UNIVERSITÄT BONN

SMART DATA ANALYTICS
FROM DATA TO KNOWLEDGE

MA-INF 4223 - Lab Distributed Big Data Analytics

Introduction

Dr. Hajira Jabeen, Gezim Sejdiu

Summer Semester 2018

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 Sejdiu

- 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. Günter Kniesel
- Dr. Hamed Shariat
- Dr. Giulio Napolitano
- …
- …Many others

Group's Research Interests

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

Projects

- Big Data Europe, EU H2020, Big Data
- Boost4.0- Big Data for Industry 4.0,
- Big Data Ocean
- 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. 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

Structured Machine Learning

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

Software Engineering for Data Science

- Leader: Dr. Günter Kniesel, 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 Collarana
  - 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/
  - Changes will be announced at the website and via mailing list

Objectives of the Lab

- At the end of the Lab we want you to be able to :
  - Program in Scala
  - Know the working model of Spark
    - Data structures and their behaviour
    - Shuffling/Cache …..
  - Create and execute parallel programs in Spark(Scala)
  - Use of Spark web UI for cluster/program analysis

Outcomes

- Given an algorithm we want you to be able to :
  - Analyze the algorithm for parallel programming
  - Implement the algorithm in distributed execution model

Lab Schedule: Tuesday 10:00 - 13:00

April 17	**Motivation**
April 24	**Spark Fundamentals I**
May 1	**No class**
May 8	**Spark Fundamentals II (Spark GraphX + Spark SQL)**
May 15	**Spark Fundamentals II (Spark ML), SANSA - Semantic Analytics Stack, Project Allocation**
May 29	**Presentation for the Project**
May 29 - July 13	**Lab work**
June 12	**Meetings**
July 13	**Project Report submission**
July 17	**Final Project Presentations**

Grading

- Test the proficiency in
  - Scala
  - Use of Spark / Scala (Parallel Programming) constructs
  - Complexity/efficiency of the developed algorithm

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

What is BigData?

