SparkR Under the Hood

How to debug your SparkR code

Hossein Falaki

June 2017

About me

- Software Engineer at Databricks Inc.
- Data Scientist at Apple Siri
- Started using Spark since 0.6
- Developed first version of Apache Spark CSV data source
- Developed Databricks R Notebooks
- Currently focusing on R experience at Databricks

About Databricks

TEAM

Started Spark project (now Apache Spark) at UC Berkeley in 2009

MISSION

Making Big Data Simple

PRODUCT

Unified Analytics Platform

About this talk

What this talk IS

- SparkR architecture
- SparkR implementation
- Common performance bottlenecks
- Common sources of error
- How to debug your code

What this talk is NOT

- Introduction to SparkR API
- Introducing new features
- How to use SparkR

Outline

- Architecture
- Implementation
- Limitations
- Common errors and problems
- How to debug your code

What is SparkR

R package distributed with Apache Spark

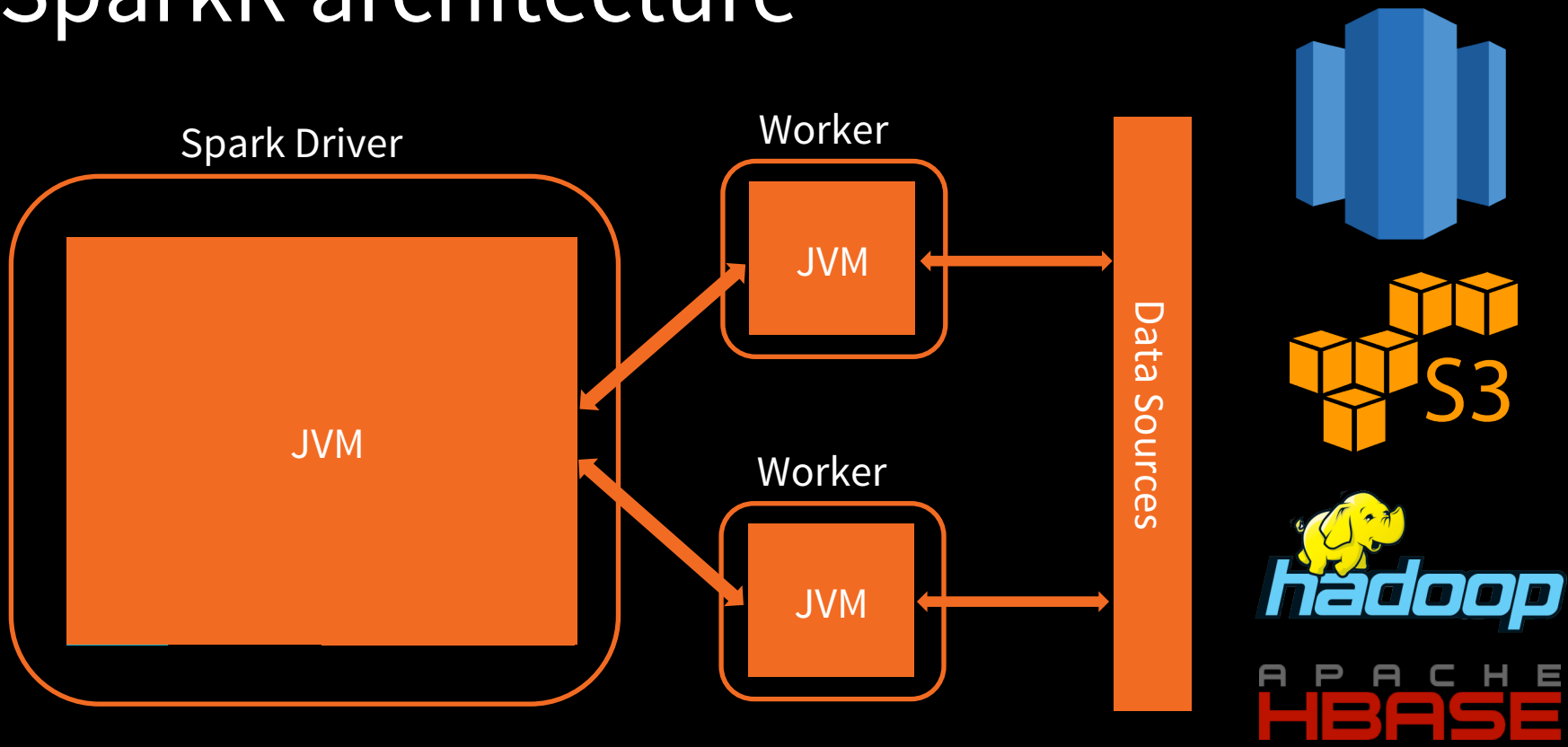
- Provides R front-end to Apache Spark
- Exposes Spark DataFrames (inspired by R & Pandas)
- Convenient interoperability between R and Spark DataFrames

