Supercharging R with Apache Spark

Hossein Falaki
@mhfalaki

databricks™

About Apache Spark and Databricks

Apache Spark is a general distributed computing engine that unifies:

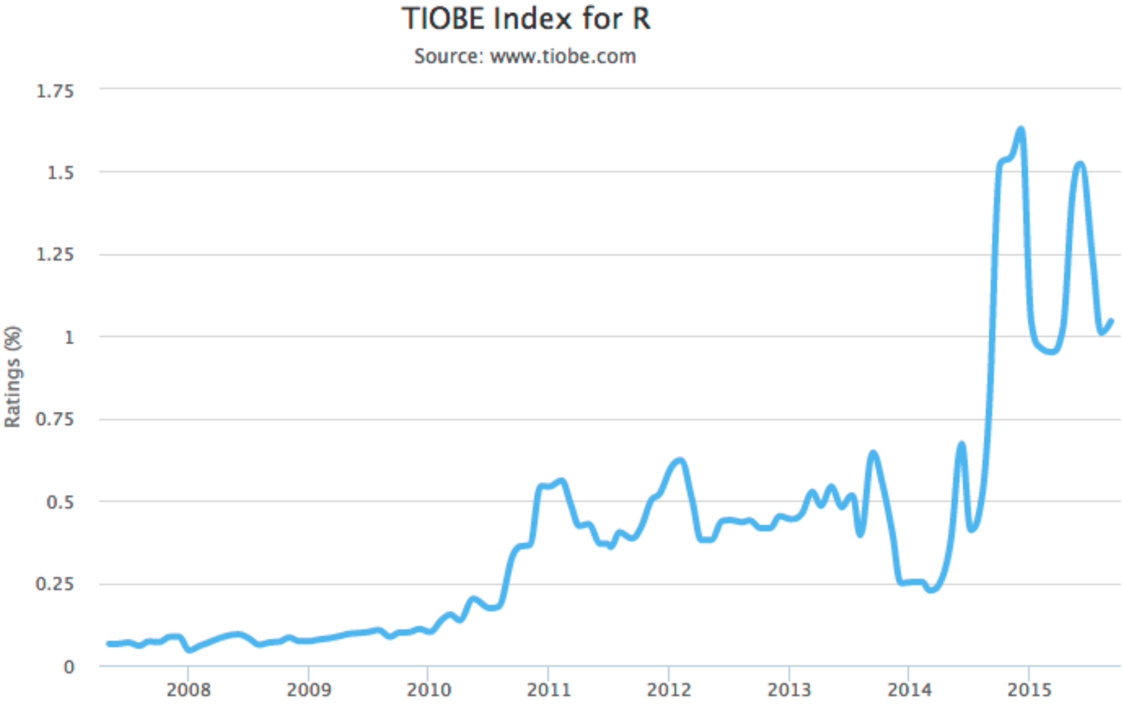
- Real-time streaming (Spark Streaming)
- Machine learning (SparkML/MLLib)
- SQL (SparkSQL)
- Graph processing (GraphX)

Databricks Inc. is a company founded by creators of Spark focused on making big data simple by offering an end to end data processing platform in the cloud

What is R?

Language and runtime

The corner stone of R is the data frame concept



Many data scientists love R

- Open source
- Highly dynamic
- Interactive environment
- Rich ecosystem of packages
- Powerful visualization infrastructure
- Data frames make data manipulation convenient
- Taught by many schools to stats and computing students

