

R for HPC and big data

Georg Heiler

Agenda

- about me
- HPC vs. Big Data
- variants of big data
- hadoop ecosystem and its history
- spark in detail
- spark integrations with R
- LAB
- downsides of non native spark integrations (R, python)
- future of big data

Georg Heiler @geoheil

- BSC Thesis about clustering of time series
- anomaly detection project spark + R for music streaming data
- co-founder of [predictr](#)
- data scientist @t-mobile austria - master's thesis about fraud prevention

HPC parallelization - number crunching

- single threaded for loops by default `for(x <- 1...100000000){callAFunction}`
- additional packages (`foreach`, `doParallel`) help to parallelize but may be platform agnostic
- scaling out over multiple machines rather cumbersome (`mpi`, need to handle communication logic by application programmer = you)
- see [cran parallel taskView](#) for more parallelization packages
- `Rcpp` for seamless C++ integration => speed up loop

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

Volume

SCALE OF DATA

It's estimated that **2.5 QUINTILLION BYTES**

[2.3 TRILLION GIGABYTES]

of data are created each day



Most companies in the U.S. have at least

100 TERABYTES

[100,000 GIGABYTES]

of data stored

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

By 2014, it's anticipated there will be

420 MILLION WEARABLE, WIRELESS HEALTH MONITORS

4 BILLION+ HOURS OF VIDEO

are watched on YouTube each month



400 MILLION TWEETS

are sent per day by about 200 million monthly active users



The New York Stock Exchange captures

1 TB OF TRADE INFORMATION

during each trading session



Velocity

ANALYSIS OF STREAMING DATA

Modern cars have close to

100 SENSORS

that monitor items such as fuel level and tire pressure



By 2016, it is projected there will be

18.9 BILLION NETWORK CONNECTIONS

— almost 2.5 connections per person on earth



1 IN 3 BUSINESS LEADERS

don't trust the information they use to make decisions



Poor data quality costs the US economy around

\$3.1 TRILLION A YEAR



27% OF RESPONDENTS

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

Veracity

UNCERTAINTY OF DATA

40 ZETTABYTES

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

2020

2005

Volume

It's estimated that
2.5 QUINTILLION BYTES
[2.3 TRILLION GIGABYTES]
of data are created each day

The FOUR V's of Big

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

150 EXABYTES
[161 BILLION GIGABYTES]



Variety

By 2014, it's anticipated
there will be

**420 MILLION
WEARABLE, WIRELESS
HEALTH MONITORS**

**4 BILLION+
HOURS OF VIDEO**
are watched on
YouTube each month

Types of big data

- fits in one machine (TB of ram are cheap)
- raw data is too large for memory of single computer; aggregation allows for analysis on single computer
- data is really too big for a single machine and aggregation not helpful (distributed ML)
- connection of (small) data silos (enterprise)

