R package *sparklyr*

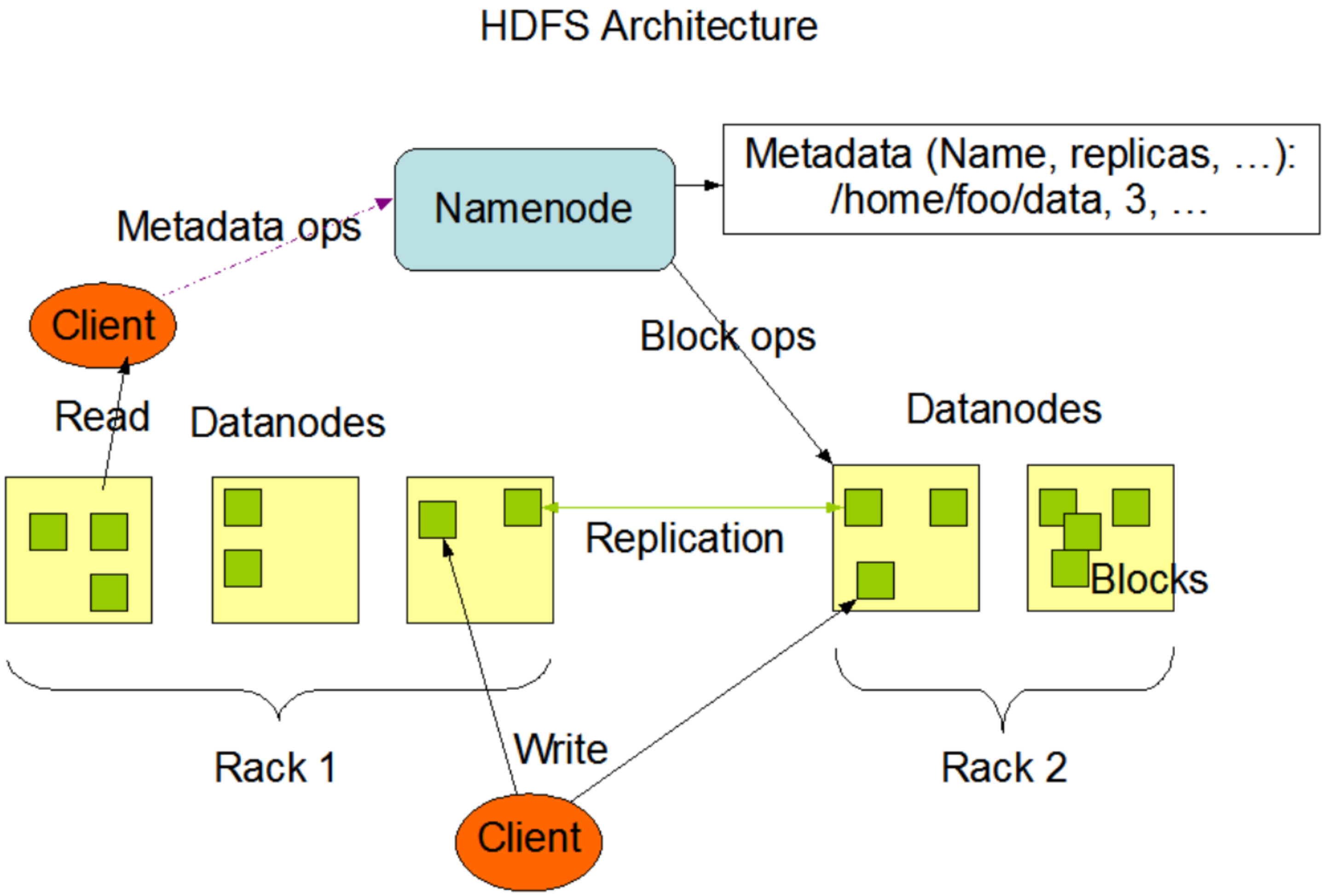
Roland Boubela

October 19, 2016

What is Apache Spark?

