Parallelizing Existing R Packages with SparkR

Hossein Falaki
@mhfalaki

databricks™

About me

- Former Data Scientist at Apple Siri
- Software Engineer at Databricks
- Started using Apache Spark since version 0.6
- Developed first version of Apache Spark CSV data source
- Worked on SparkR &Databricks R Notebook feature
- Currently focusing on R experience at

What is SparkR?

An R package distributed with Apache Spark:

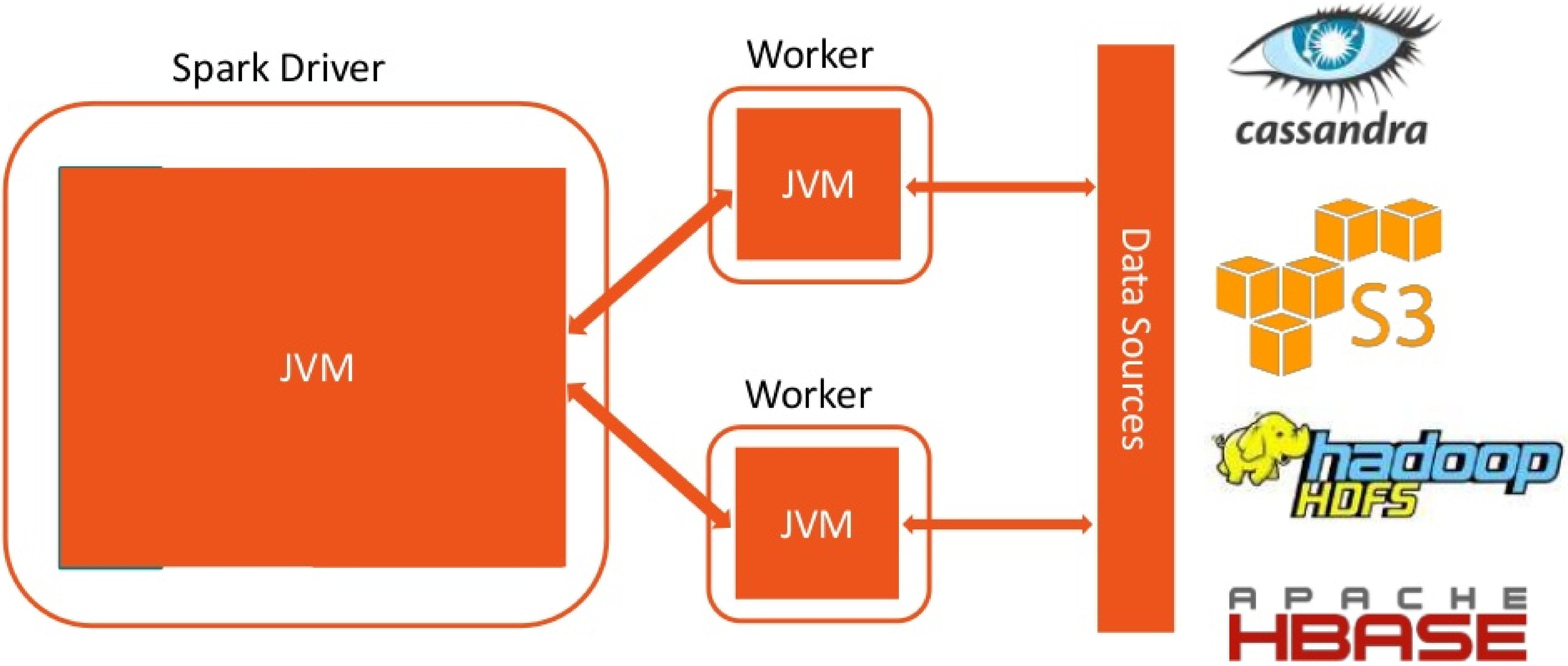
- **Provides R frontend to Spark**
- **Exposes Spark DataFrames (inspired by R and Pandas)**
- **Convenient interoperability between R and Spark DataFrames**