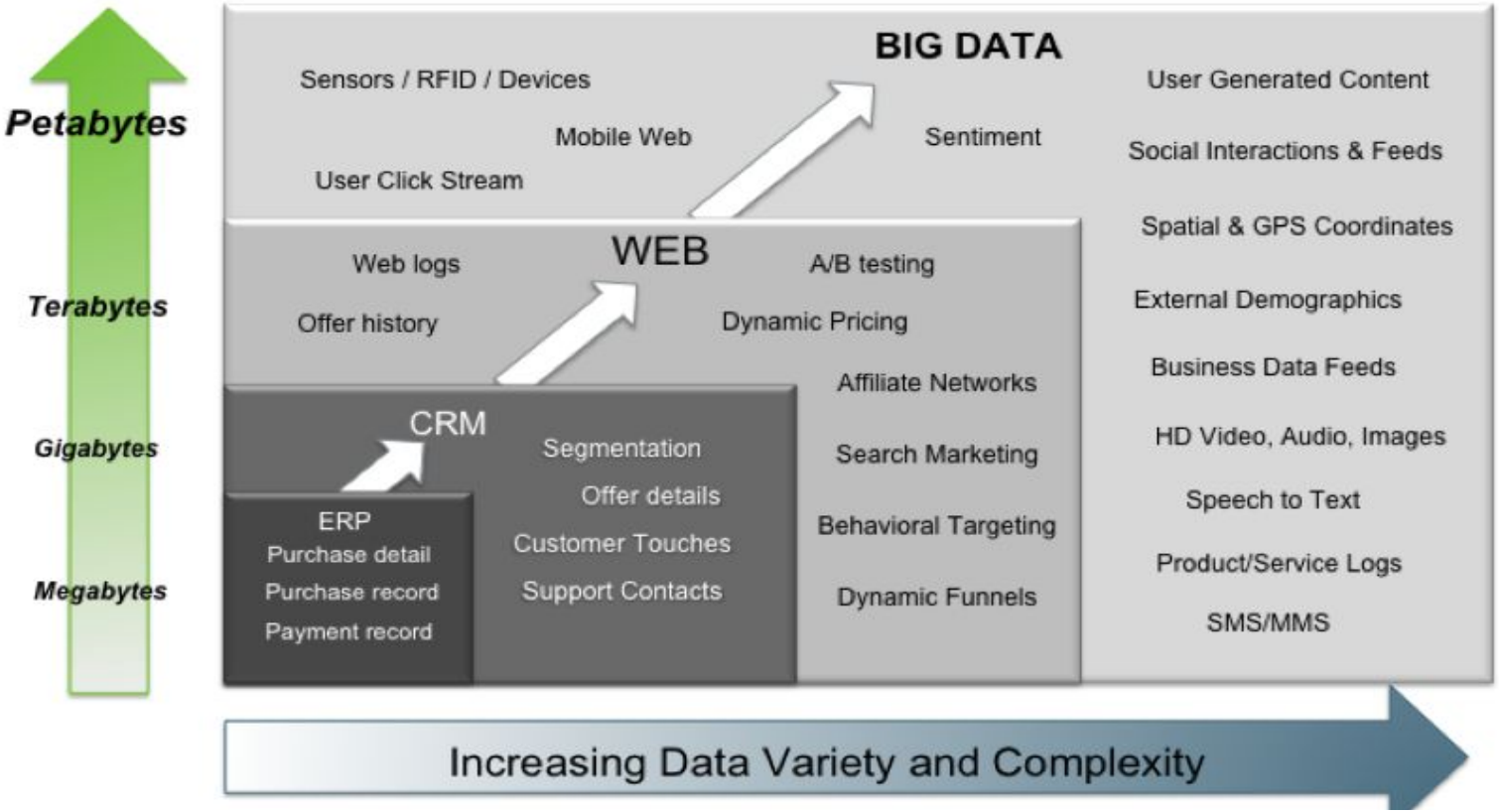
Big Data

- 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

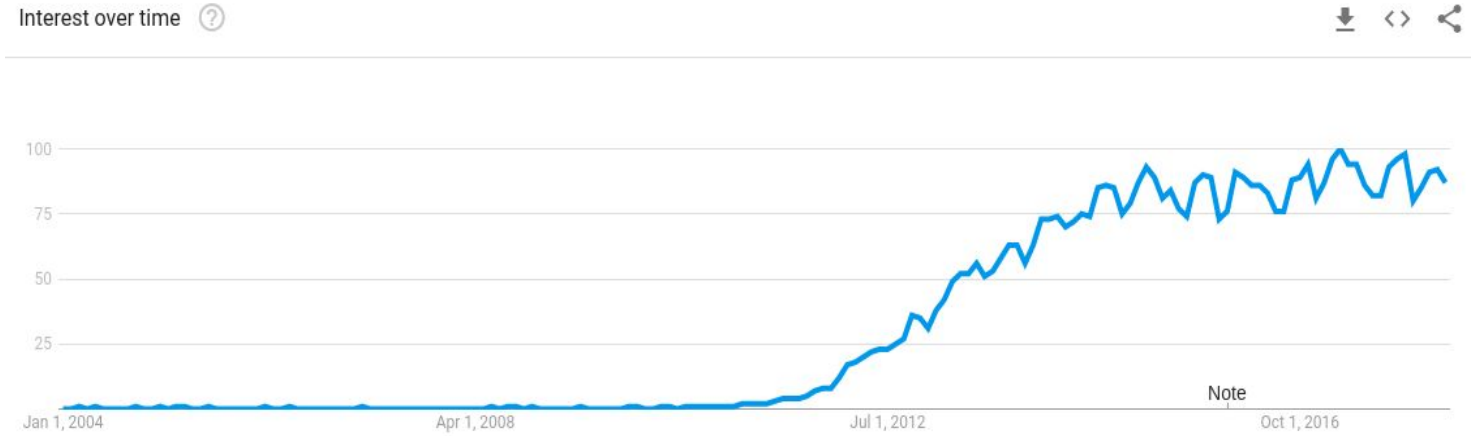
Big Data = Transactions + Interactions + Observations



Source: https://media.licdn.com/mpr/mpr/AAEAAQAAAAAAAATqAAAAJDg3ODgxMGRjLWUxOWItNDYxMC1hOTExLTMwZWIxYTdjMTQ4ZA.png

Why ‘BigData’ is so important?