Apache Hadoop

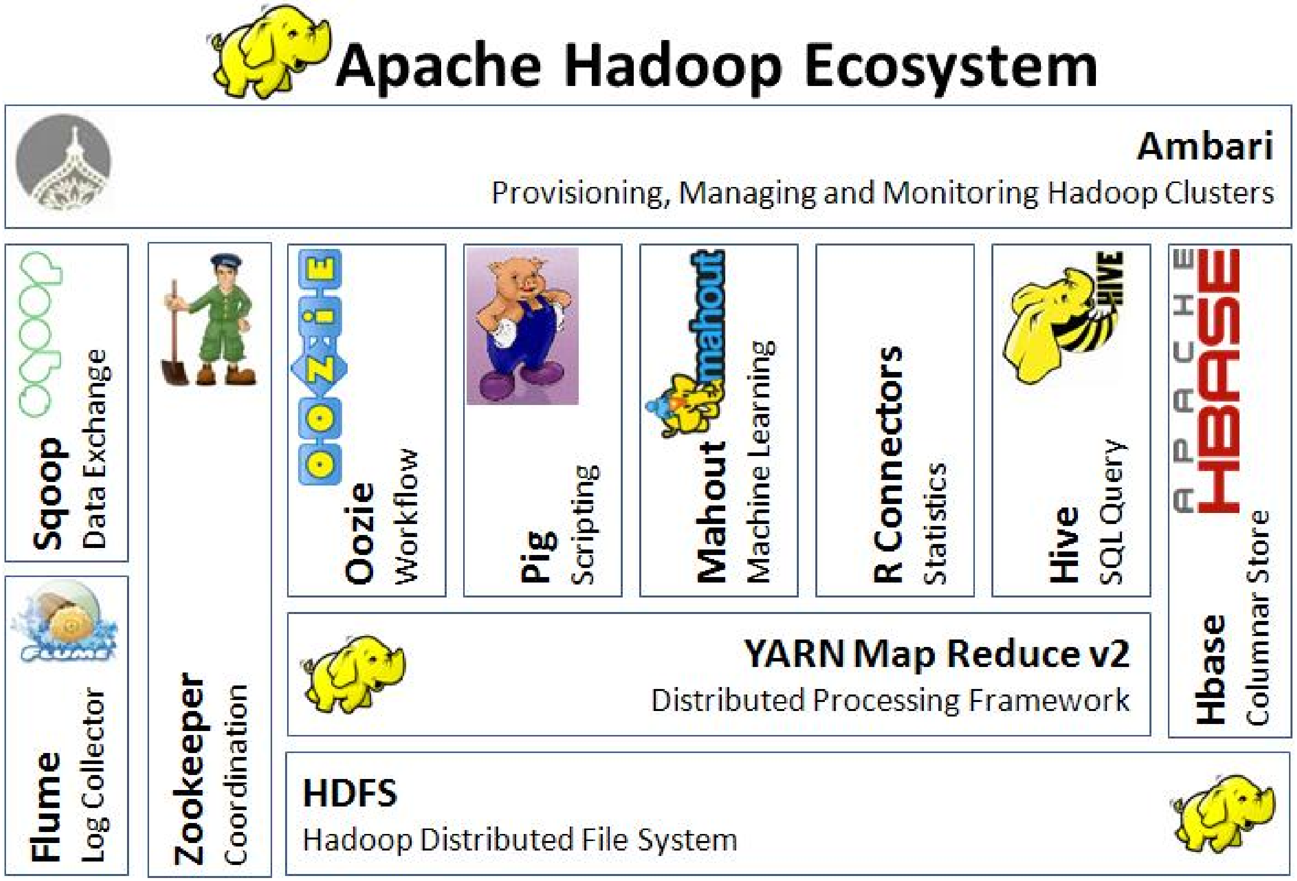
Hadoop Distributed Filesystem (HDFS)



