

# MA-INF 4223 - Lab Distributed Big Data Analytics

## Introduction

Dr. Hajira Jabeen, Gezim Sejdiu

Winter Semester 2017/18

# About Us



# Smart Data Analytics (SDA )

- ❖ Prof. Dr. Jens Lehmann
  - Institute for Computer Science , University of Bonn
  - Fraunhofer Institute for Intelligent Analysis and Information Systems (IAIS)
  - Institute for Applied Computer Science, Leipzig
- ❖ Machine learning techniques ("analytics") for Structured knowledge ("smart data")
- ❖ The group aims at covering the full spectrum of research including theoretical foundations, algorithms, prototypes and industrial applications



# Dr. Hajira Jabeen

- ❖ Senior Researcher at University of Bonn, 2016-
- ❖ PostDoc at AKSW, University of Leipzig, 2015
- ❖ Assistant Lecturer at IT-U, Copenhagen 2014
- ❖ Assistant Professor, Iqra University 2010-2013
- ❖ Research Interests :
  - *Big Data , Data Mining*
  - *Optimization*
  - *Machine Learning and Analytics*
  - *Semantic Web*
  - *Structured Machine learning*



# Gezim Sejdu

- ❖ Research Associate/PhD Student at University of Bonn, 2016-
- ❖ Guest Researcher at AKSW group at the University of Leipzig in 2015
- ❖ Research Interests :
  - Big Data, Data Mining and Data Analysis,
  - Semantic Web and Semantic Search,
  - Machine Learning,
  - Distributed Computing



# Group Members

- ❖ Dr. Asja Fischer
- ❖ Dr. Henning Petzka
- ❖ Dr. Günter Kniestel
- ❖ Dr. Alexander Kister
- ❖ Dr. Muhammad Sherif
- ❖ Dr. Hamed Shariat
- ❖ Dr. Giulio Napolitano
- ❖ ...
- ❖ ...Many others



# Group's Research Interests

- ❖ Distributed Semantic Analytics
- ❖ Semantic Question Answering
- ❖ Structured Machine Learning
- ❖ Deep Learning
- ❖ Software Engineering for Data Science
- ❖ Semantic Data Management
- ❖ Knowledge Extraction and Validation





# Projects

- ❖ Big Data Europe, EU H2020, Big Data
- ❖ GEISER, BMWi
- ❖ HOBBIT, EU H2020, Big Data
- ❖ SLIPO, EU H2020, Big Data
  - Scalable linking and integration
- ❖ QROWD, EU H2020, Big Data
  - Big data integration of cities e.g. geographic, transport, meteorological
- ❖ SAKE, BMWi
  - Semantic Analysis of Complex Events



# Projects

- ❖ Domain Specific Languages (DSLs) for Machine Learning
- ❖ Smoothed Analysis of Structured Machine Learning Algorithms from Knowledge Graphs
- ❖ Cognitive Robotics
- ❖ Experimental Analysis of Class CS Problems
- ❖ Tensor Factorisation and Visualization for Knowledge Graphs



# Software Projects

- ❖ SANSA - Distributed Semantic Analytics Stack
- ❖ AskNow - Question Answering Engine
- ❖ DL-Learner - Supervised Machine Learning in RDF / OWL
- ❖ LinkedGeoData - RDF version of OpenStreetMap
- ❖ DBpedia - Wikipedia Extraction Framework
- ❖ DeFacto - Fact Validation Framework



# Distributed Semantic Analytics



- ❖ Leader: Dr. Hajira Jabeen
  - Prof. Dr. Jens Lehmann (Mentor)
  - Dr. Hamed Shariat Yazdi
  - Dr. Luís Paulo F. Garcia
  - Dr. Henning Petzka
  - Dr. Anisa Rula
  - Claus Stadler
  - Patrick Westphal
  - Simon Bin
  - Gezim Sejdiu
  - Harsh Thakkar
  - Nilesh Chakraborty
  - Heba Ibrahim



# Semantic Question Answering



- ❖ Leader: Dr. Giulio Napolitano
  - Prof. Dr. Jens Lehmann (Mentor)
  - Dr. Ioanna Lytra (Member / Mentor)
  - Mohnish Dubey
  - Hamid Zafar
  - Debanjan Chaudhuri
  - Konrad Höffner
  - Denis Lukovnikov
  - Gaurav Maheshwari
  - Priyansh Trivedi
  - Debayan Banerjee
  - Kuldeep Singh
  - Jewgeni Rose
  - Ashwini Jaya Kumar
  - Vinay Modi



# Structured Machine Learning



- ❖ Leader: Prof. Dr. Jens Lehmann
  - Lorenz Bühmann (Deputy Leader)
  - Dr. Mohamed Sherif
  - Patrick Westphal
  - Simon Bin



# Deep Learning



- ❖ Leader: Dr. Asja Fischer
  - Prof. Dr. Jens Lehmann (Mentor)
  - Dr. Henning Petzka
  - Dr. Alexander Kister
  - Dr. Tiansi Dong
  - Denis Lukovnikov



# Software Engineering for Data Science



- ❖ Leader: Dr. Günter Kriesel, Dr. Hamed S. Yazdi
  - Prof. Dr. Jens Lehmann (Mentor)
  - Dr. Luís Paulo F. Garcia
  - Dr. Tiansi Dong
  - Afshin Sadeghi
  - Shima Ibrahim



# Semantic Data Management



- ❖ Leader: Dr. Christoph Lange, Dr. Steffen Lohmann
  - Prof. Dr. Jens Lehmann (Mentor)
  - Dr. Anisa Rula
  - Sahar Vahdati
  - Michael Galkin
  - Said Fathalla
  - Jean Claude Hernandez
  - Gabriel Gimenez
  - Diego Collaranra
  - Mohamed Nadjib Mami
  - Niklas Petersen
  - Lavdim Halilaj
  - Irlán Grangel-González
  - Mirette Elias



