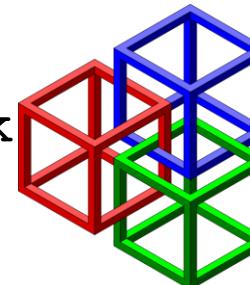
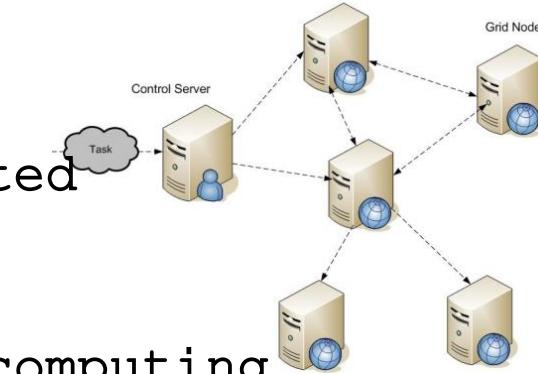
- Open-source
- General-purpose



Open Source
Initiative



- Distributed
- Cluster-computing
- Framework



Computing... ad esempio?

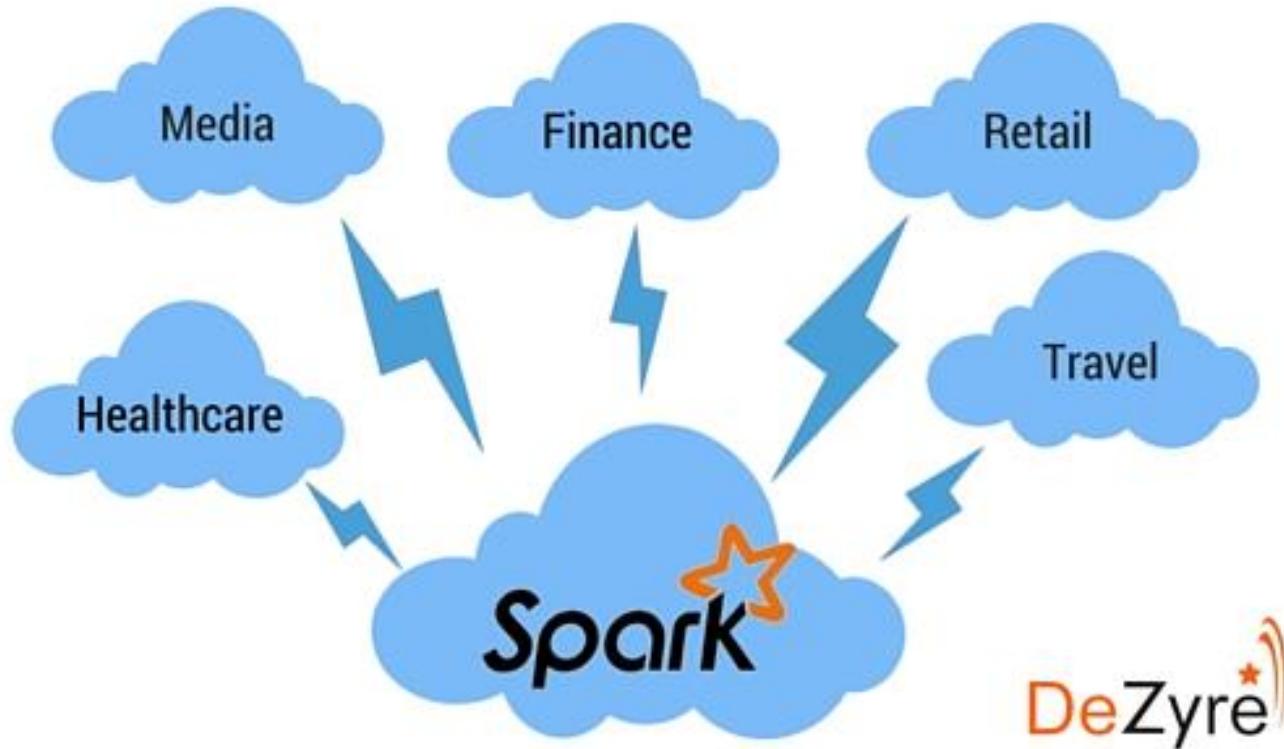
Vari utilizzi di Apache Spark

- Streaming
 - Streaming ETL (Extract, Transform, Load)
 - Data enrichment
 - Event detection (ad es. per individuare transizioni fraudolente, cambi potenzialmente pericolosi dei parametri vitali, ecc.)
 - Complex session analysis
- Machine learning
 - Marketing purposes and sentiment analysis
 - Network security
- Interactive analysis

<https://www.qubole.com/blog/apache-spark-u>

Computing... ad esempio?

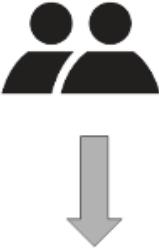
Vari utilizzi di Apache Spark



<https://www.dezyre.com/article/top-5-apache-spark->

Computing... ad esempio?

Test A/B



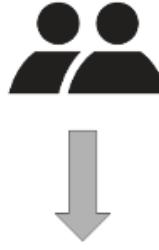
Project name Home About Contact Dropdown - Default Static top Fixed top

Welcome to our website

Click rate: 52 %

Learn more

Two user icons are positioned above a large downward-pointing arrow, which points to a screenshot of a website. The website has a navigation bar with 'Home' highlighted. The main content area displays a welcome message and a paragraph of placeholder text. A blue 'Learn more' button is at the bottom. Below the screenshot, the text 'Click rate: 52 %' is displayed.



Project name Home About Contact Dropdown - Default Static top Fixed top

Welcome to our website

Click rate: 72 %

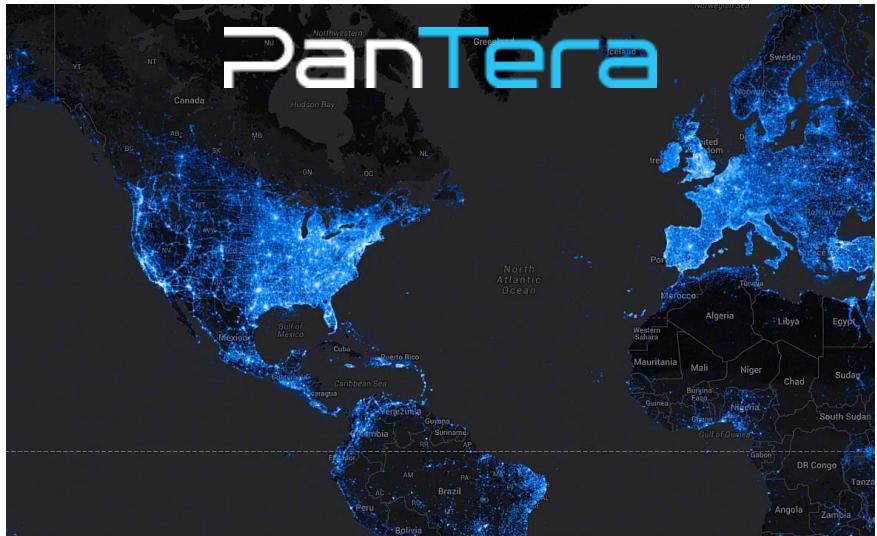
→ Learn more

Two user icons are positioned above a large downward-pointing arrow, which points to a screenshot of a website. The website has a navigation bar with 'Home' highlighted. The main content area displays a welcome message and a paragraph of placeholder text. A green '→ Learn more' button is at the bottom. Below the screenshot, the text 'Click rate: 72 %' is displayed.

Qualche esempio concreto



Jet Propulsion Laboratory
California Institute of Technology



YAHOO!

Rif: <https://spark.apache.org/powerd-by.html>

Qualche esempio concreto



Dal 2015 usa *Hadoop + Kafka + Spark (ecc.)*
100+ Petabytes con un minuto di latenza
10 000 job **Spark** / giorno

Cfr: <https://eng.uber.com/uber-big-data-platform/>



175 milioni di utenti mensilmente attivi
Test A/B
Kafka + Pinball + Spark Streaming + HBase & Memcached + Presto

Cfr: https://medium.com/@Pinterest_Engineering/building-a-new-experiment-pipeline-with-spark-streaming-and-kafka-4a2a2a2a2a2a



65 milioni di video stream / giorno

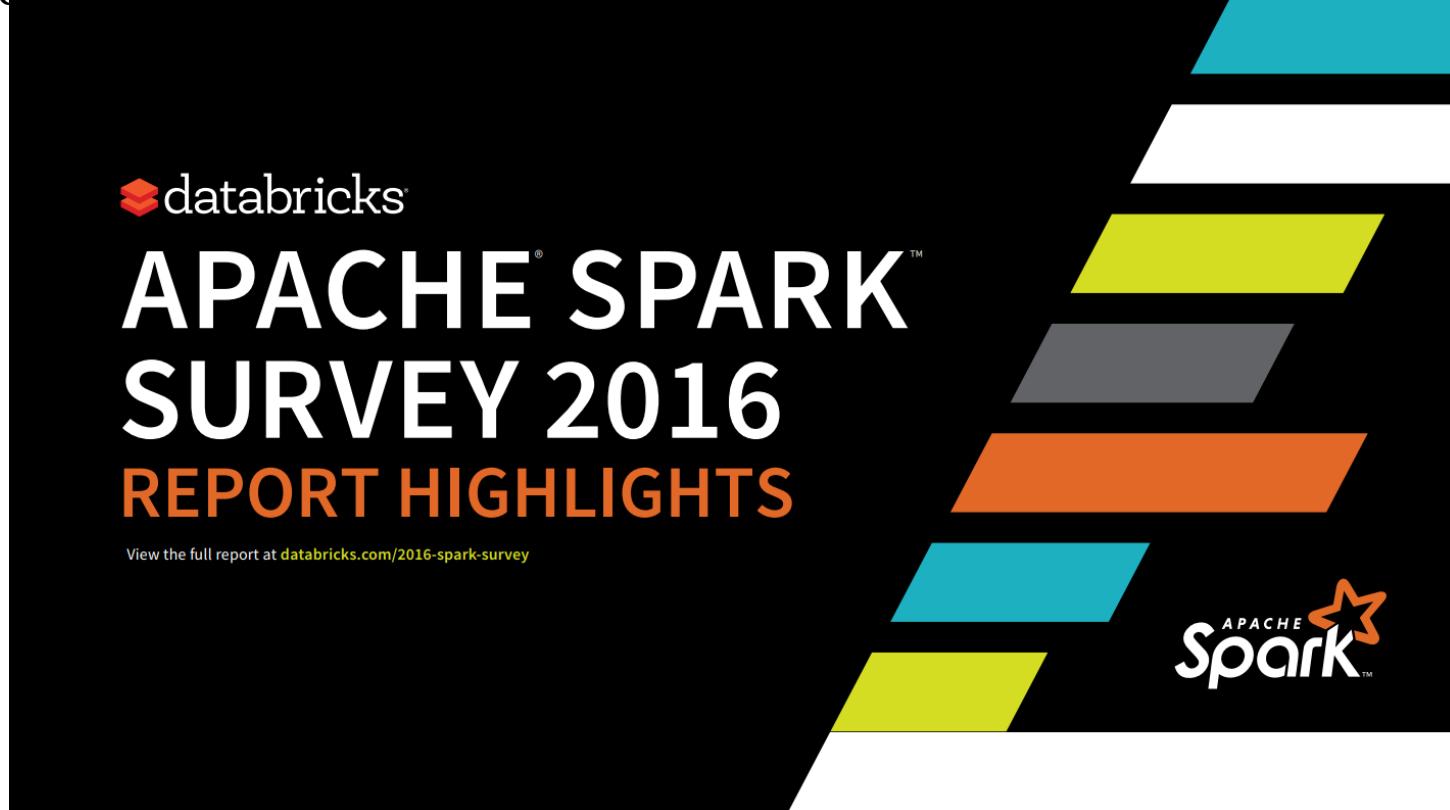
<http://ampcamp.berkeley.edu/wp-content/uploads/2012/06/dilip-joseph-amp-camp-2012-conviva-using-spark.pdf>

<https://www.qubole.com/blog/apache-spark-use-cases-for-video-processing/>

Chi usa Spark e perché

Nel 2016 i creatori di Spark hanno organizzato un sondaggio fra gli utenti.

