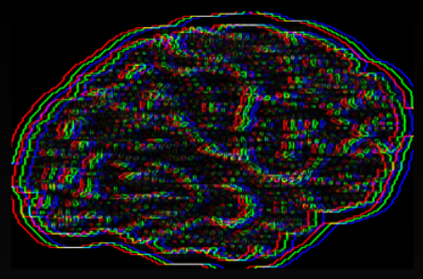


#sosc2022

#sosc22

#sosc2022

4th International School on Open Science Cloud

Distributing data analysis

SOSC22@Perugia

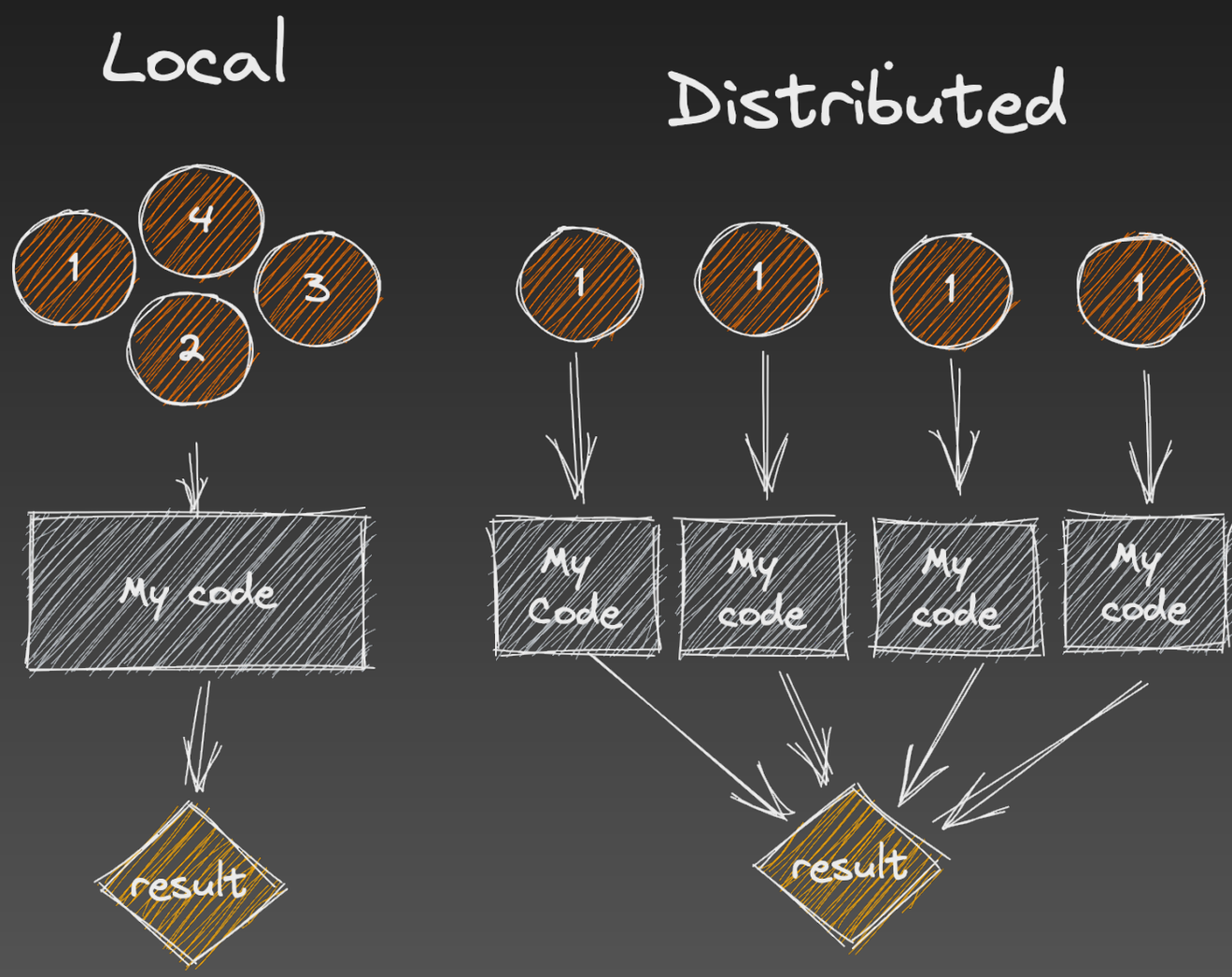
Diego Ciangottini - ciangottini@pg.infn.it