# Knowledge Extraction and Validation



- ❖ Leader: Dr. Simon Scerri, Dr. Fabrizio Orlandi
  - Prof. Dr. Jens Lehmann (Mentor)
  - Isaiah Onando Mulang'
  - Najmeh Mousavi Nejad
  - Elisa Margareth Sibarani
  - Diego Esteves
  - Fathoni A. Musyaffa

# Organisational Matters



# Organisational Matters

## ❖ Overview

- Distributed Big Data Analysis consists of two modules : lecturers/lab + project
- Mailing list : sign-up sheet
- Lecture notes will be provided at :  
<http://sda.cs.uni-bonn.de/teaching/dbda/>
- <https://sewiki.iai.uni-bonn.de/teaching/labs/dbda/2017/start>
- Changes will be announced at the website and via mailing list



# Lab Schedule: Tuesday 10:00 - 13:00

October 17	Motivation
October 24	No Class
October 31	Spark Fundamentals I
November 7	Spark Fundamentals II
November 14	RDF, Knowledge graphs and SANSA
November 21	Project Allocation
November 28- January 30	Lab Project work
January 30	Project Presentation



# Grading

- ❖ Final Project.
  - Code,
  - Documentation ,
  - Presentation

# What is BigData?



# Big Data



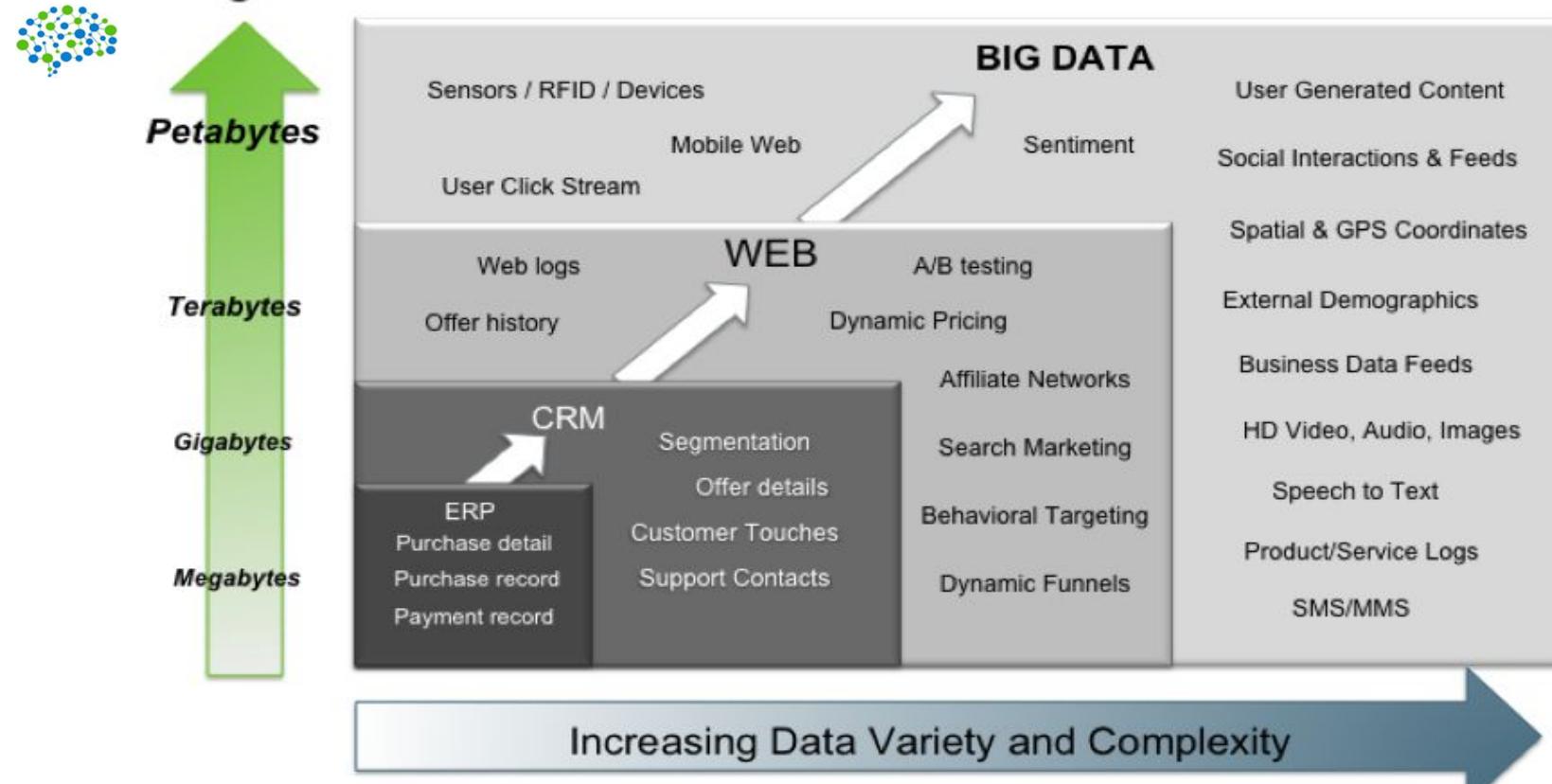
- ❖ No Single Definition
- ❖ Extremely large data sets that may be analysed computationally to reveal patterns, trends, and associations, especially relating to human behaviour and interactions
- ❖ Big data is a term for data sets that are so large or complex that traditional data processing application softwares are inadequate to deal with them