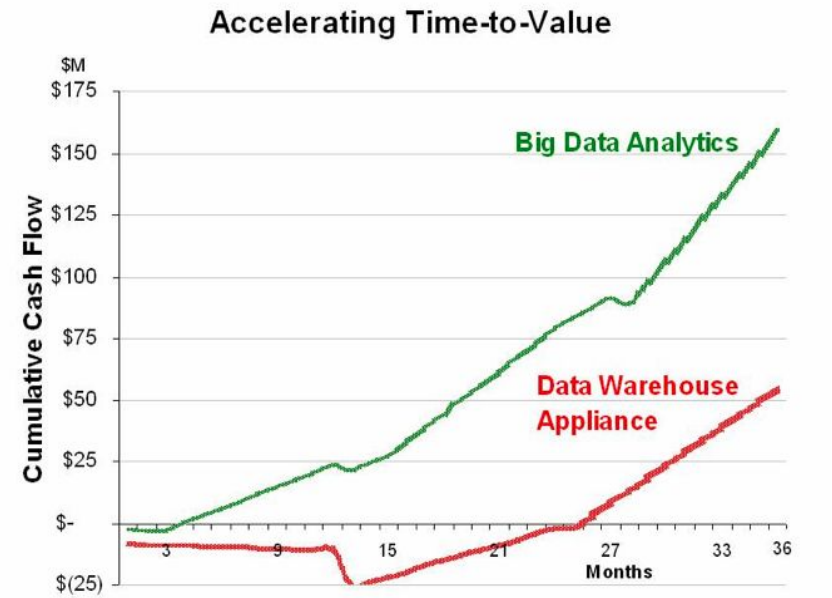
- It’s relevance is increasing drastically and Big Data Analytics is an emerging field to explore