What is distributed computing

... and why it matters

- Increasing the amount of data to process, one can think of **distributing the execution of the very same code over more than one machine**
- Results have to be collected and merged afterwards

Depending on the amount of data and the complexity of the code, **the "complexity" introduced by going distributed can be worth**.



ciangottini@pg.infn.it

What does it mean?

Pros and cons

- **PROs**
  - Faster/parallel processing
  - … not much more actually :)
- **CONs**
  - Reading data is not from your local disk anymore. It has to be even downloaded locally or streamed somehow
    - This can cause inefficiencies and errors
  - Collecting and merging data that have been processed must be managed by either the analyser or the analysis framework

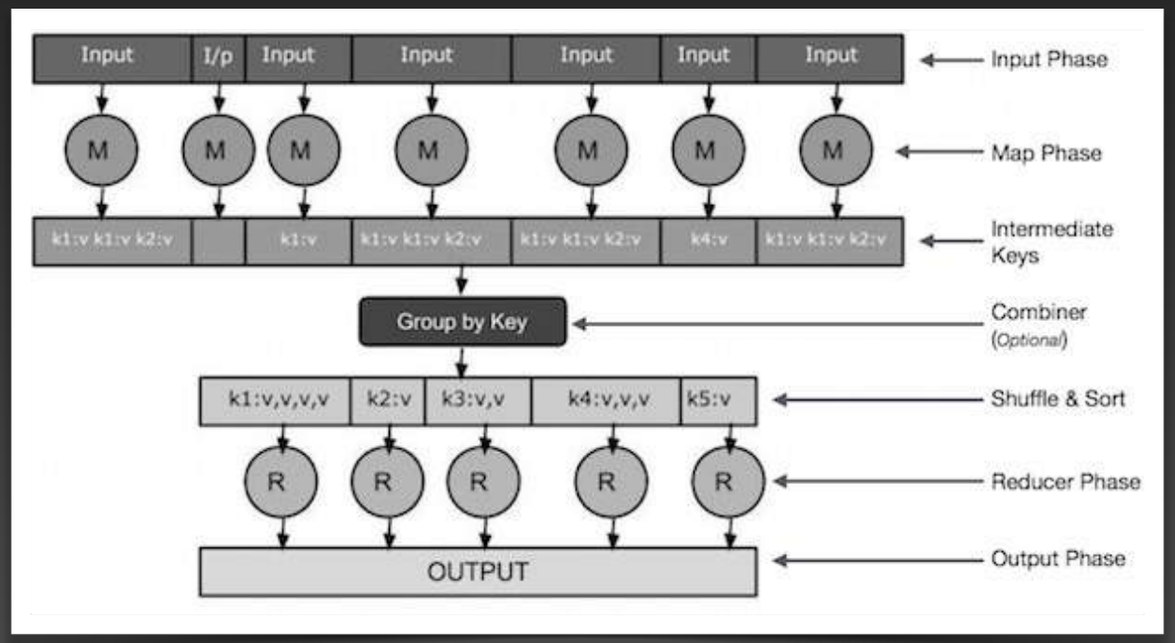
ciangottini@pg.infn.it

Big data and map-reduce success

What is a Map? And what about reduce?

- The Map task takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key-value pairs).

- The Reduce task takes the output from the Map as an input and combines those data tuples (key-value pairs) into a smaller set of tuples.



ciangottini@pg.infn.it

Main map-reduce actors

SPARK vs DASK

- Spark is written in Scala with some support for Python and R.
  - It interoperates well with other JVM code.
- Spark is an all-in-one project that has inspired its own ecosystem.
  - It integrates well with many other Apache projects.

