Improving PySpark Performance

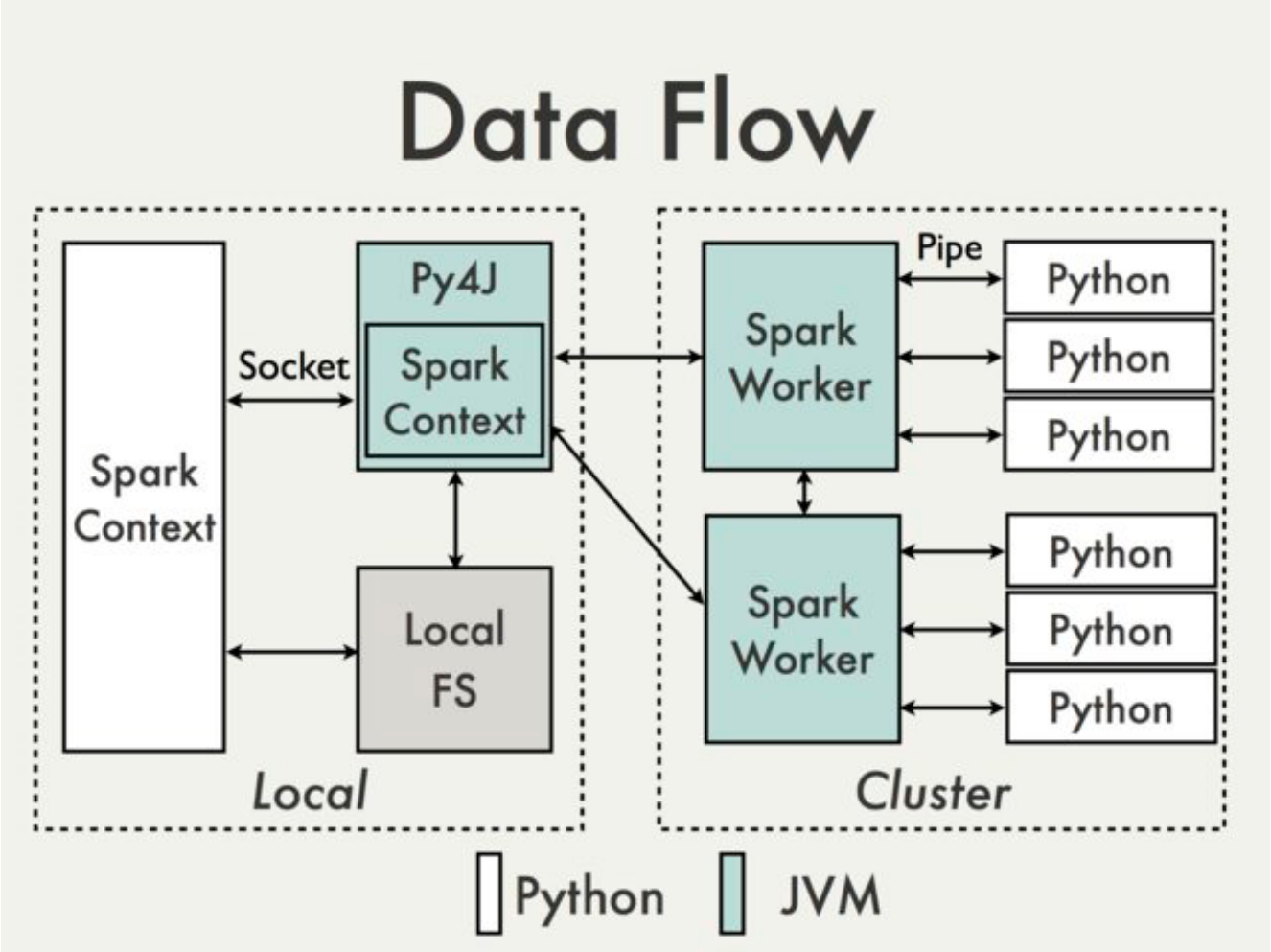
Sreeram Nudurupati

Introduction

Why does code written in PySpark perform slower compared to its Scala and Java counterparts? It is well known that Spark core and most of Spark's libraries are written in Scala, a Java Virtual Machine(JVM) based language. Then how does Python interact with Spark Core and all the various Java processes and objects.

To answer this question we will have to dig a little deeper into how Spark and Python interact and communicate.

Architecture



https://cwiki.apache.org/confluence/display/SPARK/PySpark+Internals

In the Spark driver PySpark interacts with Java Spark context using Py4J, Py4J enables Python programs to dynamically access objects in JVM.
On the worker side in the cluster, each Spark Java worker launches a corresponding Python worker and the two exchange data using unix pipelines.

Why is PySpark Slow?

- Unlike Scala/Java RDDs, Python RDDs consist of pickled objects, which need to be serialized and de-serialized while processing.
- There is already serialization involved in Spark to transfer Java objects over the network (shuffles), which yields to double-serialization when using PySpark. (Serialization/de-serialization tend to be very expensive)
- Python workers take quite a bit of time to start up.
- Python memory isn't managed by Java thus chances of exceeding container memory limits exist.

How to make PySpark faster?

The answer is using DataFrames as they provide the following benefits:

- DataFrames convert PySpark operations into a query plan and have ample opportunity to produce well optimized physical execution plan no matter the developer's coding abilities.
- They also compile down PySpark operations to Java Byte code thus eliminating the need for launching Python workers and double-serialization.
- They facilitate the use of Hive UDFs and windowed functions(rank, n-tile) that can help formulate complex logic that DataFrame DSL can't yet support.
- Using DataFrame DSL to solve complex data problems, joins, cartesian products , filters, column scans etc. before switching to traditional RDD operations.

Reference

The following video by Holden Karau, co-author of Learning Spark book and Spark contributor, explains in details the problems with using PySpark and how to mitigate some of them.

https://www.youtube.com/watch?v=WThEk88cWJQ