Spark

Cluster computing (take 2)

J. Fernando Sánchez, Joaquín Salvachúa, Gabriel Huecas

2016

Universidad Politécnica de Madrid

Schedule for today

Recap

Advanced Spark configuration

Deploying a cluster with 20+ nodes

Different orchestrators

Spark Ecosystem

Acknowledgements and useful links

Recap

Other Collection Methods

Scala collections provide many other functional methods; for example, Google for “Scala Seq”

Method on Seq[T]	Explanation
`map(f: T => U): Seq[U]`	Pass each element through f
`flatMap(f: T => Seq[U]): Seq[U]`	One-to-many map
`filter(f: T => Boolean): Seq[T]`	Keep elements passing f
`exists(f: T => Boolean): Boolean`	True if one element passes
`forall(f: T => Boolean): Boolean`	True if all elements pass
`reduce(f: (T, T) => T): T`	Merge elements using f
`groupBy(f: T => K): Map[K,List[T]]`	Group elements by f(element)
`sortBy(f: T => K): Seq[T]`	Sort elements by f(element)
`. . .`	

Spark specific methods

Some examples:

- collect: to apply all transformations and get their results
- cache: to save results for later use
- groupByKey: to group items by key
- reduceByKey[1]: group and apply reduce in the same step

[1] Prefer this to groupbykey+map+reduce. See the slides from our previous class for more information

Spark specific methods

Some examples:

- collect: to apply all transformations and get their results
- cache: to save results for later use
- groupByKey: to group items by key
- reduceByKey[1]: group and apply reduce in the same step
- keyBy: to creates [K,V] tuples by applying a function to each value

[1] Prefer this to groupbykey+map+reduce. See the slides from our previous class for more information

Spark specific methods

Some examples:

- collect: to apply all transformations and get their results
- cache: to save results for later use
- groupByKey: to group items by key
- reduceByKey[1]: group and apply reduce in the same step
- keyBy: to creates [K,V] tuples by applying a function to each value
- ...

[1] Prefer this to groupbykey+map+reduce. See the slides from our previous class for more information

Closer look at our demo

Now, I’ll start using the terminal.

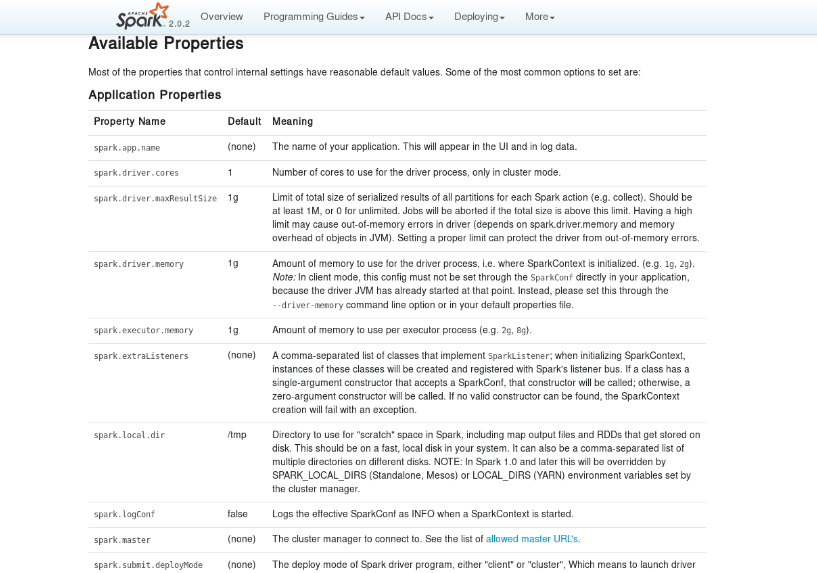
Closer look at our demo

Now, I'll start using the terminal.
Don't freak out

Apologies in advance.