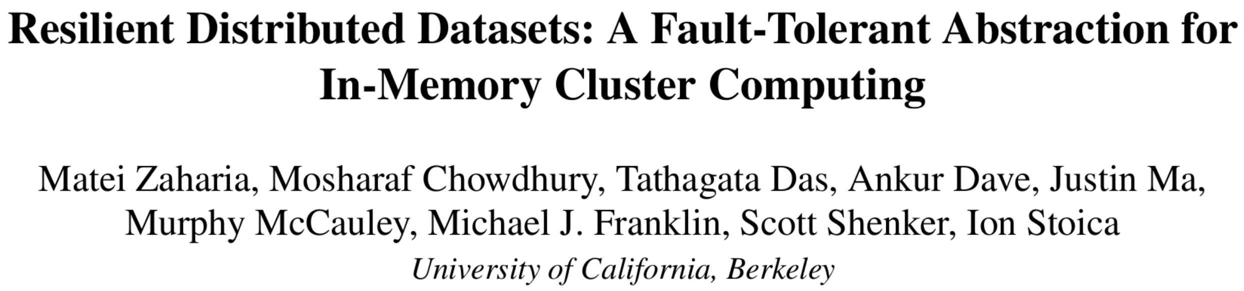
Map Reduce Pardigm

- MapReduce: Simplified Data Processing on Large Clusters (Dean and Gemawat 2004, OSDI)
- Splits computations in map and reduce phase
- Handles
  - Details of input data partitioning
  - Scheduling program’s execution across a set of machines
  - Machine failures
  - Required inter-machine communication

Hadoop Ecosystem



Resilient Distributed Datasets



Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing

Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma,
Murphy McCauley, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley

Spark Concept

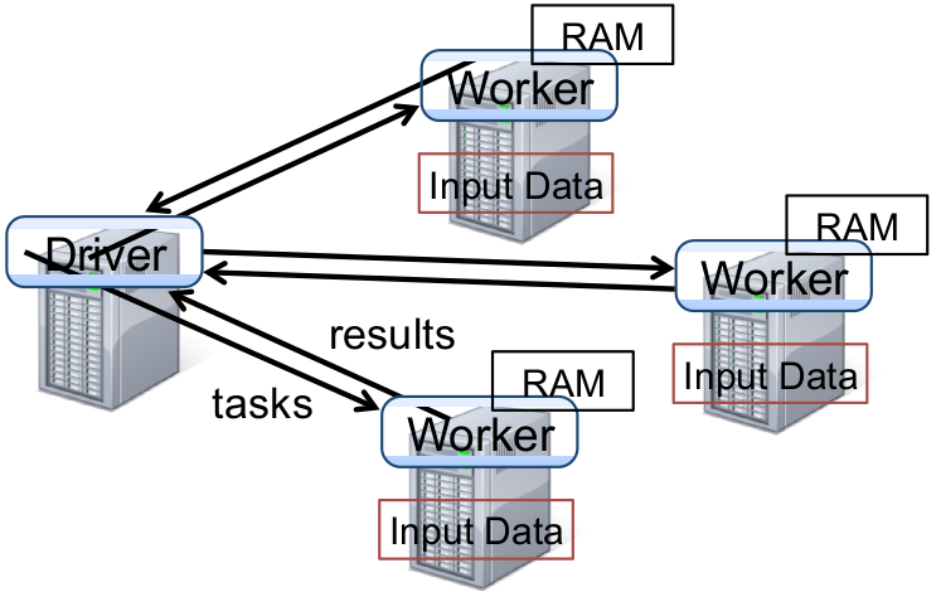
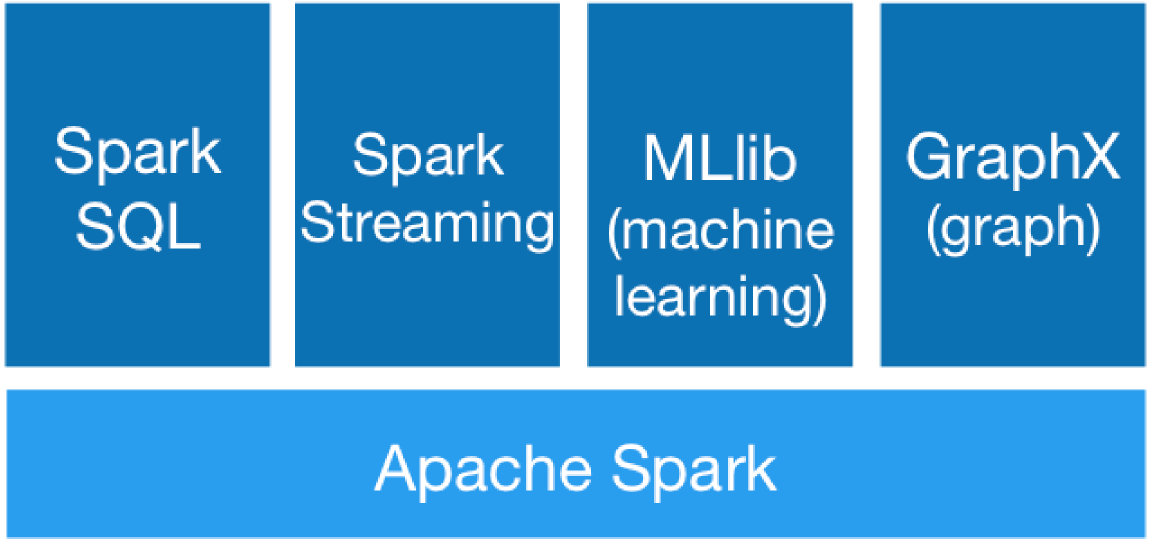


