BIG DATA PLAFORM

GDAF

Gaia Data Analysis Framework

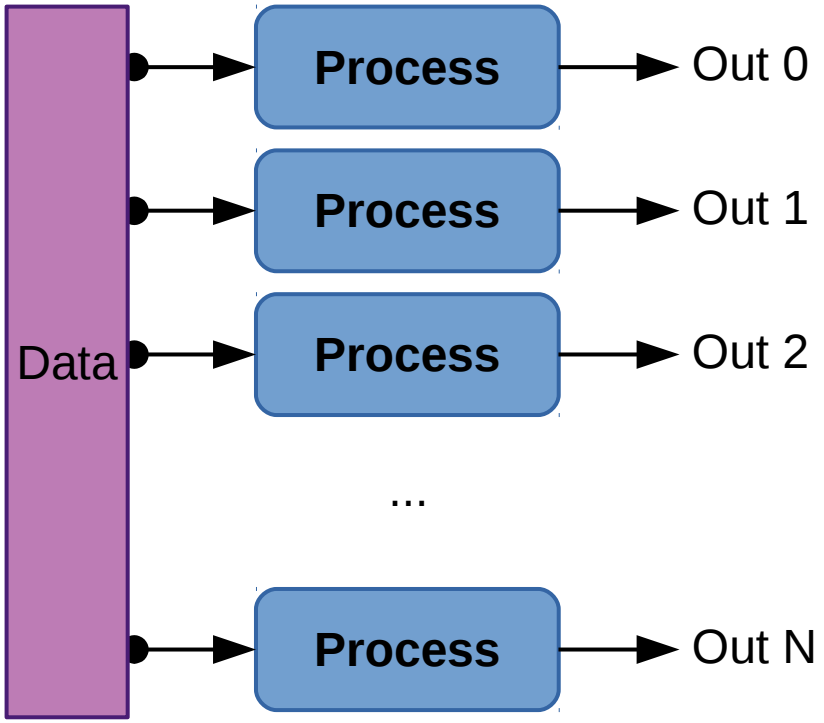
Sergio Soria, Pau Castro, Pau Madrero

Big Data – Why?

- Astronomical catalogues:
  - **Tycho2** (~2.5 million sources) ~ 600 MB
  - **Hipparcos** (~120 thousand sources) ~ 100 MB
  - **GAIA**
    - DR1 (~1.2 billion sources) ~1 TB
    - DR2 (>1.5 billion sources) ~2 TB
    - DR3 ~10 TB
    - ....
    - Final release, ~100 TB

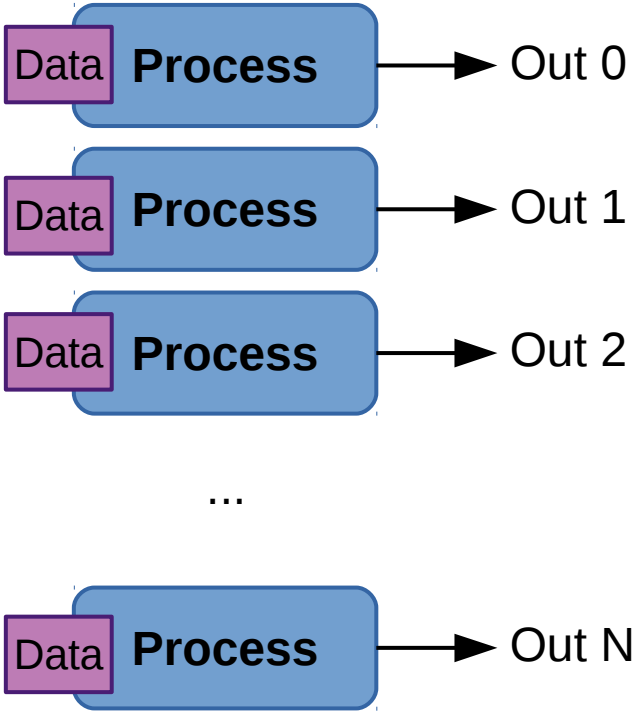
Big Data – How?