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

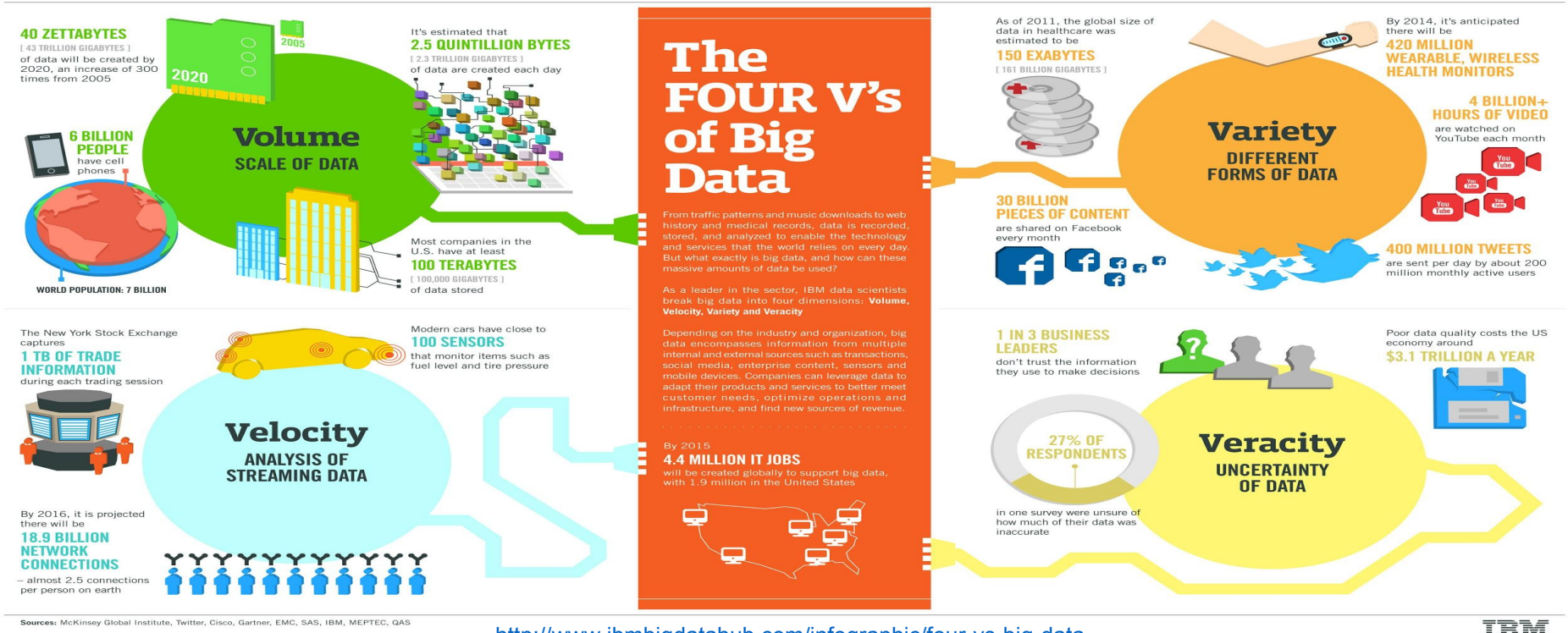
Big Data Analytics

- 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



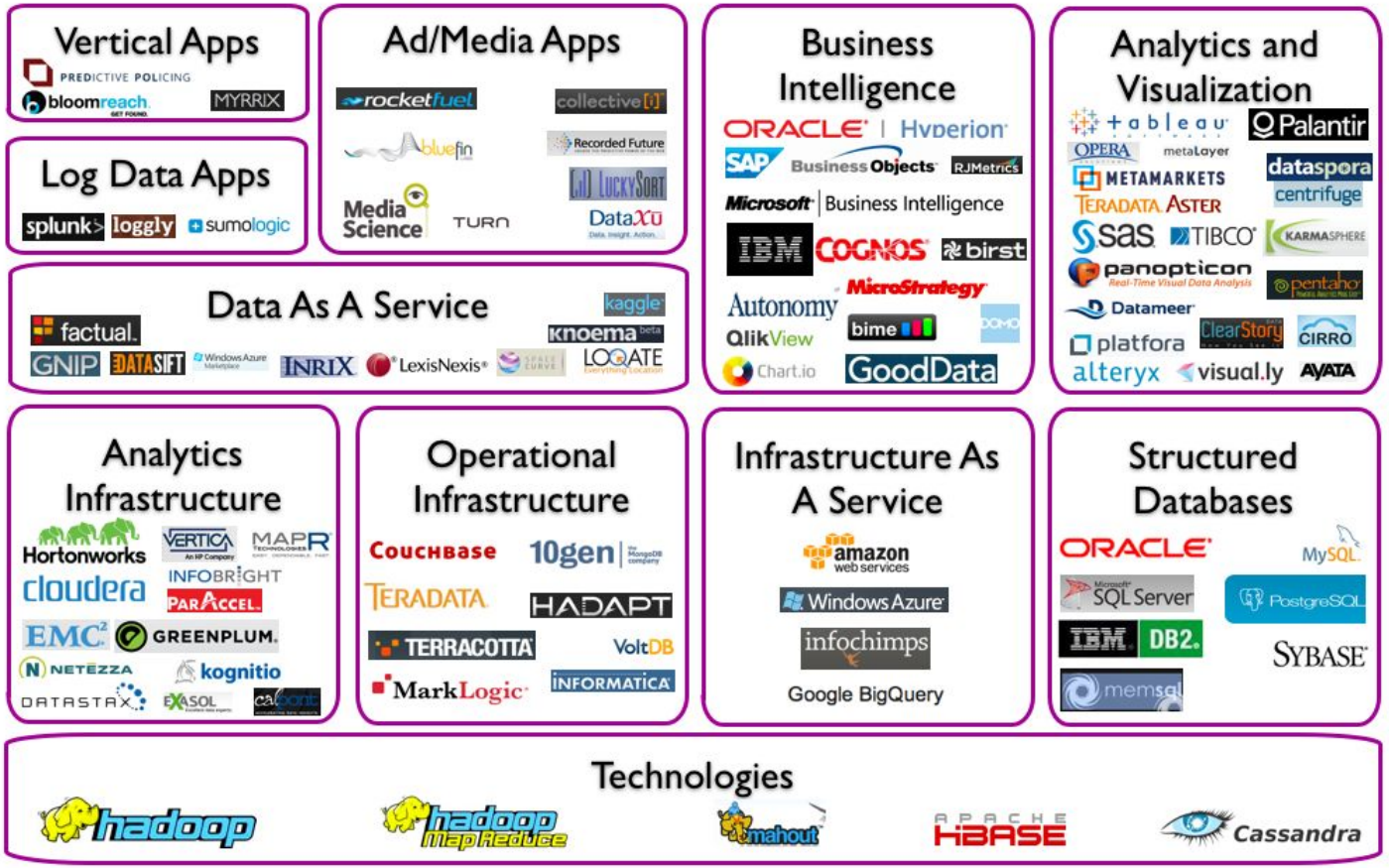
Source: http://wikibon.org/wiki/v/Enterprise_Big-data