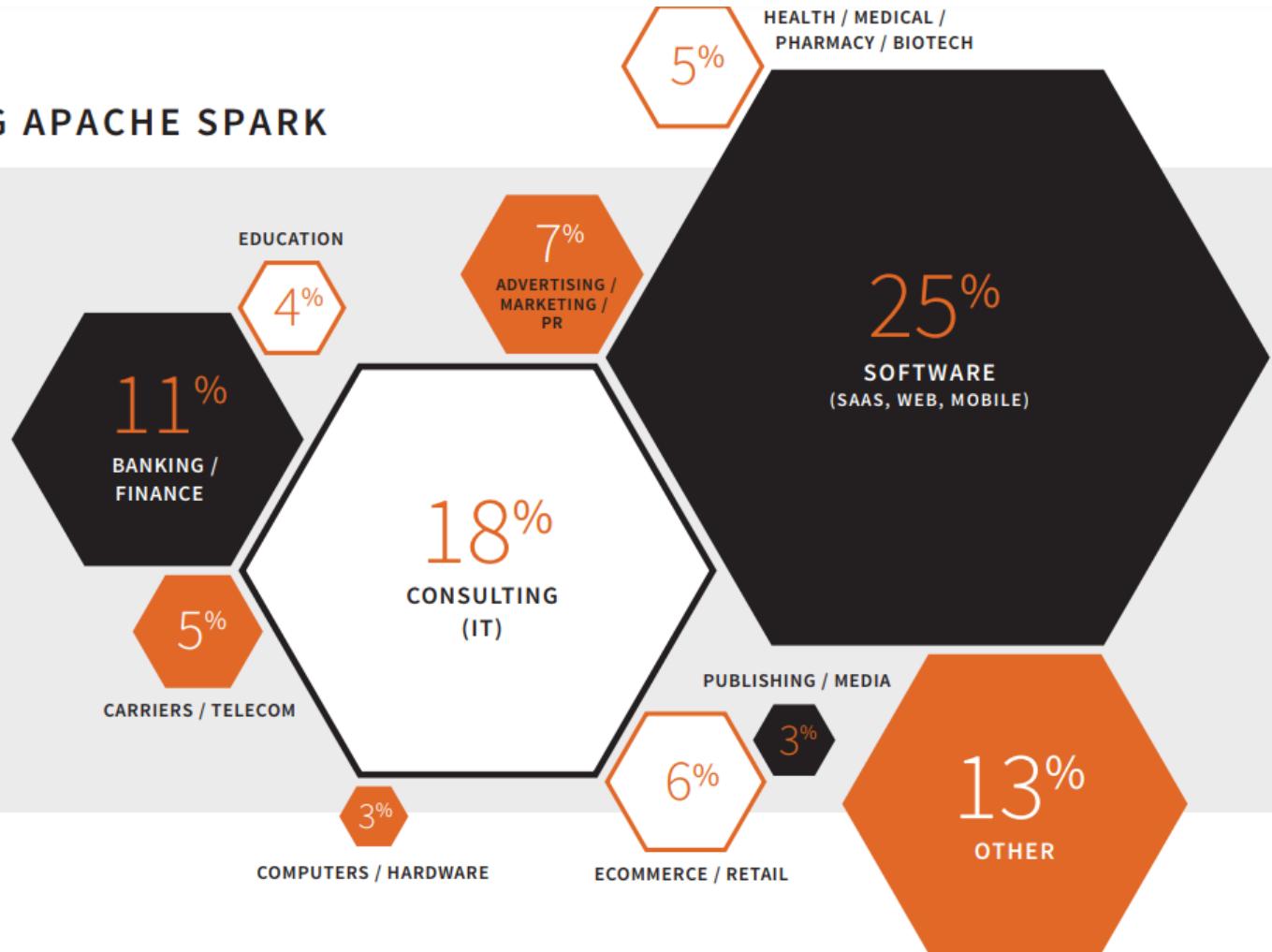
(1615 risposte da parte di più di 900 organizzazioni)



http://pages.databricks.com/rs/094-YMS-629/images/2016_Spark_Infographic.pdf

Chi usa Spark e perché

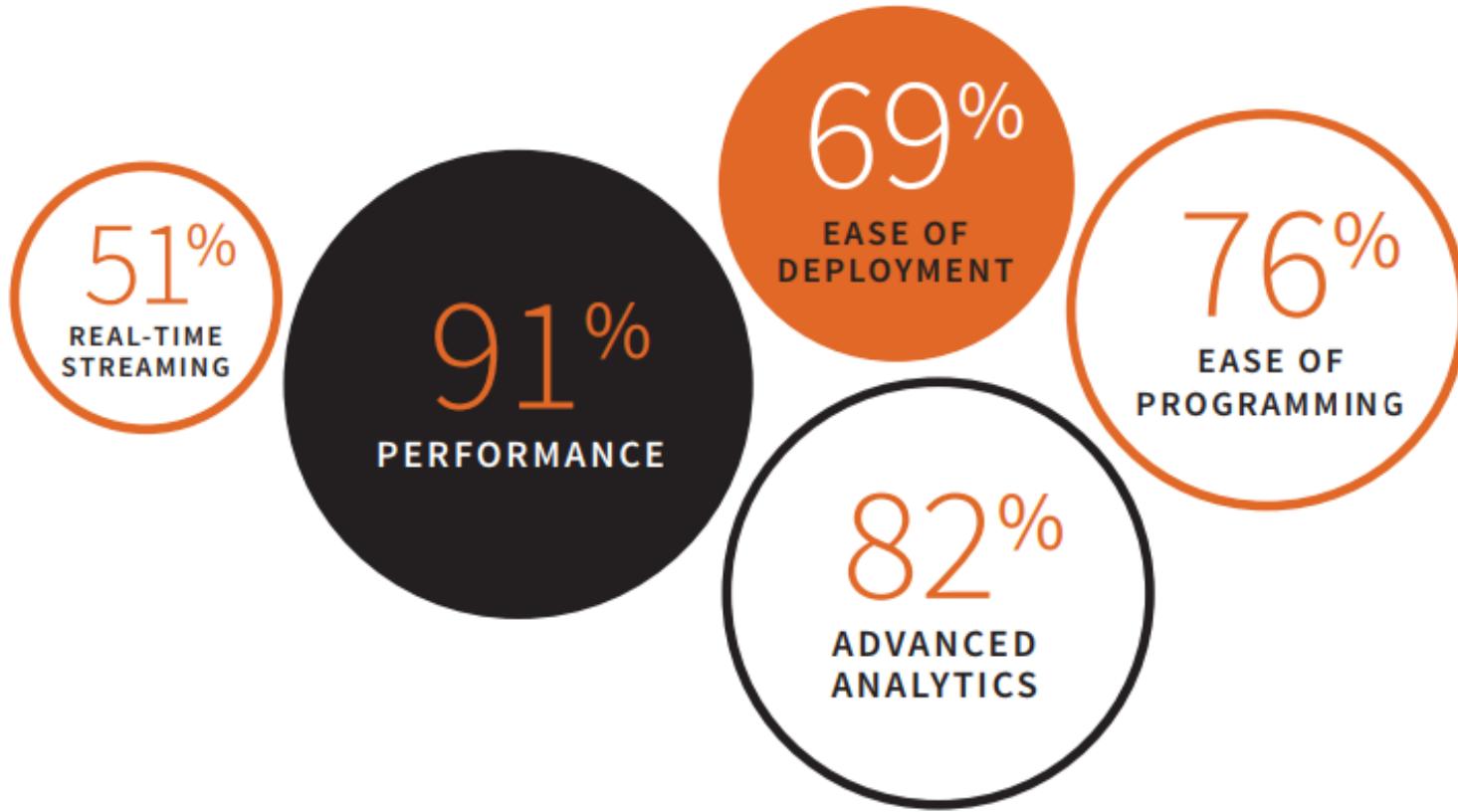
KEY INDUSTRIES USING APACHE SPARK



Chi usa Spark e perché

FEATURES USERS CONSIDER IMPORTANT

Respondents were allowed to select more than one feature.



Perché Spark?

Principali punti di forza

- Veloce
- Agnostico rispetto alla piattaforma
- Versatile
- Facile
- Capace di scalare
- Le «3 S» di Spark:
 - **Speed** = Velocità
 - **Simplicity** = Facilità di uso
 - **Support** Vari linguaggi e sistemi di storage;
Provider commerciali + comunità grande,
attiva e internazionale

<https://bigdata-madesimple.com/what-is-3ss-of-spark-and-its-effect-on-big-data/>

Punti di forza: Velocità

- Ingegnerizzato dal basso per ottenere le migliori performance
- 100x più veloce di *Hadoop MapReduce* grazie all'elaborazione in memoria (e altre ottimizzazioni)
- Veloce anche su disco.
- Record del mondo nel 2014 e 2016 per ordinamento su disco

New CloudSort Benchmark

Cost to sort 100TB of data



<https://databricks.com/blog/2016/11/14/setting-new-world-record-apache-spark-cloudsort-benchmark/>

Punti di forza: Facilità d'uso

- Approccio dichiarativo di alto livello ed API per agire su grandi datasets
- >100 operatori per trasformare data
- API basate su *DataFrame* e *Dataset* che permettono di manipolare facilmente dati strutturati e semi-strutturati

```
val errorCount = spark.read
    .text(fn)
    .filter($"value".like("%ERROR%"))
    .count()
```

Punti di forza: Facilità d'uso

WordCount in MapReduce (Java)

```
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class WordCount {

    public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        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);
            }
        }
    }

    public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
        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));
        }
    }

    public static void main(String[] args) throws Exception {
        JobConf conf = new JobConf(WordCount.class);
        conf.setJobName("WORDCOUNT");
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(Map.class);
        conf.setCombinerClass(Reduce.class);
        conf.setReducerClass(Reduce.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

WordCount in Spark (S

```
val file = spark.textFile("hdfs://...")
val counts = file
    .flatMap(line => line.split(" "))
    .map(word => (word, 1)).reduceByKey(_ + _)
counts.saveAsTextFile("hdfs://...")
```

Punti di forza: Facilità d'uso

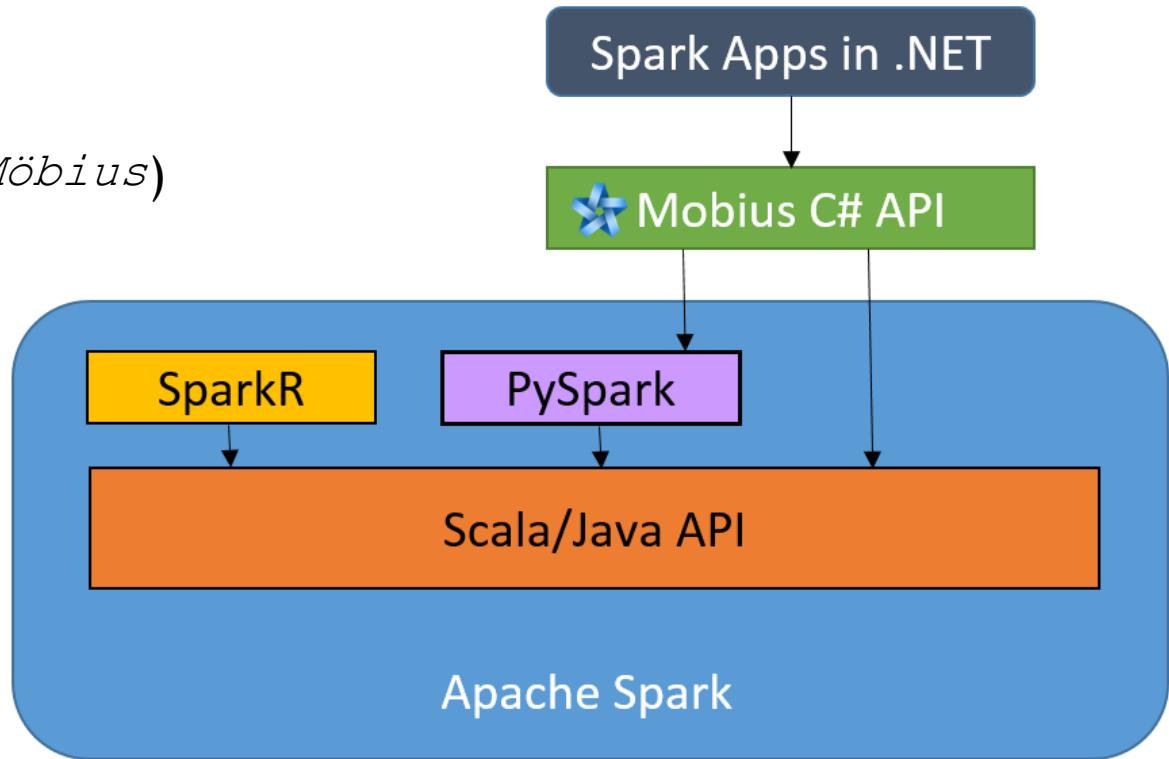
- Può essere eseguito interattivamente dalla shell o utilizzando un *NoteBook*
- Supporto per diversi linguaggi: Scala, Java, Python, R, SQL



E stanno arrivando altri linguaggi!

(progetti di terze parti)

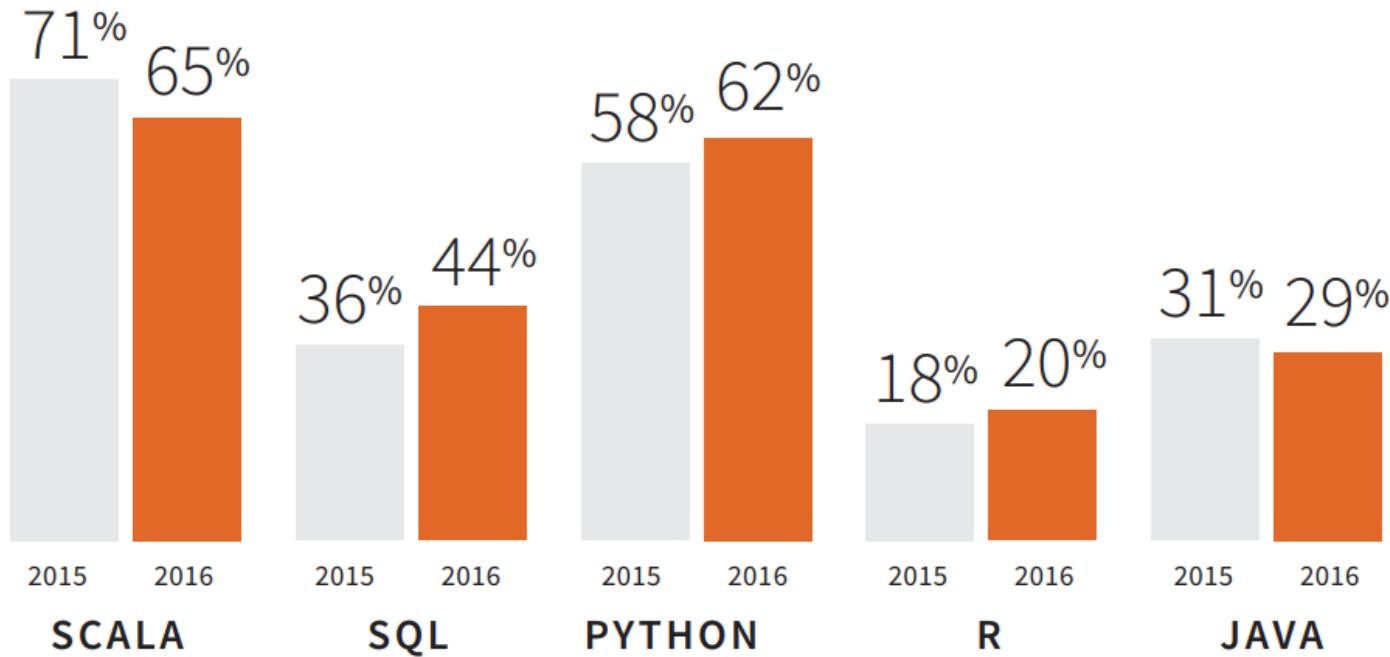
- *Clojure*
- *C#/.net* (progetto *Möbius*)
- *Groovy*
- *Julia*



Quale linguaggio usare?

LANGUAGES USED IN APACHE SPARK

Respondents were allowed to select more than one language.



http://pages.databricks.com/rs/094-YMS-629/images/2016_Spark_Infographic.pdf

Quale linguaggio usare?

Java



Java™

PRO:

- Potente e completo
- Molto diffuso, conosciuto e utilizzato: probabilmente il linguaggio più usato per i Big Data
- Alte prestazioni

CONTRO:

- Verboso
- Non supporta *Read-Evaluate-Print-Loop*

Quale linguaggio usare?

R



PRO:

- Il linguaggio di riferimento per Statistici e Analisti di Dati
- Buone librerie di visualizzazione

CONTRO:

- Non è considerato un linguaggio di utilizzo generale
- Prestazioni non alte
- C'è un certo ritardo nell'integrazione con Spark

Quale linguaggio usare?

SQL



PRO:

- Utilizzabile anche da non programmatori
- Trend di utilizzo in crescita

CONTRO:

- Inadatto per alcuni tipi di applicazioni

Quale linguaggio usare?

Python



PRO:

- Linguaggio molto noto e diffuso, anche nel mondo dei Big Data
- Facile da imparare
- Molto conciso ed espressivo
- Buone librerie di visualizzazione

CONTRO:

- Più lento
- «*Duck typing*». Controllo sui tipi a run-time

Quale linguaggio usare?

Scala



PRO:

- Spark è scritto in Scala: ogni feature del framework è immediatamente disponibile
- Compatibile con le librerie Java
- Controllo sui tipi durante la compilazione
- Alte prestazioni

CONTRO:

- Meno diffuso di Java o Python
- Marginalmente meno facile di Python

In questo corso useremo:



```
object HelloWorld {  
    def main(args: Array[String]): Unit = {  
        println("Hello, world!")  
    }  
}
```

Spark ha qualche punto debole

- Non ha un suo sistema di storage
- Grande consumo di risorse, in particolare la memoria ⇒ è dispendioso
- Problemi con dataset molto grandi
- Non ha supporto per il *real-time processing* (*Spark Streaming* usa micro-batches)
- Tende a generare tanti file relativamente piccoli. Questo è un male, ad es. per *Hadoop HDFS* o *Amazon S3*.
- Può richiedere ottimizzazione manuale
- Ha una latenza considerevole

Non solo Apache Spark!

Alcune alternative:



Apache Flink

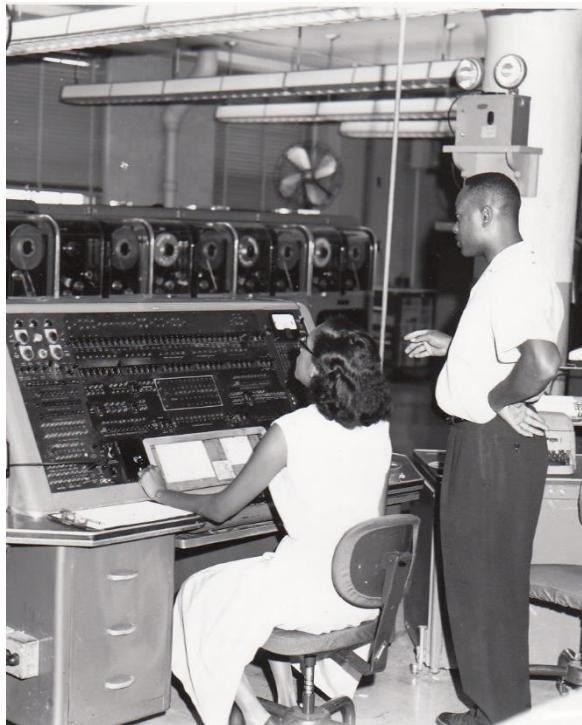


IBM InfoSphere Streams

Il contesto
(e un po' di
storia)

Contesto

La potenza di calcolo ha continuato a crescere ininterrottamente per ottant'anni



Univac I (1951)

~ 1 000 000\$

1905 operazioni al secondo
1 GFLOPS



Cray-1 (1976)

~ 10 000 000\$, 115 Kw < 1 000\$

iPad 2 (2011)

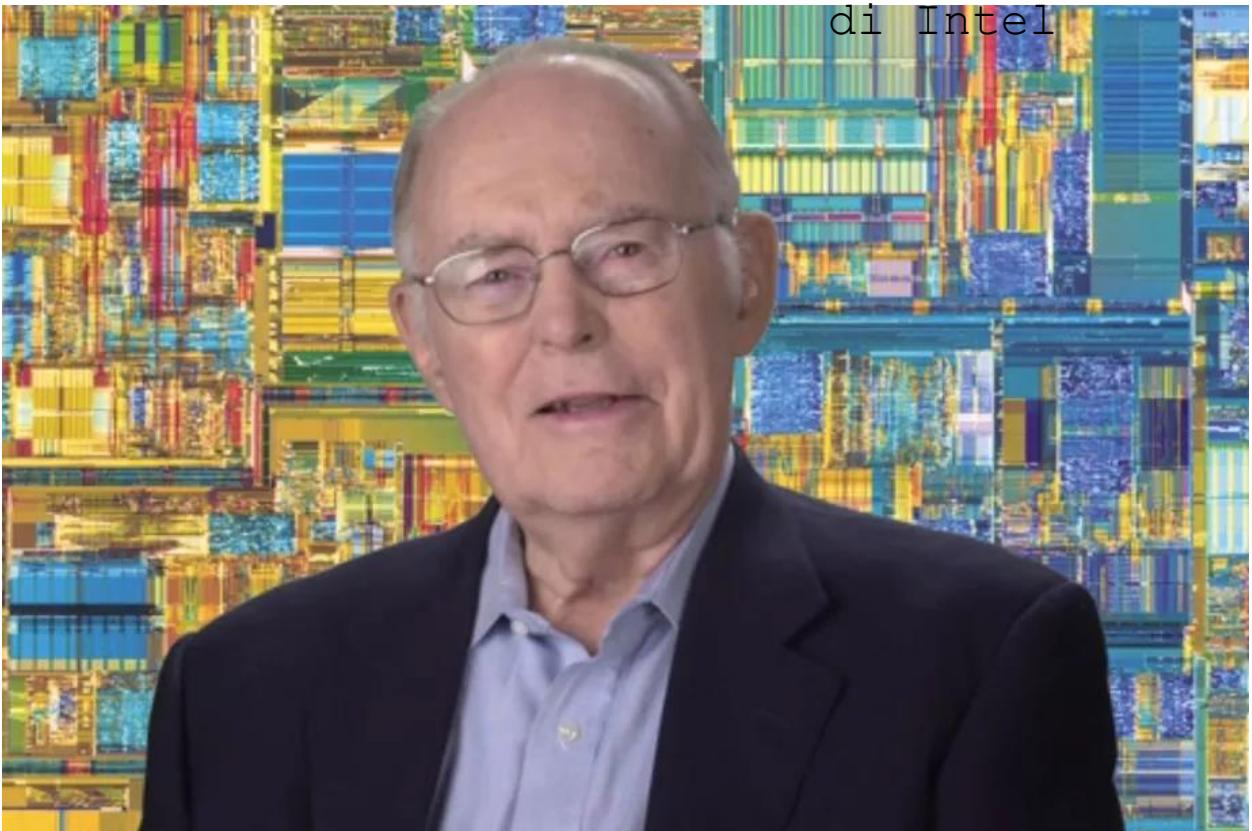
1.6 GFLOPS



La legge di Moore

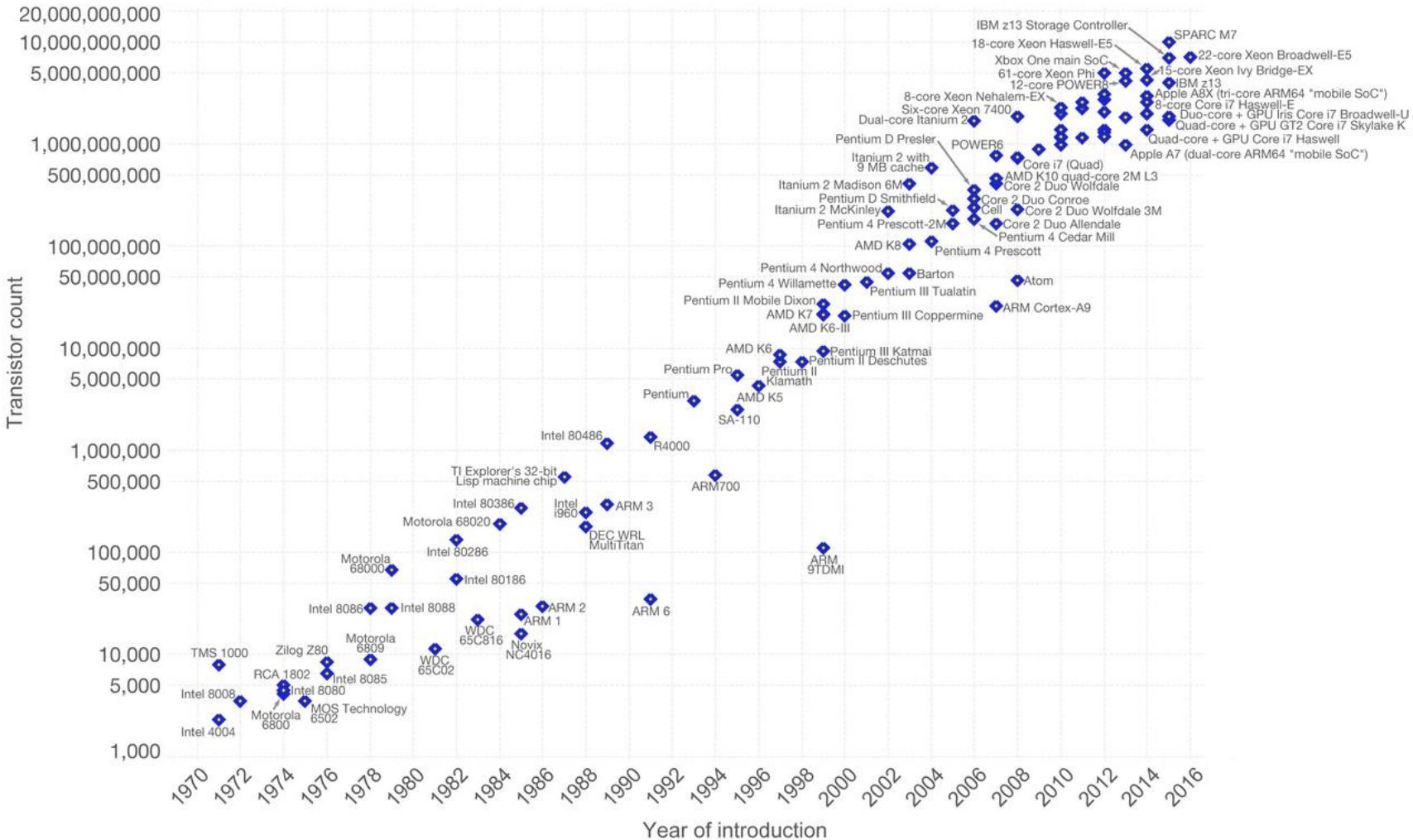


Gordon Earle Moore, nato nel 1929 fondatore



Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)

The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Licensed under CC-BY-SA by the author Max Roser.

La legge di Moore è ancora valida?

DEATH NOTICE

Long beloved by both engineers and computer scientists because of ongoing performance benefits ceaselessly and seemingly effortlessly achieved. From the age of fifty, Moore's Law began to age rapidly, passing into senescence and then, at the beginning of this month, into oblivion. Moore's Law leaves a thriving multi-trillion dollar global semiconductor and electronics industry, along with a growing set of questions about how that industry will survive its passing. In lieu of flowers, donations should be lavished on Intel shares.

Un'altra strategia per andare veloci: il parallelismo

Largest
Amazon EC2
instance: 128
virtual CPUs

Azul Systems Vega 3
Cores per chip: 54
Cores per system: 864

The Future is Parallel

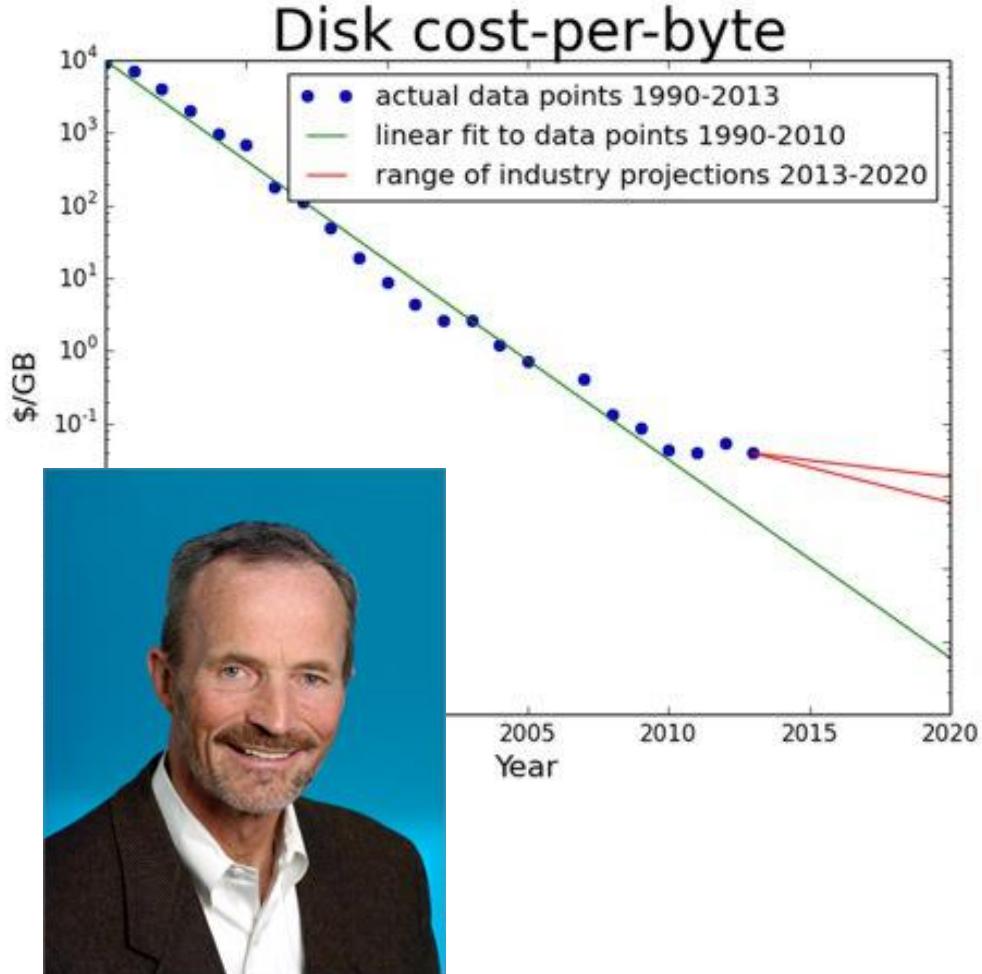
Intel Xeon
24 cores
48 threads

AMD
Opteron
16 cores

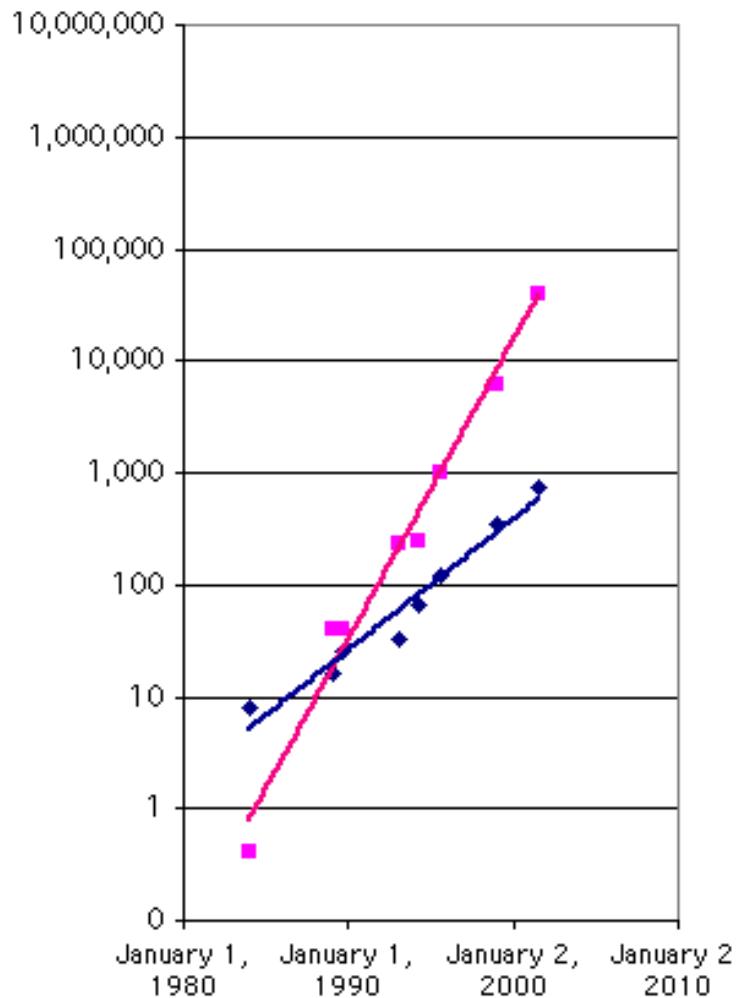
Tilera Gx-
3000
100 cores

La legge di Kryder

Le prestazioni dei dischi crescono più velocemente della potenza di calcolo
Legge di Moore (blu)



Legge di Kryder (viola)
VS.
Legge di Moore (blu)



Le prestazioni dei dischi crescono più velocemente della potenza di calcolo

Hardware Trends

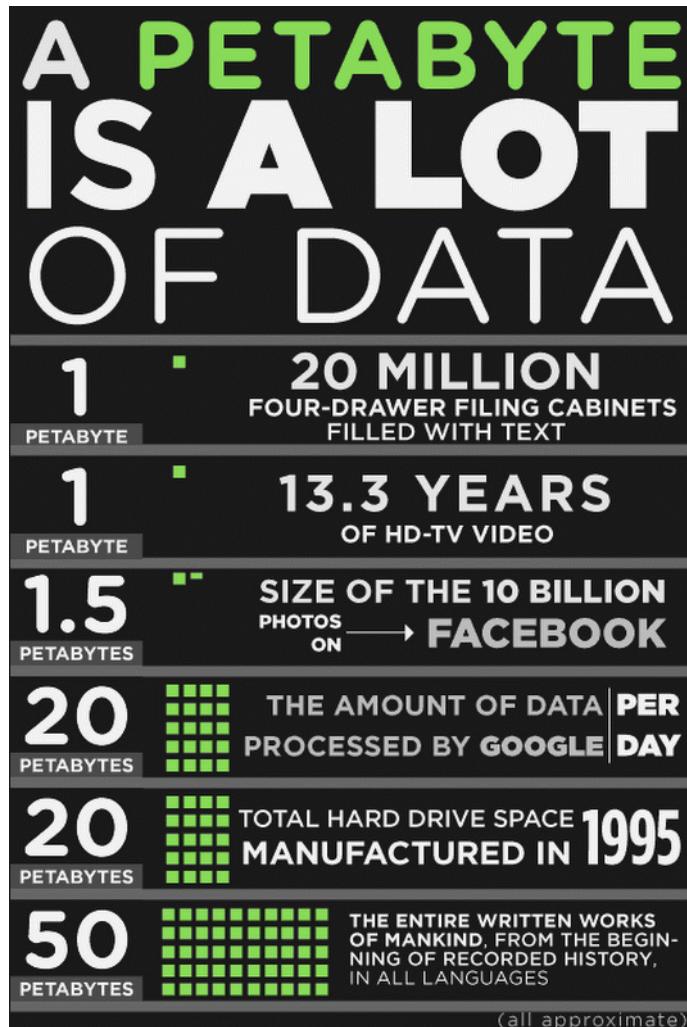
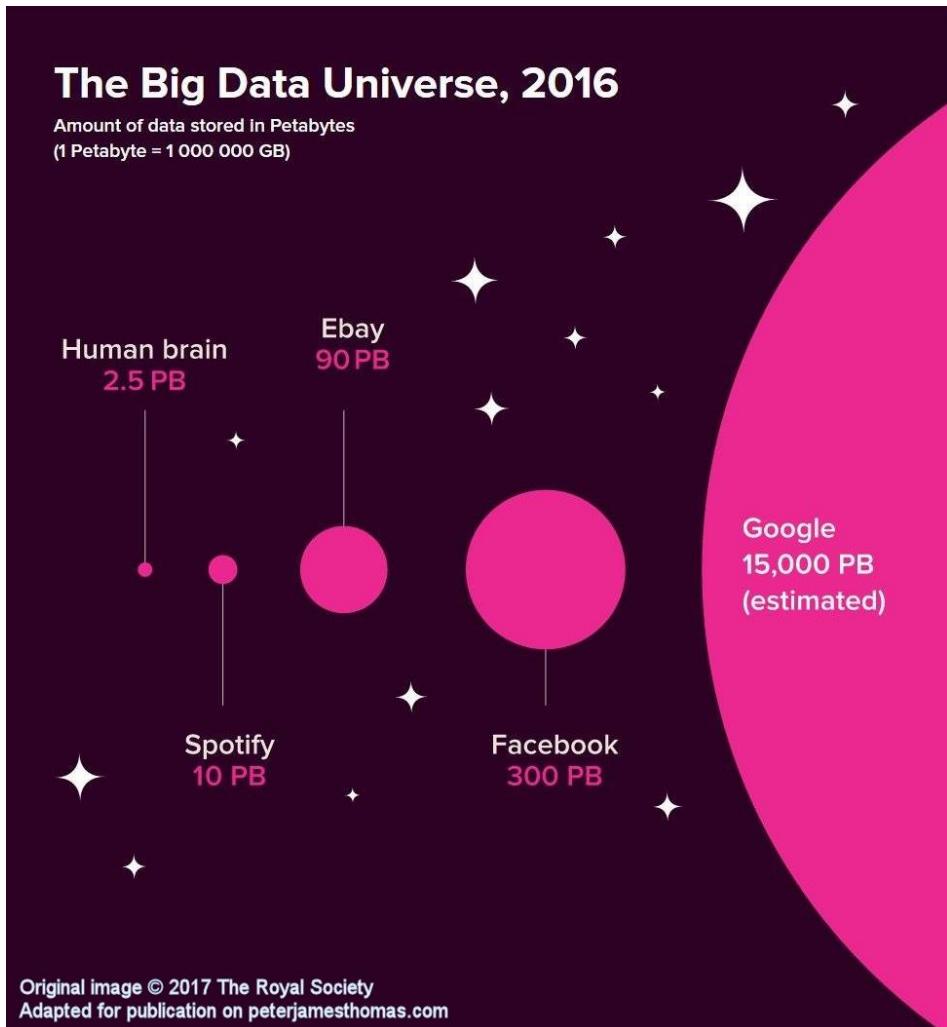
	2010	2015	
Storage	50+MB/s (HDD)	500+MB/s (SSD)	10X
Network	1Gbps	10Gbps	10X
CPU	~3GHz	~3GHz	:(

L'avvento dei BIG Data

BIG DATA:

- 500 milioni di tweets al giorno
- Traffico di quasi 2 GB/mese sui telefoni cellulari
- Amazon vende 600 prodotti al secondo
- 200 miliardi di email vengono scambiate ogni giorno
- MasterCard esegue 74 miliardi di transazioni ogni anno
- Ci sono quasi 6000 voli di linea al giorno.

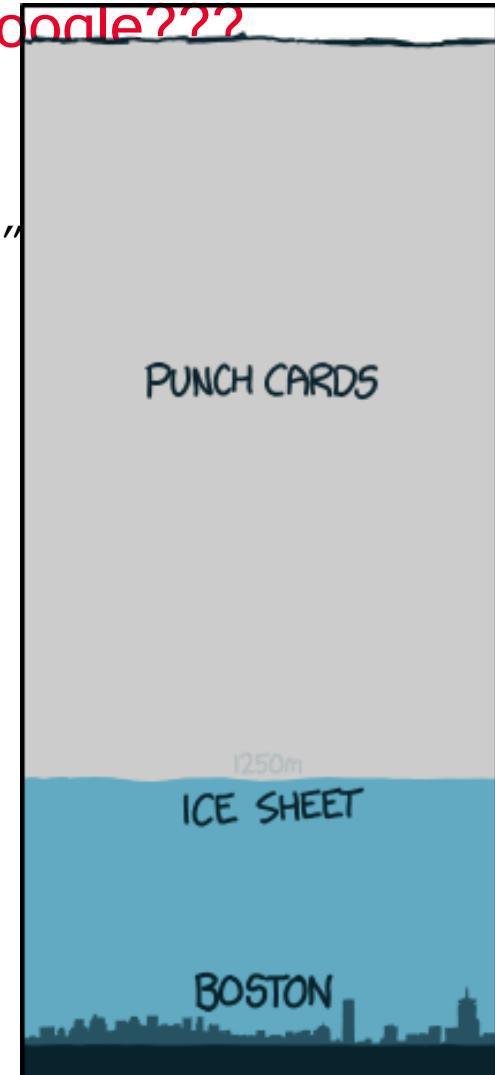
Quanto sono «*BIG*» i Big Data?



Intermezzo : aspetta... quanti dati ha Google???

“If all digital data were stored on punch cards, how big would Google’s data warehouse be?”

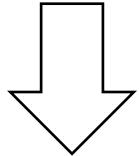
Randall Munroe



<https://blog.ted.com/using-serious-math-to-answer-weird-questions-randall-munroe-at-ted2014/>

Riassumendo

- La velocità di calcolo cresce, ma non tanto quanto vorremmo
- Lo storage cresce molto più velocemente
- Con uno storage economico e virtualmente illimitato la quantità di dati



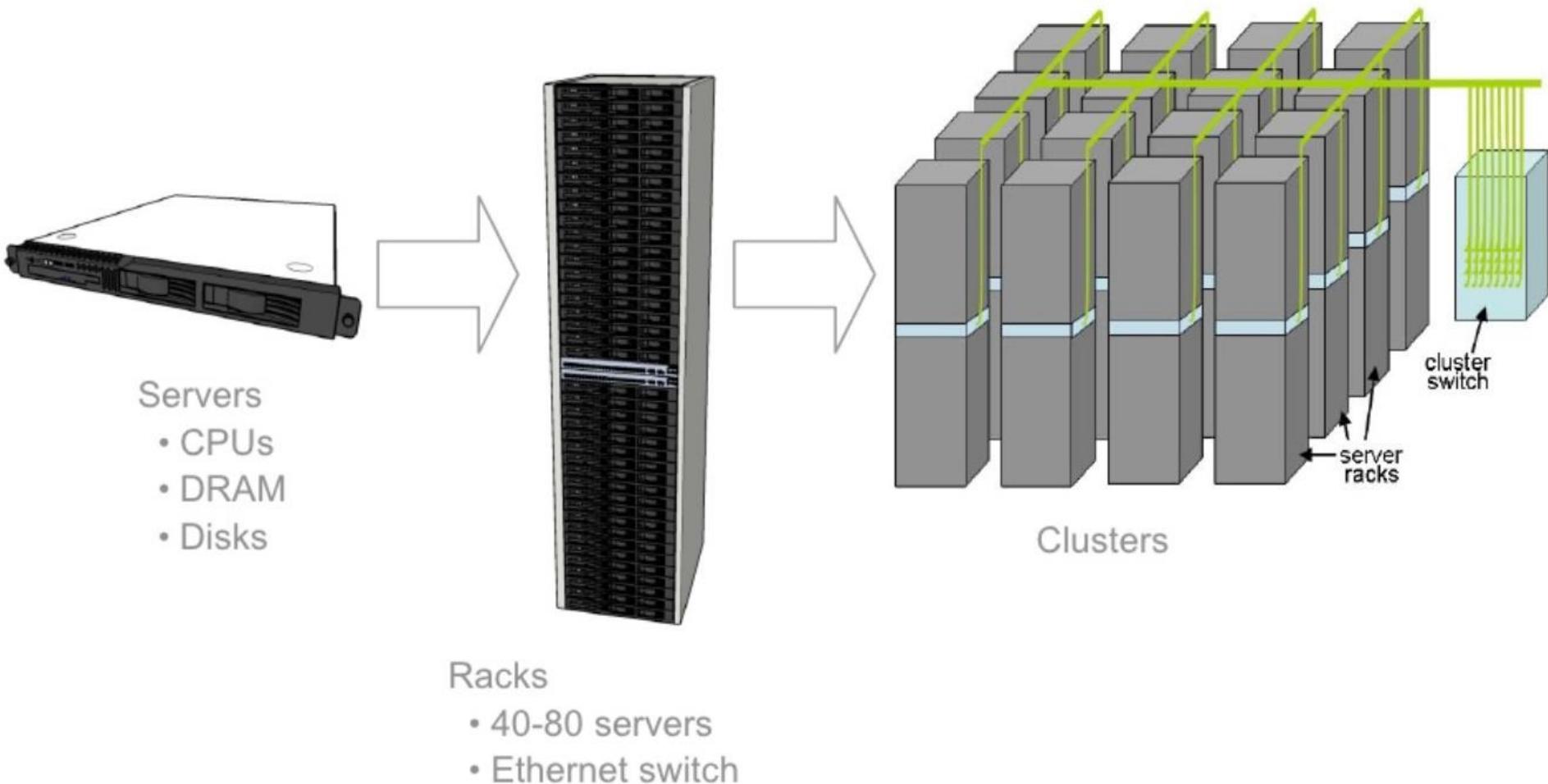
Il modo più efficace di analizzare questa enorme massa di dati è il *clustering*.

Esistono cluster con centinaia e anche migliaia di nodi!

Come si programma questa roba?



Come si programma questa roba?



Nel primo decennio del secolo Google pubblica tre articoli fondamentali

2003

GFS - Google File System

The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung
Google

ABSTRACT

We have designed and implemented the Google File System (GFS) to meet the rapidly growing demands of Google's distributed data processing system. GFS is a highly reliable system running on inexpensive commodity hardware, and it delivers high bandwidth and low latency access to large amounts of data.

While sharing many of the core goals as previous distributed file systems, our design has been driven by characteristics of Google's data processing needs. These requirements, both current and anticipated, reflect a marked departure from traditional distributed file system design. This has led us to reexamine traditional choices and explore radically different approaches.

The file system must not store transient data. It is used for the generation and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

In this paper, we present file system interface extensions designed to support distributed applications. Data may appear in multiple locations, and clients can move data from local micro-benchmarks and real world use.

Categories and Subject Descriptors

D.4.3 - Distributed file systems

General Terms

Design, reliability, performance, measurement

Keywords

Fault tolerance, scalability, data storage, clustered storage

The authors can be reached at the following addresses: {sanjay, gobi, shun}@csail.mit.edu; google.com.