robust distributed processing, data source, off-memory data

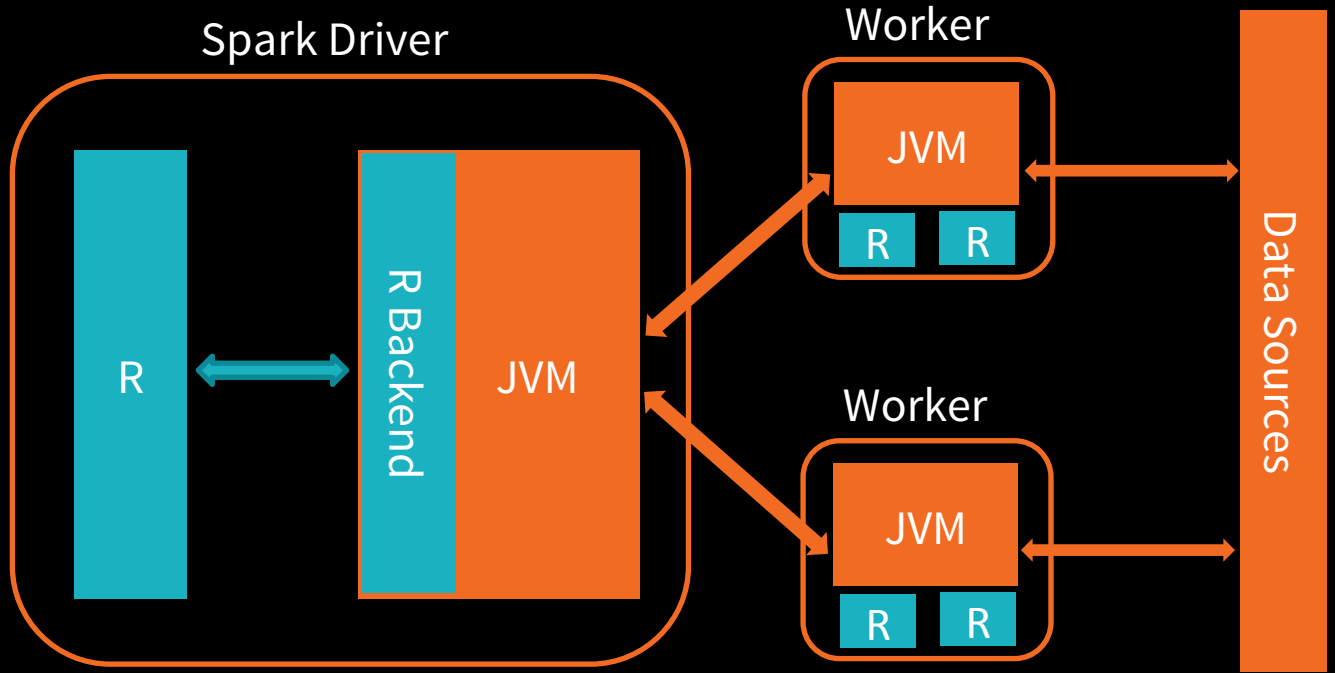
+

dynamic environment, interactivity, +10K packages, visualizations

SparkR architecture



SparkR architecture (2.x)