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

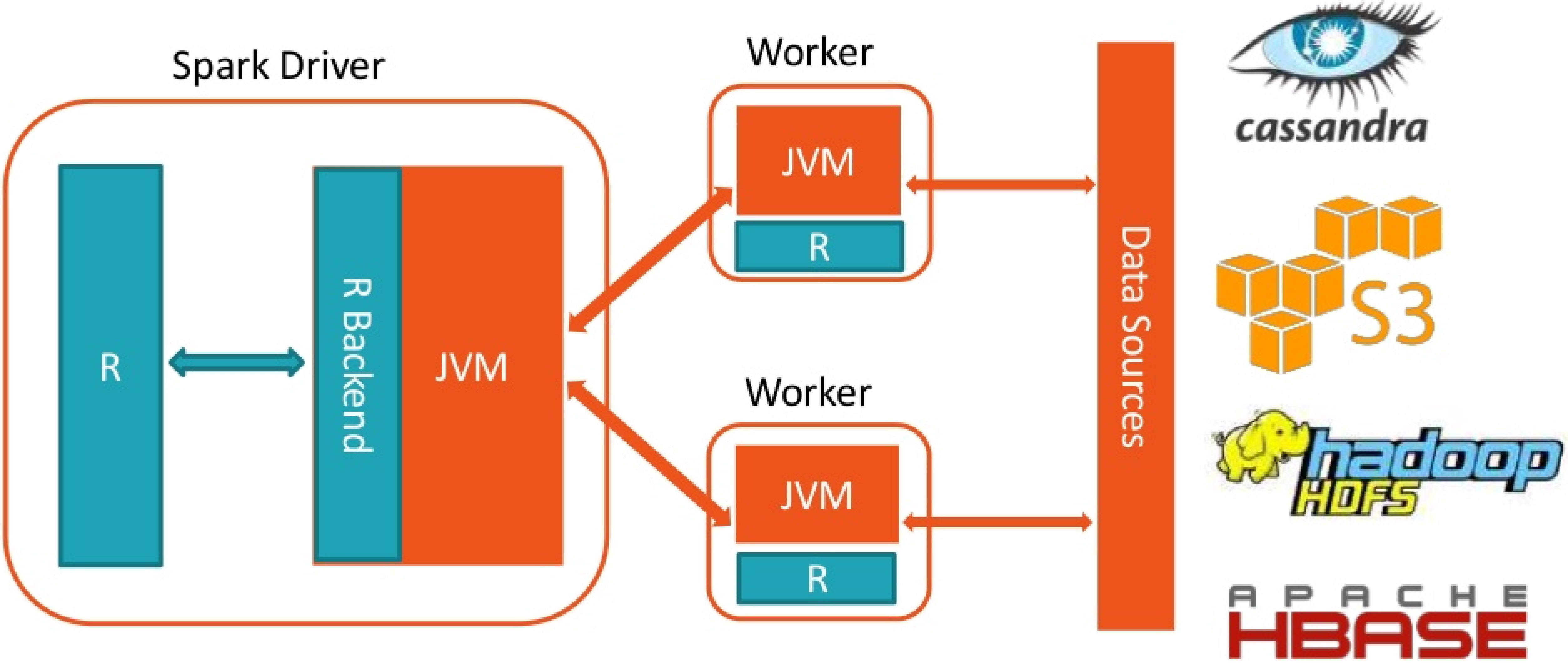
+

Dynamic environment, interactivity, packages, visualization

SparkR architecture



SparkR architecture (since 2.0)