- HPC



- Hardware dependent
- Data not distributed

- MapReduce



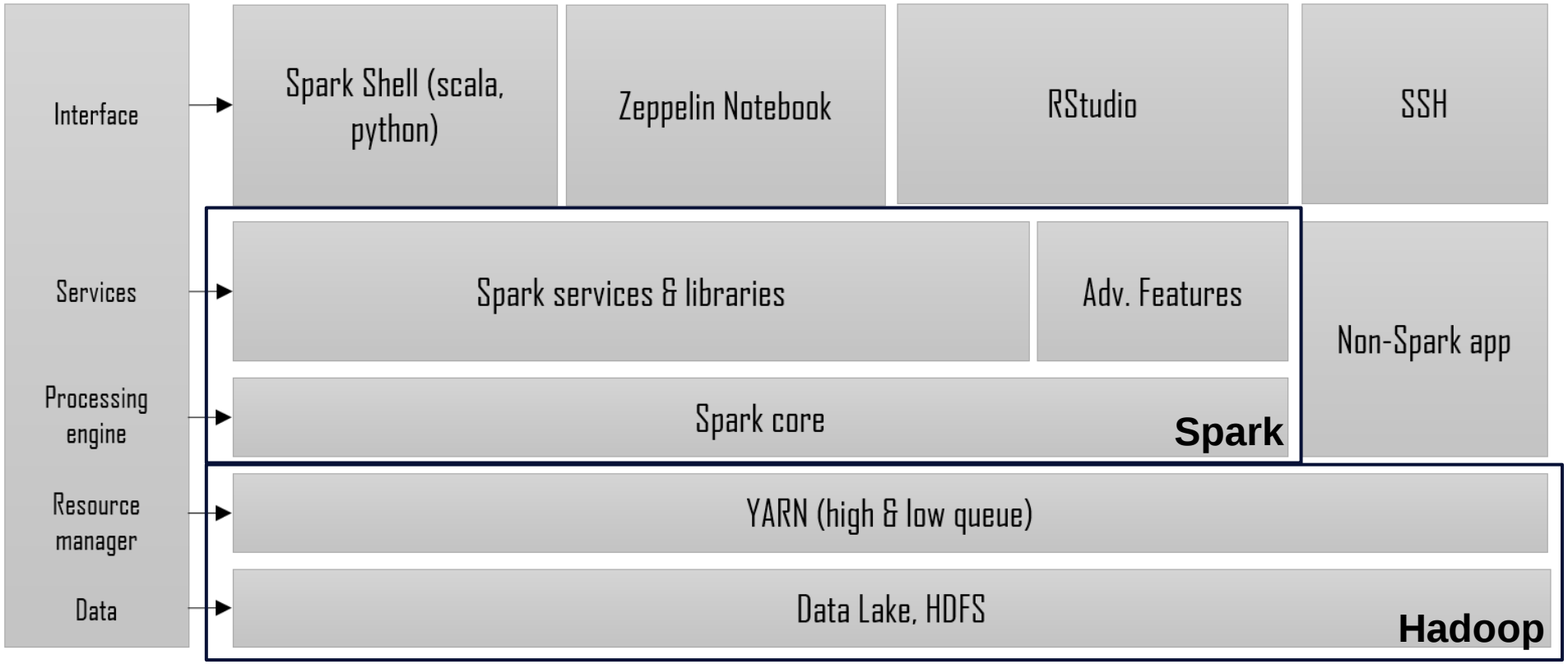
- Commodity hardware
- Brings code to data

GDAF – Environment

- 6 Nodes
  - 96 Cores (2 x 8 Intel Xeon 2,6 GHz each)
  - 4 TFLOPs
  - 384 GB RAM (8 x 8 GB DDR4 each)
  - 72 TB disk (12 x 1 TB HD, SATA 6 Gb/s each)