Figure 2: Spark runtime. The user's driver program launches multiple workers, which read data blocks from a distributed file system and can persist computed RDD partitions in memory.

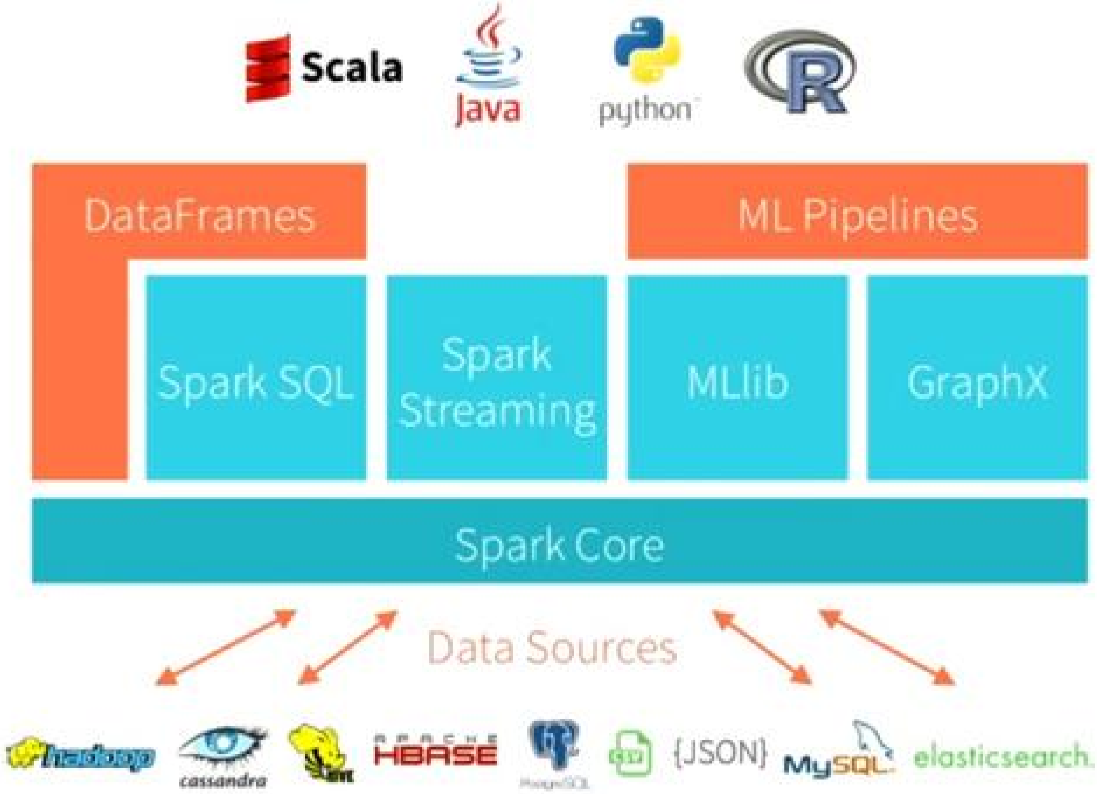
RDD Properties

- Resilient, distributed collections
- Immutable
- Transformations
    - map, filter, reduceByKey, join, …
- Actions
    - reduce, collect, count, foreach, …