Driver implementation

2. SparkR establishes socket connections

3. Each SparkR call sends serialized data over the socket and waits for response

1. RBackend opens a server port and waits for connections

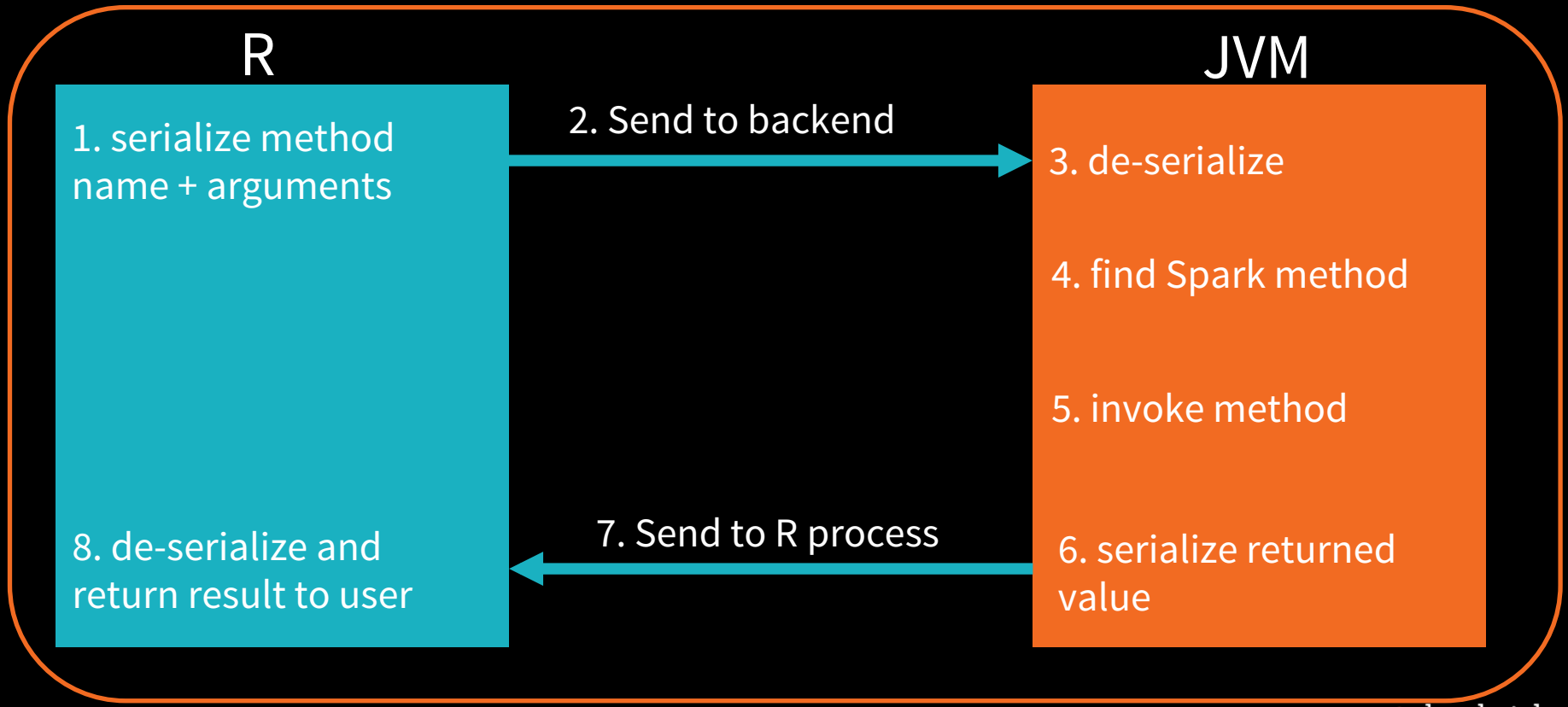
4. RBackendHandler handles and process requests