Advanced Spark configuration

Spark conf



Available Properties

Most of the properties that control internal settings have reasonable default values. Some of the most common options to set are:

Application Properties

Property Name	Default	Meaning
`spark.app.name`	(none)	The name of your application. This will appear in the UI and in log data.
`spark.driver.cores`	1	Number of cores to use for the driver process, only in cluster mode.
`spark.driver.maxResultSize`	1g	Limit of total size of serialized results of all partitions for each Spark action (e.g. collect). Should be at least 1M, or 0 for unlimited. Jobs will be aborted if the total size is above this limit. Having a high limit may cause out-of-memory errors in driver (depends on spark.driver.memory and memory overhead of objects in JVM). Setting a proper limit can protect the driver from out-of-memory errors.
`spark.driver.memory`	1g	Amount of memory to use for the driver process, i.e. where SparkContext is initialized. (e.g. `1g`, `2g`). *Note:* In client mode, this config must not be set through the `SparkConf` directly in your application, because the driver JVM has already started at that point. Instead, please set this through the `--driver-memory` command line option or in your default properties file.
`spark.executor.memory`	1g	Amount of memory to use per executor process (e.g. `2g`, `8g`).
`spark.extraListeners`	(none)	A comma-separated list of classes that implement `SparkListener`; when initializing SparkContext, instances of these classes will be created and registered with Spark's listener bus. If a class has a single-argument constructor that accepts a SparkConf, that constructor will be called; otherwise, a zero-argument constructor will be called. If no valid constructor can be found, the SparkContext creation will fail with an exception.
`spark.local.dir`	/tmp	Directory to use for "scratch" space in Spark, including map output files and RDDs that get stored on disk. This should be on a fast, local disk in your system. It can also be a comma-separated list of multiple directories on different disks. NOTE: In Spark 1.0 and later this will be overridden by SPARK_LOCAL_DIRS (Standalone, Mesos) or LOCAL_DIRS (YARN) environment variables set by the cluster manager.
`spark.logConf`	false	Logs the effective SparkConf as INFO when a SparkContext is started.
`spark.master`	(none)	The cluster manager to connect to. See the list of allowed master URL's.
`spark.submit.deployMode`	(none)	The deploy mode of Spark driver program, either "client" or "cluster", Which means to launch driver

- Application, Runtime, UI and RDD settings
- Highlights
    - Master URL and Port
    - CPU and memory per worker
    - Parallelism: default partition size

Deploying a cluster with 20+ nodes

Our big data cluster

We will illustrate what we've already covered by deploying a cluster of more than 20 nodes.

Our big data cluster

We will illustrate what we've already covered by deploying a cluster of more than 20 nodes.

Top of the line servers in Azure and Amazon.

Our big data cluster

We will illustrate what we've already covered by deploying a cluster of more than 20 nodes.

Top of the line servers in Azure and Amazon.

Backed up by BlueMix's Spark infrastructure.

Cinco años de recortes en la Universidad pública

La universidad no se ha librado. La crisis ha hecho mella en los campus públicos de toda España en forma de recortes presupuestarios y de personal

Our brain-powered cluster

This is embarrassing...

Our brain-powered cluster

This is embarrassing...

We'll have to make do with our brains and paper.

Our brain-powered cluster

This is embarrassing...

We'll have to make do with our brains and paper.

That's probably how Amazon Mechanical Turk was born.

Input

We have a log of users that know or ask about a topic. It looks like this:

```
Alice knows Scala
Bob asks about Scala
Caroline asks about Java
Don knows about Scala
 ...
```

In other words, lines have this format:

```
{student} {action} {topic}
```

Your task

Answer the questions:

- How many questions were asked about each topic?

Your task

Answer the questions:

- How many questions were asked about each topic?
- How many times did each student ask about each topic?

Your task

Answer the questions:

- How many questions were asked about each topic?
- How many times did each student ask about each topic?
- (Harder) Is there any topic with questions that no other student knows about?

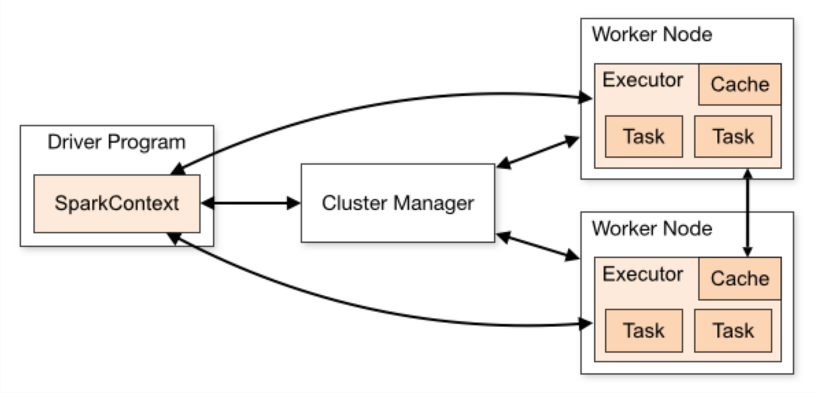
Your task

Answer the questions:

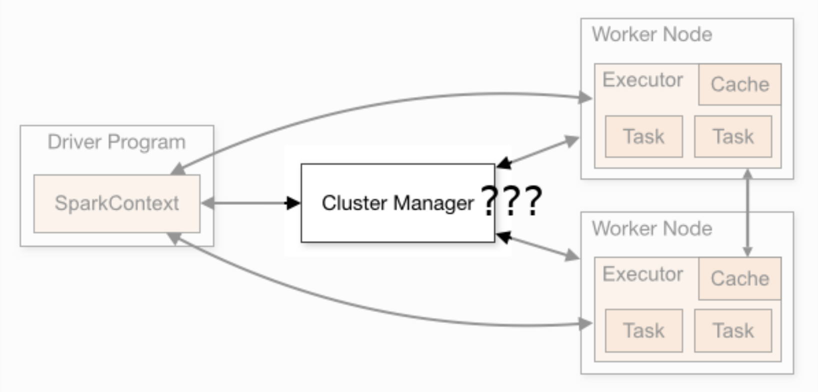
- How many questions were asked about each topic?
- How many times did each student ask about each topic?
- (Harder) Is there any topic with questions that no other student knows about?
- (Even harder) Pair students that know about a topic with students that don't

Different orchestrators

Spark architecture



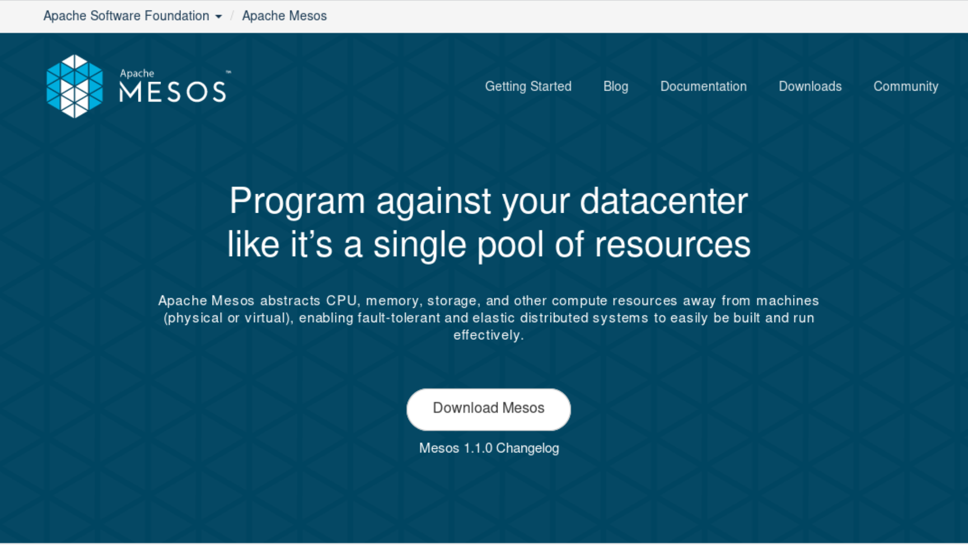
Spark architecture



Cluster managers

Cluster managers (either Spark's own standalone cluster manager, Mesos or YARN), which allocate resources across applications. Once connected, Spark acquires executors on nodes in the cluster, which are processes that run computations and store data for your application.