By 2016, it is projected
there will be

**18.9 BILLION
NETWORK
CONNECTIONS**

— almost 2.5 connections
per person on earth



with 1.9 million in the United States



OF DATA

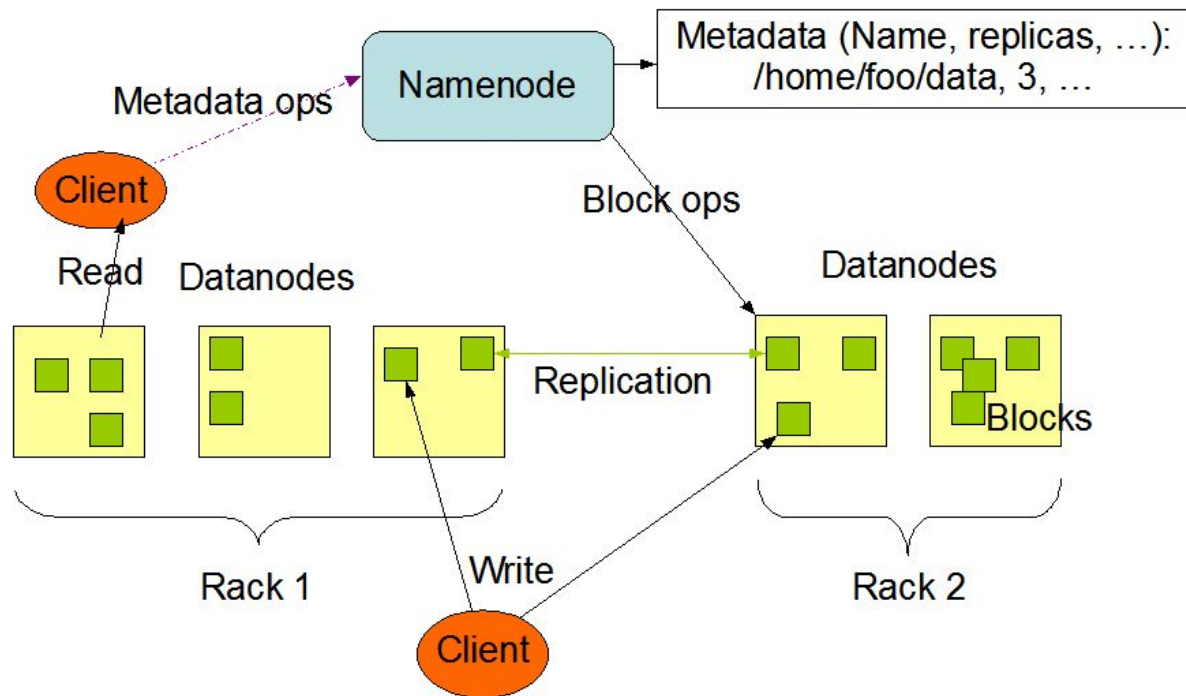
in one survey were unsure of
how much of their data was
inaccurate