SparkR Serialization

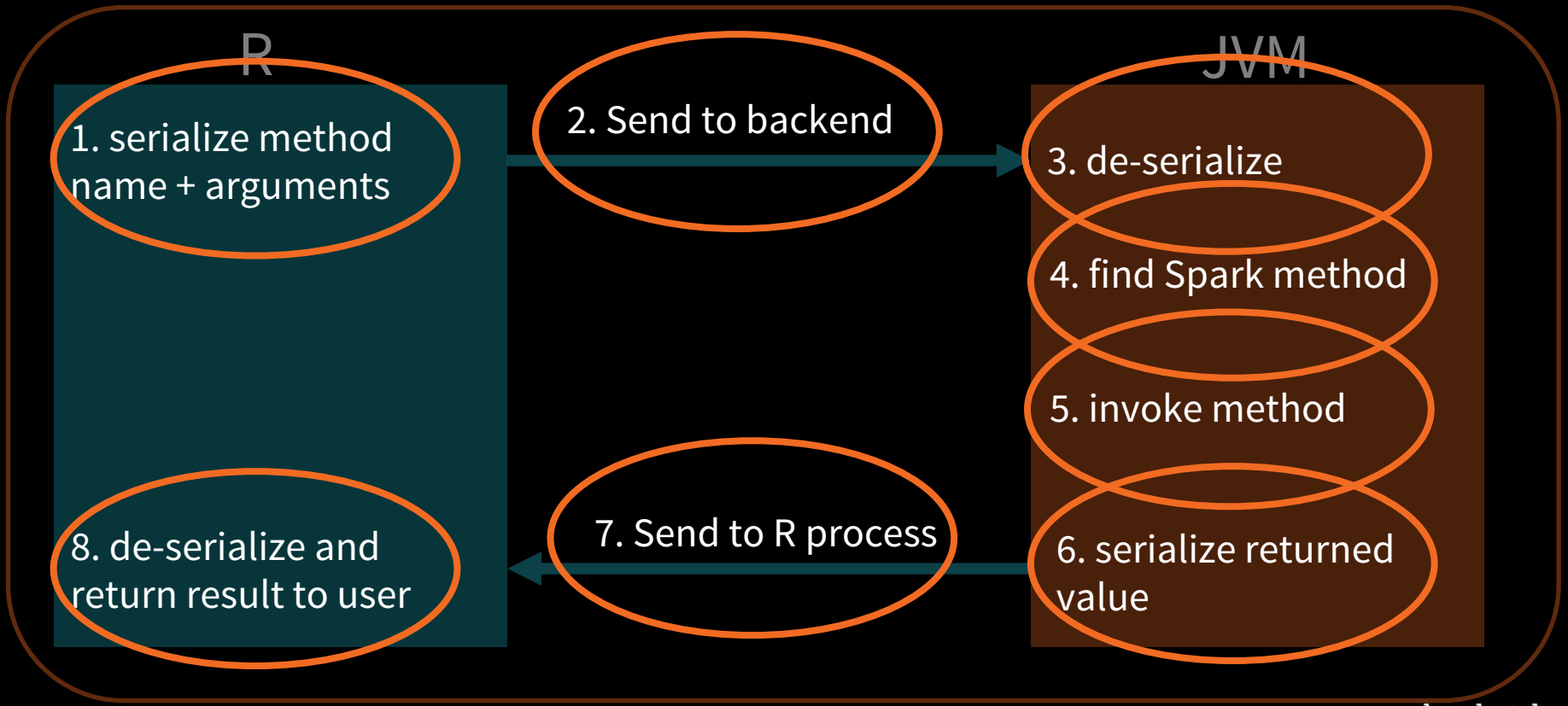
R and JVM use a proprietary serialization format as wire protocol.

Basic type	type	binary data	
Lists	type	size	element 1, element 2, element 3, ...

A simple SparkR query



What can go wrong?



Serialization & deserialization

Memory allocation in R

```
Error in writeBin(batch, con, endian = “big”)

  attempting to add too many elements to raw vector
```

De-serialization in JVM

```
ERROR Executor: Exception in task 0.0 in stage 1.0 (TID 1)
java.lang.NegativeArraySizeException
org.apache.spark.api.r.SerDe$.readStringBytes(SerDe.scala:110)
    at org.apache.spark.api.r.SerDe$.readString(SerDe.scala:
119)
```

Serialization & deserialization

Corner case with types

```
Lost task 0.3 in stage 52.0 (TID 10114, 10.0.229.211):
java.lang.RuntimeException: java.lang.Double is not a valid
external type for schema of date
```