Performance Limitations of R

R language

- R’s dynamic design imposes restrictions on optimization

R runtime

- Single threaded
- Everything has to fit in memory

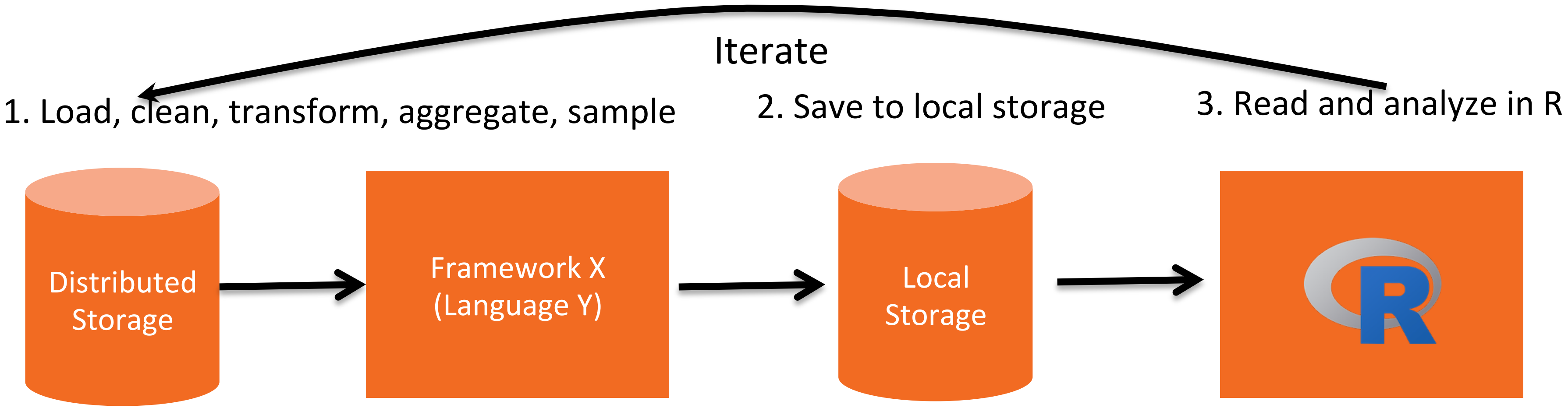
What would be ideal?

Seamless manipulation and analysis of very large data in R

- R’s flexible syntax
- R’s rich package ecosystem
- R’s interactive environment
- Scalability (scale up and out)
- Integration with distributed data sources / storage

Augmenting R with other frameworks

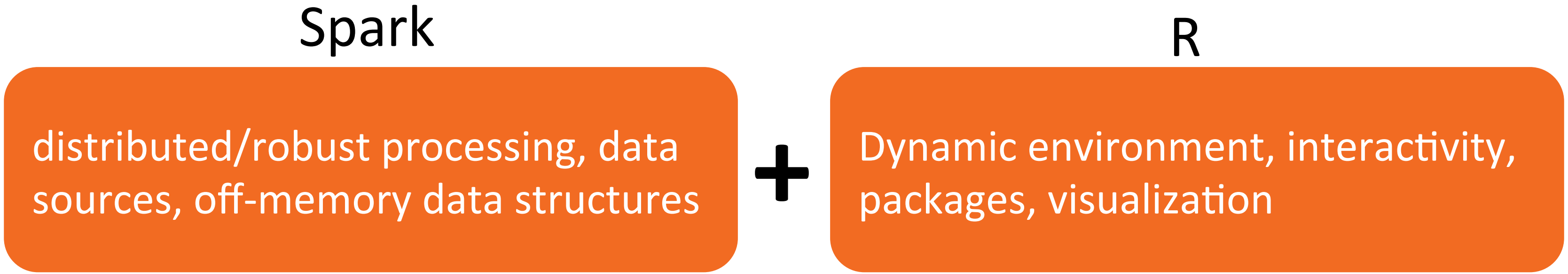
In practice data scientists use R in conjunction with other frameworks (Hadoop MR, Hive, Pig, Relational Databases, etc)



What is SparkR?

An R package distributed with Apache Spark:

- Provides R frontend to Spark
- Exposes Spark Dataframes (inspired by R and Pandas)
- Convenient interoperability between R and Spark DataFrames