Mesos

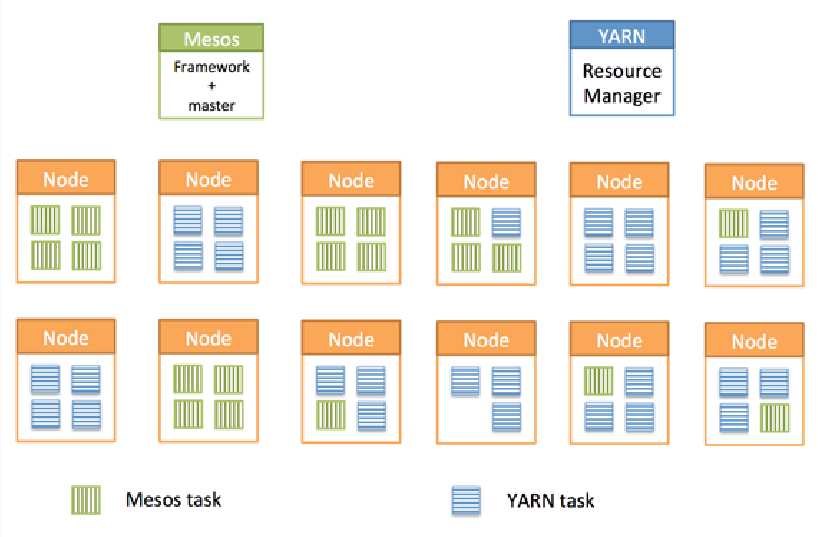


Apache Software Foundation / Apache Mesos

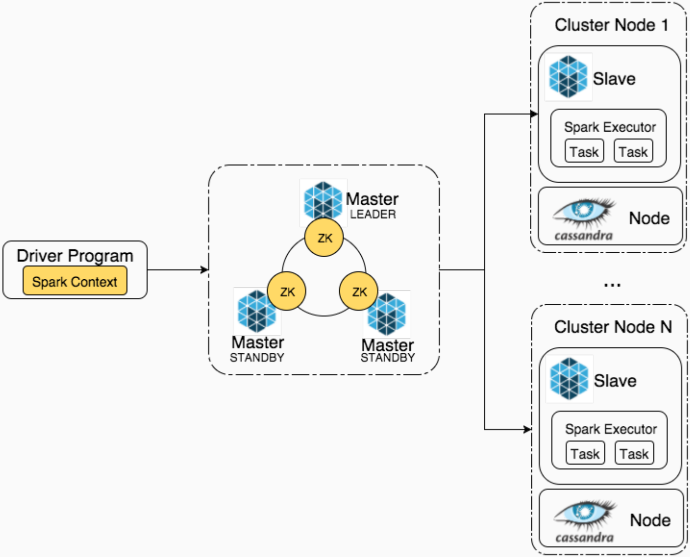
Program against your datacenter like it’s a single pool of resources

Apache Mesos abstracts CPU, memory, storage, and other compute resources away from machines (physical or virtual), enabling fault-tolerant and elastic distributed systems to easily be built and run effectively.