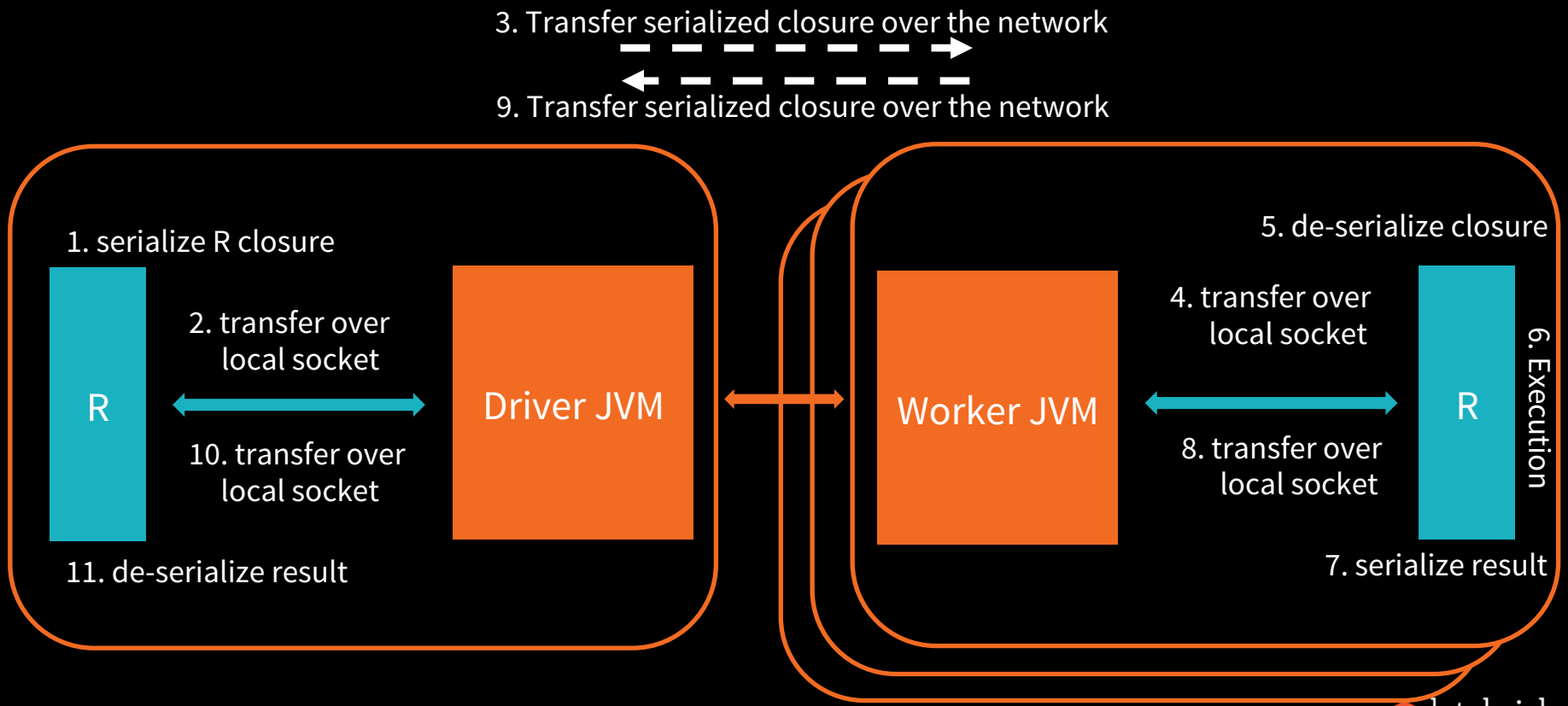
Corner case with types

```
org.apache.spark.SparkException: Job aborted due to stage
failure:

java.lang.IllegalArgumentException   at
java.sql.Date.valueOf(Date.java:143)    at
org.apache.spark.api.r.SerDe$.readDate(SerDe.scala:128)  at
org.apache.spark.api.r.SerDe$.readTypedObject(SerDe.scala:77)
```