Permission to make digital or hard copies of all or part of this work for personal use or internal distribution as you grant the provided copy are not made or distributed for profit or commercial advantage and the copy is not sold in whole or in part. Copyright © 2003 ACM 1581137375/03/0606-0001 \$15.00.
Copyright 2003 ACM 1581137375/03/0606-0001 \$15.00.

1. INTRODUCTION

We have designed and implemented the Google File System (GFS) to meet the rapidly growing demands of Google's distributed data processing system. GFS is a highly reliable system running on inexpensive commodity hardware, and it delivers high bandwidth and low latency access to large amounts of data. Our design has been driven by key observations of our application workloads and technological constraints, both current and anticipated. These requirements, both current and anticipated, reflect a marked departure from traditional distributed file system design. This has led us to reexamine traditional choices and explore radically different approaches.

The file system must not store transient data. It is used for the generation and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

In this paper, we present file system interface extensions designed to support distributed applications. Data may appear in multiple locations, and clients can move data from local micro-benchmarks and real world use.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

The file system interface extensions designed for distributed applications are the same as those designed for the general and processing of data used by our services as well as research and development efforts that require long-term data storage. Thus, we have required thousands of terabytes of storage across thousands of disks on over 1000 machines. We have also required high availability by hundreds of clients.

2004

MapReduce

MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat
jeff@google.com, sanja@google.com
Google, Inc.

Abstract

MapReduce is a programming model and an associated runtime system for processing large data sets. Users specify a map function to generate a set of intermediate key/value pairs, and a reduce function to aggregate those values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown in our benchmarks.

Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The system hides the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and performing bulk processing on disk. This allows programmers without any experience in parallel and distributed systems to easily express complex data analysis programs.

Our implementation of MapReduce runs on a large cluster of commodity machines and is highly scalable. It processes petabytes of data on thousands of machines. Programmers can use MapReduce to implement any algorithm that can be expressed as a sequence of map and reduce operations. We describe how MapReduce programs have been used to implement a variety of applications, ranging from web search to machine learning. We present the design and implementation of MapReduce, and discuss some lessons learned.

MapReduce is a simple and powerful tool for distributed data processing. It is well suited for applications that are amenable to a functional programming style, such as data mining and machine learning. We believe that MapReduce will find many uses in the future.

To appear in OSDI 2004

2006

BigTable

Bigtable: A Distributed Storage System for Structured Data

Jeff Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber
(to jeff@google.com, sanja@google.com, wilson@google.com, deborah@google.com, mburrows@google.com, tushar@google.com, andrew@google.com, rgruber@google.com)

Abstract

Bigtable is a distributed storage system for managing structured data. It is designed for data with very large size: petabytes of data across thousands of commodity servers. Many projects at Google store data in this form, including Google News, Google Images, Google Finance, and Google Books. These applications place very different demands on the underlying storage system. Bigtable allows clients to reason about the locality properties of the data represented in the underlying storage. Data is indexed by row and column, and can be updated incrementally. Bigtable also treats data as uninterpreted strings. Data can be read sequentially or in random order. Data can be partitioned and semi-structured into these strings. Clients can control the locality of their data through careful choice of row and column identifiers. We provide dynamic control over data layout and format, and we describe the design and implementation of Bigtable.

1 Introduction

Over the last two and a half years we have designed, implemented, and deployed a distributed storage system for managing structured data. This paper describes the design and implementation of Bigtable, and provides some initial measurements of its performance. Section 2 provides the fundamentals of the Bigtable implementation. Section 3 provides measurements of Bigtable's performance. We describe the design and implementation of Bigtable in Section 4, and discuss some lessons we learned in Section 5. Finally, we conclude with some future directions in Section 6.

2 Data Model

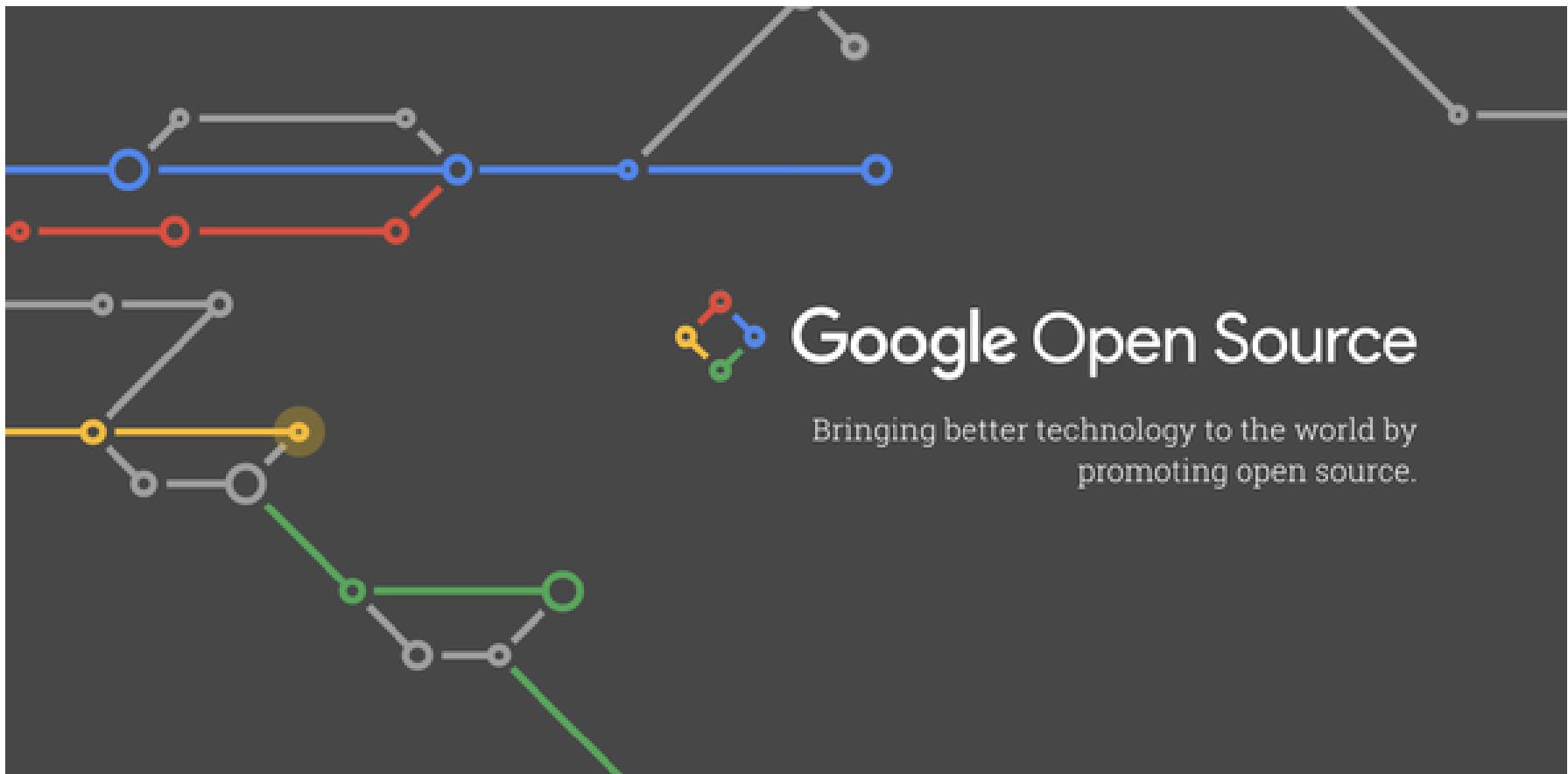
A Bigtable is a sparse, distributed, persistent multi-dimensional sorted map. The map is indexed by a row key, column key, and a timestamp; each value in the map is an uninterpreted array of bytes.