# Big Data

- ❖ Every day, there are 2.5 quintillion bytes of data created - so much that 90% of the data in the world today has been created in the last two years alone
- ❖ It is not only about data collection, or data querying, its is about learning from this tremendous data for informed decision making

# Big Data = Transactions + Interactions + Observations

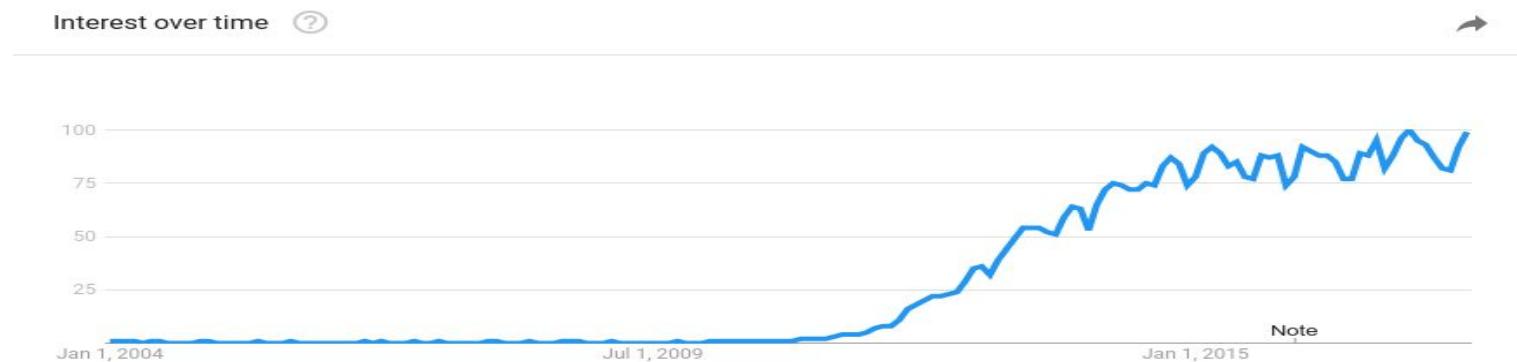


Source: <https://media.lcdn.com/mpr/mpr/AEAAQAAAAAAAATgAAAAJDg3ODgxMGRjLWUxOWItNDYxMC1hOTExLTMwZWIxYTdjMTQZA.png>



# Why 'BigData' is so important?

- ❖ Its relevance is increasing drastically and Big Data Analytics is an emerging field to explore

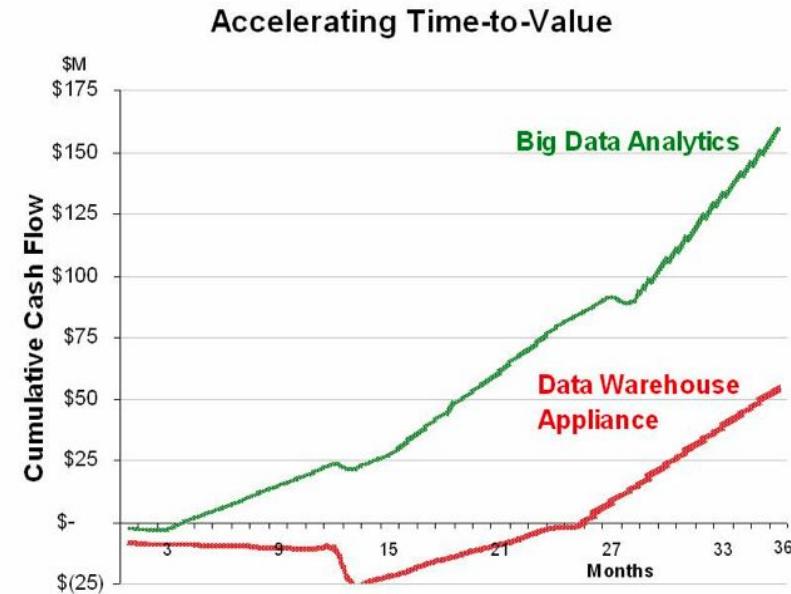


<https://www.google.com/trends/explore?date=all&q=%22big%20data%22>



# Big Data Analytics

- ❖ Big data is more real-time in nature than traditional DW applications
- ❖ Traditional DW architectures (e.g. Exadata, Teradata) are not well-suited for big data apps
- ❖ Shared nothing, massively parallel processing, scale out architectures are well-suited for big data apps



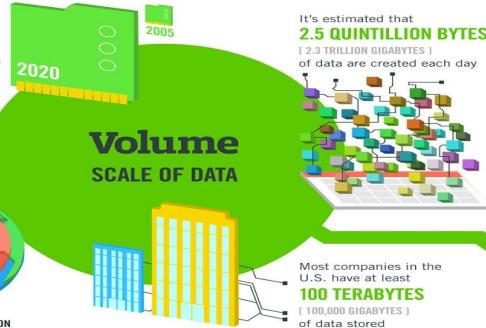
Source: [http://wikibon.org/wiki/v/Enterprise\\_Big-data](http://wikibon.org/wiki/v/Enterprise_Big-data)



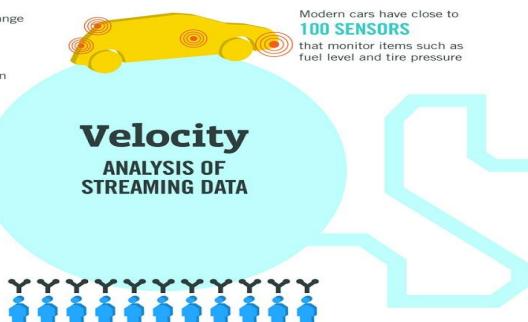
# Big Data Dimensions

**40 ZETTABYTES**  
[ 43 TRILLION GIGABYTES ]  
of data will be created by  
2020, an increase of 300  
times from 2005

**6 BILLION PEOPLE**  
have cell phones  
**WORLD POPULATION: 7 BILLION**



The New York Stock Exchange captures  
**1 TB OF TRADE INFORMATION** during each trading session



