Introduction to Spark

Quách Đình Hoàng

Big Data and Distributed Computing at Google

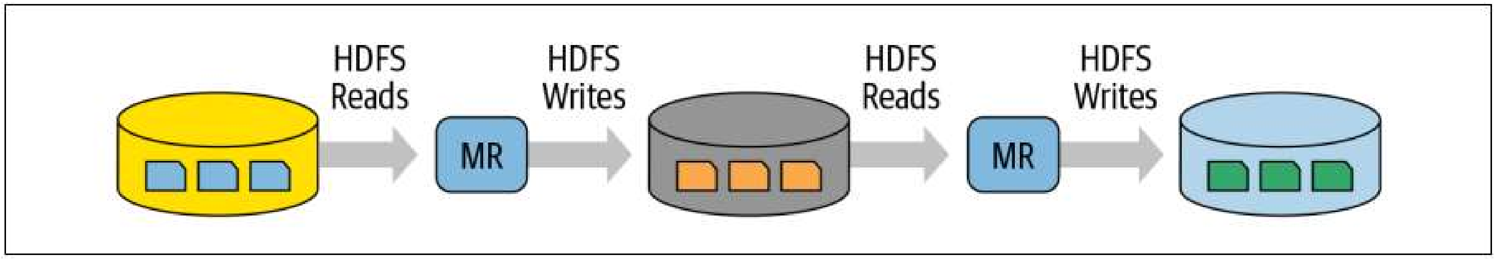
- Three important papers of Google for big data challenges of scale for its search engine
  - Google File System (GFS): a fault-tolerant and distributed file system across many commodity hardware servers in a cluster farm
  - Bigtable: a scalable storage of structured data across GFS
  - MapReduce (MR): a new parallel programming paradigm, based on functional programming, for large-scale processing of data distributed over GFS and Bigtable.

Hadoop at Yahoo!

- Google's GFS and MapReduce papers provided a blueprint for
  - the Hadoop File System (HDFS), and
  - MapReduce implementation as a framework for distributed computing
- Yahoo! donated HDFS and MapReduce implementation for Apache Software Foundation (ASF) in April 2006
- Apache Hadoop framework:
  - Hadoop Common,
  - MapReduce,
  - HDFS, and
  - Apache Hadoop YARN

Hadoop shortcomings

- The MapReduce framework on HDFS had a few shortcomings
  - It was complex to learn and hard to manage
  - MapReduce API was verbose (a lot of code) and low fault tolerance
  - Intermediate results are written to the local disk for the subsequent stage → not good for iterative computing jobs because high cost of I/O operations
  - It good for batch processing, but not good for combining other workloads such as machine learning, streaming, or interactive SQL-like queries
    - → Apache Hive, Storm, Impala, Giraph, Drill, Mahout, etc. → complexity