GDAF - Architecture

Hadoop distribution from cloudera



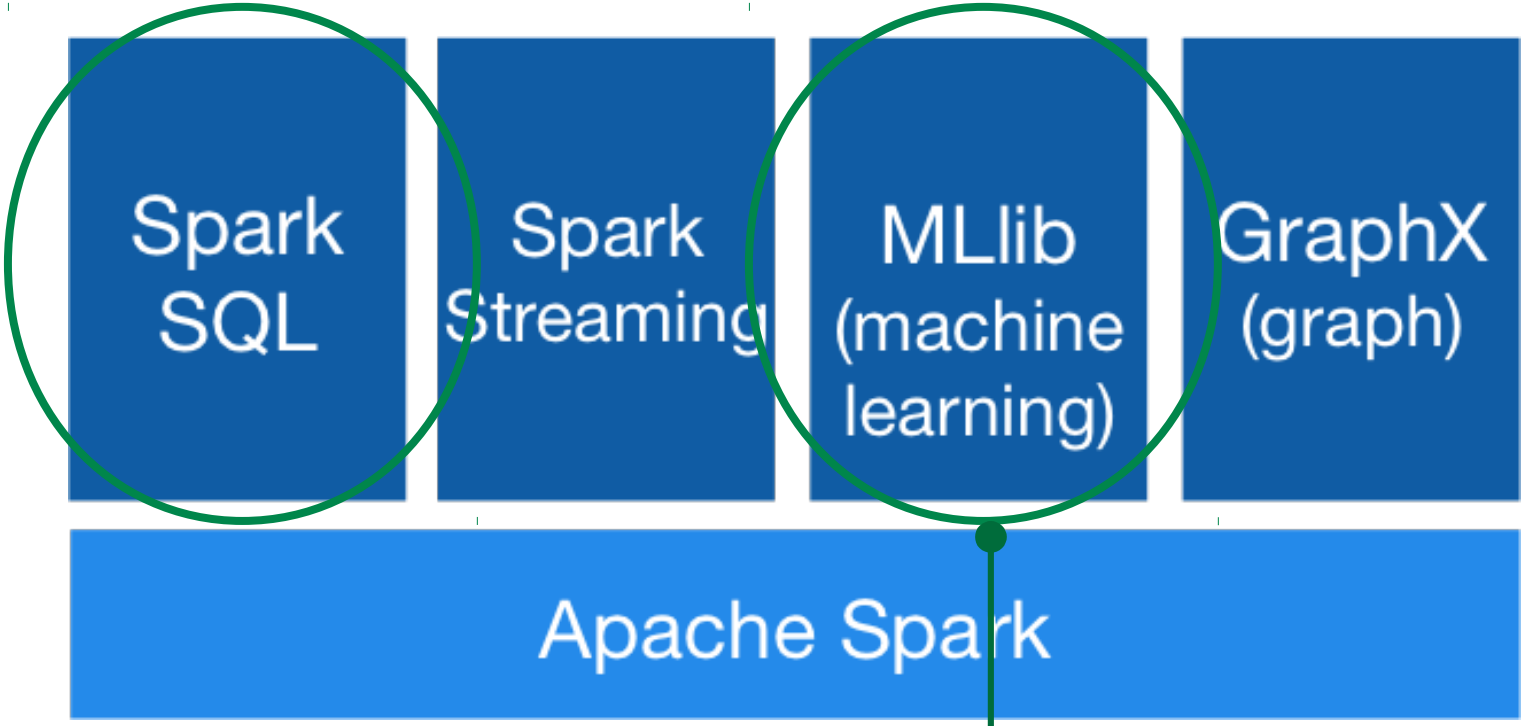
GDAF – HDFS Data

- TGAS, GDR1 and GDR2 in HDFS
  - Access through **hadoop** commands
- Formats
  - **ASCII (Coma-Separated Value)**
    - Header and data in different files
    - Well known and user readable
    - Slow
  - **Parquet**
    - Column-based
    - ~5 times faster than ASCII
    - Efficient for simple data schema