Big Data Dimensions



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

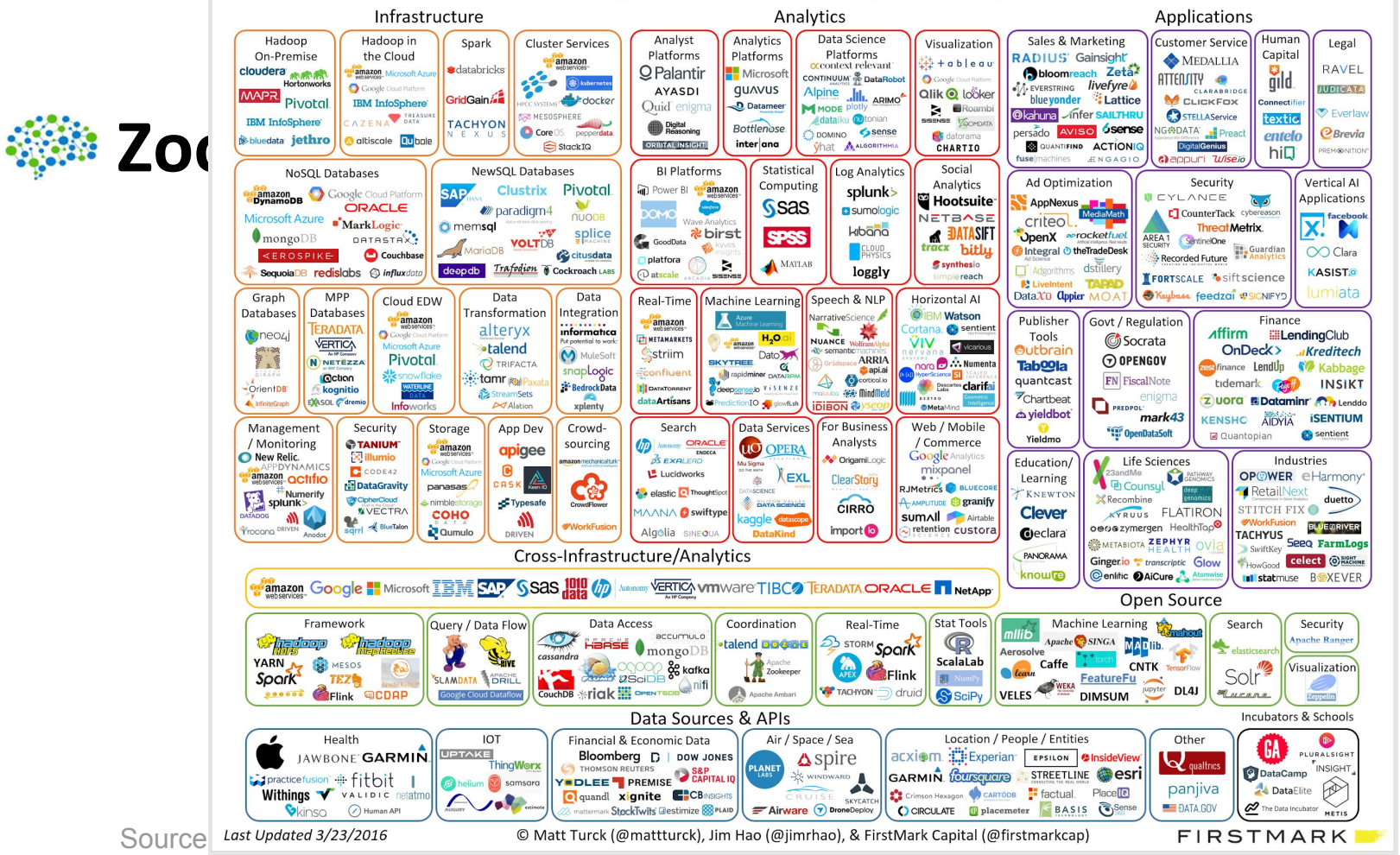
Big Data Landscape



dave@vcdave.com

blogs.forbes.com/davefeinleib

© Matt Turck (@mattturck) and Shivon Zilis (@shivonz), Bloomberg Ventures


Infrastructure
Analytics
Applications
Hadoop On-Premise
Hadoop in the Cloud
Spark
Cluster Services
Analyst Platforms
Analytics Platforms
Data Science Platforms
Visualization
Sales & Marketing
Customer Service
Human Capital
Legal
NoSQL Databases
NewSQL Databases
BI Platforms
Statistical Computing
Log Analytics
Social Analytics
Ad Optimization
Security
Vertical AI Applications
Graph Databases
MPP Databases
Cloud EDW
Data Transformation
Data Integration
Real-Time
Machine Learning
Speech & NLP
Horizontal AI
Publisher Tools
Govt / Regulation
Finance
Management / Monitoring
Security
Storage
App Dev
Crowd-sourcing
Search
Data Services
For Business Analysts
Web / Mobile / Commerce
Education / Learning
Life Sciences
Industries
Cross-Infrastructure/Analytics
Open Source
Framework
Query / Data Flow
Data Access
Coordination
Real-Time
Stat Tools
Machine Learning
Search
Security
Visualization
Data Sources & APIs
Incubators & Schools
Health
IOT
Financial & Economic Data
Air / Space / Sea