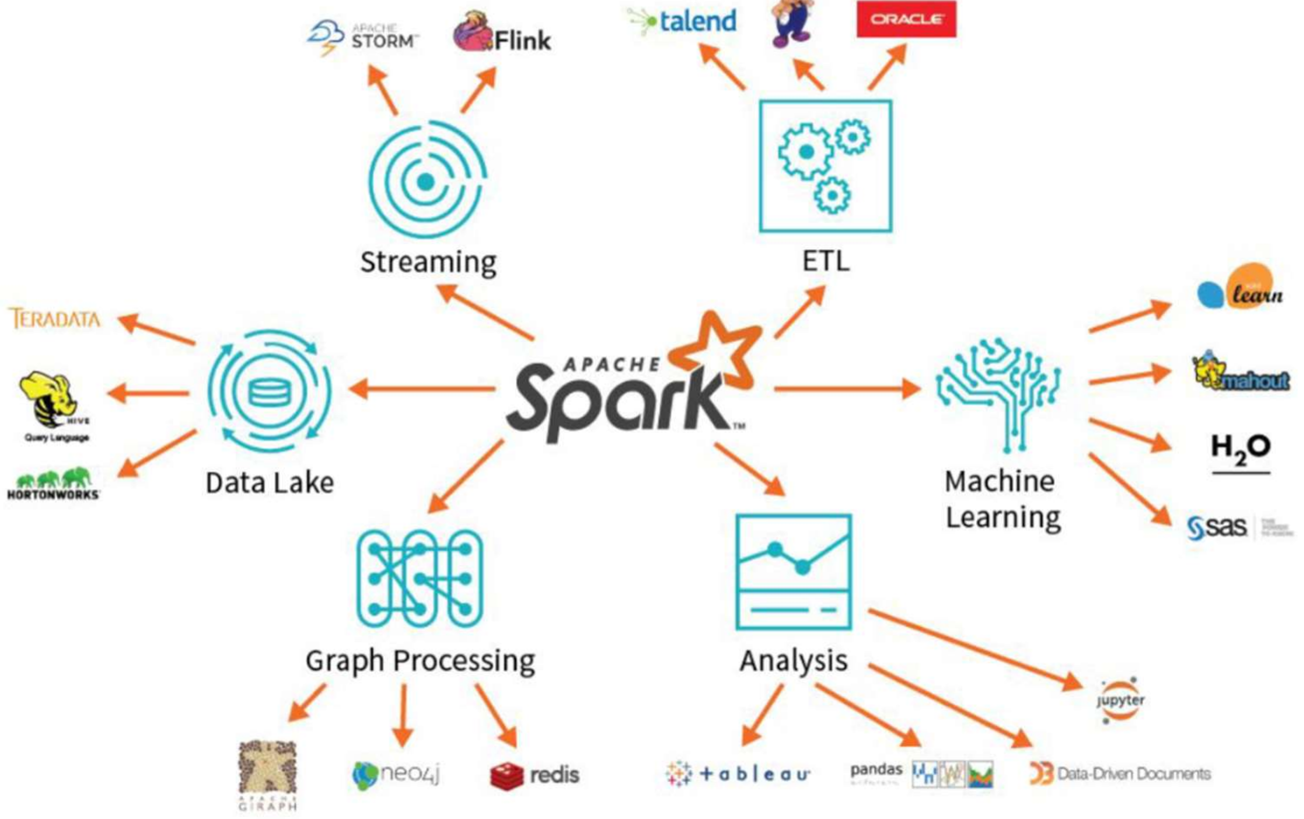
Spark's Early Years at AMPLab

- Researchers at UC Berkeley created Spark to deal with Hadoop shortcomings
  - Make it simpler, faster, and easier
  - started in 2009 at the RAD Lab → the AMPLab → RISELab
- 2010 Spark paper demonstrated that it was 10 to 20 times faster than Hadoop MapReduce for certain jobs
- 2013 Spark had gained widespread use and its original creators donated the Spark project to the ASF and formed a company called Databricks.
  - Spark original creators: Matei Zaharia, Ali Ghodsi, Reynold Xin, Patrick Wendell, Ion Stoica, and Andy Konwinski
- Spark 1.0 released in May 2014 (the first major release) by Databricks and the community of open source developers

Apache Spark

- A unified engine designed for large-scale distributed data processing, on premises in data centers or in the cloud.
- Spark provides in-memory storage for intermediate computations, making it much faster than Hadoop MapReduce.
- Spark's design philosophy centers around four key characteristics:
  - Speed
  - Ease of use
  - Modularity
  - Extensibility

Apache Spark – A Unified Analytics Engine



Speed

- The framework is optimized to take advantage of memory, multiple cores, multithreading and parallel processing
- Spark builds its query computations as a directed acyclic graph (DAG)
  - computations is decomposed into tasks and executed in parallel
- Spark retained all the intermediate results in memory → huge performance boost

Ease of Use

- Spark achieves simplicity by providing a abstraction
  - Low level simple logical data structure: Resilient Distributed Dataset (RDD)
  - higher-level structured data abstractions: DataFrame and Dataset
- Spark offers a simple programming model by a set of transformations and actions as operations

