SparkOscope: Enabling Apache Spark Optimization Through Cross-Stack Monitoring and Visualization

Yiannis Gkoufas
IBM Research Dublin,Ireland
High Performance Systems

whoami

- Research Software Engineer in **IBM Research, Ireland** since 2012
- Work on **Analytics Foundations Middleware**
  - Distributed Frameworks, Anything Java/Scala based, Web-based POCs
- **High Performance Systems Group**: Kostas, Andrea, Khalid, Michael, Mustafa, Pier, Sri

Spark Experience

- We love developing in Spark our analytical workloads and **fully embraced** it since the early 1.0.x versions
- Last few years, used it to run jobs on large volume of **energy-related sensor data**

Jobs on Daily Basis

- Once we managed to develop the needed jobs, they were executed in a **recurring fashion**
- We were receiving a new batch of **data every day**

Fighting Bugs

- When there was a bug on our code, it was very easy to discover it the **Spark Web UI**
- We could easily retrieve information about the **job, stage and line number** in our source code

Fighting bottlenecks

- However we couldn't easily spot which **jobs and stages** were causing a **slow down**
- **What was the part of our code that was the bottleneck?**

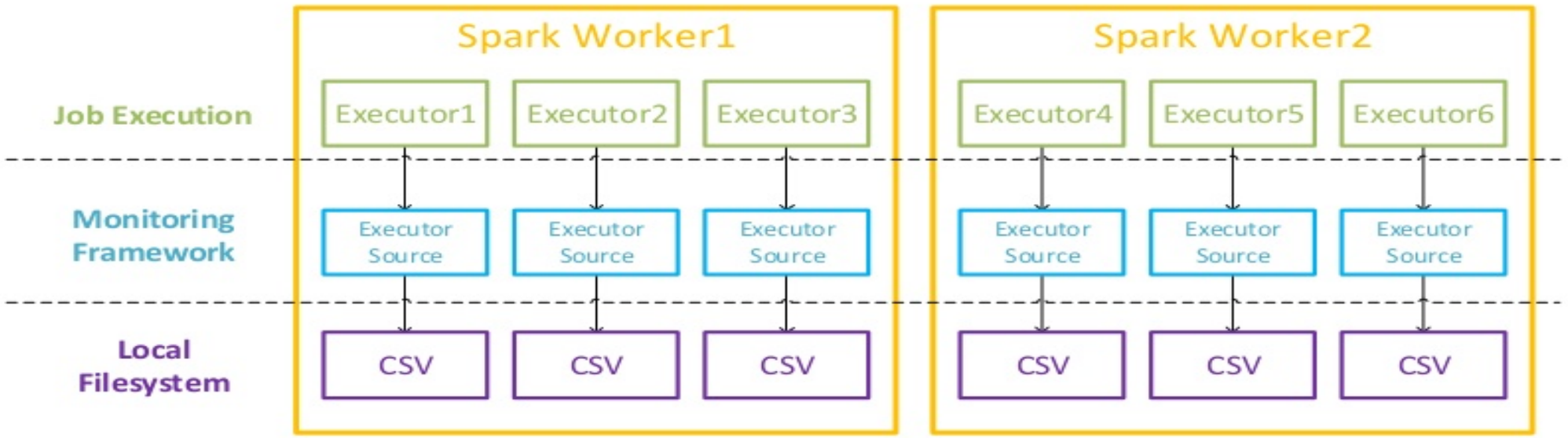
Ganglia Extension

- We had the option to use the **Ganglia Extension** to export the metrics but:
  - We need to maintain/configure **yet another external system**
  - There is **no association** with the Spark jobs/stages/source code

Spark Monitoring Framework

- We could use the built-in **Spark Monitoring Framework** but:
  - **Collecting CSVs** from the worker nodes and **aggregating** them seems **cumbersome**
  - Again we couldn't easily **extract associations with our source code** of the job