In this paper we describe the data model and many implementation strategies with databases, Parallel databases [14] and main-memory databases [13].

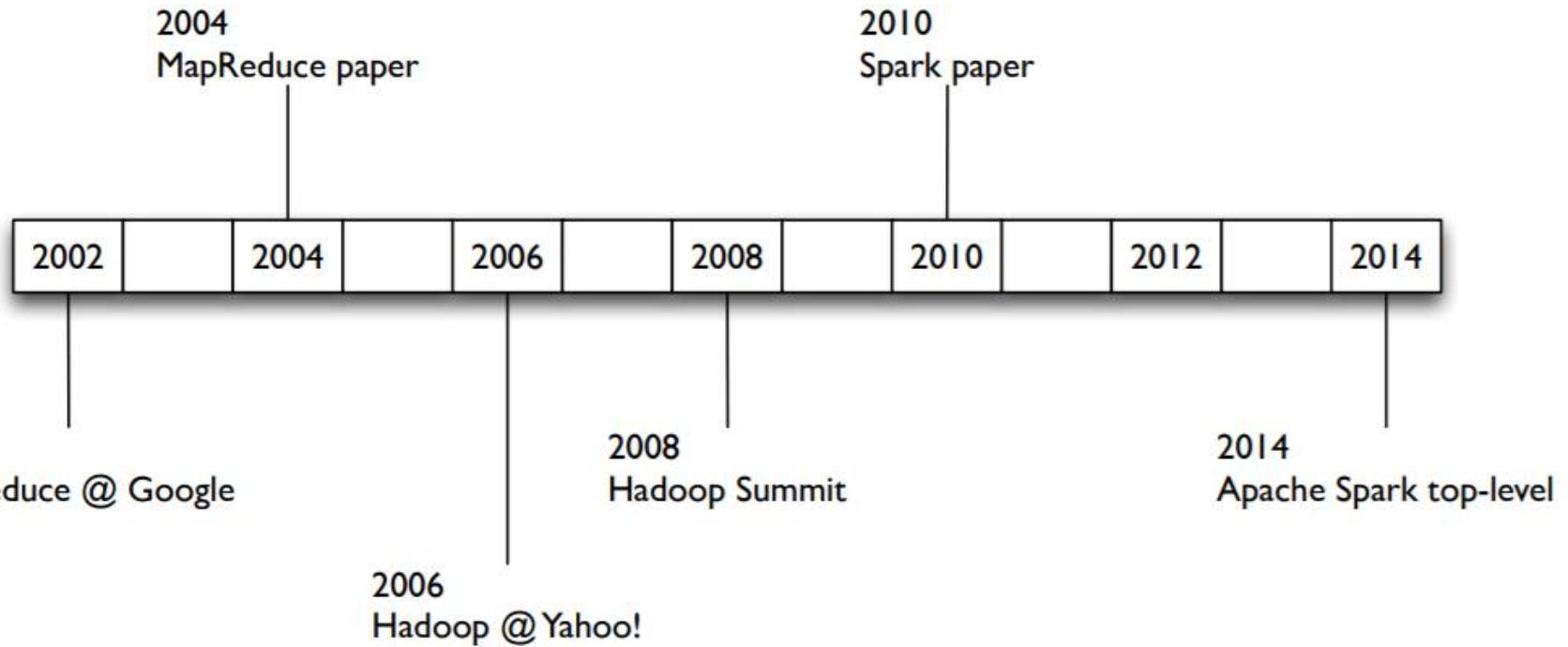
To appear in OSDI 2006

ma non pubblica il codice dell'implementazione

... poi (un po' in ritardo) abbraccia l'Open Source



Nel frattempo nascono Hadoop e Spark



Subito prima della nascita di Spark

Hadoop è la soluzione di riferimento per i Big Data:

- HDFS per lo storage e
- MapReduce per il calcolo

MapReduce ha delle limitazioni: è relativamente lento e difficile da programmare.

Inoltre gli utenti cominciano a richiedere:

- Algoritmi più complessi, basati su passi multipli
- Query ad-hoc interattive
- Processing in real time

Queste feature sono difficilmente implementabili in MapReduce

Subito prima della nascita di Spark

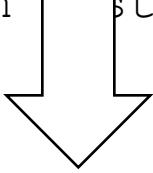
Per ovviare a queste limitazioni vengono sviluppati dei sistemi *specializzati*, uno per ogni tipo di esigenza:

- Pregel
- Giraph
- Dremel
- Drill
- Impala
- Presto
- Storm
- S4
- ...



Subito prima della nascita di Spark

Problemi con i sistemi specializzati

- Molti sistemi da controllare, configurare e installare
- Difficile combinarli fra loro anche se spesso servirebbe (ad es. leggere dati con SQL e poi usare degli algoritmi di machine learning)
⇒ In molti casi il trasferimento dei dati fra due sistemi diventa un  dominante

Spark nasce come un engine unificato

2009 : Nascita di Spark

Un gruppo di ricerca a UC Berkeley

Il gruppo lavora su un framework per il controllo di cluster che possa sostenere diversi tipi di cluster computing system. Spark nasce come esempio.

All'inizio focalizzato sul *machine learning*, poi diventa uno strumento generale



An Architecture for Fast and General Data Processing
on Large Clusters

Matei Zaharia



Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-12
<http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-12.html>

February 3, 2014

<https://www.kdnuggets.com/2015/05/interview-matei-zaharia-creator-apache-spark.html>

2009 : Nascita di Spark

Un gruppo di ricerca a UC Berkeley

Due idee chiave:

- Tenere i risultati intermedi in memoria
- Fare uno strumento generale che possa essere utilizzato in contesti differenti (es. streaming e graph processing)

Nel 2013 si muove in Apache

Nel 2014 diventa un *Top-Level Project* (TLP)

<https://www.kdnuggets.com/2015/05/interview-matei-zaharia-creator-apache-spark.html>



An Architecture for Fast and General Data Processing
on Large Clusters

Matei Zaharia



Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-12
<http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-12.html>

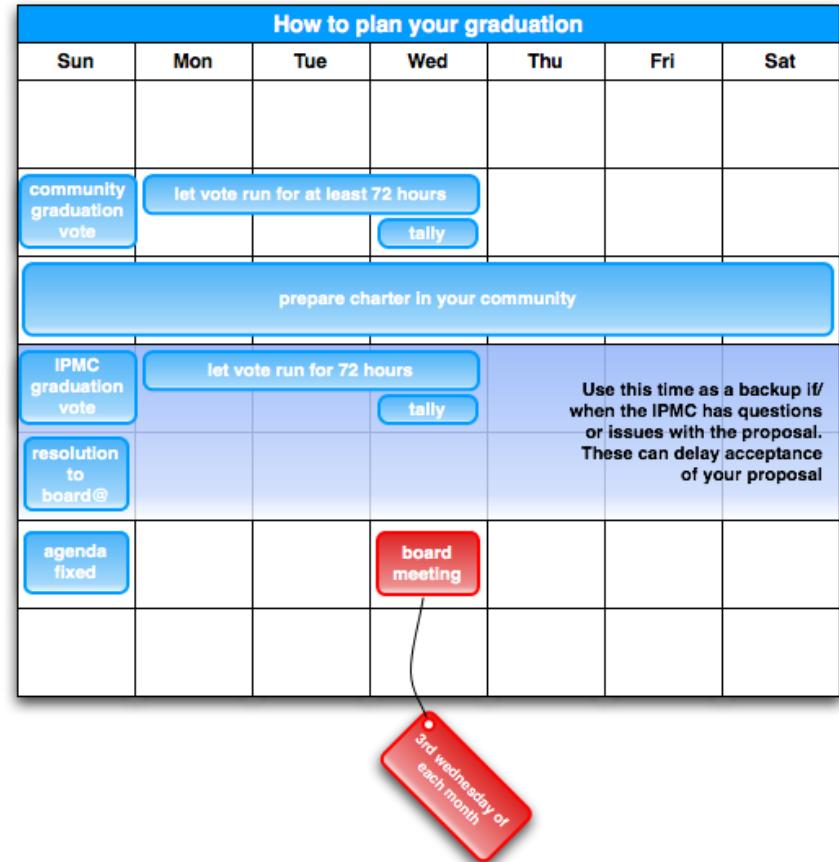
February 3, 2014

2014: Spark è Apache Top Level Project



Diventare TPL è un processo complicato e laborioso.

Quando un progetto diventa TLP può contare su una comunità grande e attiva e su una grande visibilità



Apache is a well recognized brand



GOVERNOR ARNOLD SCHWARZENEGGER

November 5, 2009

Apache Software Foundation

It is a great pleasure to extend my greetings to all those attending ApacheCon and congratulations on your tenth anniversary.

I applaud your incredible work over the past decade and appreciate you choosing California as the place to celebrate this fantastic milestone. Our state is a land of innovation, and you have likewise fostered great technological advancements that have touched the lives of millions of people around the world.

Whether managing financial systems, positioning satellites or powering websites through the Apache HTTP Server, your open source projects play key roles in making our information age possible. Thank you for your extraordinary accomplishments and commitment to discovery.

On behalf of all Californians, I send my gratitude to everyone in attendance for your participation, and I offer my best wishes for a rewarding conference and continued success.

Sincerely,

Arnold Schwarzenegger

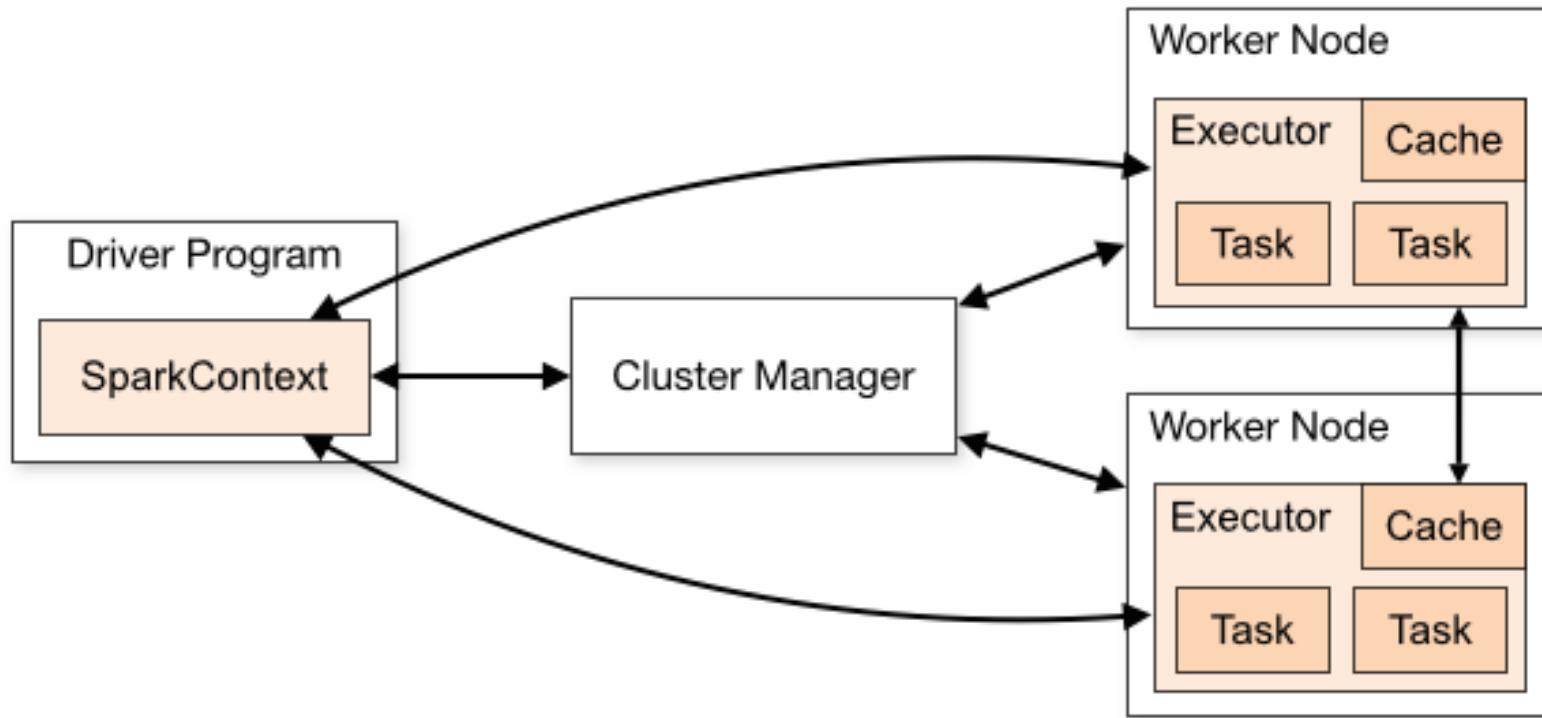
Big Data Transformation

Spark Fundamentals & Query Fundamentals

Spark,
com'è fatto?

Struttura di un programma Spark

L'elaborazione è distribuita su più nodi



L'ecosistema di Spark

Cluster manager



Storage



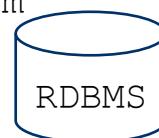
amazon
S3



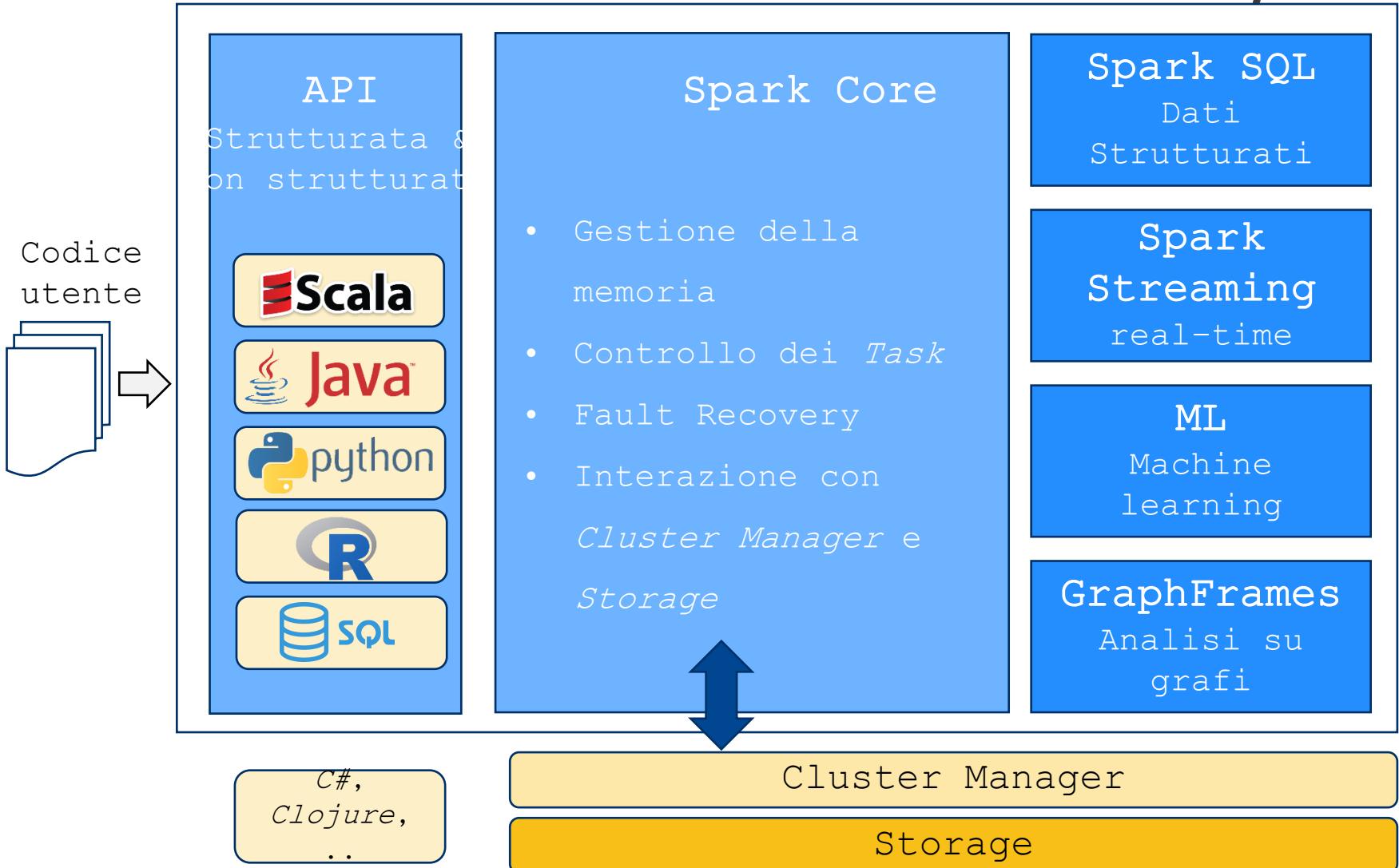
cassandra



Google Cloud Storage



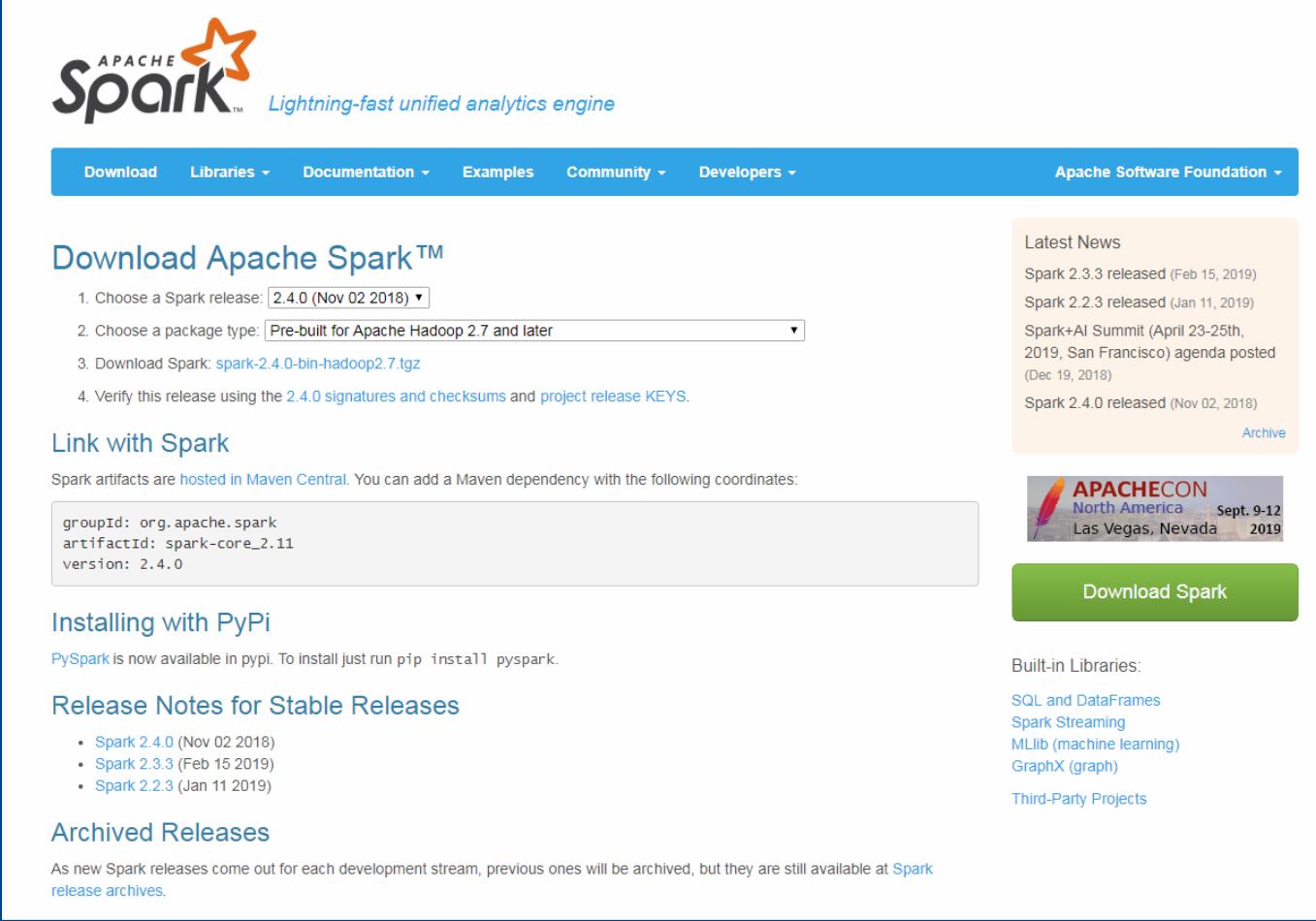
L'ecosistema di Spark



In pratica?

Per fare degli esperimenti a casa

<https://spark.apache.org/downloads.html>



The screenshot shows the Apache Spark download page. At the top, there's the Apache Spark logo with the tagline "Lightning-fast unified analytics engine". Below the logo is a navigation bar with links for Download, Libraries, Documentation, Examples, Community, Developers, and Apache Software Foundation. The main content area has a heading "Download Apache Spark™" and a numbered list of steps:

- Choose a Spark release: 2.4.0 (Nov 02 2018) ▾
- Choose a package type: Pre-built for Apache Hadoop 2.7 and later
- Download Spark: spark-2.4.0-bin-hadoop2.7.tgz
- Verify this release using the 2.4.0 signatures and checksums and project release KEYS.

Below this, there's a section titled "Link with Spark" with Maven dependency coordinates:

```
groupId: org.apache.spark  
artifactId: spark-core_2.11  
version: 2.4.0
```

There's also a section for "Installing with PyPi" and "Release Notes for Stable Releases" which lists releases from Nov 02 2018 to Jan 11 2019. A "Archived Releases" section notes that previous versions are archived. To the right, there's a "Latest News" sidebar with links to recent releases, an "Archive" link, an "APACHECON North America" event card for Sept. 9-12, 2019 in Las Vegas, Nevada, and a large green "Download Spark" button.

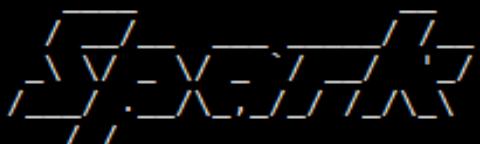
Per fare degli esperimenti a casa

Spark gira anche su un singolo computer

Lo stesso codice funziona su un cluster di 1000 nodi!

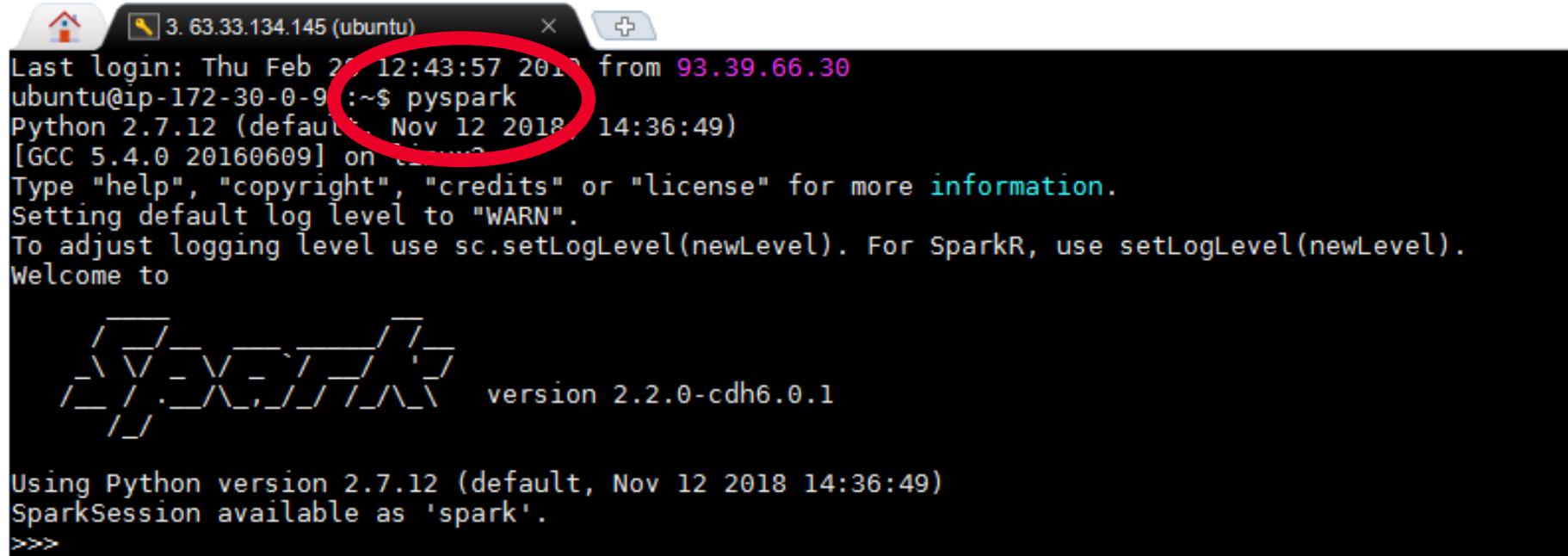
Come useremo Spark nel laboratorio di oggi?

La Shell di Spark