Spark

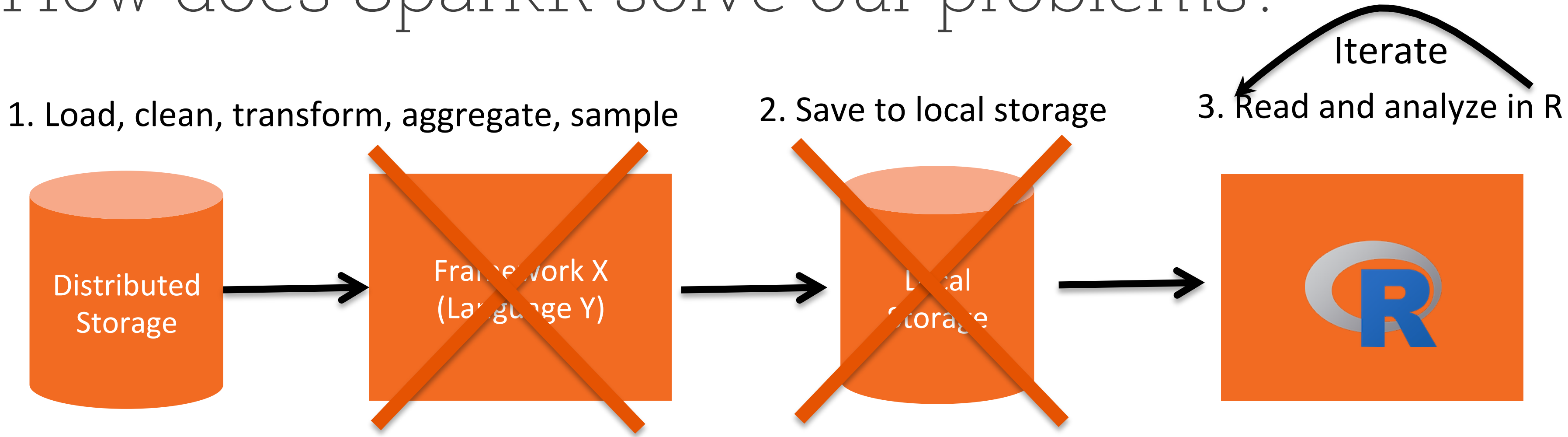
distributed/robust processing, data sources, off-memory data structures

+

R

Dynamic environment, interactivity, packages, visualization

How does SparkR solve our problems?



No local storage involved

Write everything in R

Use Spark's distributed cache for interactive/iterative analysis at speed of thought

Example SparkR program

```
# Loading distributed data
df <- read.df(“hdfs://bigdata/logs”, source = “json”)

# Distributed filtering and aggregation
errors <- subset(df, df$type == “error”)
counts <- agg(groupBy(errors, df$code), num = count(df$code))

# Collecting and plotting small data
qplot(code, num, data = collect(counts), geom = "bar", stat =
   "identity") + coord_flip()
```