Sources: McKinsey Global Institute, Twitter, Cisco, Gartner, EMC, SAS, IBM, MEPTEC, QAS

<http://www.ibmbigdatahub.com/infographic/four-vs-big-data>

## The FOUR V's of Big Data

From traffic patterns and music downloads to web history and medical records, data is recorded, stored, and analyzed to enable the technology and services that the world relies on every day. But what exactly is big data, and how can these massive amounts of data be used?

As a leader in the sector, IBM data scientists break big data into four dimensions: **Volume**, **Velocity**, **Variety** and **Veracity**.

Depending on the industry and organization, big data encompasses information from multiple internal and external sources such as transactions, social media, enterprise content, sensors and mobile devices. Companies can leverage data to adapt their products and services to better meet customer needs, optimize operations and infrastructure, and find new sources of revenue.

By 2015  
**4.4 MILLION IT JOBS** will be created globally to support big data, with 1.9 million in the United States



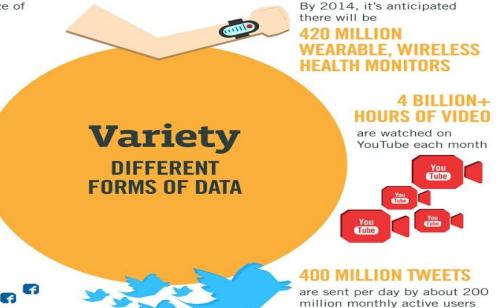
As of 2011, the global size of data in healthcare was estimated to be