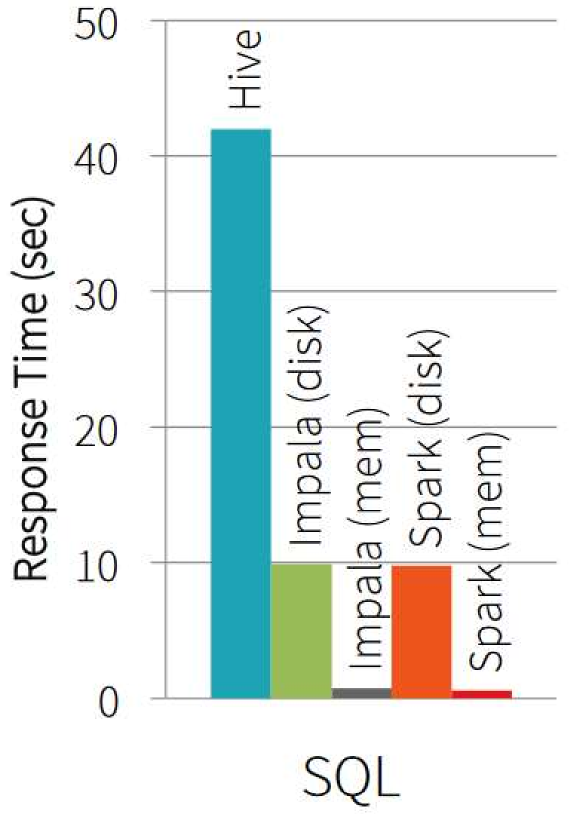
Modularity

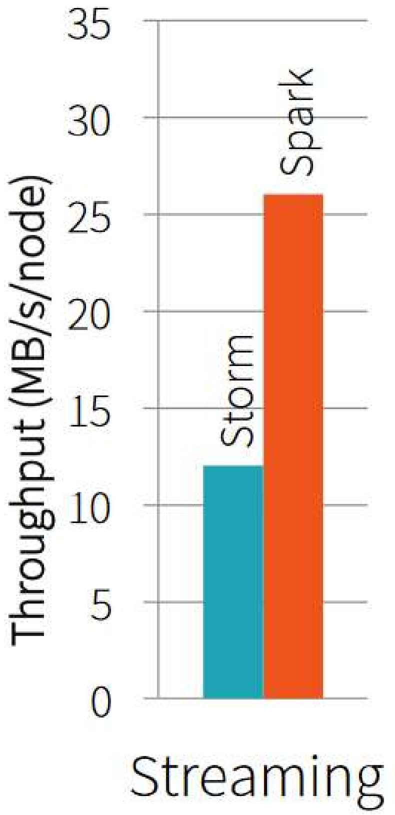
- Spark operations can be applied across many types of workloads and expressed in many programming languages
    - Scala, Java, Python, SQL, and R
- Spark offers unified libraries with well-documented APIs
    - Spark SQL, Spark Structured Streaming, Spark MLlib, and GraphX

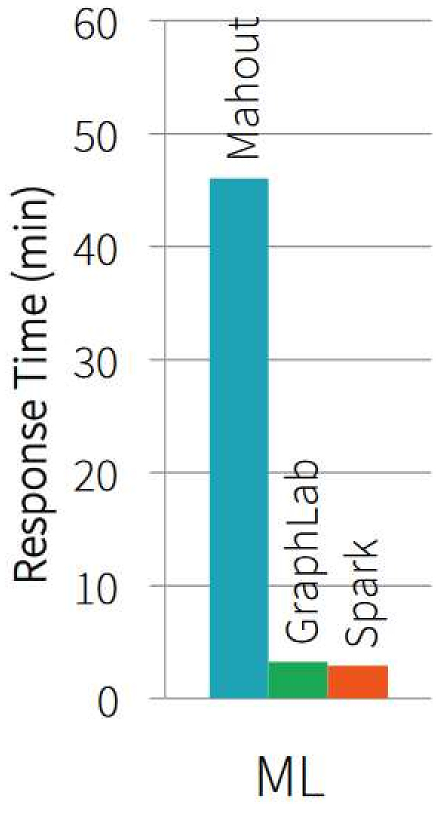
Extensibility

- Spark focuses on its fast, parallel computation engine rather than on storage
  - Hadoop included both storage and compute, Spark decouples the two
- Spark can read data stored in many sources and process it all in memory
  - Apache Hadoop, Cassandra, HBase, MongoDB, Hive, RDBMSs, …
- Spark ecosystem is rich, the community of Spark developers is large