- Dask is written in Python and only really supports Python.
  - It interoperates well with C/C++/Fortran/LLVM
- Dask is a component of the larger Python ecosystem
  - It couples with and enhances other libraries like NumPy, pandas, and Scikit-learn.

ciangottini@pg.infn.it

Main map-reduce actors

SPARK vs DASK

- Spark is more focused on traditional business intelligence operations like SQL and lightweight machine learning.
- Spark DataFrame has its own API and memory model. It also implements a large subset of the SQL language
  - Spark includes a high-level query optimizer for complex queries.

- Dask is applied more generally both to business intelligence applications, as well as a number of scientific and custom situations.
- Dask DataFrame reuses the Pandas API and memory model. It implements neither SQL nor a query optimizer.
  - It is able to do random access, efficient time series operations, and other Pandas-style indexed operations.

ciangottini@pg.infn.it

What to choose?

SPARK vs DASK

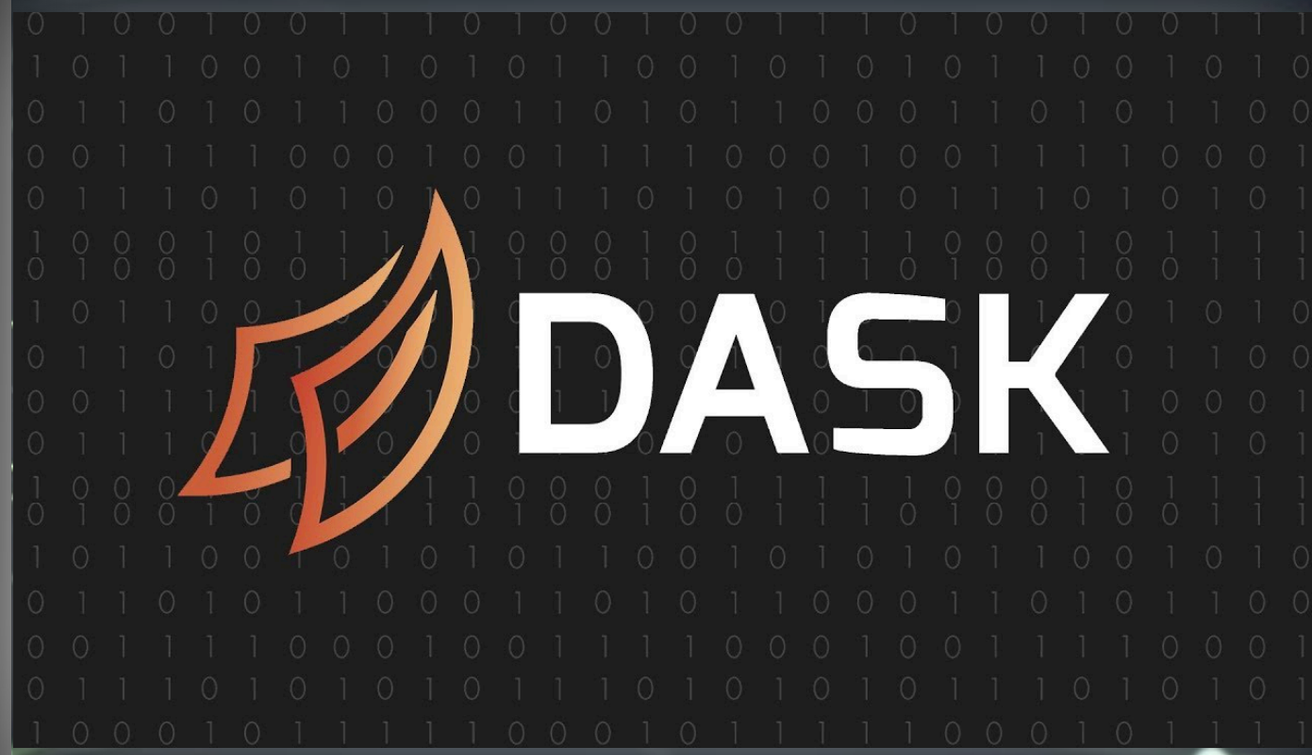
- You prefer Scala or the SQL language
- You have mostly JVM infrastructure and legacy systems
- You want an established and trusted solution for business
- You are mostly doing business analytics with some lightweight machine learning
- You want an all-in-one solution

