Apache Spark

A Primer

databricks™

What is Apache Spark?

Apache Spark™ is an open source data processing engine built for speed, ease of use, and sophisticated analytics. Since its release, Spark has seen rapid adoption by enterprises across a wide range of industries. Internet powerhouses such as Yahoo, Baidu, and Tencent, have eagerly deployed Spark at massive scale, collectively processing multiple petabytes of data on clusters of over 8,000 nodes. Meanwhile it has become the largest open source community in big data, with over 500 contributors from 200+ organizations. To help Spark achieve this growth, Databricks continues to contribute broadly throughout the project, both with roadmap development and with community evangelism.

What is Spark used for?

Spark is a general-purpose engine used for many types of data processing. Spark comes packaged with support for ETL, interactive queries (SQL), advanced analytics (e.g. machine learning) and streaming over large datasets. For loading and storing data, Spark integrates with many storage systems (e.g. HDFS, Cassandra, HBase, S3). Spark is also pluggable, with dozens of third party libraries and storage integrations.

Additionally, Spark supports a variety of popular development languages including R, Java, Python and Scala.

" At Databricks, we're working hard to make Spark easier to use and run than ever, through our efforts on both the Spark codebase and support materials around it. All of our work on Spark is open source and goes directly to Apache. "

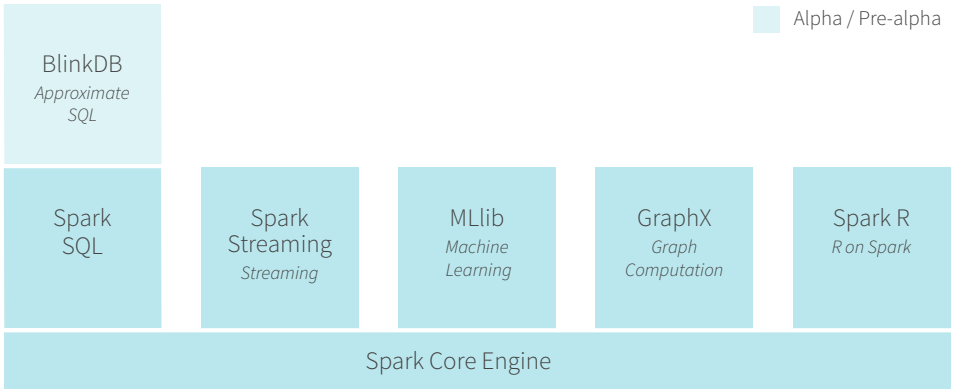
—Matei Zaharia, VP, Apache Spark, Founder & CTO, Databricks

How does Spark work?

Spark takes programs written in a high-level concise language, and distributes execution onto many machines. It achieves this through an API built on Resilient Distributed Datasets (RDDs) – a distributed dataset abstraction that performs calculations on large clusters in a fault-tolerant manner.

Spark's architecture differs from earlier approaches in several ways that improves its performance significantly. To begin with, Spark allows users to take advantage of memory centric computing architectures by persisting RDD datasets in-memory, enabling fast iterative processing use cases such as interactive querying or machine learning. Spark's high level API's also enable further intelligent optimization of user programs. As a testament to its performance, Spark currently holds the world record in disk-based sorting, winning the Daytona GraySort 2014 contest.

However, sheer performance is not the only distinctive feature of Spark, its true power lies in flexibility. Spark unifies previously disparate functionalities including batch processing, advanced analytics, interactive exploration, and real-time stream processing in a single unified data processing framework.



The Spark Ecosystem. Image courtesy of Databricks.

What are the benefits of Spark?

Spark was initially designed for interactive queries and iterative algorithms, as these were two major use cases not well served by batch frameworks like MapReduce. Consequently Spark excels in scenarios that require fast performance, such as iterative processing, interactive querying, large-scale batch computations, streaming, and graph computations.

Developers and enterprises typically deploy Spark because of its inherent benefits:

Simple

Easy-to-use APIs for operating on large datasets. This includes a collection over 100 operators for transforming data and familiar data frame APIs for manipulating semi-structured data.

Fast

Engineered from the bottom-up for performance, running 100x faster than Hadoop by exploiting in memory computing and other optimizations. Spark is also fast when data is stored on disk, and currently holds the world record for large-scale on-disk sorting.

Unified Engine

Packaged with higher-level libraries, including support for SQL queries, streaming data, machine learning and graph processing. These standard libraries increase developer productivity and can be seamlessly combined to create complex workflows.

Broadly Compatible

Built-in support for many data sources, such as HDFS, RDBMS, S3, Cassandra, and MongoDB.

" MapReduce is an implementation of a design that was created more than 15 years ago. Spark is a from-scratch reimagined or re-architecting of what you want out of an execution engine given today's hardware."

—Patrick Wendell, Founding committer, Apache Spark & co-founder, Databricks

Is Spark competitive to Hadoop?

In the broad context of Hadoop the ecosystem, Spark is not competitive with Hadoop. In fact, Spark is designed to interoperate seamlessly with the Hadoop stack. Spark can read from any input source that MapReduce supports, ingest data directly from Apache Hive warehouses, and runs on top of the Apache YARN resource manager.

In the narrow context of the Hadoop MapReduce processing engine, Spark represents a modern alternative to MapReduce, based on a more performance oriented and feature rich design. In many organizations, Spark has succeeded MapReduce as the engine of choice for new projects, especially for projects involving multiple processing models or where performance is mission critical. Spark is also evolving much more rapidly than MapReduce, with significant feature additions occurring on a regular basis.

Spark is also widely used outside of Hadoop environments, since the Spark engine has no required dependency on the Hadoop stack. For instance, companies use Spark to crunch data in “NoSQL” data stores such as Cassandra and MongoDB, cloud storage offerings like Amazon S3, or traditional RDBMS data warehouses.

What are some common Spark use cases?

Because of its unique combination of performance and versatility, Spark is being deployed by over 500 organizations in many industry verticals, across a wide range of use cases. While innovators are constantly deploying Spark in creative and disruptive ways, common use cases include:

Data integration and ETL

Cleansing and combining data from diverse sources for visualization or processing in the future. Examples include Palantir's data analytics platform in the government and financial sectors.

Interactive analytics or business intelligence

Gaining insight from massive data sets to inform product or business decisions in ad hoc investigations or regularly planned dashboards. Examples include Goldman Sachs' analytics platform in the financial sector and Huawei's query platform in the telecom sector.

High performance batch computation

Reducing time to run complex algorithms against large scale data. Examples include Novartis' genomic research in the pharma sector and MyFitnessPal / UnderArmour's food database in the health & wellness sector.

Machine learning and advanced analytics

The application of sophisticated algorithms to predict outcomes, inferring hidden information, or making decisions based on input data. Examples include Alibaba's analysis of its marketplace in the retail sector and Spotify's music recommendation engine in the media sector.

Real-time stream processing

Capturing and processing data continuously with low latency and high reliability. Examples include Netflix's streaming recommendation engine in the media sector and British Gas' connected homes in the energy sector.

Who is Databricks?

Databricks was founded by the team behind Apache Spark, the most active open source project in the big data ecosystem today. Our mission at Databricks is to dramatically simplify big data processing and free users to focus on turning their data into value. We do this through our product, Databricks, that is powered by Spark.

For more information on Databricks, download the Databricks Primer.

Take Spark for a test drive on Databricks with a free trial account.

databricks.com/registration