**150 EXABYTES**  
[ 161 BILLION GIGABYTES ]

**30 BILLION PIECES OF CONTENT** are shared on Facebook every month



### Variety DIFFERENT FORMS OF DATA



**400 MILLION TWEETS** are sent per day by about 200 million monthly active users

**1 IN 3 BUSINESS LEADERS** don't trust the information they use to make decisions



**27% OF RESPONDENTS**

in one survey were unsure of how much of their data was inaccurate

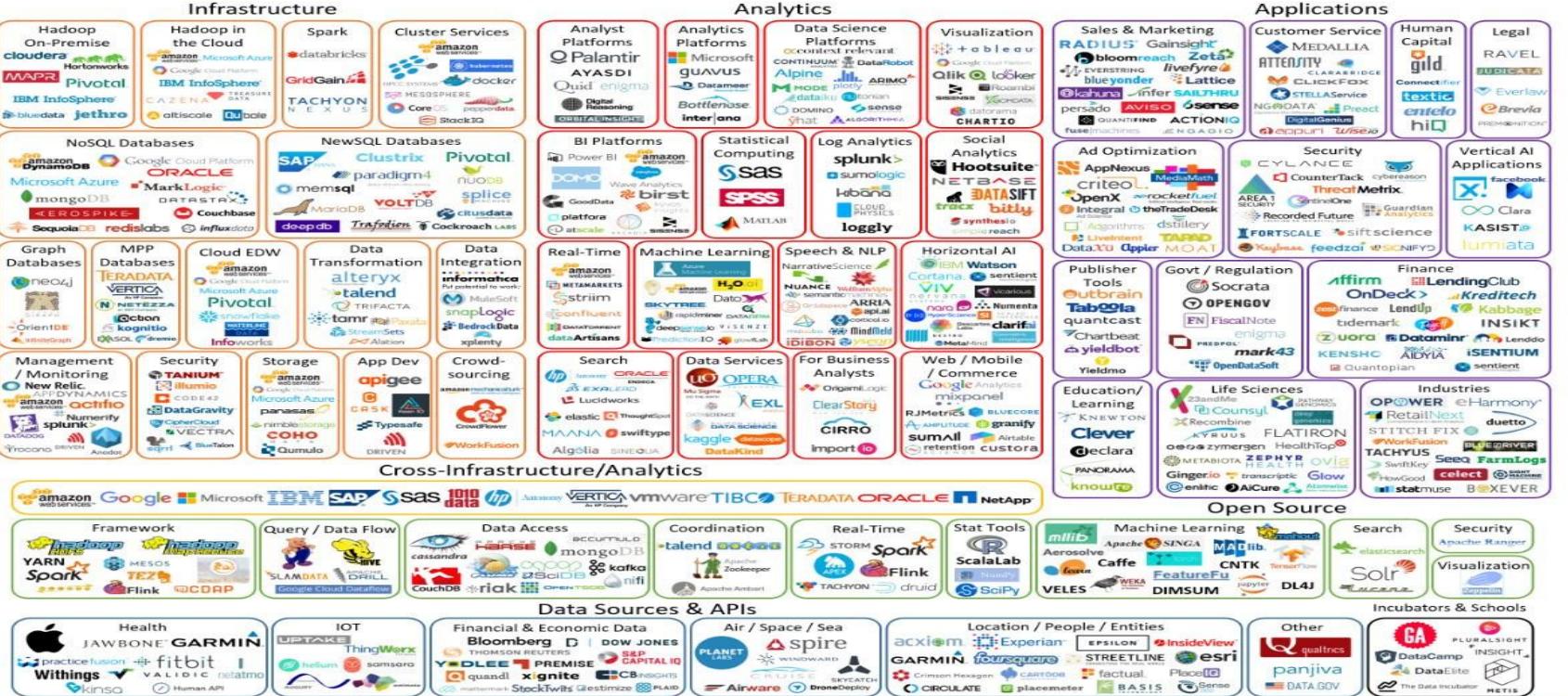
### Veracity UNCERTAINTY OF DATA

Poor data quality costs the US economy around **\$3.1 TRILLION A YEAR**





# Zoo of Big Data tools



Source: Big Data Universe v3. Matt Turck, Jim Hao & FirstMark Capital



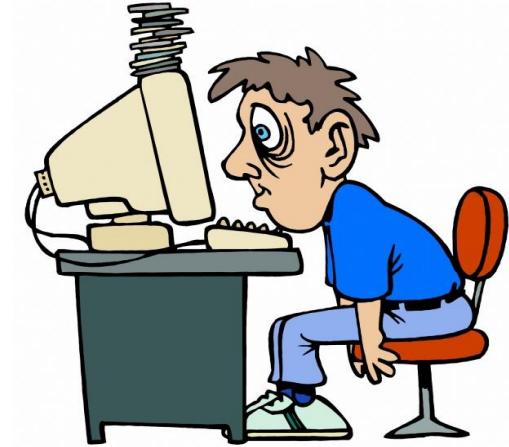
# Big Data Ecosystem

<b>File system</b>	HDFS, NFS
<b>Resource manager</b>	Mesos, Yarn
<b>Coordination</b>	Zookeeper
<b>Data Acquisition</b>	Apache Flume, Apache Sqoop
<b>Data Stores</b>	MongoDB, Cassandra, Hbase, Project Voldemort
<b>Data Processing</b>	<ul style="list-style-type: none"><li>• Frameworks Hadoop MapReduce, Apache Spark, Apache Storm, Apache FLink</li><li>• Tools Apache Pig, Apache Hive</li><li>• Libraries SparkR, Apache Mahout, MLlib, etc</li></ul>
<b>Data Integration</b>	<ul style="list-style-type: none"><li>• Message Passing Apache Kafka</li><li>• Managing data heterogeneity SemaGrow, Strabon</li></ul>
<b>Operational Frameworks</b>	<ul style="list-style-type: none"><li>• Monitoring Apache Ambari</li></ul>



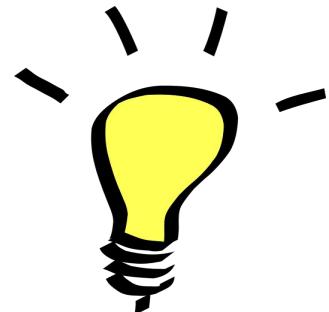
# Data Flow Models

- ❖ Data flow vs. traditional network programming
  - Message Passing Interface
  - Programmer managed data locality and Code
  - Hardware failure
  - Slow Hardware
  - Data Communication is slow over the network





# Map reduce



- ❖ First popular data flow model comprises the programmer's ability to control functionality in order to handle the problems
- ❖ In the Map-Reduce model, the user provides two functions (map and reduce)
  - Map() must output key-value pairs, and as a result
  - reduce() is guaranteed that its input is partitioned by key across machines



# Drawbacks of Mapreduce

- ❖ Force data analysis workflow into a map and a reduce phase
  - You might need
    - Join
    - Filter
    - Sample
  - Complex workflows that do not fit into map/Reduce
  - Mapreduce relies on reading data from disk
    - Performance bottleneck
    - Especially for iterative algorithms



# Solution:

- ❖ A tool that works in the same environment
- ❖ Provides an Interactive shell
- ❖ Compatible with the existing environment
- ❖ No need to replace the stack, but replace map--reduce



# Data Flow Engines

- ❖ The Data flow engines are useful because
  - They save time and effort on the part of the programmer by providing abstractions
  - They scale well
  - Many algorithms have been redesigned to fit into the paradigm
  - They have become common on clusters



- ❖ Rich programming interface
- ❖ More than 20 efficient, distributed operations
- ❖ Easier for data scientists to create custom data analysis pipeline in Spark
- ❖ Data can be cached in Memory
- ❖ ML Pipelines have gained 10 to 100 times speedup



# Spark

- ❖ Fast
  - in-memory computations
- ❖ General-purpose,
  - Mapreduce jobs, Iterative jobs, Queries, Streaming
- ❖ Easy to use
  - Simple APIs for Scala, Python, Java, R
  - General:
    - Spark runs on Hadoop clusters such as Hadoop YARN or Apache Mesos,
    - As a standalone with its own scheduler.



# Spark

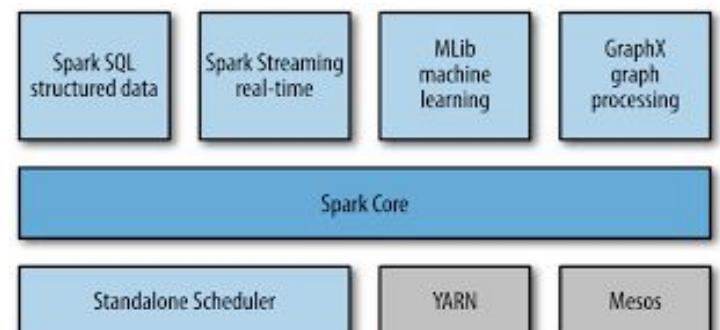
- ❖ Spark provides
  - parallel distributed processing,
  - fault tolerance on commodity hardware,
  - Scalability
- ❖ Spark adds to the concept by
  - low latency,
  - Aggressively cached in-memory distributed Computing,
  - high level APIs
  - stack of high level tools
  - This saves time and money.