Performance vs Specialized Systems



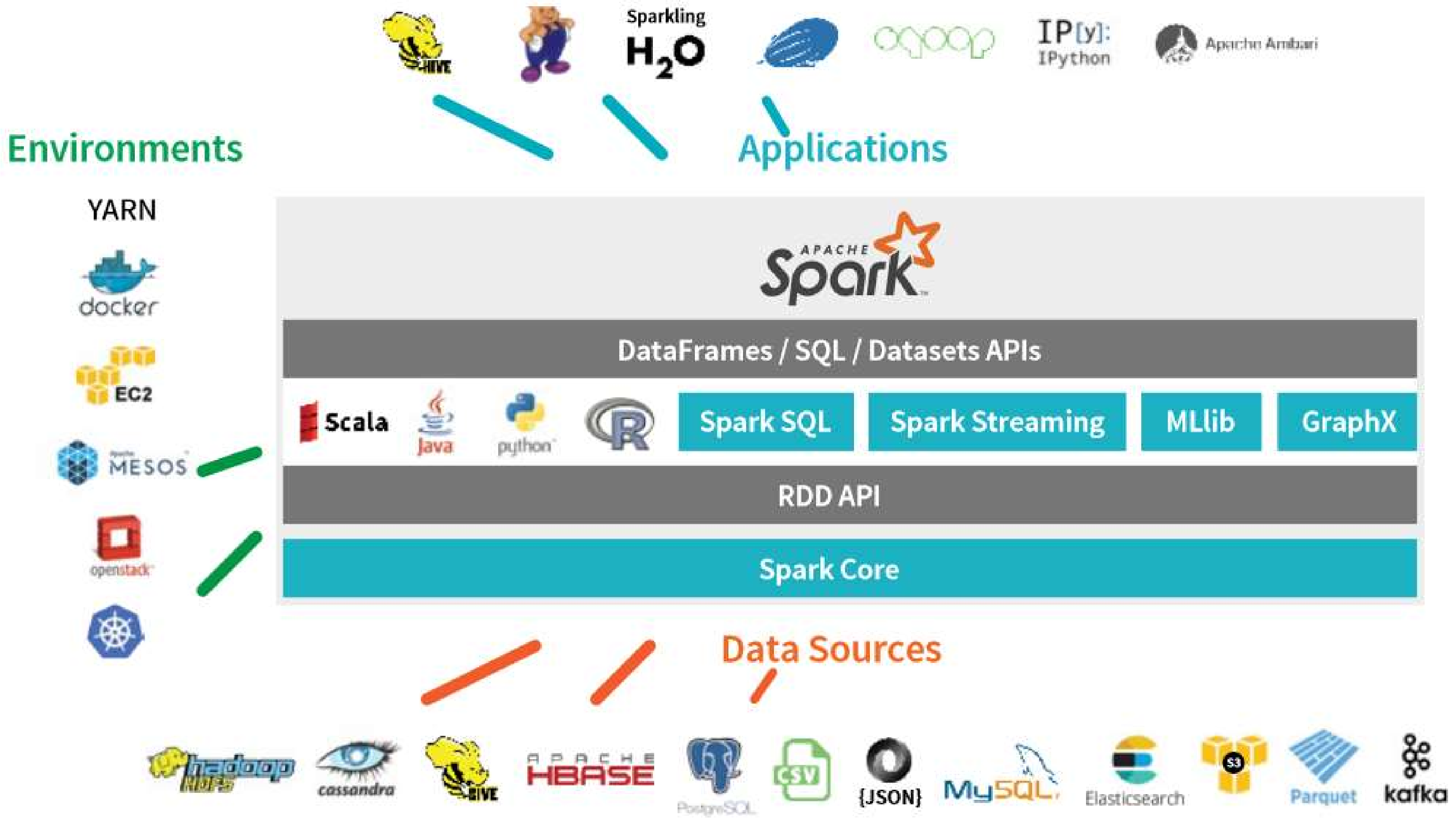




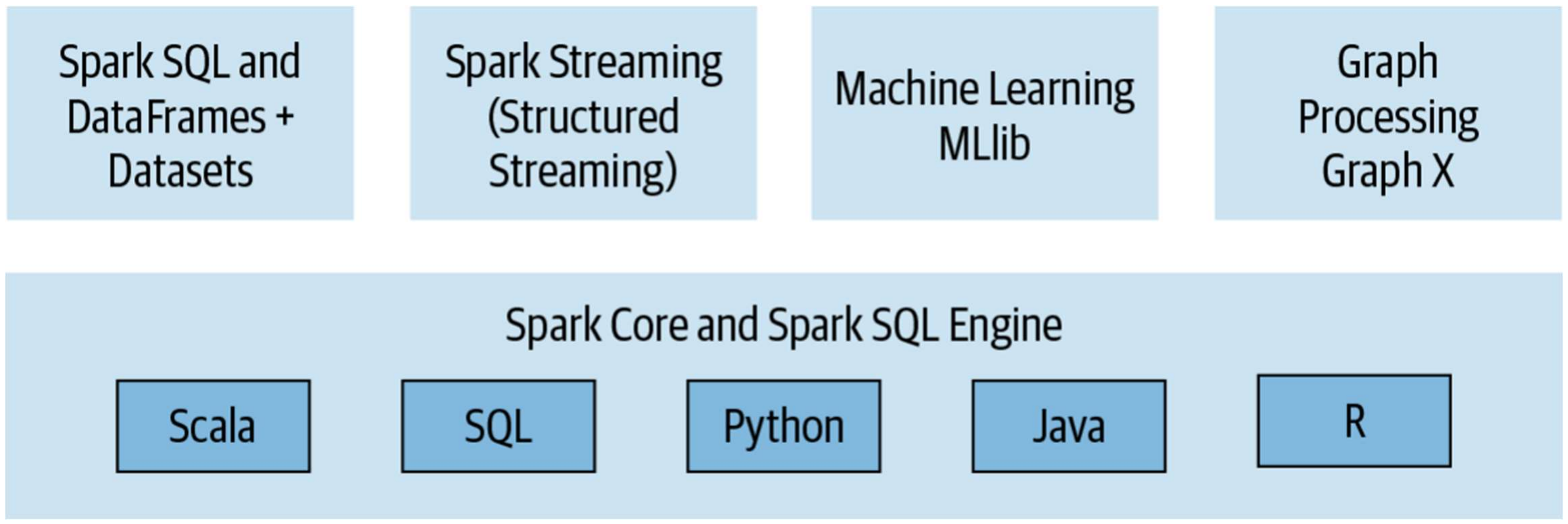
Spark Community

- Over 1000 deployments, clusters up to 8000 nodes