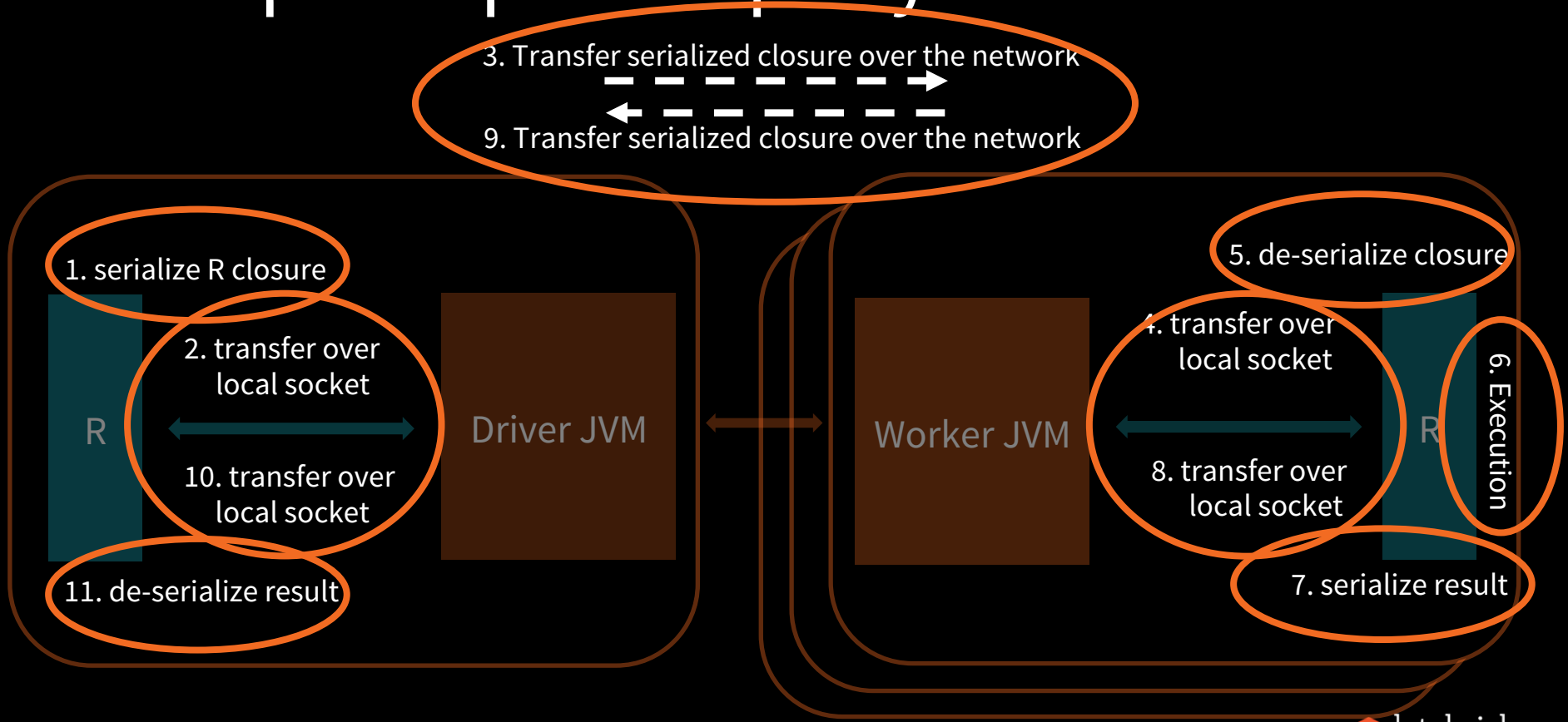
Method signature matching and invocation

```
RBackendHandler: dfToCols on
org.apache.spark.sql.api.r.SQLUtils failed

java.lang.Exception: No matched method found for class
org.apache.spark.sql.api.r.SQLUtils.dfToCols
```

A complex SparkR query



A complex SparkR query



Common problems when using UDFs

- Skew in data
  - Are partitions evenly sized?
- Packing too much data in the closure
- Auxiliary data
  - Can be joined with input DataFrame
  - Can be distributed to all the workers
- Returned data schema

Practical guide to debug SparkR code

Get used to reading Java stack traces

- Often the root cause is at the bottom of the stack trace
- Stack trace includes both driver and executor exceptions
- In many cases the R worker error is included in the exception message

data.frame vs. DataFrame

- ... doesn't know how to deal with data of class SparkDataFrame
- no method for coercing this S4 class to a ...
- Expressions other than filtering predicates are not supported in the first parameter of extract operator.

R function vs. SparkSQL expression

Expressions translate to JVM calls, but functions run in R process of driver or workers

- `filter(logs$type == “ERROR”)`
- `ifelse(df$level > 2, “deep”, “shallow”)`

```
dapply(logs, function(x) {
  subset(x, type == “ERROR”)
}, schema(logs))
```

Special characters in schema names

- ‘.’ is a special character in Spark
- Sometimes SparkR automatically converts ‘.’ to ‘_’ in column names

```
In FUN(X[[i]], ...) :
  Use Sepal_Length instead of Sepal.Length  as column name
```

- Sometimes, names are not transformed and you may end up with ‘.’ in column names

Packing too much into the closure

```
Error in invokeJava(isStatic = FALSE, objId$id,
methodName, ...):

  org.apache.spark.SparkException: Job aborted due to stage
failure: Serialized task 29877:0 was 520644552 bytes, which
exceeds max allowed: spark.rpc.message.maxSize (268435456
bytes).
```

Workers returning empty results

```
Job aborted due to stage failure:
java.lang.ArrayIndexOutOfBoundsException

Driver stacktrace:  at
org.apache.spark.scheduler.DAGScheduler.org$apache$spark
$scheduler$DAGScheduler$
$failJobAndIndependentStages(DAGScheduler.scala:1435)

...

Caused by: java.lang.ArrayIndexOutOfBoundsException
```

Try Apache Spark in Databricks!

UNIFIED ANALYTICS PLATFORM

Collaborative cloud environment
Free version (community edition)

DATABRICKS RUNTIME 3.0

Apache Spark - optimized for the cloud
Caching and optimization layer - DBIO
Enterprise security – DBES
Support for sparklyr

Try for free today.
databricks.com

Thank You

Hossein Falaki @mhfalaki