GDAF – Spark – Key Concepts

- User cannot load data into memory! **Lazy loading**
- **RDD**s and **dataframes**
  - Middle steps data “in-memory”
- Offers interactivity through **shells** (python, scala)

GDAF – Spark Libraries (I)



- K-Means
- Gaussian Mixture

GDAF – Spark Libraries (II)

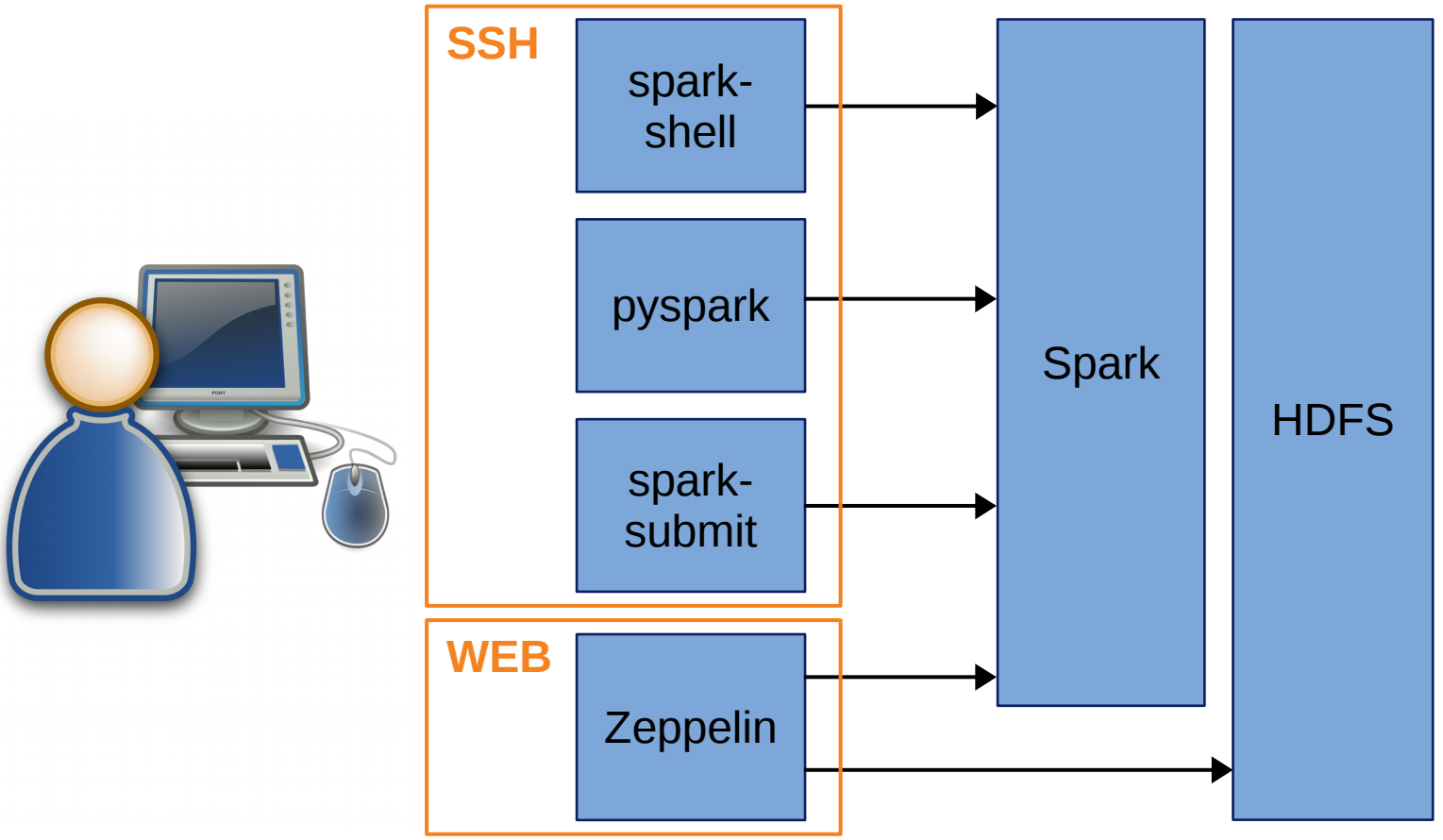
- **DBScan** (scala)
  - Integrated and tested for few dimensions
- **HMAC** (scala)
  - Integrated and intial testing
- **Cassandra** (scala)
  - Integrated and tested
- For future
  - SOM
  - MultiNest
  - SparkR