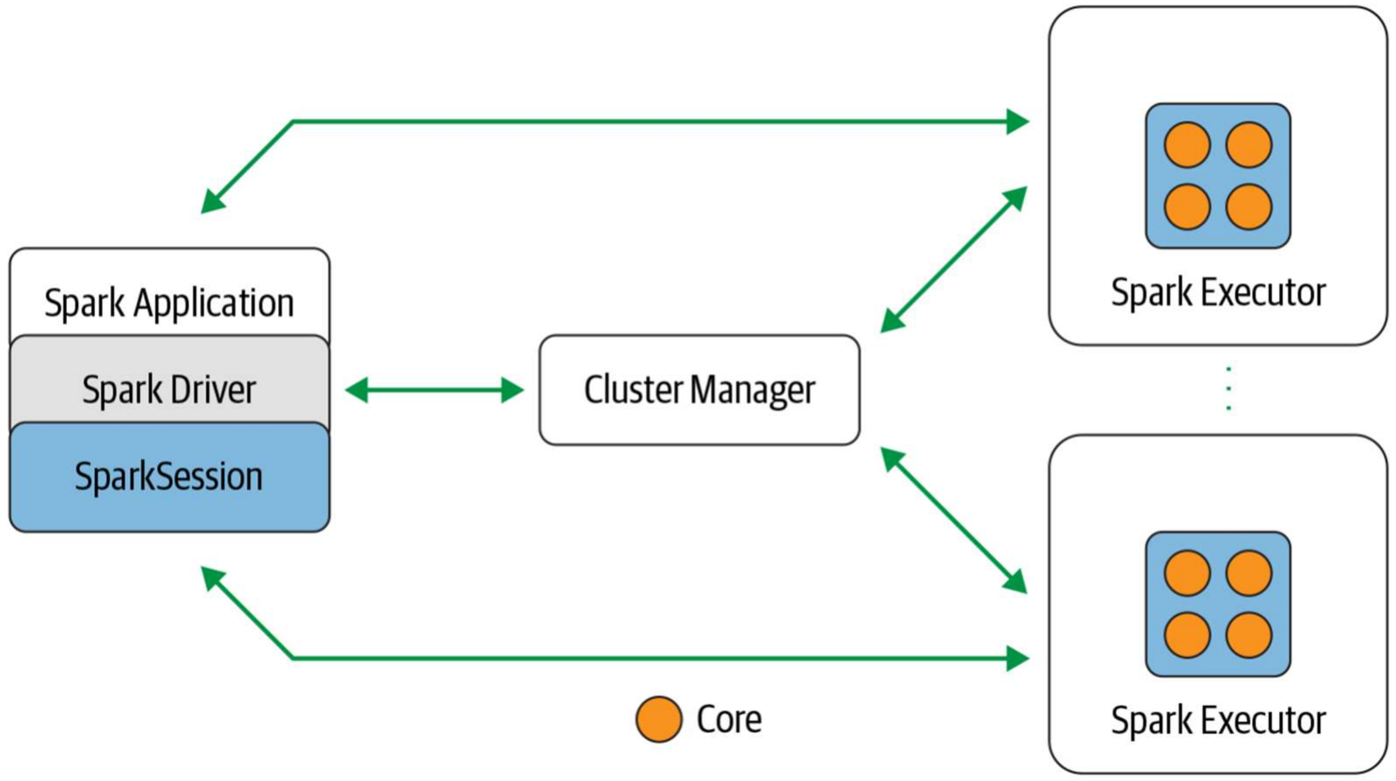
Apache Spark's ecosystem



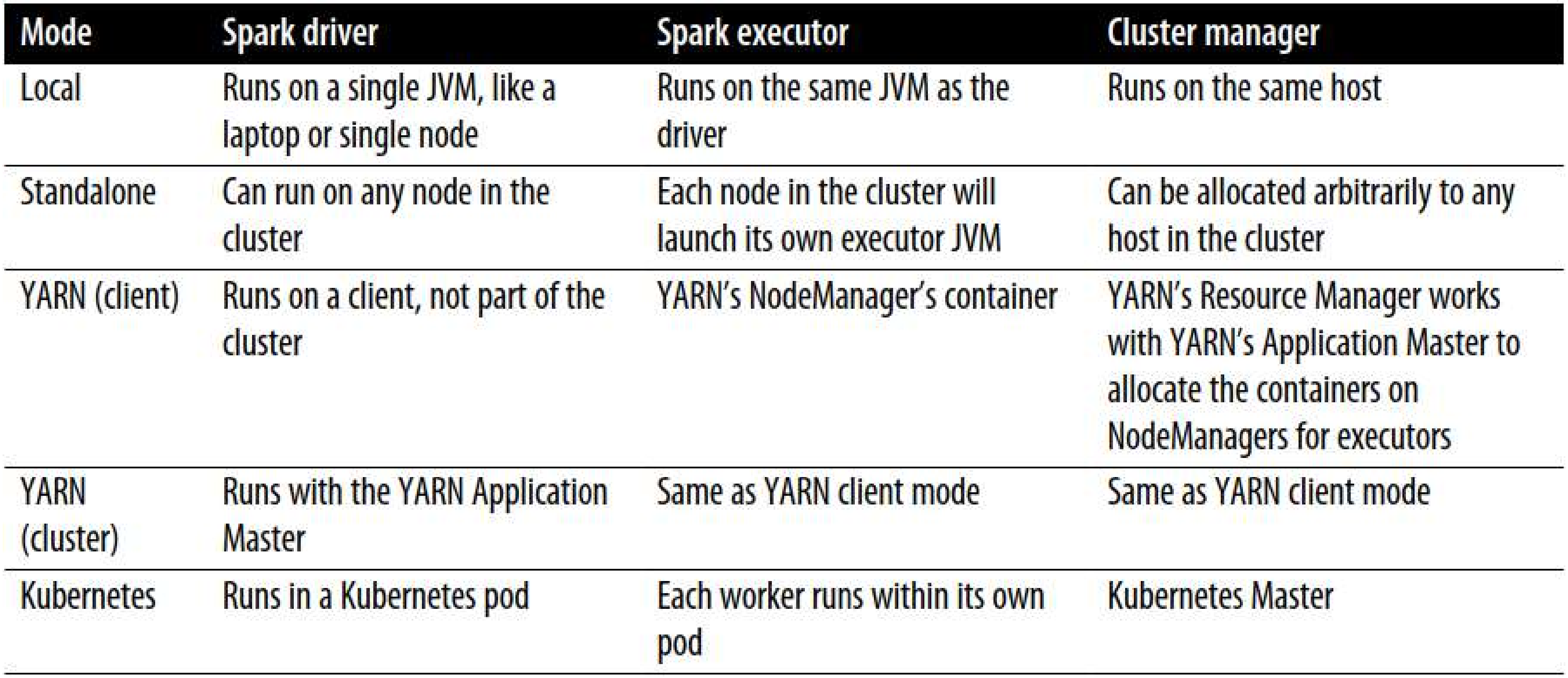
Apache Spark components and API stack


Spark SQL and DataFrames + Datasets
Spark Streaming (Structured Streaming)
Machine Learning MLlib
Graph Processing Graph X
Spark Core and Spark SQL Engine
Scala
SQL
Python
Java
R