Location / People / Entities
Other
Last Updated 3/23/2016
© Matt Turck (@mattturck), Jim Hao (@jimrhao), & FirstMark Capital (@firstmarkcap)
FIRSTMARK

Source

BIG DATA LANDSCAPE 2017

V2 – Last updated 5/3/2017

Big Data Ecosystem

File system	HDFS, NFS
Resource manager	Mesos, Yarn
Coordination	Zookeeper
Data Acquisition	Apache Flume, Apache Sqoop
Data Stores	MongoDB, Cassandra, Hbase, Project Voldemort
Data Processing	
• Frameworks	Hadoop MapReduce, Apache Spark, Apache Storm, Apache FLink
• Tools	Apache Pig, Apache HIve
• Libraries	SparkR, Apache Mahout, MlLib, etc
Data Integration	
• Message Passing • Managing data heterogeneity	Apache Kafka SemaGrow, Strabon
Operational Frameworks	
• Monitoring	Apache Ambari

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

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

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

Spark

- 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.

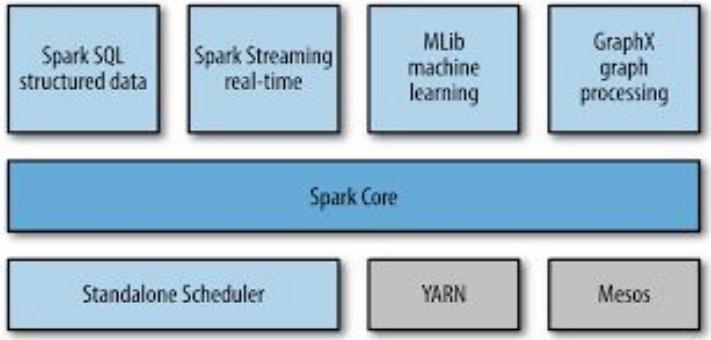
Spark

- 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

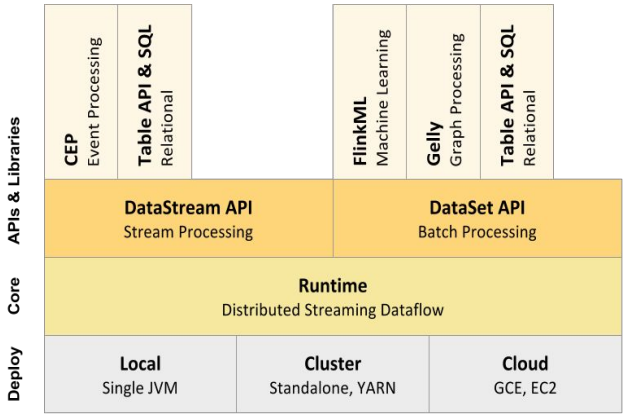
Apache Spark

- **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 structured
  - 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

