Apache SPARK

Docente: Patrizio Dazzi
Email: patrizio.dazzi@isti.cnr.it
patrizio.dazzi@gmail.com

HISTORY OF SPARK (SHORT)

A few milestones about MapReduce and Spark:
2002: MapReduce @ Google
2004: MapReduce paper
2006: Hadoop @ Yahoo! The most popular open source implementation of MapReduce.
2010: Spark paper
2014: Spark top-level becomes an Apache Software Foundation

PRELIMINARIES

"At a high level, every Spark application consists of a driver program that launches various parallel operations on a cluster. The driver program contains your application's *main function* and *defines distributed datasets* on the cluster, then *applies operations* to them.“

In the following examples, the driver program was the Spark shell. "Driver programs access Spark through a SparkContext object, which represents a connection to a computing cluster. In the shell, a SparkContext is automatically created for you as the variable called sc ."

RESILIENT DISTRIBUTED DATASET (RDD)

This is the main Spark ingredient. Spark supports massive parallel computations based on RDD. RDD is where you put your data. An RDD can be viewed as a vector/list of data stored on several machine, where each machine can *access* and *process* part of it. An RDD is fault tolerant: if a machine fails, the RDD is not "broken".

Your job is to build an RDD with your data, and then ask Spark to run operations on that data.

TRANSFORMATIONS AND ACTIONS

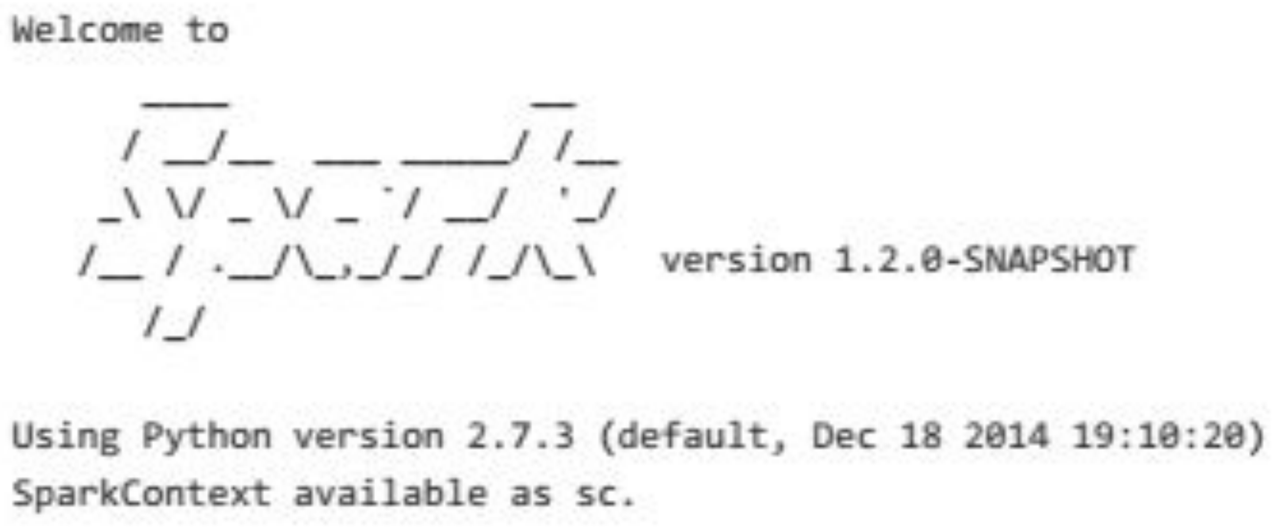
Spark allows to work on RDD with transformations and actions:

- transformations create a new RDD from an input RDD (e.g., filtering out some unwanted elements).
- actions return a (list of) value(s) to the driver program or write to the (distributed) file system.

Spark exploits a *lazy evaluation* of transformations. Transformation are not applied immediately, but they are *accumulated*. When an action occurs, i.e., some output is required, the Spark framework re-organizes transformations and actions into a parallel execution plan.

SPARK SHELL (INTERACTIVE SHELL)

- From terminal launch the command pyspark . This will open a python shell with a spark environment.



```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 1.2.0-SNAPSHOT
      /_/

Using Python version 2.7.3 (default, Dec 18 2014 19:10:20)
SparkContext available as sc.
```

FIRST RDD

Let's create our first RDD

```
>>> data = range(20)
>>> myrdd = sc.parallelize(data)
>>> data
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

```
>>> myrdd
ParallelCollectionRDD[0] at parallelize at PythonRDD.scala:364
```

As expected myrdd is *ParallelCollectionRDD*, which allows us to run Spark parallel operations on it.

TRANSFORMATIONS

- transformations return a new RDD.
- transformations are not executed immediately, but only when Sparks decides so. (This is called *lazy evaluation*).