Spark components and architecture

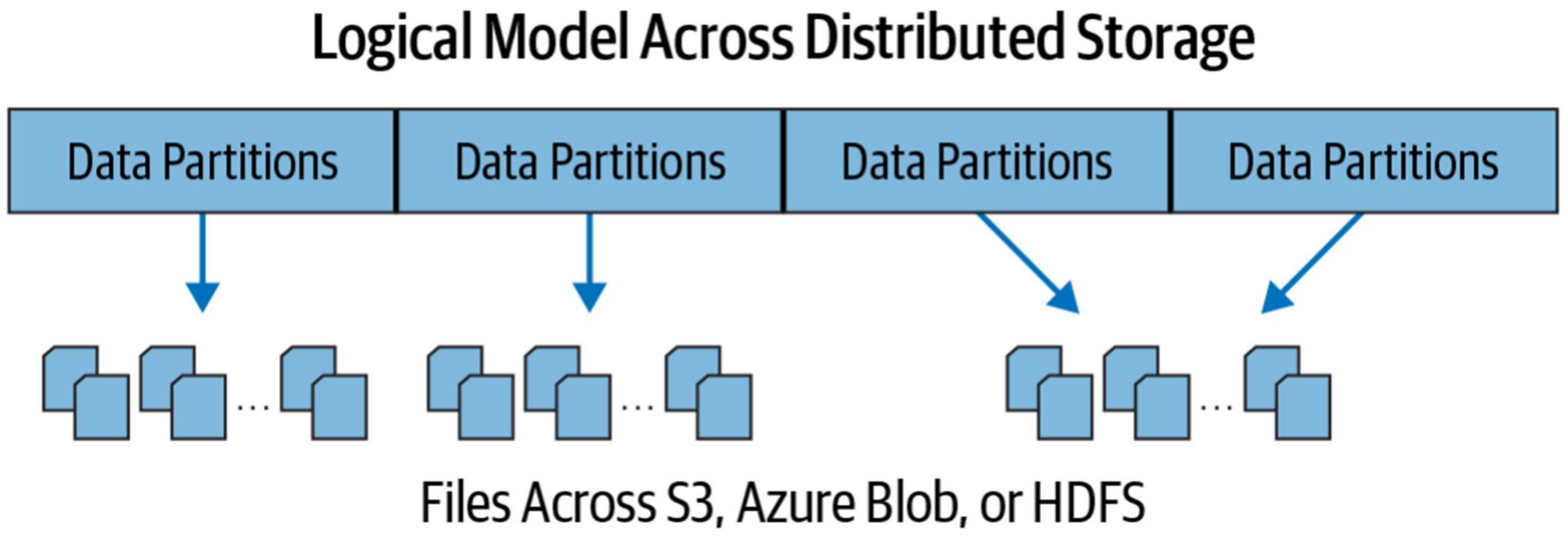


Spark deployment modes

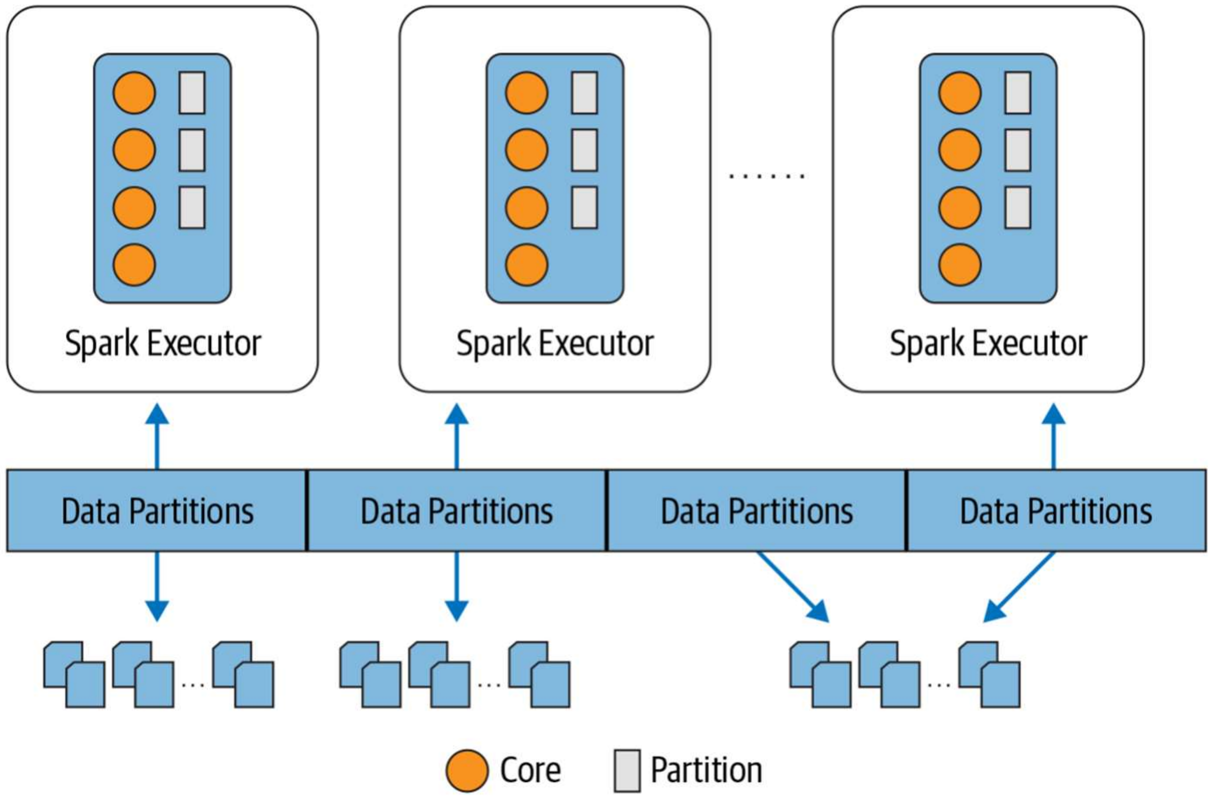


Mode	Spark driver	Spark executor	Cluster manager
Local	Runs on a single JVM, like a laptop or single node	Runs on the same JVM as the driver	Runs on the same host
Standalone	Can run on any node in the cluster	Each node in the cluster will launch its own executor JVM	Can be allocated arbitrarily to any host in the cluster
YARN (client)	Runs on a client, not part of the cluster	YARN's NodeManager's container	YARN's Resource Manager works with YARN's Application Master to allocate the containers on NodeManagers for executors
YARN (cluster)	Runs with the YARN Application Master	Same as YARN client mode	Same as YARN client mode
Kubernetes	Runs in a Kubernetes pod	Each worker runs within its own pod	Kubernetes Master

Data is distributed across physical machines


Logical Model Across Distributed Storage
Data Partitions
Data Partitions
Data Partitions
Data Partitions
...
...
...
Files Across S3, Azure Blob, or HDFS