Current Monitoring Architecture

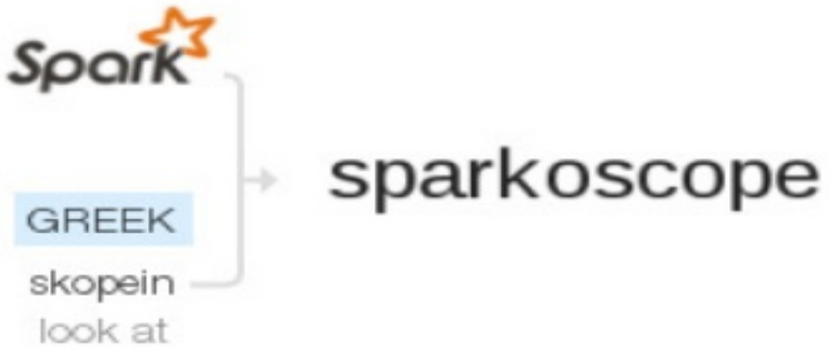


Enter SparkOscope

sparkoscope

/spɑ:kəskəʊp/

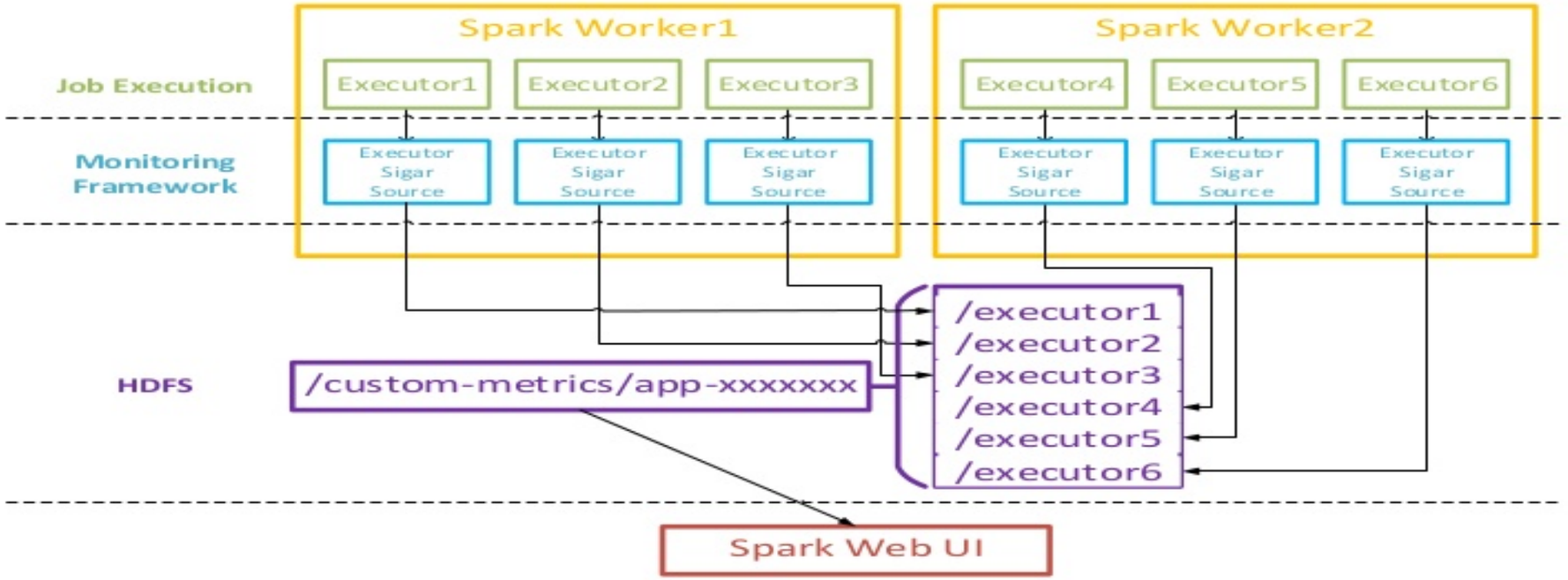
Origin



SparkOscope Overview

- Extension to enrich Spark's Monitoring Framework with **OS-level Metrics**
- Enhancement of the Web UI to **plot all the available metrics** + the newly developed OS-level metrics

SparkOscope High-level Architecture



SparkOscope Basic Installation

- Clone the git repo: https://github.com/ibm-research-ireland/sparkoscope
- Build Spark
- Modify the configuration files:

metrics.properties

```
executor.sink.hdfs.class=org.apache.spark.metrics.sink.HDFSSink

executor.sink.hdfs.pollPeriod = 20

executor.sink.hdfs.dir = hdfs://localhost:9000/custom-metrics

executor.sink.hdfs.unit = seconds
```

spark-defaults.conf

```
spark.hdfs.metrics.dir          hdfs://127.0.0.1:9000/custom-metrics

spark.eventLog.enabled          true

spark.eventLog.dir              hdfs://127.0.0.1:9000/spark-logs
```