GDAF – Non-Spark Libraries

- Scikit
- Numpy
- Matplotlib
- AstroABC

¡NOT FOR DISTRIBUTED COMPUTING!

GDAF – UI



2 execution queues

GDAF – Zeppelin (I)

- Intepreters
  - Scala, SQL, pyspark, bash, cassandra, R, python, etc.
- Built-in visualization tool
- Multi user support
- Security management and authentication integrated
- Last stabel version 0.7.3
  - Waiting for 0.8.0

GDAF – Zeppelin (II)

Load Data FINISHED

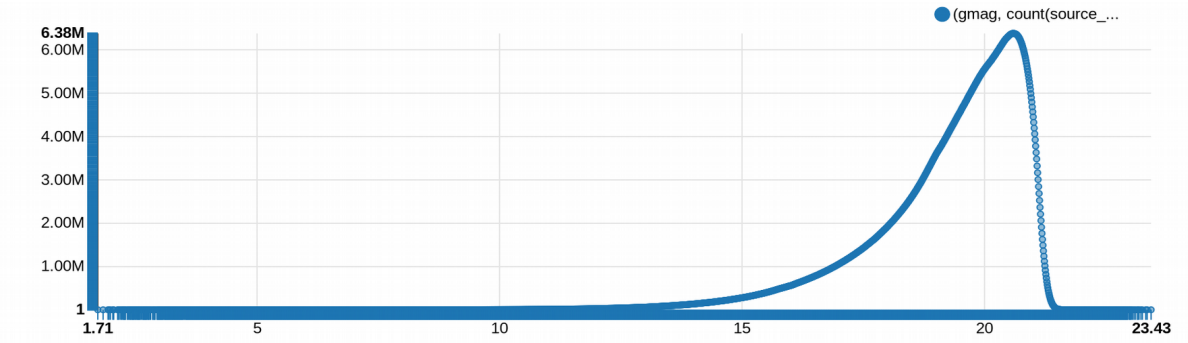
```
%pyspark

gdr2 = spark.read.parquet("/archive/parquet/gaia/gdr2/gaiaSource")
gdr2.registerTempTable("gdr2")
```

Built-in plotting FINISHED

```
%spark

val selection = sqlContext.sql("select count(source_id), round(phot_g_mean_mag,2) as gmag from gdr2 group by gmag")
z.show(selection)
```



GDAF – Zeppelin (III)

Zeppelin Notebook Job

06-DBScan Head

```
val clusterIds = model.points.filter(_.clusterId > 0)
                             .map(_.clusterId)
                             .distinct()
                             .collect()
```

```
res41: Long = 2
clusterIds: Array[Int] = Array(100001, 100002)
```

Took 27 sec. Last updated by gaia at September 12 2017, 9:34:14 AM. (outdated)