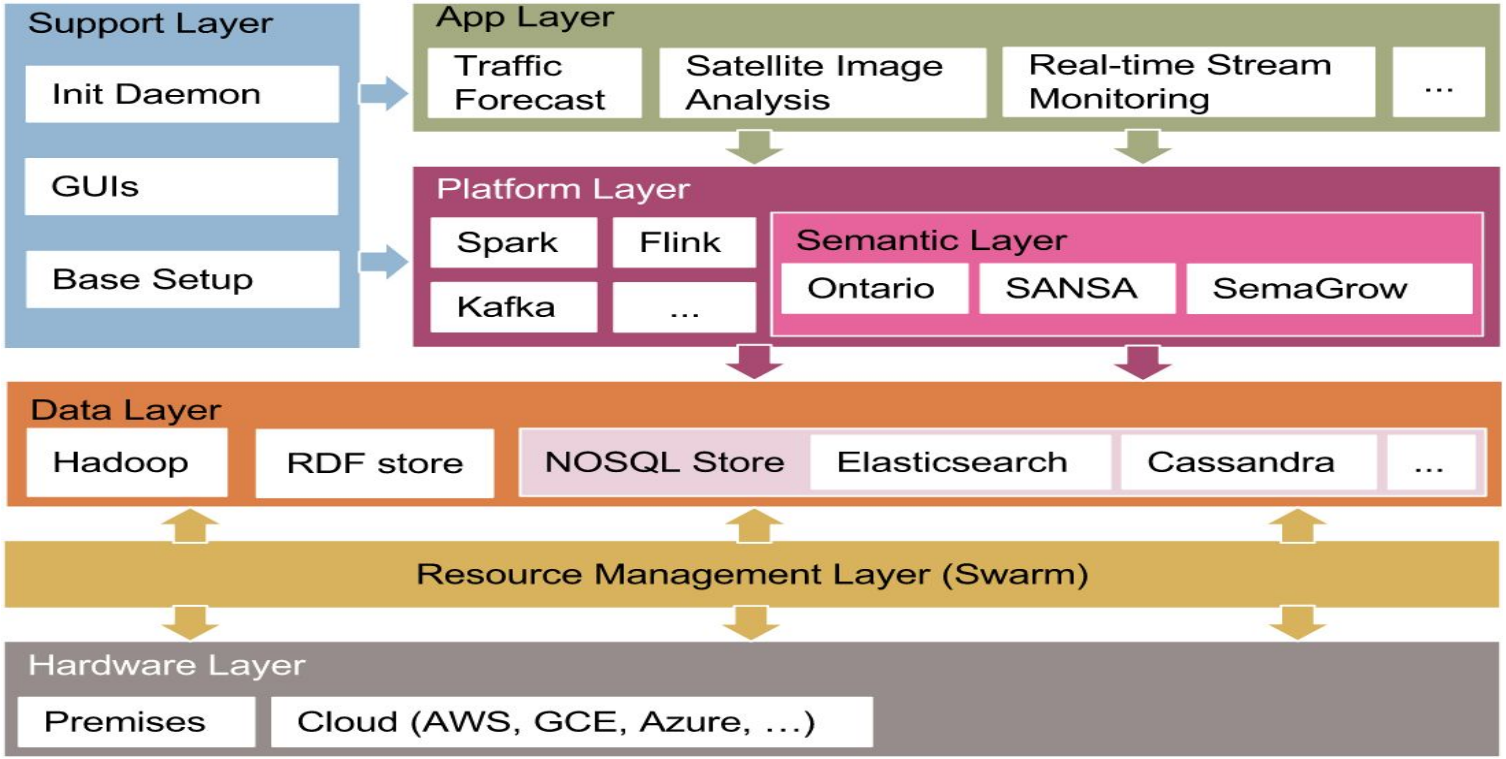


Big Data Projects

Big Data Projects

- Related to this lab
  - Big Data Europe
  - SANSA
  - Big Data Ocean
  - Boost4.0