Apache hadoop

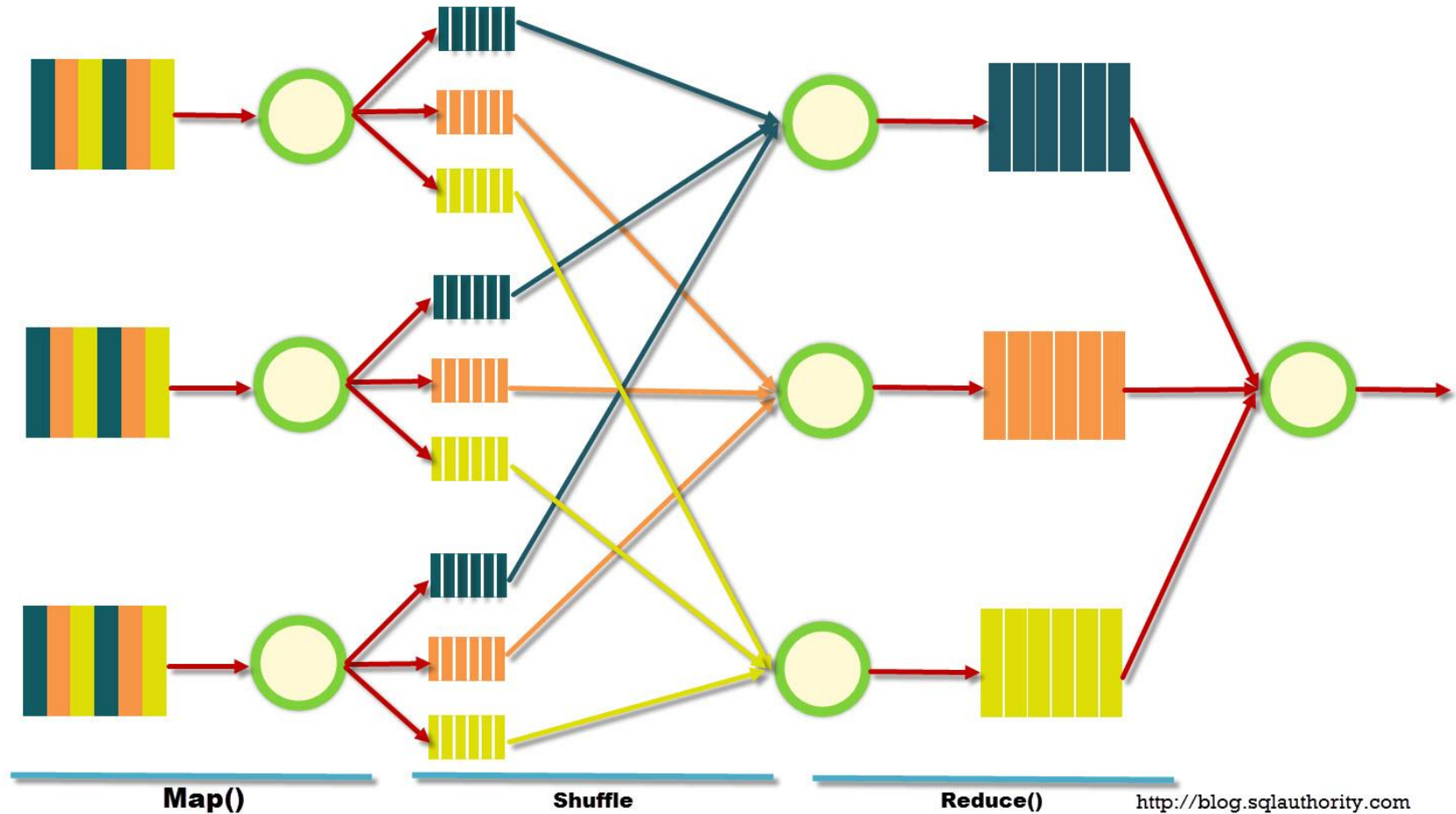


Hadoop Distributed File System (HDFS)

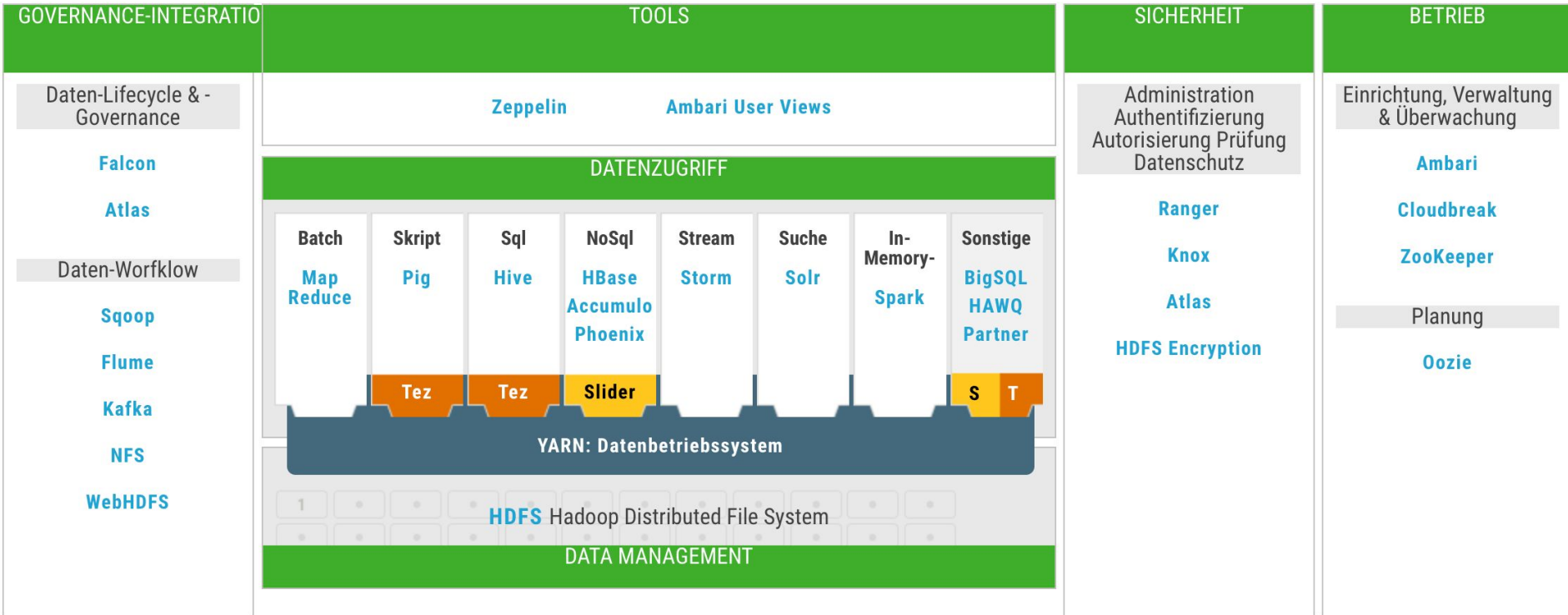
HDFS Architecture



How MapReduce Works?