# Big Data distributed frameworks



# Apache Spark

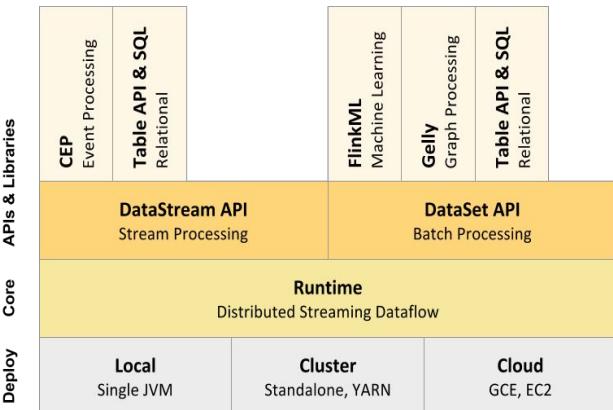
- ❖ **Apache Spark** is an open-source distributed and highly scalable in-memory data processing and analytics system. It provides APIs in Scala, Java, Python and R which try to simplify the programming complexity by introducing the abstraction of Resilient Distributed Datasets (RDD), i.e. a logical collection of data partitioned across machines.
- ❖ On top of its core, Spark provides 4 libraries:
  - [Spark SQL](#) for SQL and unstructured data processing
  - [Spark Streaming](#) stream processing
  - [MLlib](#) machine learning algorithms
  - [GraphX](#) graph processing.





# Apache Flink

- ❖ Apache Flink is an open-source stream processing framework for distributed, high-performing, always-available, and accurate data streaming applications
- ❖ Flink provides:
  - [DataStream API](#) for unbounded streams
  - [DataSet API](#) for static data
  - [Table API & SQL](#) with a SQL-like expression
- ❖ It bundles libraries for domain-specific use cases:
  - [CEP](#), a complex event processing library,
  - [Machine Learning library](#), and
  - [Gelly](#), a graph processing API and library