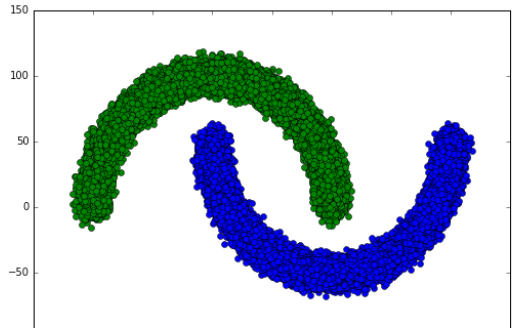
```
%pyspark

import matplotlib.pyplot as plt

data = sc.textFile("/tmp/dbscan-output/output1")
parsedData = data.map(lambda line: [float(x) for x in line.split(',')])

colors = ['go','bo','yo','mo','yo','ko','ro']
clusterIds = [100001, 100002]
for i in range (0,2):
    clusterId = clusterIds[i]
    cluster = parsedData.filter(lambda point: point[2] == clusterId)
    x = cluster.map(lambda point: point[0]).collect()
    y = cluster.map(lambda point: point[1]).collect()
    plt.plot(x,y,colors[i])
```

```
[<matplotlib.lines.Line2D object at 0x7fa75a127fd0>]
[<matplotlib.lines.Line2D object at 0x7fa75a4b3b90>]
```



Load coordinates and densities FINISHED

```
%pyspark

src = "/tmp/plots/HeatMap/*"
points = sqlContext.read.parquet(src)
x = points.select("alpha").collect()
y = points.select("delta").collect()
z = points.select("score").collect()
```

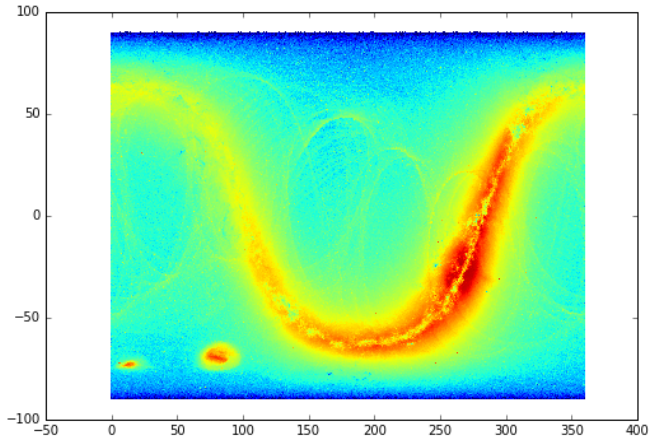
Took 1 min 46 sec. Last updated by gaia at November 15 2017, 5:24:01 PM.

Plot heatmap FINISHED

```
%pyspark

from gaia.dpac.cu9.gdaf import HeatMap

hm = HeatMap()
hm.plotHeatMap(x,y,z)
```



Took 3 min 1 sec. Last updated by gaia at November 15 2017, 5:27:04 PM.

GDAF – Virtual Machine

- Ready for VirtualBox
- 16 GB dynamically allocated disk space
- 4 GB RAM
- 2 CPUs
- Only TGAS

Don’t hesitate, use it!

Astronomy & Astrophysics manuscript no. aanda
May 25, 2018

BGM FASt: Besançon Galaxy Model for Big Data

Inferring the IMF, the Solar Neighbourhood population density and its SFH

R. Mor[1], A.C. Robin[2], F. Figueras[1], and T. Antoja[1]

[1] Dept. Física Quàntica i Astrofísica, Institut de Ciències del Cosmos, Universitat de Barcelona (IEEC-UB), Martí Franquès 1, E08028 Barcelona, Spain. e-mail: rmor@fqa.ub.edu
[2] Institut Utinam, CNRS UMR6213, Université de Bourgogne Franche-Comté, OSU THETA , Observatoire de Besançon, BP 1615, 25010 Besançon Cedex, France

- Open Clusters using DBSCAN
- Source distances

Future

- Platform ready
- Expecting more use cases
  - New DM libraries
  - API requirements
- Preparing for DR3
  - Don’t wait until last time
- User local tests
  - Virtual Machine
  - VirtualEnv

Questions