Hadoop ecosystem



Spark concept

Resilient Distributed Datasets (RDD)

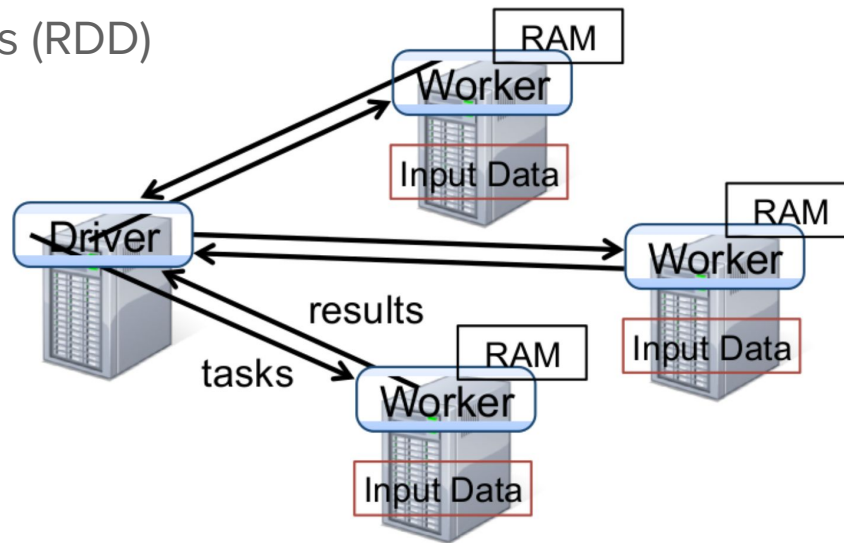


Figure 2: Spark runtime. The user's driver program launches multiple workers, which read data blocks from a distributed file system and can persist computed RDD partitions in memory.

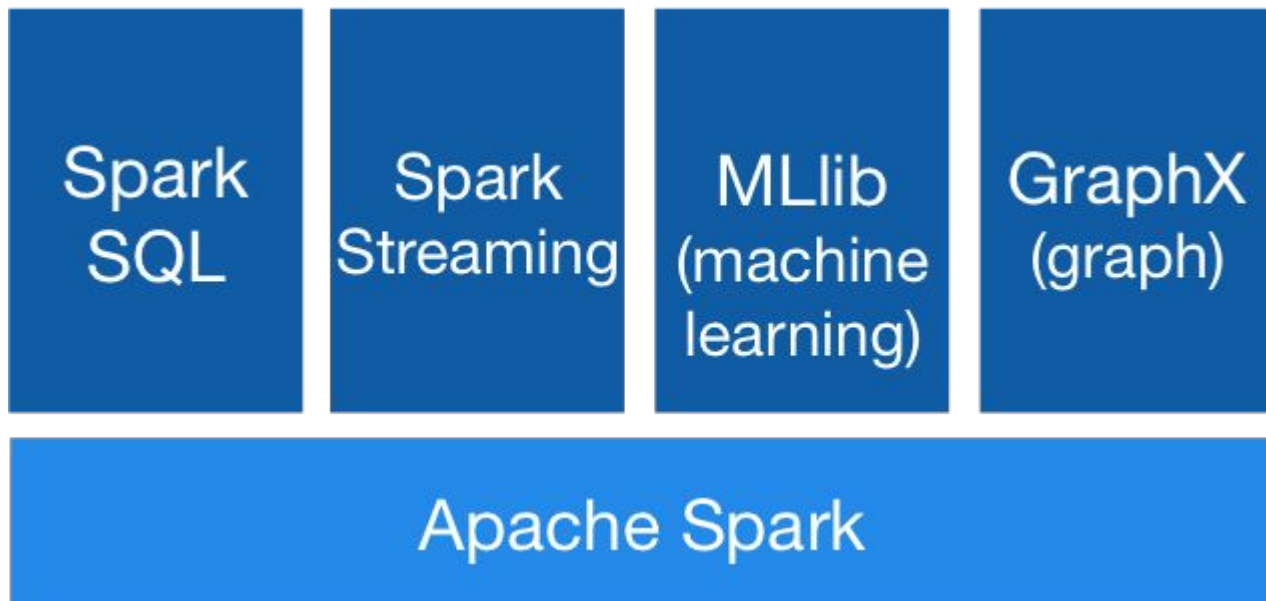
RDD properties

- resilient, distributed collections
- immutable
- transformations (map, filter, reduceByKey join,...)
- actions (reduce,collect,count,foreach,...)