# Big Data Projects

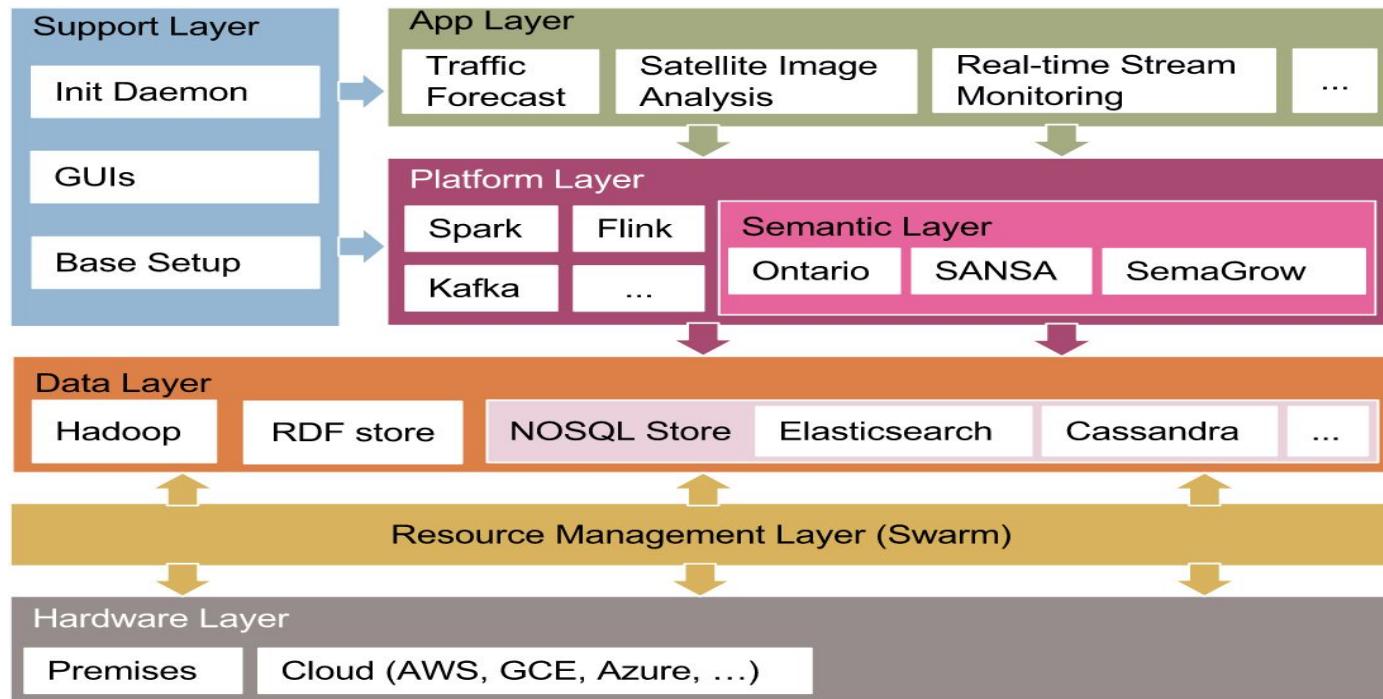


# Big Data Projects

- ❖ Related to this lab
  - Big Data Europe
  - SANSA
  - Big Data Ocean



# BDE Platform

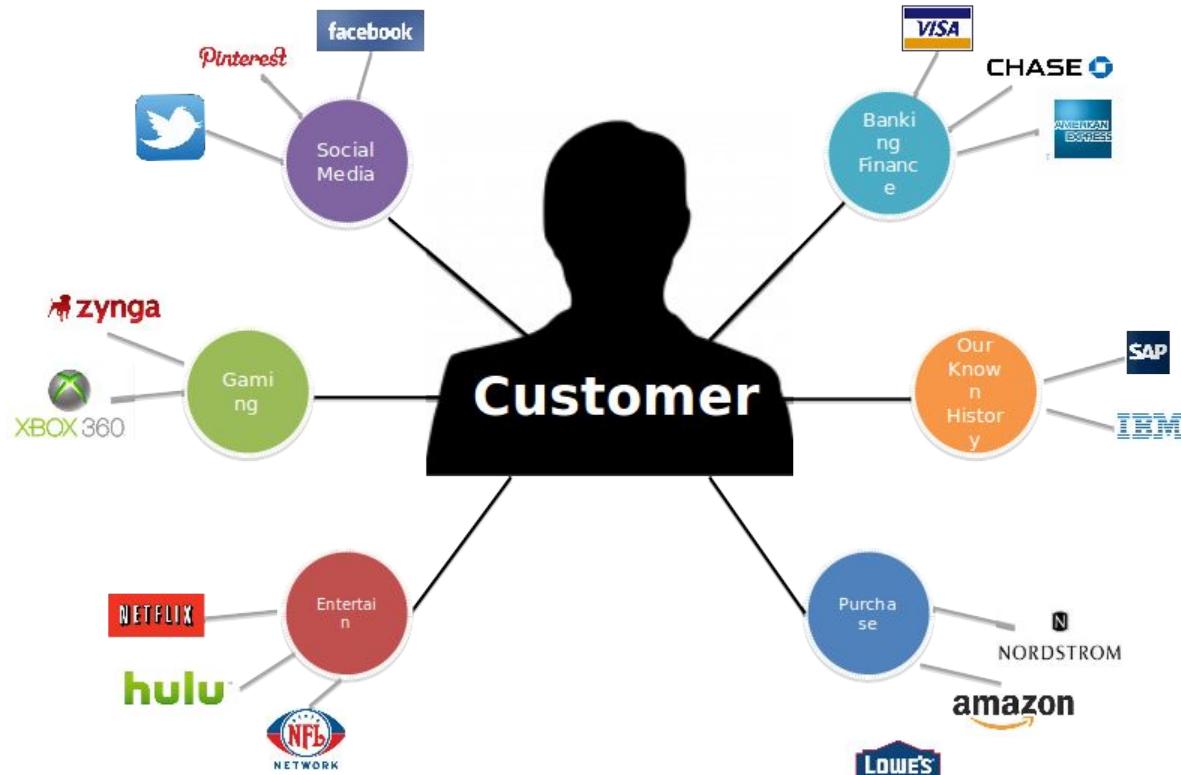




# Key Observation From BDE

- ❖ Heterogeneity AKA Variety

# A Single View to the Customer

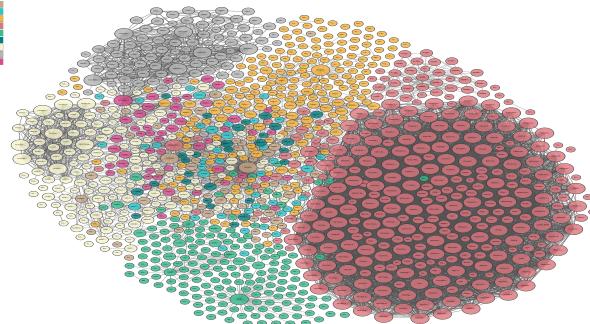
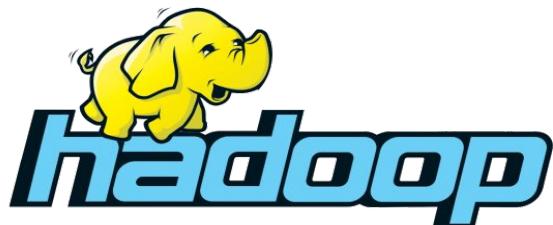


# Smart Big Data



# Smart Big Data

- ❖ Over the last years, the size of the Semantic Web has increased and several large-scale datasets were published



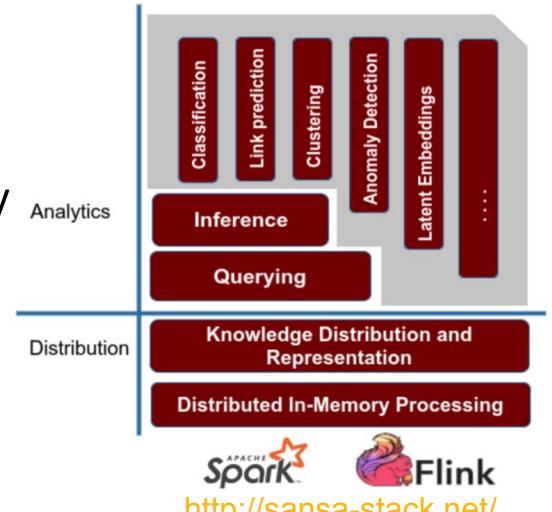
Source: LOD-Cloud (<http://lod-cloud.net/>)

- ❖ Hadoop ecosystem has become a standard for Big Data applications → use this infrastructure for Semantic Web as well



# SANSA - Semantic Analytics Stack

- ❖ Knowledge Graphs become increasingly popular (via graph databases and semantic technologies)
- ❖ But:
  - Most ML algorithms work on simple feature input (not graphs)
  - Advanced algorithms for knowledge graphs usually do not scale horizontally
- ❖ SANSA is a suite of APIs for distributed reading, querying, inferencing and analysing of RDF knowledge graphs



