Apache Hadoop (and MapReduce)

Reference:
[1] Apache Hadoop 2.9.0 Map Reduce Tutorial
https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html, last viewed 2018.

[2] Liveramp: https://liveramp.com/engineering/tracking-mapreduce-job-performance-with-counters/, last viewed 2018.

Where is Hadoop used? Everywhere. It has become a part of Bigdata infrastructure for any company with Bigdata pipelines, Bigdata analytics and Bigdata products, in general. [2]

From [1]:

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

A MapReduce *job* usually splits the input data-set into independent chunks which are processed by the *map tasks* in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the *reduce tasks*. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

Typically the compute nodes and the storage nodes are the same, that is, the MapReduce framework and the Hadoop Distributed File System (see HDFS Architecture Guide) are running on the same set of nodes. This configuration allows the framework to effectively schedule tasks on the nodes where data is already present, resulting in very high aggregate bandwidth across the cluster.

The MapReduce framework consists of a single master `ResourceManager`, one slave `NodeManager` per cluster-node, and `MRAppMaster` per application.

Minimally, applications specify the input/output locations and supply *map* and *reduce* functions via implementations of appropriate interfaces and/or abstract-classes. These, and other job parameters, comprise the *job configuration*.

The Hadoop *job client* then submits the job (jar/executable etc.) and configuration to the `ResourceManager` which then assumes the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.

Although the Hadoop framework is implemented in Java™, MapReduce applications need not be written in Java.