```
2. 63.33.134.145 (ubuntu) +  
Last login: Fri Feb 22 08:34:07 2019 from 93.39.66.30  
ubuntu@ip-172-30-0-97:~$ spark-shell  
Setting default log level to "WARN"  
To adjust logging level use setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
Spark context Web UI available at http://172.30.0.97:4040  
Spark context available as 'sc' (master = yarn, app id = application_1550659209716_0030).  
Spark session available as 'spark'.  
Welcome to  
 version 2.2.0-cdh6.0.1  
Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_141)  
Type in expressions to have them evaluated.  
Type :help for more information.  
scala>
```

Come useremo Spark nel laboratorio di oggi?

Si può usare anche Python (pyspark)



```
Last login: Thu Feb 21 12:43:57 2019 from 93.39.66.30
ubuntu@ip-172-30-0-9:~$ pyspark
Python 2.7.12 (default, Nov 12 2018 14:36:49)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to

    \_____
   /       \
  /  _    _ \
 /  /\  /\ \
/  /  \  \_ \
 \  \  /  \  \
  \  \  /  \  \
   \  \  /  \  \
    \  \  /  \  \
     \  \  /  \  \
      \  \  /  \  \
       \  \  /  \  \
        \  \  /  \  \
         \  \  /  \  \
          \  \  /  \  \
           \  \  /  \  \
            \  \  /  \  \
             \  \  /  \  \
              \  \  /  \  \
               \  \  /  \  \
                \  \  /  \  \
                 \  \  /  \  \
                  \  \  /  \  \
                   \  \  /  \  \
                    \  \  /  \  \
                     \  \  /  \  \
                      \  \  /  \  \
                       \  \  /  \  \
                        \  \  /  \  \
                         \  \  /  \  \
                          \  \  /  \  \
                           \  \  /  \  \
                            \  \  /  \  \
                             \  \  /  \  \
                              \  \  /  \  \
                               \  \  /  \  \
                                \  \  /  \  \
                                 \  \  /  \  \
                                  \  \  /  \  \
                                   \  \  /  \  \
                                    \  \  /  \  \
                                     \  \  /  \  \
                                      \  \  /  \  \
                                       \  \  /  \  \
                                        \  \  /  \  \
                                         \  \  /  \  \
                                          \  \  /  \  \
                                           \  \  /  \  \
                                            \  \  /  \  \
                                             \  \  /  \  \
                                              \  \  /  \  \
                                               \  \  /  \  \
                                                \  \  /  \  \
                                                 \  \  /  \  \
                                                  \  \  /  \  \
                                                   \  \  /  \  \
                                                    \  \  /  \  \
                                                     \  \  /  \  \
                                                      \  \  /  \  \
                                                       \  \  /  \  \
                                                        \  \  /  \  \
                                                         \  \  /  \  \
                                                          \  \  /  \  \
                                                           \  \  /  \  \
                                                            \  \  /  \  \
                                                             \  \  /  \  \
                                                              \  \  /  \  \
                                                               \  \  /  \  \
                                                                \  \  /  \  \
                                                                 \  \  /  \  \
                                                                  \  \  /  \  \
                                                                   \  \  /  \  \
                                                                    \  \  /  \  \
                                                                     \  \  /  \  \
                                                                      \  \  /  \  \
                                                                       \  \  /  \  \
                                                                        \  \  /  \  \
                                                                         \  \  /  \  \
                                                                          \  \  /  \  \
                                                                           \  \  /  \  \
                                                                            \  \  /  \  \
                                                                             \  \  /  \  \
                                                                              \  \  /  \  \
                                                                               \  \  /  \  \
                                                                                \  \  /  \  \
                                                                                 \  \  /  \  \
                                                                                  \  \  /  \  \
                                                                                   \  \  /  \  \
                                                                                   version 2.2.0-cdh6.0.1