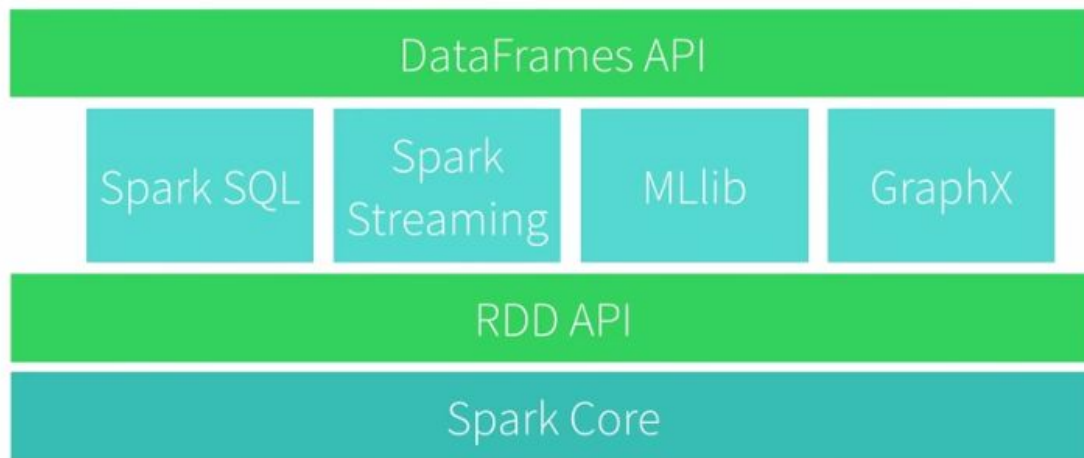
Word count

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


Apache Spark Stack



Apache Spark integrations



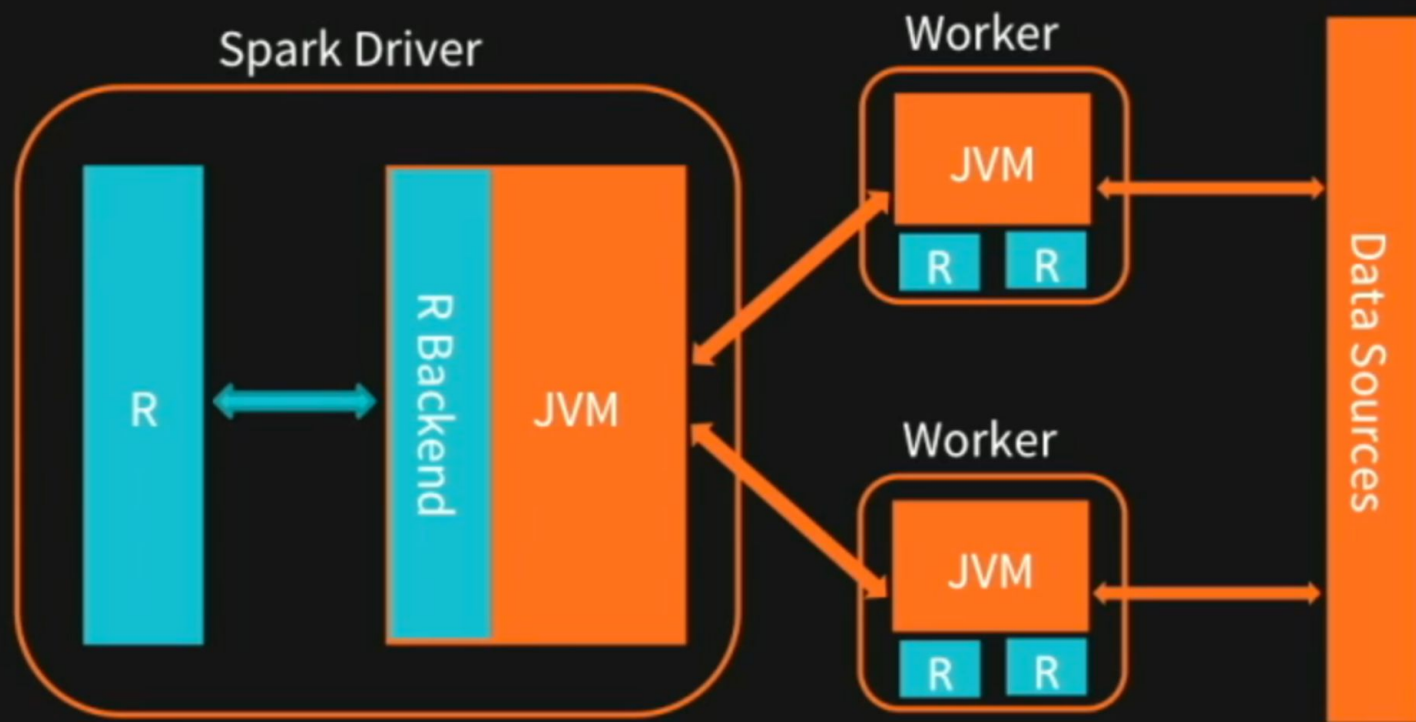
Integrations of R with Apache Spark

- sparkR
- sparklyR
- unix pipe

sparkR package

- part of main spark project
- dataframe api (SQL like)
- UDFs
 - **dapply** for each partition of a sparkDataFrame
 - **gapply** for each group of a sparkDataFrame
- run local R function distributed
 - **spark.lapply** (like doParallel or lapply)

SparkR architecture (2.x)



sparkDataFrame operations

```
#Create the SparkDataFrame
```

```
df <- as.DataFrame(faithful)
```

```
#Filter to retain rows with wait times shorter than 50 mins
```

```
head(filter(df,df$waiting<50))
```

```
# Aggregate and count the number of times each waiting time  
appears
```

```
head(summarize(groupBy(df,df$waiting),count=n(df$waiting)))
```


sparkDataFrame machine learning

```
irisDF<-suppressWarnings(createDataFrame(iris))
```

```
#Fit a generalized linear model of family "gaussian" with spark.glm