Overview of SparkR API

http://spark.apache.org/docs/latest/api/R/

IO

read.df / write.df /
createDataFrame / collect

Caching

cache / persist / unpersist /
cacheTable / uncacheTable

SQL

sql / table / saveAsTable /
registerTempTable / tables

ML Lib

glm / kmeans / Naïve Bayes
Survival regression

DataFrame API

select / subset / groupBy /
head / avg / column / dim

UDF functionality (since 2.0)

spark.lapply / dapply /
gapply / dapplyCollect

SparkR UDF API

spark.lapply

Runs a function over a list of elements

spark.lapply()

dapply

Applies a function to each partition of a SparkDataFrame

dapply()

dapplyCollect()

gapply

Applies a function to each group within a SparkDataFrame

gapply()

gapplyCollect()

spark.lapply

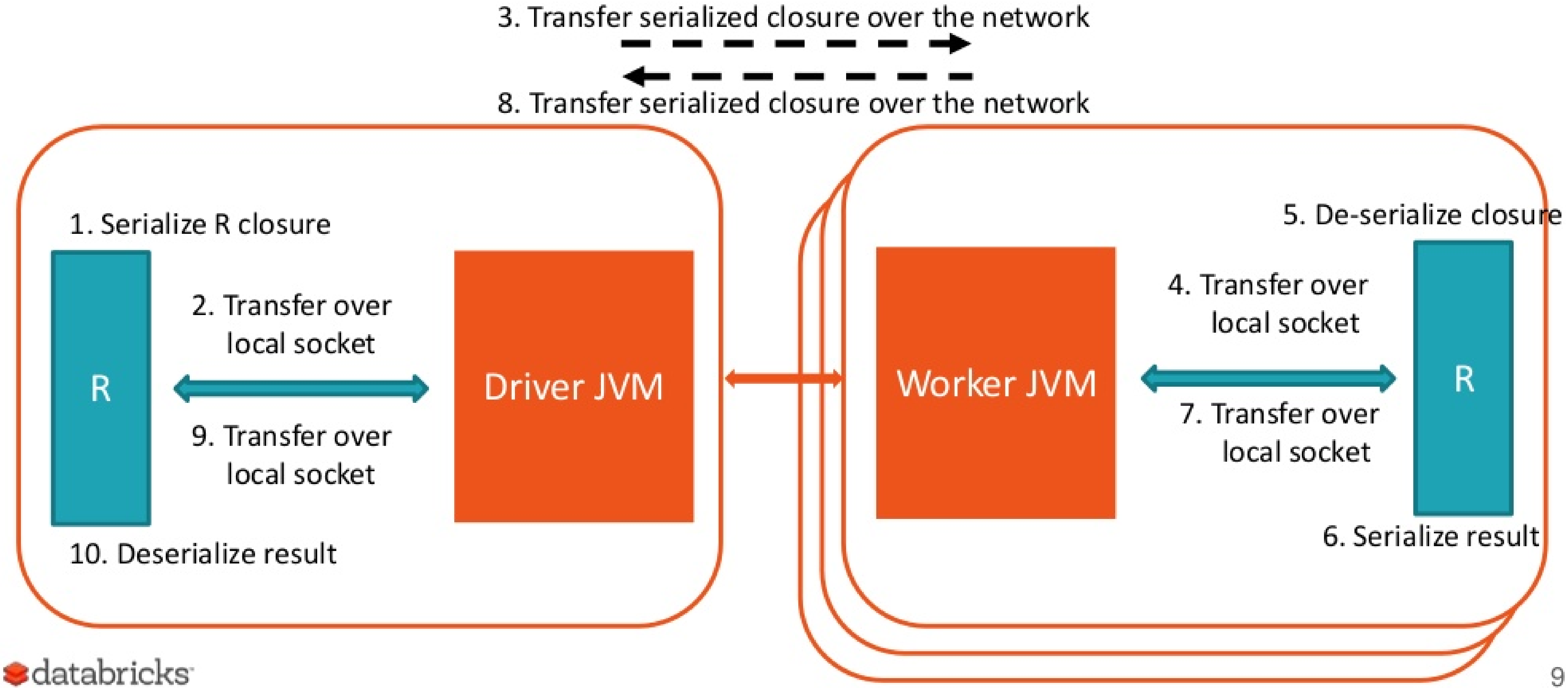
Simplest SparkR UDF pattern

For each element of a list:

1. Sends the function to an R worker
2. Executes the function
3. Returns the result of all workers as a list to R driver

```
spark.lapply(1:100, function(x) {
  runBootstrap(x)
}
```

spark.lapply control flow


3. Transfer serialized closure over the network
8. Transfer serialized closure over the network
1. Serialize R closure
R
2. Transfer over local socket
9. Transfer over local socket
Driver JVM
10. Deserialize result
5. De-serialize closure
Worker JVM
4. Transfer over local socket
7. Transfer over local socket
R
6. Serialize result

databricks

dapply

For each partition of a Spark DataFrame

1. collects each partition as an R data.frame
2. sends the R function to the R worker
3. executes the function

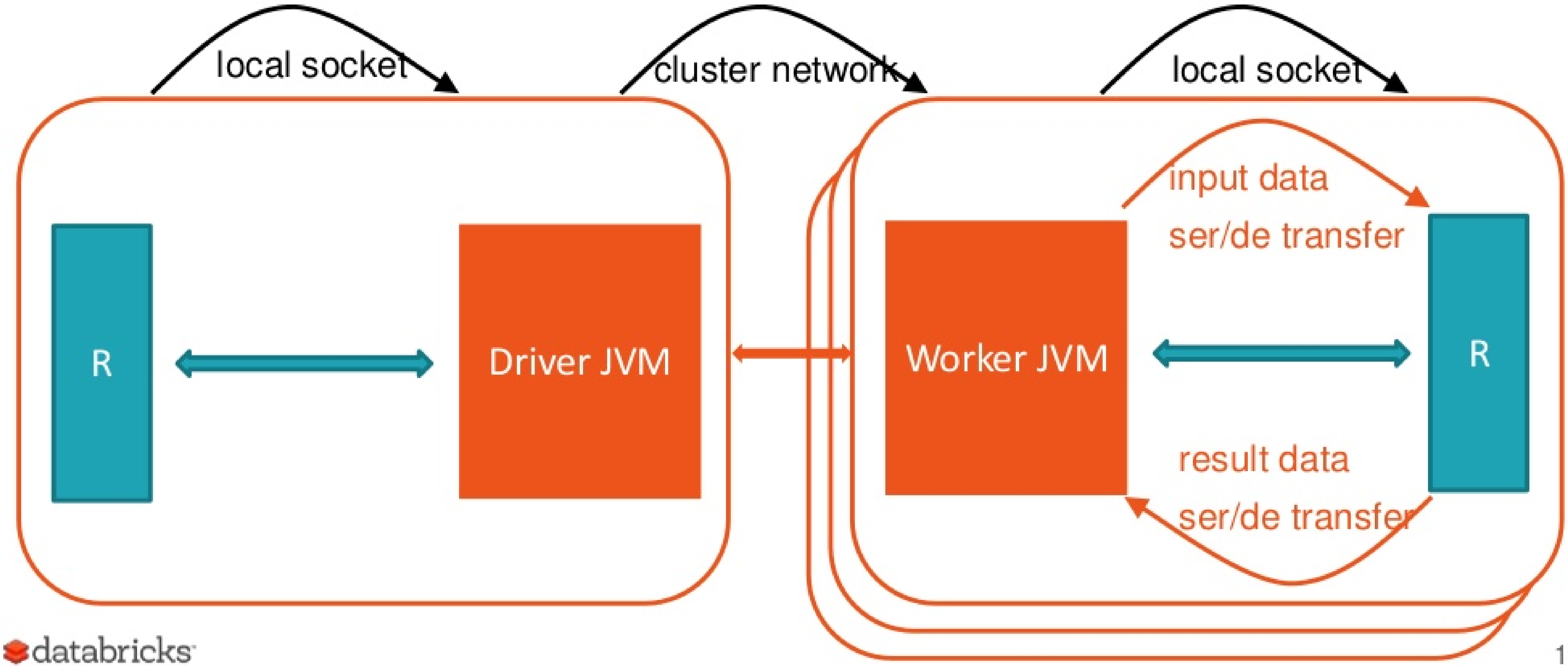
dapply(sparkDF, func, schema)

combines results as DataFrame with provided schema

dapplyCollect(sparkDF, func)