SparkOscope OS-level Metrics

- Download the Hyperic Sigar library to all the slave nodes
- Extract it anywhere in the system
- Modify the configuration files

metrics.properties

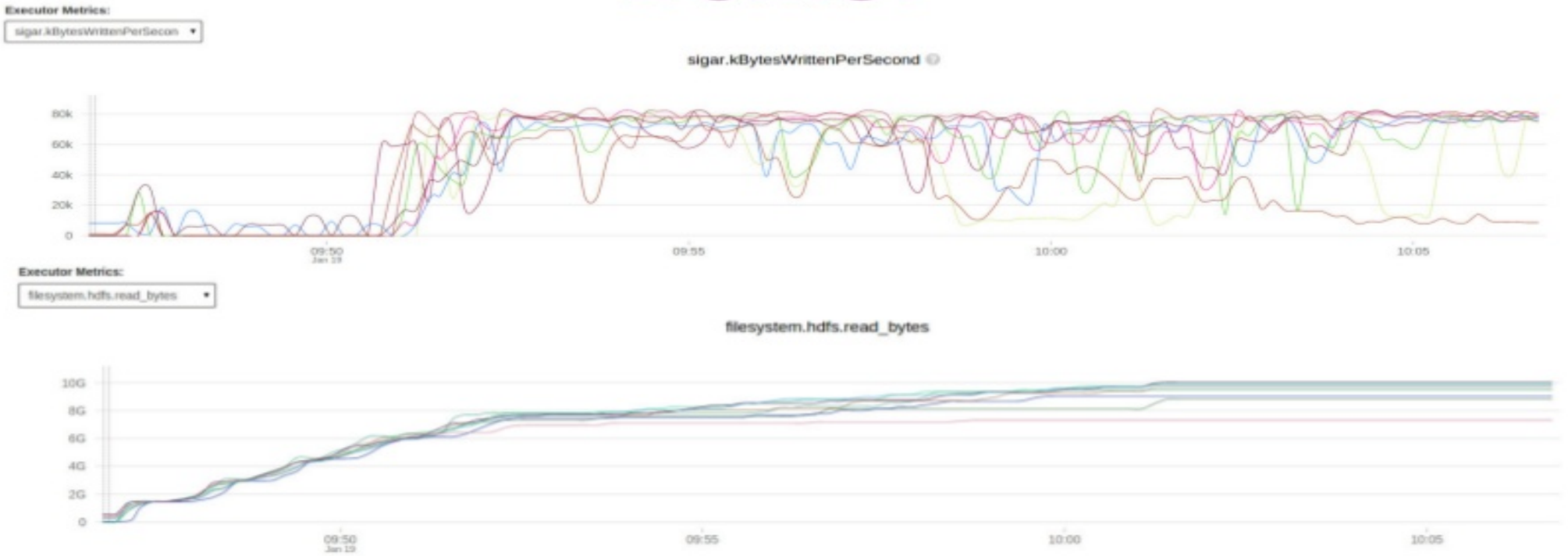
```
executor.source.jvm.class=org.apache.spark.metrics.source.SigarSource
```

spark-env.sh

```
HADOOP_CONF_DIR=/path/to/hadoop/etc/hadoop
```

```
LD_LIBRARY_PATH=/path/to/hyperic-sigar-1.6.4/sigar-bin/lib/:$LD_LIBRARY_PATH
```

Demo!


Executor Metrics:
sigar.kBytesWrittenPerSecon
sigar.kBytesWrittenPerSecond
80k
60k
40k
20k
0
09:50
Jan 19
09:55
10:00
10:05
Executor Metrics:
filesystem.hdfs.read_bytes
filesystem.hdfs.read_bytes
10G
8G
6G
4G
2G
0
09:50
Jan 19
09:55
10:00
10:05

Roadmap

- Pluggable storage mechanism (hbase,mongodb, etc)
- Smart recommendations on infrastructure needs derived from patterns of resource utilization of jobs
- Work with the opensource ecosystem to improve it and target more use cases

THANK YOU.

Questions?

email: yiannisg@ie.ibm.com