```

```
gaussianDF <- irisDF gaussianTestDF <- irisDF gaussianGLM <- spark.glm(gaussianDF,  
Sepal_Length~Sepal_Width+Species, family="gaussian")
```

```
#Modelsummary
```

```
summary(gaussianGLM)
```

```
#Prediction
```

```
showDF(predict(gaussianGLM,gaussianTestDF))
```

Why sparklyR

tidy universe

unique grammar for data
manipulation helping
reproducible research

```
library(dplyr)
```

```
starwars %>%
```

```
  filter(species ==  
         "Droid")
```



lab

sparklyR, java, dplyR, <https://www.rstudio.com> (own laptop)

databricks community cloud <https://databricks.com/product/faq/community-edition> (cloud)

lab code

<https://github.com/RSummerSchool/R-for-HPC-and-big-data>

Q&A

open questions for the lab?

downsides of non spark-native languages (R, python)

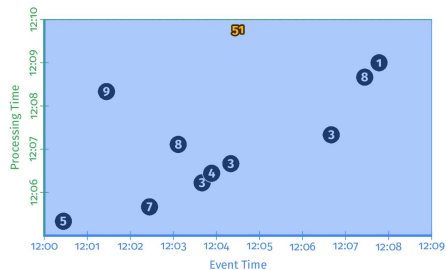
- don't get the latest features (ml, ml-pipelines, streaming ...)
- exceptions will occur on multiple layers
- performance bottlenecks (serialization)
- deployment on local cluster (!)
- no good integration to [spark-packages](#)