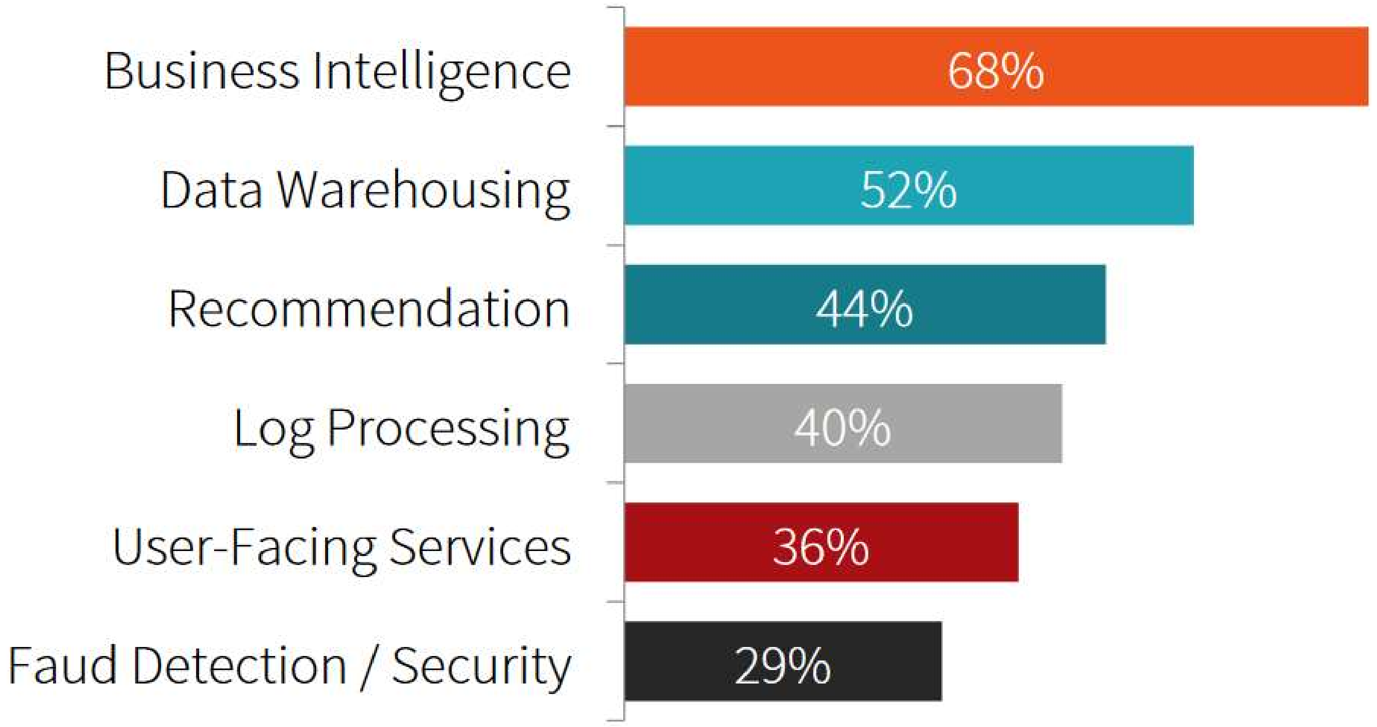
Each executor's core gets a partition of data to work on



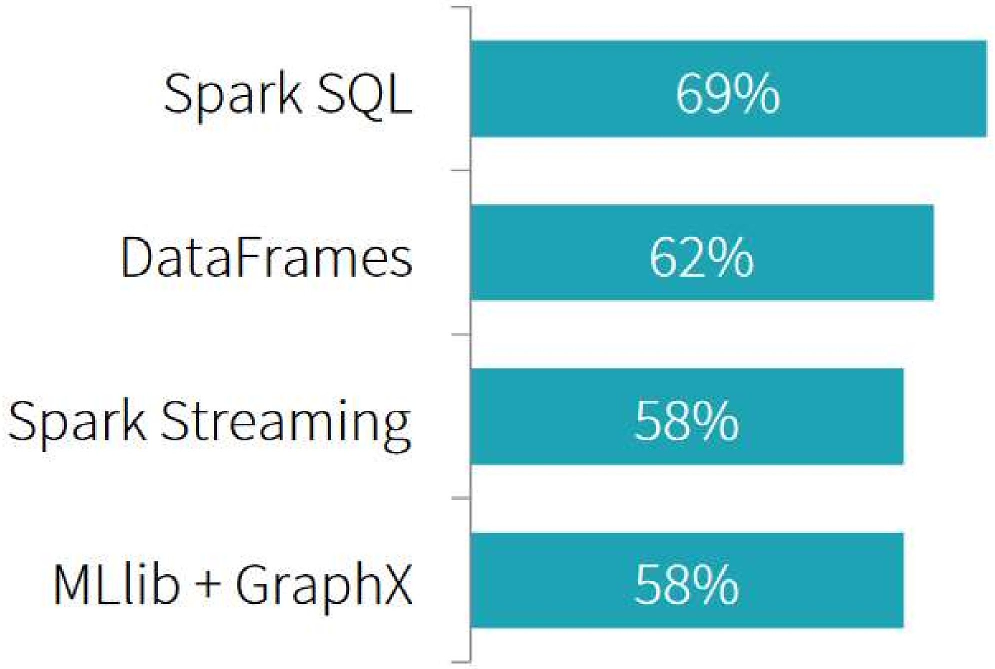
Popular Spark use cases

- Processing in parallel large data sets distributed across a cluster
- Performing ad hoc or interactive queries to explore and visualize data sets
- Building, training, and evaluating machine learning models using MLlib
- Implementing end-to-end data pipelines from myriad streams of data
- Analyzing graph data sets and social networks

Top Applications



Spark Components Used



of users use more than one component

References

- Jules S. Damji, Brooke Wenig, Tathagata Das & Denny Lee, *Learning Spark: Lightning-Fast Data Analytics, 2nd Edition*, O’Reilly, 2020.
- Matei Zaharia, *Making Big Data Processing Simple with Spark*, ACM Techtalks, December 17, 2015, "Making Big Data Processing Simple with Spark," Matei Zaharia - YouTube