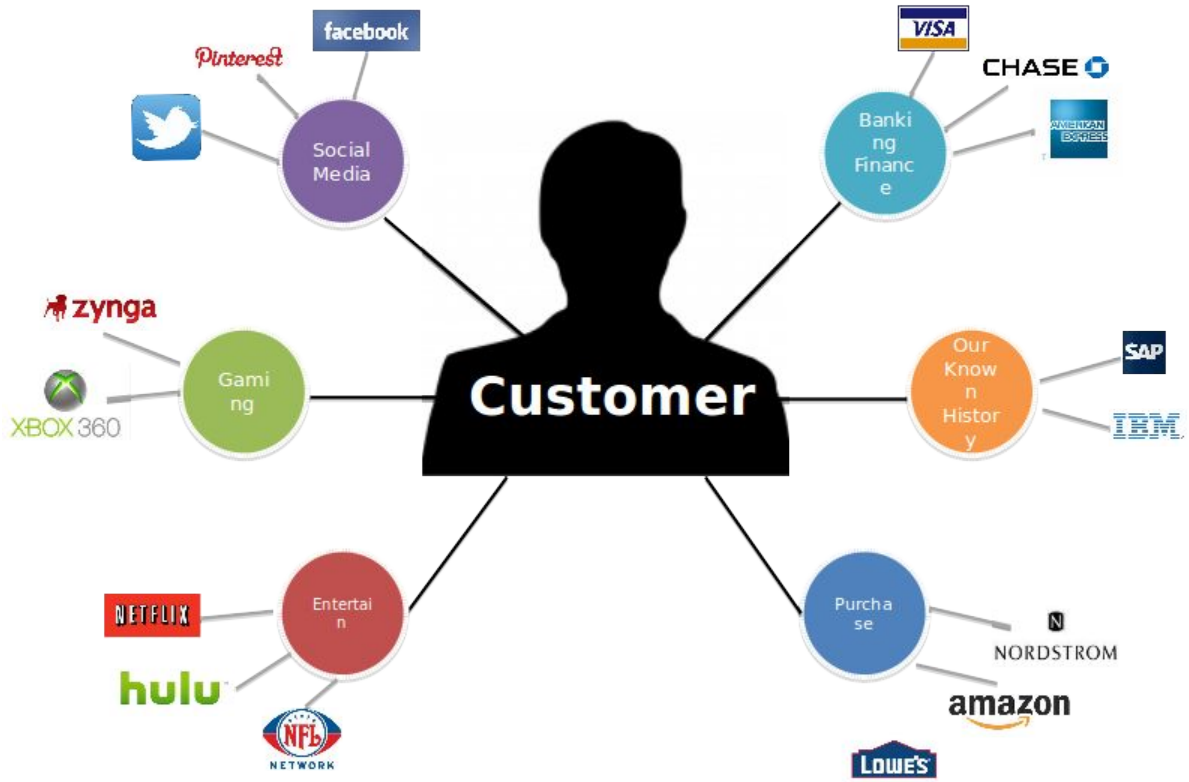
BDE Platform



Key Observation From BDE

- Heterogeneity AKA Variety

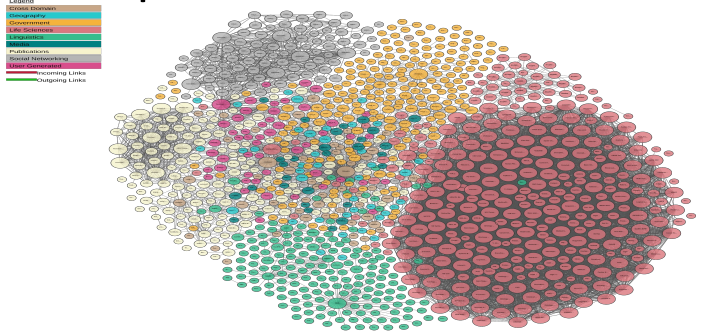
A Single View to the Customer



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

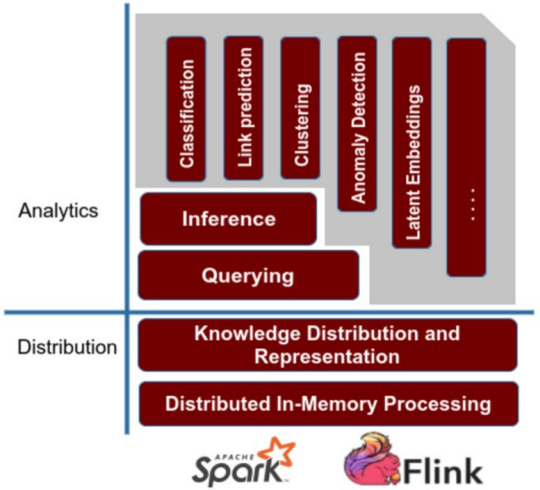


Source: LOD-Cloud (http://lod-cloud.net/)

- 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 analysis of RDF knowledge graphs



http://sansa-stack.net/

Introduction to Scala

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

Scala

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

Scala

- Everything is an object
  - Primitive types e.g. numbers, bool
  - Functions
- Numbers are objects:
  - 1+2*3 -> (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;
    }
println(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 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
```

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

Anonymous 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

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

UNIVERSITÄT BONN

SMART DATA ANALYTICS
FROM DATA TO KNOWLEDGE

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
Room 1.062 (Appointment per e-mail)

Gezim Sejdiu
sejdiu@cs.uni-bonn.de
Room 1.052 (Appointment per e-mail)