future of big data

R and streaming analytics

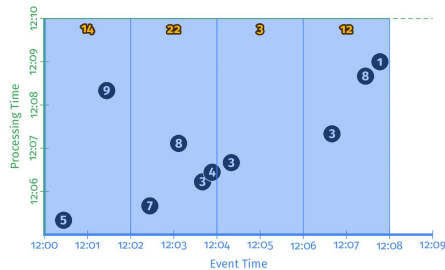
- no spark integration
- PMML

BEAM: converging big data engines **What Where When How**



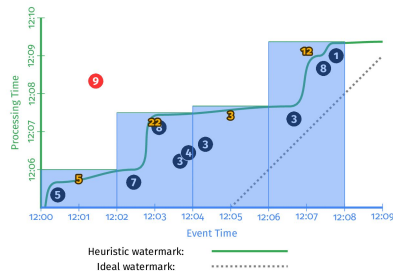
1

**Classic
Batch**



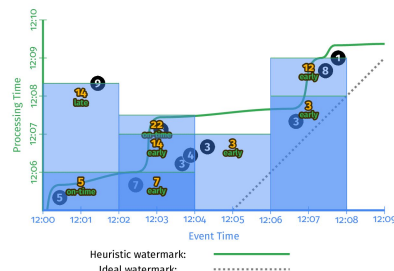
2

**Windowed
Batch**



3

Streaming



4

**Streaming
+ Accumulation**

A Containerized Data Lake

Pachyderm lets you store and analyze your data using containers.

Learn More

Try on GitHub

Pachyderm v1.4 is out and ready for production use!

Pachyderm is a data lake that offers complete **version control for data** and leverages the container ecosystem to provide **reproducible data processing**.

Version control for data

Pachyderm version controls all your data, similar to what Git does with code. You can view diffs of your data and collaborate with teammates using Pachyderm commits and branches. [Learn more](#) →



R/python

- Reproducibility
- Data Provenance
- Collaboration
- Incrementality
- Data Scientist
- Autonomy
- Infrastructure
- Agnostic
- less complexity
- a different package in same direction: [partools](#)

Q&A

Georg Heiler | @geoheil

get in contact if you are interested in an internship or thesis with T-Mobile
Austria



H₂O Prediction Engine

SDK/API

Rapids Query R-engine

Nano Fast Scoring Engine

In-Mem Map Reduce
Distributed fork/join

Memory Manager
Columnar Compression

Deep Learning

Cluster	Classify	Regression	Trees	Boosting	Forests	Solvers	Gradients
---------	----------	------------	-------	----------	---------	---------	-----------

Ensembles



HDFS

S3

SQL

NoSQL

R APIs for Spark

- SparkR
- ASF, Apache-licensed
- Ships with Apache-Spark since 1.4x
- SparkSQL and SparkML support through RPC
- UDF support through gapply, dapply, spark.lapply

- sparklyr
- On CRAN/github (0.5)
- Apache-licensed
- SparkSQL and SparkML support through DBI, dplyr and RPC
- UDF support announced ([PR#78](#))
- Remote execution with Livy
- Extensions!

- RxSpark
- Commercially licensed
- Deploy RevoScaleR in Spark clusters
- Interop with sparklyr
- PEMA support
- Consume Parquet, and Hive Tables
- Robust UDF support with rxDataStep, rxExec, rxExecBy, and foreach
- Current version: 9.1

CRAN-R

- a.k.a., GNU-R
- Single-threaded
- In-memory
- 10K+ packages
- Interfaces to C++, C, and Fortran for speed
- Cross-platform

MRO

- GNU-R + Intel MKL = multi-threaded linear algebra
- 100% CRAN-R compatible
- Reproducible R toolkit with the checkpoint package
- Open Source

R Client

- HPA with the RevoScaleR and MicrosoftML
- Multi-threaded support (two cores)
- Deployment with mrsdeploy
- Free, Win + Linux, Docker image on my github

In-memory distributions

Microsoft R Server

- Enterprise-class high-performance analytics distribution
- *Parallel external memory algorithms* with the RevoScaleR library
 - Scale to all compute resources
- *Out-of memory* computation with the XDF file format
- Deployment capabilities through the mrsdeploy package
- *Commercial support*
- Available in Linux (Red Hat, CentOS), Windows, *SQL Server, Spark, Hadoop and Teradata*
- Battle-tested Microsoft Research libraries for *state-of-the-art machine learning* with MicrosoftML