# Introduction to Scala



- ❖ A functional Programming Language that follows a rich concise syntax
- ❖ Multiparadigm ( you can also write OO code)
- ❖ Interoperates with Java
- ❖ Simplifies the concurrent programming , ( No threads/Locks)
- ❖ Static typing, Immutable objects, Closure, elegant pattern matching
- ❖ Strongly Typed language, with functional and concurrency support



- ❖ Conciseness ( 100 lines in Java = 10-15 lines of code in scala)
- ❖ Sometimes harder to understand
  - Currying,
  - Function passing,
  - High order functions



- ❖ Everything is an object
  - Primitive types e.g. numbers, bool
  - Functions
- ❖ Numbers are objects:
  - $1+2*3 \rightarrow (1).+(2).* (3)$
- ❖ Functions are objects:
  - Pass functions as arguments
  - Store them in variables
  - Return them from other functions
- ❖ Function declaration
  - Def functionName ( [list of parameters]):[return type]



# Scala REPL

- ❖ Read evaluate and Print Loop.
- ❖ Interactive shell session,
- ❖ In interactive mode, the REPL reads expressions at the prompt, wraps them in an executable template, and then compiles and executes the result.



# Basics

```
scala> val msg = "Hello, world!"  
msg: String = Hello, world!
```

```
scala> println(msg)
```

```
Hello, world!
```

```
scala> def max(x: Int, y: Int): Int  
= if (x < y) y else x  
max: (x: Int, y: Int)Int
```

```
scala> max(3, 5)  
res1: Int = 5
```

<http://www.artima.com/scalazine/articles/steps.html>



# Anonymous Functions

```
>val plusOne = (x: Int) => x + 1
>(x: Int) => x * x * x // returns cube
    >(x: Int, y: Int) => x + y // sums two numbers
//Alternatively
{
def f(x: Int, y: Int) = x + y;
}
printIn(plusOne(0)) // Prints: 1
anotherFunction(plusOne(20)) // Prints: 21
```



# High Order Functions

- ❖ First order functions :
  - Acts on simple data types
- ❖ High order Function:
  - Act on other functions
  - Sum function in the next example is a High order function
  - Functions are treated as first-class values
    - passed as parameters
    - returned as a value
  - Flexible way to compose programs



# High order Functions

```
def sum(f: Int => Int, a: Int, b: Int): Int =  
    if (a > b) 0  
    else f(a) + sum(f, a + 1, b)
```

We can then write:

```
def sumInts(a: Int, b: Int)      = sum(id, a, b)  
def sumCubes(a: Int, b: Int)     = sum(cube, a, b)  
def sumFactorials(a: Int, b: Int) = sum(fact, a, b)
```

where

```
def id(x: Int): Int    = x  
def cube(x: Int): Int = x * x * x  
def fact(x: Int): Int = if (x == 0) 1 else fact(x - 1)
```



# High order Functions

```
def sum(f: Int => Int, a: Int, b: Int): Int =  
  if (a > b) 0  
  else f(a) + sum(f, a + 1, b)
```

We can then write:

```
def sumInts(a: Int, b: Int)      = sum(id, a, b)  
def sumCubes(a: Int, b: Int)     = sum(cube, a, b)  
def sumFactorials(a: Int, b: Int) = sum(fact, a, b)
```

where

**Can we use anonymous functions, how ?**

```
def id(x: Int): Int  = x  
def cube(x: Int): Int = x * x * x  
def fact(x: Int): Int = if (x == 0) 1 else fact(x - 1)
```

Ref: Slides by Martin Odersky, coursera



# High order Functions

```
def sum(f: Int => Int, a: Int, b: Int): Int =  
  if (a > b) 0  
  else f(a) + sum(f, a + 1, b)
```

We can then write:

```
def sumInts(a: Int, b: Int) = sum(x => x, a, b)  
def sumCubes(a: Int, b: Int) = sum(x => x * x * x, a, b)
```

where

**Can we use anonymous functions, how ?**

```
def id(x: Int): Int = x  
def cube(x: Int): Int = x * x * x  
def fact(x: Int): Int = if (x == 0) 1 else fact(x - 1)
```

Ref: Slides by Martin Odersky, coursera



# High order Functions and Tail recursion

```
def sumInts(a: Int, b: Int): Int =  
    if (a > b) 0 else a + sumInts(a + 1, b)  
  
def sum(f:Int => Int, a:Int, b:Int) = { // f is a function  
def loop(a:Int, acc:Int):Int =  
    if(a>b) acc  
    else loop(a+1, f(a) + acc)  
        loop(a, 0) // return value  
}  
sum(x =>x*x, 3, 5) // 50
```



# References

- ❖ "Linking Open Data cloud diagram 2017, by Andrejs Abele, John P. McCrae, Paul Buitelaar, Anja Jentzsch and Richard Cyganiak.  
<http://lod-cloud.net/>".
- ❖ <http://www.artima.com/scalazine/articles/steps.html>
- ❖ <http://sda.cs.uni-bonn.de/>
- ❖ <https://github.com/SANSA-Stack>
- ❖ <https://github.com/big-data-europe>
- ❖ <https://github.com/SmartDataAnalytics>

# THANK YOU !

<http://sda.cs.uni-bonn.de/teaching/dbda/>

- <http://sda.cs.uni-bonn.de/>
- <https://github.com/SANSA-Stack>
- <https://github.com/big-data-europe>
- <https://github.com/SmartDataAnalytics>



**Dr. Hajira Jabeen**  
[jabeen@cs.uni-bonn.de](mailto:jabeen@cs.uni-bonn.de)

Room A108 (Appointment per e-mail)



**Gezim Sejdiu**  
[sejdiu@cs.uni-bonn.de](mailto:sejdiu@cs.uni-bonn.de)

Room A120 (Appointment per e-mail)