Using Python version 2.7.12 (default, Nov 12 2018 14:36:49)
SparkSession available as 'spark'.
>>>
```

Come useremo Spark nel laboratorio di oggi?

Zeppelin Notebook

Basic Features (Spark) - Zeppelin - Mozilla Firefox

Cloudera Live : Welcome... Basic Features (Spark) +

localhost:8080/#/notebook/2A94M5J1Z

Cloudera Zeppelin Hue Hadoop HBase Impala Spark Solr Oozie Cloudera Manager >

```
import sqlContext.implicits._  
import org.apache.commons.io.IOUtils  
import java.net.URL  
import java.nio.charset.Charset  
bankText: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[28] at parallelize at <console>:24  
defined class Bank  
bank: org.apache.spark.sql.DataFrame = [age: int, job: string ... 3 more fields]
```

Took 5 sec. Last updated by anonymous at February 19 2019, 4:25:13 AM. (outdated)

sql select age, count(1) value from bank where age < 30 group by age order by age

maxAge 35

marital single

value

103 231 105

20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105

20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105

19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69

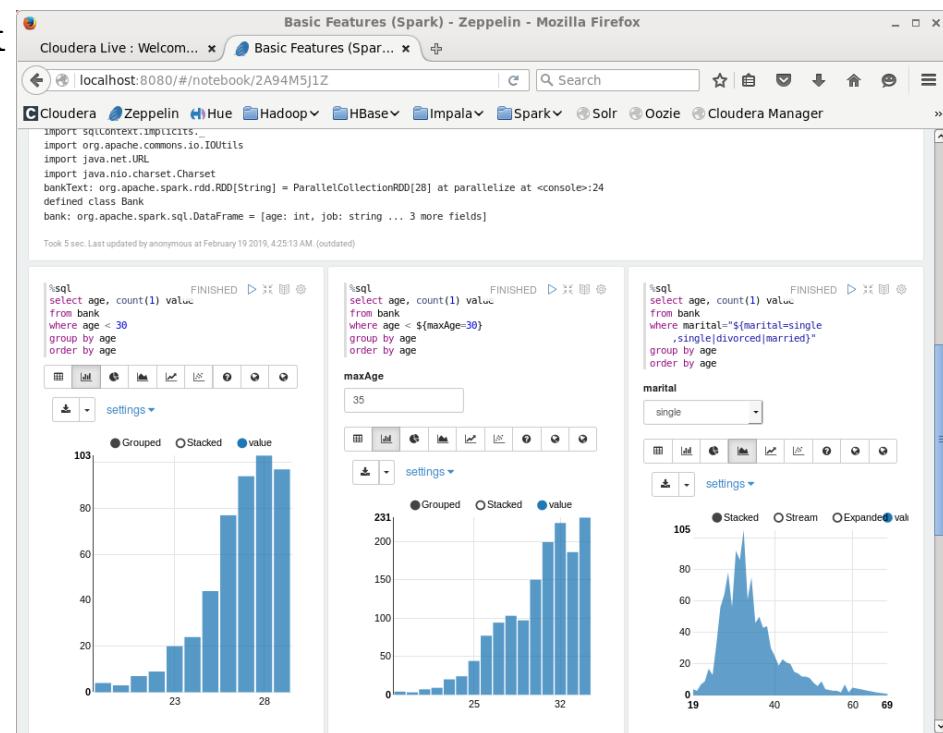
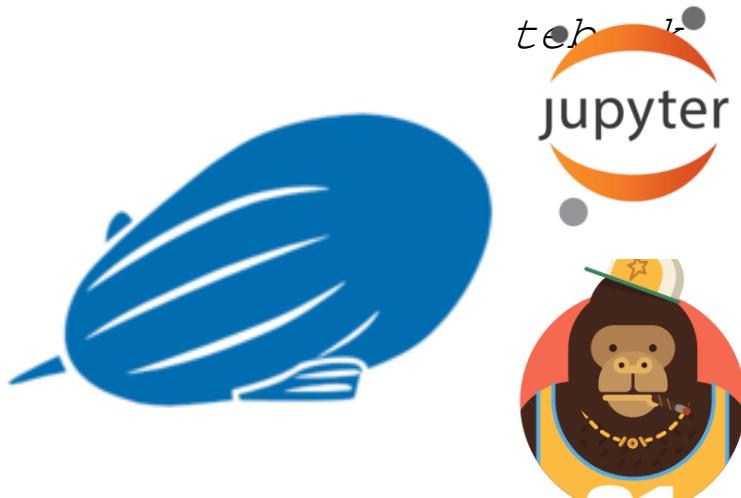
I Notebook

Un ambiente per il «literate programming»

Un'interfaccia che permette di mescolare frammenti eseguibili di programma con grafici, testo e altri contenuti.

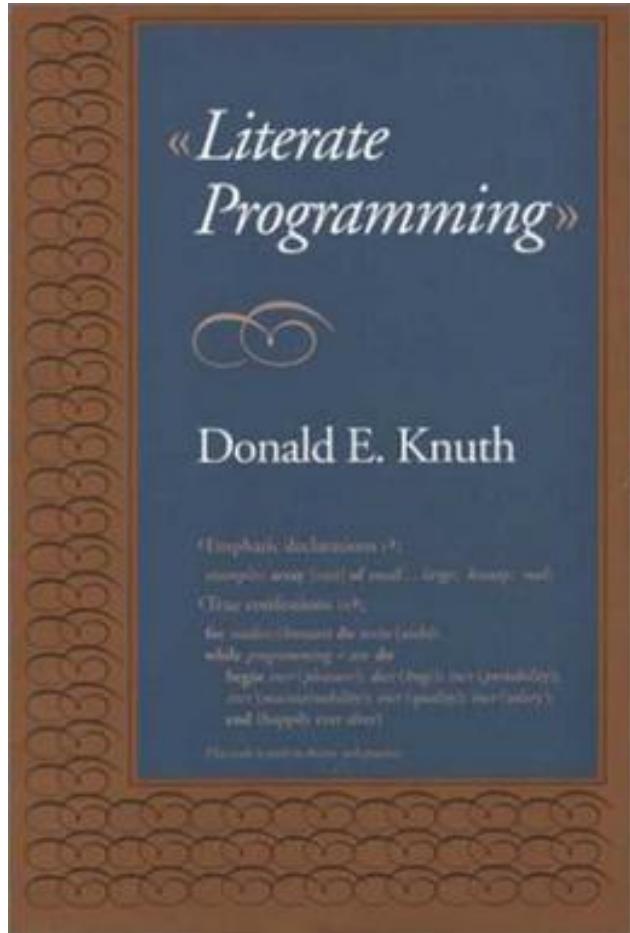
I notebook per Apache Spark

- *Jupyter Notebook*
- *Apache Zeppelin*



Literate programming

Donald E. Knuth



Laboratorio

Obiettivi per questa mattina

- Ripasso di Scala
- Prendere familiarità con *Zeppelin Notebook*



Scala Cheat Sheet

variances

Given $T <: P$	
When C is covariant then $C[T] <: C[P]$	
When C is contravariant then $C[T] >: C[P]$	
class C[+T]	C is covariant
class C[-T]	C is contravariant
class C[T]	C is invariant

variables and functions

val x = 1	constant, evaluated immediately
var y = 2	variable
def z = 2	evaluated when called
lazy val w = 2	evaluated once when needed
def f(x: Int) = { x*x }	define function
def f(x: R)	call by value
def f(x: => R)	call by name (lazy parameters)
(1 to 5).map(_*2)	anonymous function
(1 to 5).map(2*)	(positionally matched arguments)
(1 to 5).map(2*)	anonymous function
(1 to 5).map(x => x*x)	(bound infix method)
def mapmake[T](g:T=>T)(seq: List[T]) =	anonymous function
seq.map(g)	(to use an arg twice, have to name it)
def sum(args: Int*) = args.reduceLeft(_+_)	generic type
varargs	

control constructs

if (condition) this else that	if then else
while (x < 5) { println(x); x += 1}	while loop
do { println(x); x += 1 } while (x < 5)	do while loop
for (x <- xs if x%2 == 0) yield x*10	for comprehension
for (i <- 1 to 5) { println(i) }	for loop

object orientation

class C(x: R)	class with private param
class C(val x: R)	class with public param
new { ... }	anonymous class
abstract class D { ... }	abstract class
class C extends D { ... }	inheritance
class C(x: R) extends D(x)	inheritance and constructor params
object O extends D { ... }	singleton
trait T { ... }	interfaces with implementation
class C extends D with T	using multiple traits

gutefrage.net

Zeppelin Notebook

Manuale online

The screenshot shows two side-by-side browser windows. The left window displays the 'Install' page of the Apache Zeppelin documentation, featuring a sidebar with 'Requirements' and a main content area with a welcome message and a 'Requirements' section. The right window displays a 'Labeled Property Graph Data Model' visualization, showing nodes for authors (John Le Carre, Graham Greene) and books (Tinker Tailor Soldier Spy, Our Man in Havana) connected by edges labeled 'WROTE' and 'PURCHASED'.

Install

- Requirements
 - Downloading Binary Package
 - Building Zeppelin from source
- Starting Apache Zeppelin
- Start Apache Zeppelin with a service manager
- Next Steps

Welcome to Apache Zeppelin! On this page are instructions to help you get started.

Requirements

Apache Zeppelin officially supports and is tested on the following environments:

Name	Value
Oracle JDK	1.7 (set <code>JAVA_HOME</code>)
OS	Mac OSX Ubuntu 14.X CentOS 6.X Windows 7 Pro SP1

Labeled Property Graph Data Model

```
graph LR; A((name: John Le Carre)) -- WROTE --> B((title: Tinker, Tailor, Soldier, Spy)); A -- WROTE --> C((title: Our Man in Havana)); D((name: Ian)) -- PURCHASED --> B; D -- PURCHASED --> C; E((name: Alan)) -- PURCHASED --> C;
```

What are the APIs?

The new NETWORK visualization is based on json with the following params:

<https://zeppelin.apache.org/>



**KEEP
CALM
AND
START
CODING**