Cluster managers



Spark architecture



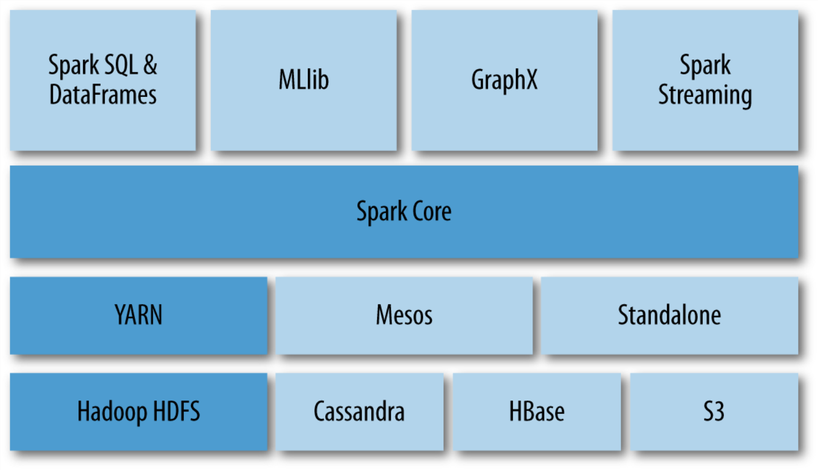
Spark on mesos

Once again, Spark has astoundingly good documentation

http://spark.apache.org/docs/latest/running-on-mesos.html

Spark Ecosystem

Ecosystem



Spark SQL & DataFrames

MLlib

GraphX

Spark Streaming

Spark Core

YARN

Mesos

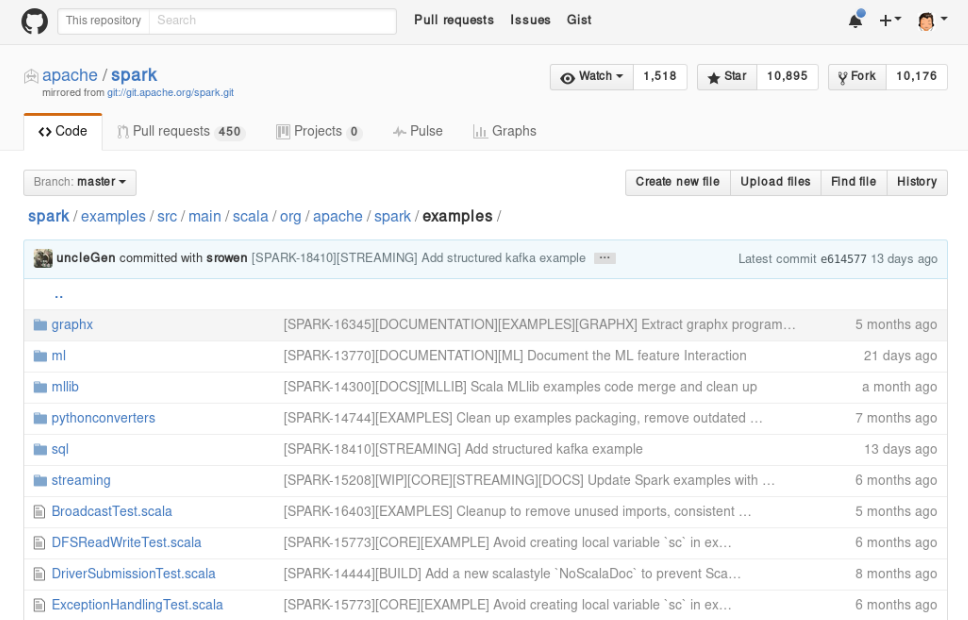
Standalone

Hadoop HDFS

Cassandra

HBase

S3

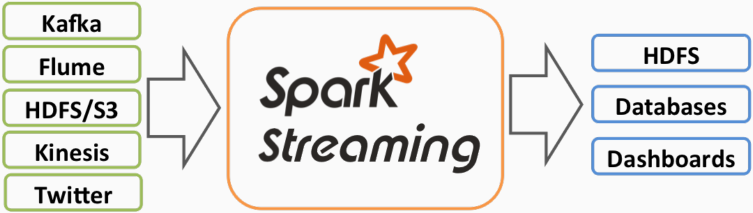


spark / examples / src / main / scala / org / apache / spark / **examples** /

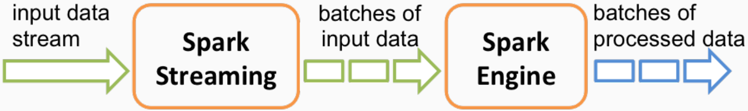
uncleGen committed with **srowen** [SPARK-18410][STREAMING] Add structured kafka example ··· Latest commit e614577 13 days ago

..		
graphx	[SPARK-16345][DOCUMENTATION][EXAMPLES][GRAPHX] Extract graphx program…	5 months ago
ml	[SPARK-13770][DOCUMENTATION][ML] Document the ML feature Interaction	21 days ago
mllib	[SPARK-14300][DOCS][MLLIB] Scala MLlib examples code merge and clean up	a month ago
pythonconverters	[SPARK-14744][EXAMPLES] Clean up examples packaging, remove outdated …	7 months ago
sql	[SPARK-18410][STREAMING] Add structured kafka example	13 days ago
streaming	[SPARK-15208][WIP][CORE][STREAMING][DOCS] Update Spark examples with …	6 months ago
BroadcastTest.scala	[SPARK-16403][EXAMPLES] Cleanup to remove unused imports, consistent …	5 months ago
DFSReadWriteTest.scala	[SPARK-15773][CORE][EXAMPLE] Avoid creating local variable `sc` in ex…	6 months ago
DriverSubmissionTest.scala	[SPARK-14444][BUILD] Add a new scalastyle `NoScalaDoc` to prevent Sca…	8 months ago
ExceptionHandlingTest.scala	[SPARK-15773][CORE][EXAMPLE] Avoid creating local variable `sc` in ex…	6 months ago

Spark Streaming i



Spark Streaming ii



Demo!

We'll show a demo of a modified Spark Streaming task.

Here is a quick "video" of the demo:
https://asciinema.org/a/6kp49z5m3hq9vja9r7x4rjq26

All the code and instructions are available in our repository (in several branches):

https://github.com/balkian/docker-spark

Acknowledgements and useful links

- Spark programming guide
- Databricks introducing apache spark datasets
- Data Analytics with Hadoop: In-Memory Computing with Spark
- Understanding RDD operations, transformations and actions
- Spark Streaming programming guide