Overview of SparkR API

IO

- read.df / write.df
- createDataFrame

Caching

- cache / persist / unpersist
- cacheTable / uncacheTable

Utility functions

- dim / head / take
- names / rand / sample

ML Lib

- glm / predict

DataFrame API

select / subset / groupBy

head / collect / showDF

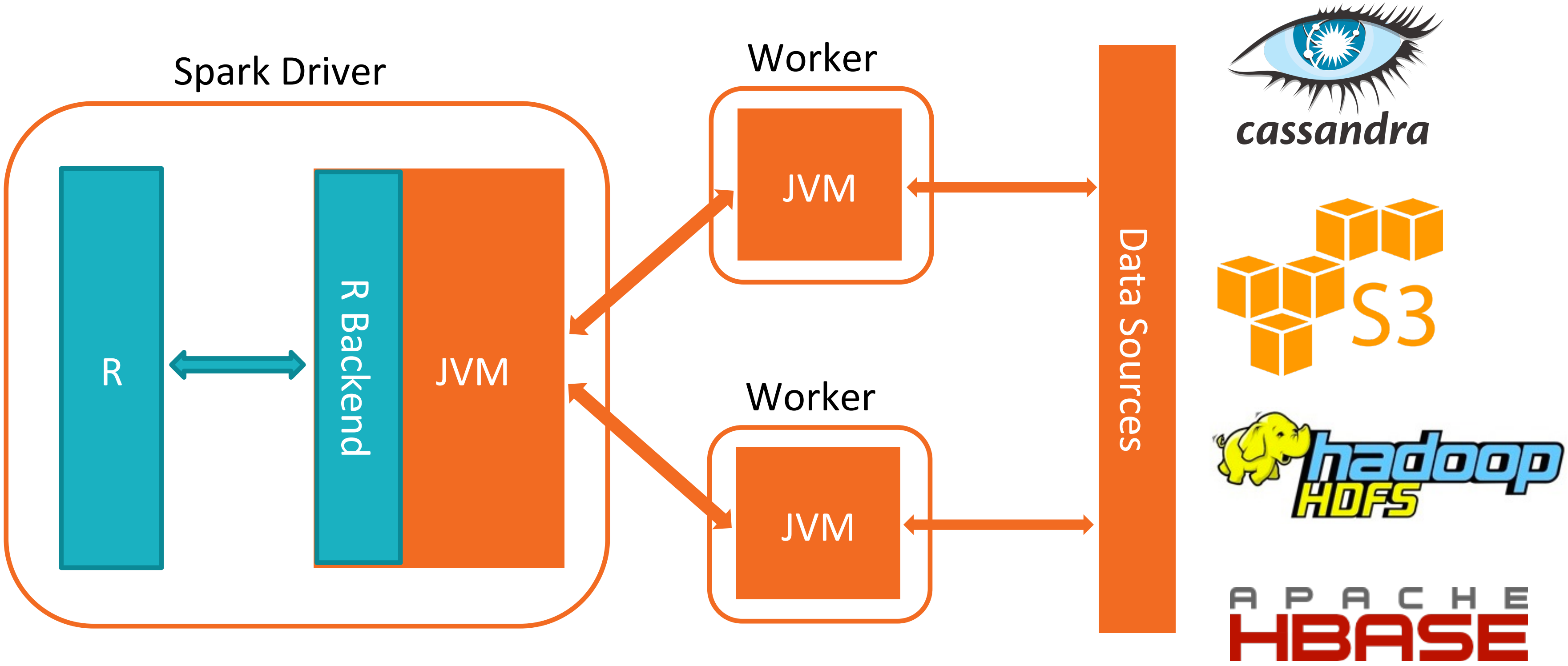
unionAll / agg / avg / column

SQL

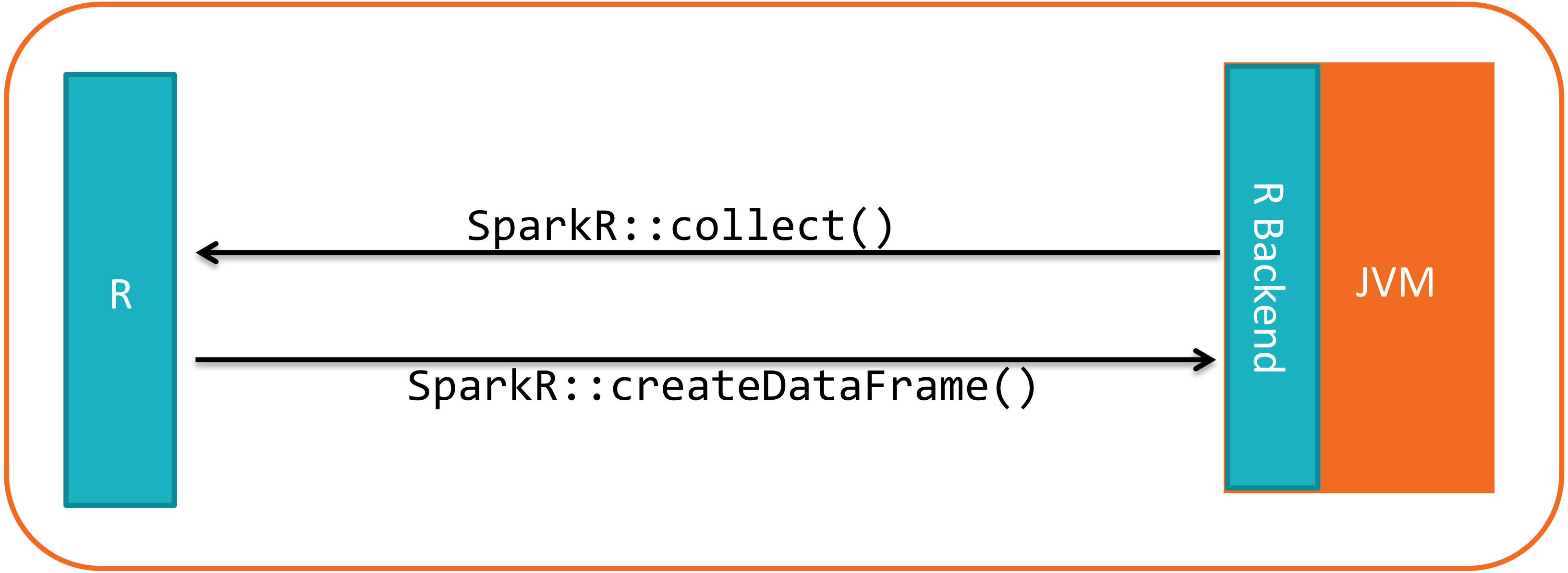
sql / table / saveAsTable

registerTempTable / tables

SparkR architecture

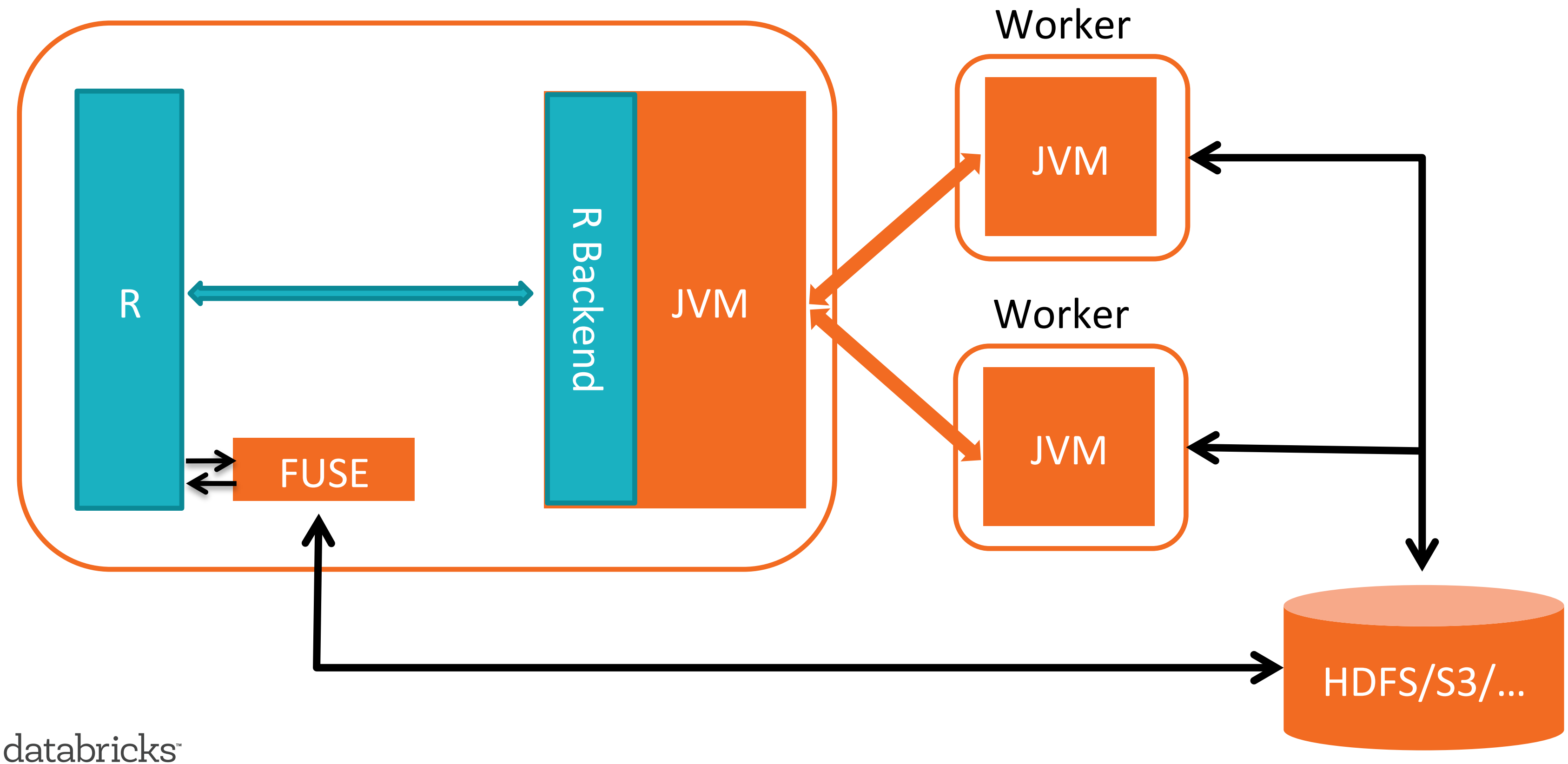


Moving data between R and JVM

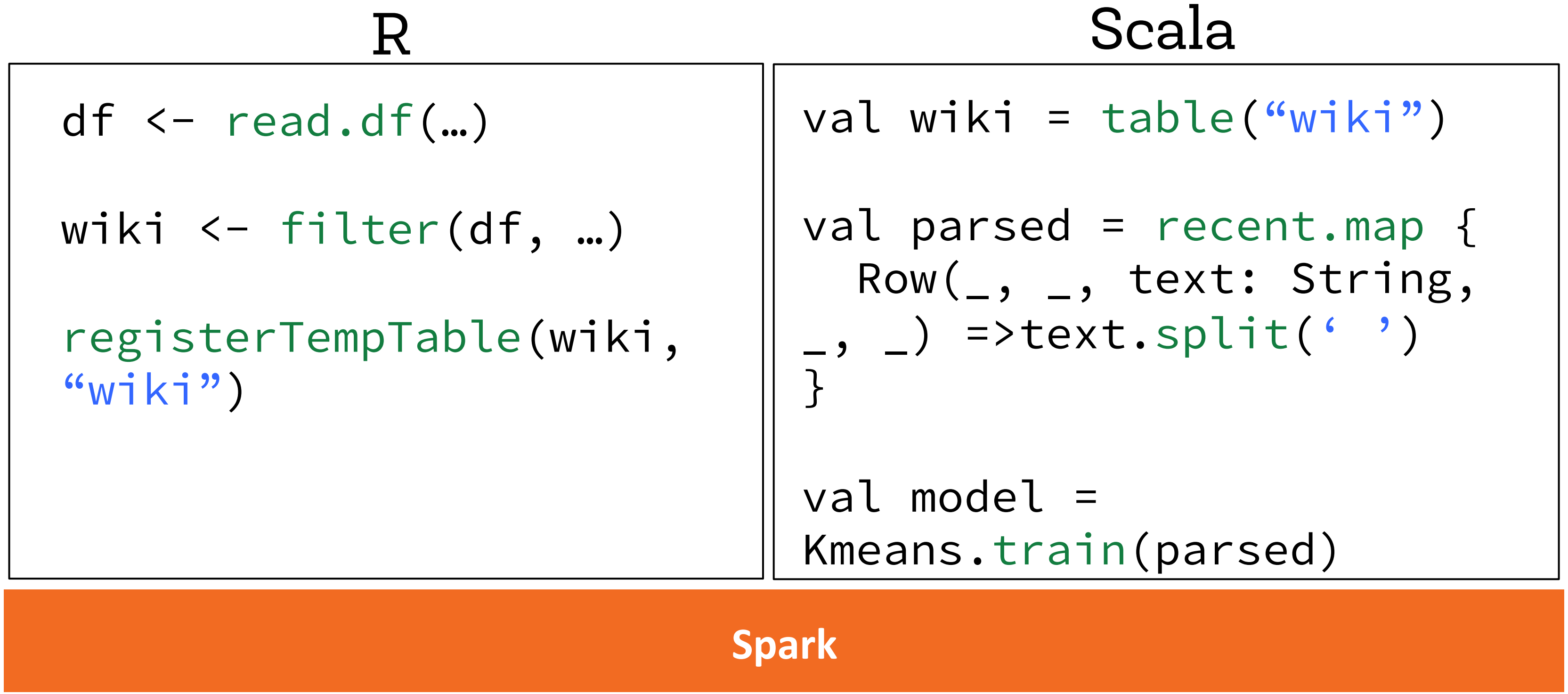


databricks™

Moving data between R and JVM