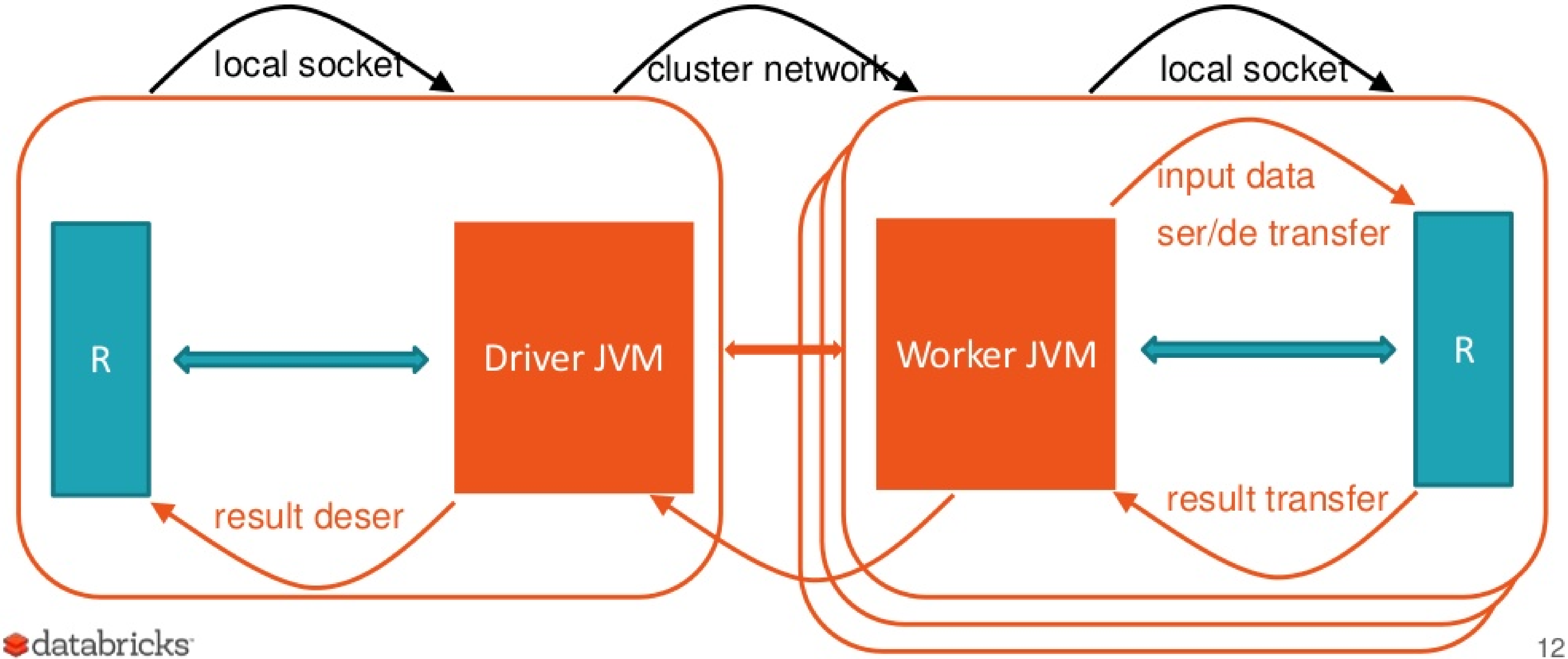
combines results as R data.frame

dapply control & data flow



databricks

dapplyCollect control & data flow



gapply

Groups a Spark DataFrame on one or more columns

1. collects each **group** as an R data.frame
2. sends the R function to the R worker
3. executes the function