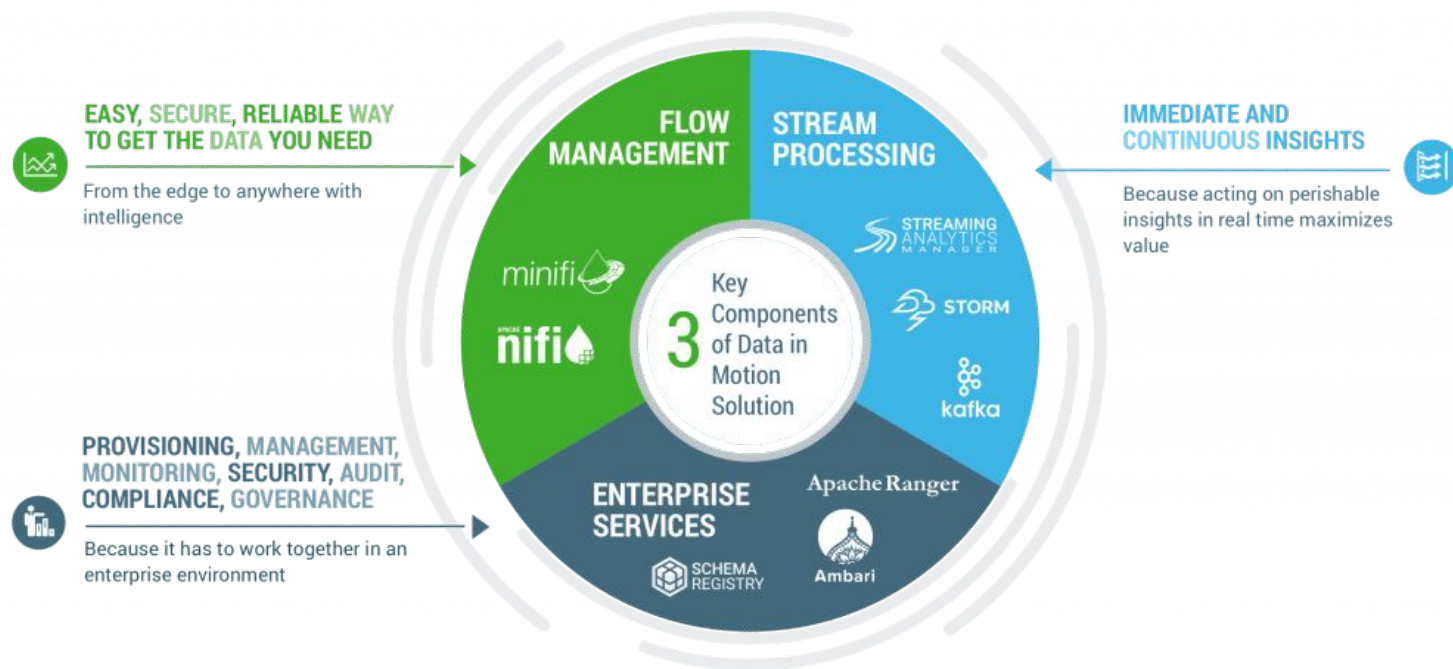
R cloud native

```
library(doAzureParallel)

# generate a credentials json file "cluster", then...
registerDoAzureParallel(cluster)

# Run 5 million option pricing simulations
closingPrices <- foreach(i = 1:50, .combine='c') %dopar% {
  replicate(100000, getClosingPrice())
}
```

Hadoop streaming



Machine Learning on spark with R

h2o.ai / [rsparkling](https://github.com/h2oai/rsparkling)

[microsoftR](https://github.com/microsoftR)

Some interesting links

- <https://longhowlam.wordpress.com/2017/02/15/r-formulas-in-spark-and-un-nesting-data-in-sparklyr-nice-and-handly/>
- <https://spark-summit.org/2017/events/extending-the-r-api-for-spark-with-sparklyr-and-microsoft-r-server/>
- <https://spark-summit.org/2017/events/apache-sparkr-under-the-hood-how-to-debug-your-sparkr-applications/>
- <http://tfe/2017/05/managing-spark-data-handles-in-r/>
- <https://de.slideshare.net/HadoopSummit/the-next-generation-of-data-processing-and-open-source>
- https://docs.google.com/presentation/d/1SHie3nwe-pqmjGum_QDznPr-B_zXCjJ2VBDGdafZme8
- <http://multithreaded.stitchfix.com/blog/2017/06/15/beware-r-in-production/>
- <http://www.win-vector.com/blog/2017/07/working-with-r-and-big-data-use-replyr/>
- <https://blogs.msdn.microsoft.com/rserver/2017/05/04/performance-rxexecby-vs-gapply-on-spark/>