Apache Spark Stack



Apache Spark



There is SparkR

SparkR Package

- Part of the Apache Spark project
- Main feature: *SparkDataFrame* operations
- Hive support
- Applying user-defined function
  - **dapply** (to each partition of a SparkDataFrame)
  - **gapply** (to each group of a SparkDataFrame)
- Running local R functions distributed
  - **spark.lapply** (like *doParallel* or *lapply*)

SparkDataFrame Operations

```
# Create the SparkDataFrame
df <- as.DataFrame(faithful)

# Get basic information about the SparkDataFrame
df
## SparkDataFrame[eruptions:double, waiting:double]

# Select only the "eruptions" column
head(select(df, df$eruptions))
##  eruptions
##1     3.600
##2     1.800
##3     3.333
```

SparkDataFame Operations ctd.

```
# You can also pass in column name as strings
head(select(df, "eruptions"))

# Filter the SparkDataFrame to only retain rows
# with wait times shorter than 50 mins
head(filter(df, df$waiting < 50))
##  eruptions waiting
##1     1.750      47
##2     1.750      47
##3     1.867      48
```

SparkDataFame Operations: Grouping

```
# We use the `n` operator to count the number of times
# each waiting time appears
head(summarize(groupBy(df, df$waiting), count = n(df$waiting)))
##  waiting count
##1      70     4
##2      67     1
##3      69     2

# We can also sort the output from the aggregation
# to get the most common waiting times
waiting_counts <- summarize(groupBy(df, df$waiting),
                            count = n(df$waiting))
head(arrange(waiting_counts, desc(waiting_counts$count)))
##   waiting count
##1      78    15
##2      83    14
##3      81    13
```

Spark MLlib

```
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")

# Model summary
summary(gaussianGLM)

# Prediction
gaussianPredictions <- predict(gaussianGLM, gaussianTestDF)
showDF(gaussianPredictions)
```

Why *sparklyr*?

Getting Started

```
install.packages("sparklyr")

library(sparklyr)
spark_install(version = "1.6.2")
```

RStudio Integration

```
library(sparklyr)
sc <- spark_connect(master = "local")
```

- demo

dplyr Interface to SparkSQL

```r
library(dplyr)
iris_tbl <- copy_to(sc, iris)
flights_tbl <- copy_to(sc, nycflights13::flights, "flights")
batting_tbl <- copy_to(sc, Lahman::Batting, "batting")

src_tbls(sc)

# filter by departure delay
flights_tbl %>% filter(dep_delay == 2)
```

dplyr in Action

```
delay <- flights_tbl %>%
  group_by(tailnum) %>%
  summarise(count = n(),
            dist = mean(distance),
            delay = mean(arr_delay)) %>%
  filter(count > 20,
         dist < 2000,
         !is.na(delay)) %>%
  collect()

# plot delays
library(ggplot2)
ggplot(delay, aes(dist, delay)) +
  geom_point(aes(size = count), alpha = 1/2) +
  geom_smooth() +
  scale_size_area(max_size = 2)
```

Using SQL

```
library(DBI)

iris_preview <- dbGetQuery(sc, "SELECT * FROM iris LIMIT 10")
iris_preview
```

Machine Learning

- Spark MLlib functionality
- Distributed machine learning using **H2O Sparkling Water**
    - *rsparkling*
    - *h2o*
- . . . another meetup session

Summary

SparkR vs. sparklyr

- SparkR
  - spark.lapply
- sparklyr
  - easy installation of Spark
  - dplyr interface
  - h2o, rsparkling (should also work with SparkR)

- **Thanks for your attention!**
- ?