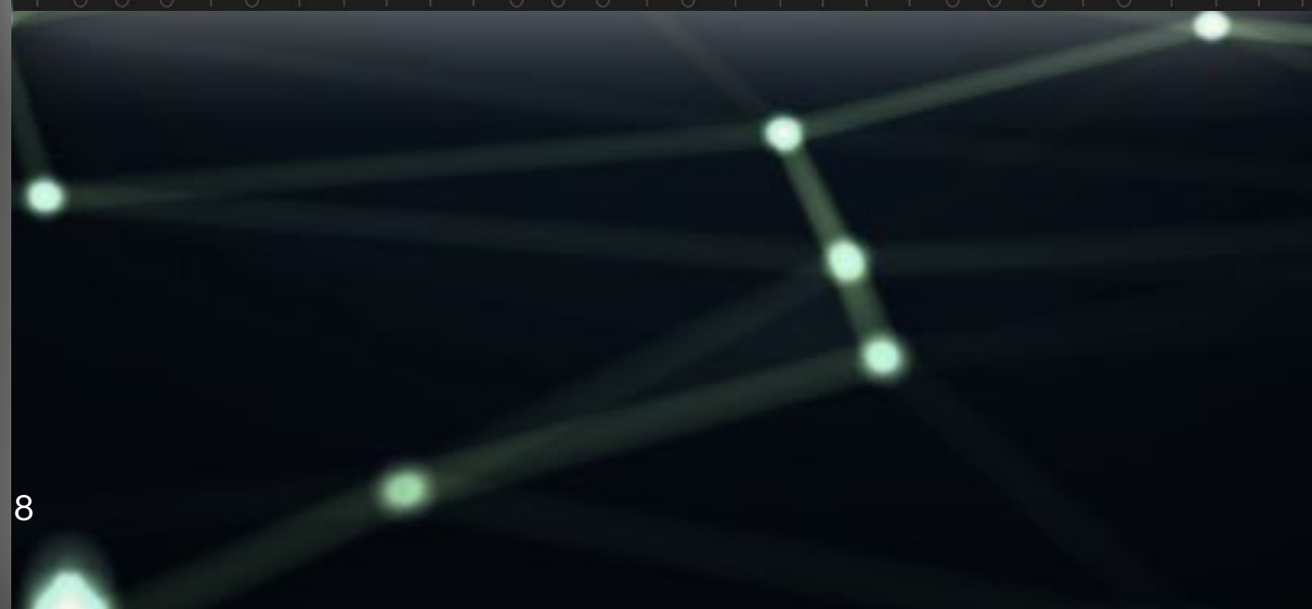
- You prefer Python or native code, or have large legacy code bases that you do not want to entirely rewrite
- Your use case is complex or does not cleanly fit the Spark computing model
- You want a lighter-weight transition from local computing to cluster computing
- You want to interoperate with other technologies and don’t mind installing multiple packages

ciangottini@pg.infn.it

For all these reasons we are going to focus on DASK

Deep dive by examples





Interactive workflows with Dask

Key concepts and their implementation

- Arrays
  - coordinate many Numpy arrays, arranged into chunks within a grid. They support a large subset of the Numpy API.
- Bag
  - Dask Bag implements operations like `map`, `filter`, `groupby` and aggregations on collections of Python objects.
- Dataframes
  - coordinate many Pandas dataframes, partitioned along an index. They support a large subset of the Pandas API.
- Delayed
  - a simple and powerful way to parallelize existing code. It allows users to delay function calls into a task graph with dependencies.

ciangottini@pg.infn.it