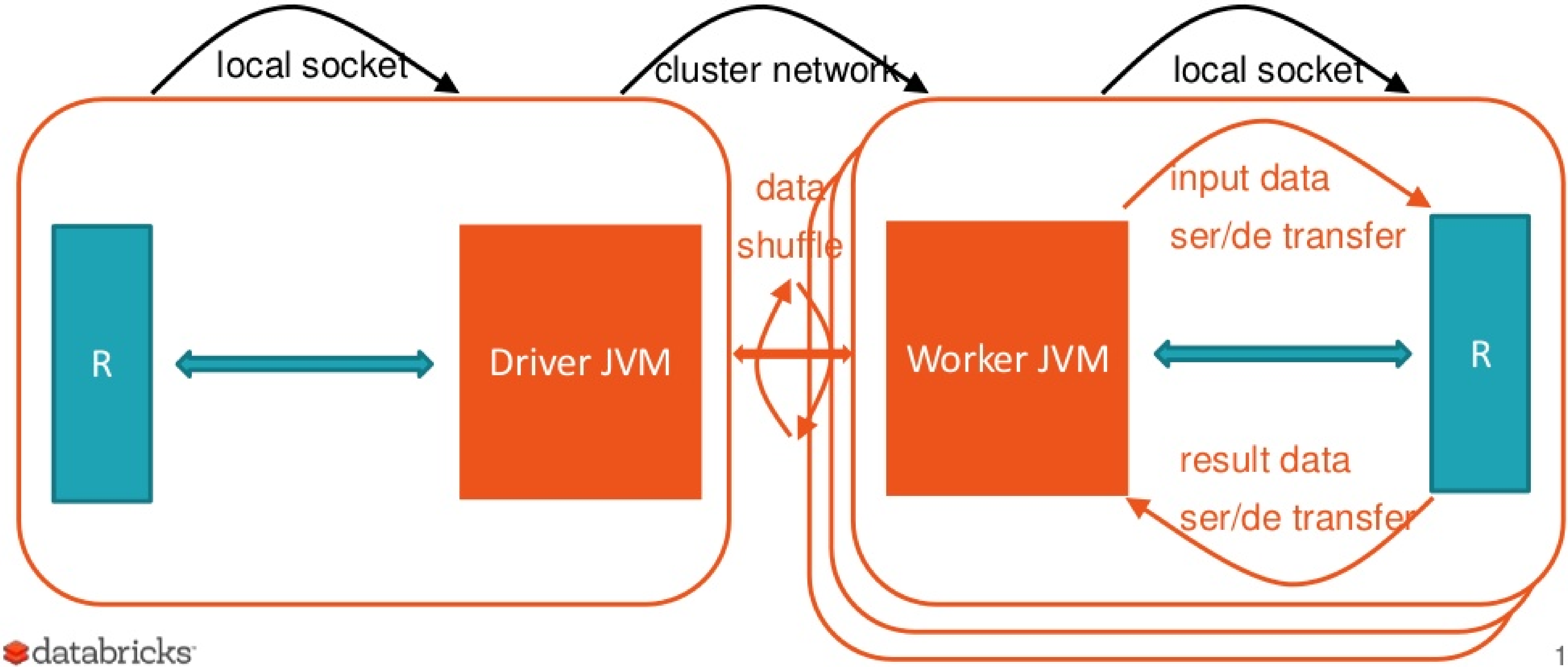
gapply(sparkDF, cols, func, schema)

combines results as DataFrame with provided schema

gapplyCollect(sparkDF, cols, func)

combines results as R data.frame

gapply control & data flow



dapply vs. gapply

	gapply	dapply

Parallelizing data

- Do not use spark.lapply() to distribute large data sets
- Do not pack data in the closure
- Watch for skew in data
  - Are partitions evenly sized?
- Auxiliary data
  - Can be joined with input DataFrame
  - Can be distributed to all the workers

Packages on workers

- SparkR closure capture does not include packages
- You need to import packages on each worker inside your function
- If not installed install packages on workers out-of-band
- spark.lapply() can be used to install packages

Debugging user code

1. Verify your code on the Driver
2. Interactively execute the code on the cluster
   - When R worker fails, Spark Driver throws exception with the R error text
3. Inspect details of failure reason of failed job in spark UI
4. Inspect stdout/stderror of workers

Demo

http://bit.ly/2krYMwC

http://bit.ly/2ltLVKs

Thank you!

databricks™