Moving between languages (notebooks)



R

```
df <- read.df(…)

wiki <- filter(df, …)

registerTempTable(wiki,
“wiki”)
```

Scala

```
val wiki = table(“wiki”)

val parsed = recent.map {
  Row(_, _, text: String,
_, _) =>text.split(‘ ’)
}

val model =
Kmeans.train(parsed)
```

Spark

Example use case: exploratory analysis

- Data pipeline implemented in Scala/Python
- New files are appended to existing data partitioned by time
- Table scheme is saved in Hive metastore
- Data scientists use SparkR to analyze and visualize data
  1. `refreshTable(sqlConext, “logsTable”)`
  2. `logs <- table(sqlContext, “logsTable”)`
  3. Iteratively analyze/aggregate/visualize using Spark & R DataFrames
  4. Publish/share results

Demo

Summary

1. SparkR is an R frontend to Apache Spark
2. Distributed data resides in the JVM
3. Workers are not running R process (yet)
4. Distinction between Spark DataFrames and R data frames

Further pointers

http://spark.apache.org

http://www.r-project.org

http://www.ggplot2.org

https://cran.r-project.org/web/packages/magrittr

https://databricks.com/blog/2015/09/22/large-scale-topic-modeling-improvements-to-lda-on-spark.html

www.databricks.com

Office hour: 2:55 – 3:35 at Table A (O'Reilly Booth) / Databricks booth

Thank you

Spark

databricks™