Technical Brief

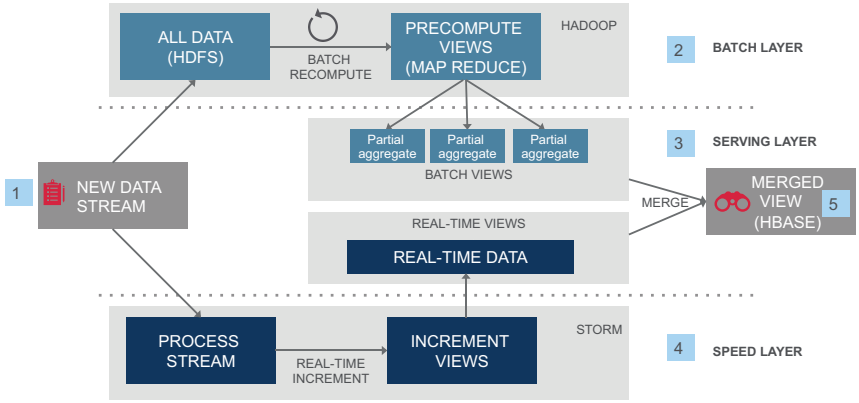
Stream Processing with MapR

Stream Processing with MapR

Introduction

MapR enables the Enterprise Data Hub—a storage and processing infrastructure for big data that not only provides complete data life-cycle functionality from ingestion to archival but also supports a wide breadth of data-driven analytical and operational applications that deliver real-time business value. Being able to process data streams at scale and in a reliable manner is a fundamental requirement of the Enterprise Data Hub. This tech-brief delves into data stream processing on Apache™ Hadoop® in the context of the Lambda Architecture[1]- a useful framework to think through the architectural layout of big data systems—where there is a dedicated "speed" layer, entirely focusing on data stream processing. See Figure 1.

Figure 1. The Lambda Architecture for big data systems



1. All data entering the system is dispatched to both the batch layer and the speed layer for processing.
2. The **batch layer** has two functions:
   a. Manage the master dataset, an immutable, append-only set of raw data.
   b. Pre-compute the batch views.
3. The **serving layer** indexes the batch views so that they can be queried in low-latency, ad-hoc way.
4. The **speed layer** compensates for the high latency of updates to the serving layer and deals with recent data only, providing real-time views.
5. Any incoming query can be answered by merging results from batch views and real-time views.

Note that there are several methods available to realize the functionality provided by the different layers described above. In real-world deployments the speed layer is often realized using event processing software such as Spark Streaming and Storm[2], a distributed and fault-tolerant real-time computation system, which we will discuss in greater detail below.

1 http://lambda-architecture.net/

2 http://storm-project.net/

Data Sources and Use Cases Overview

The data streams can potentially come from a variety of sources, depending on the use case and business requirements. Example data streams include:

- Smartphones such as iPhone or Android devices that have a dozen of physical sensors including GPS and accelerometer
- Wearables such as Google Glass that have greater sensor density and volume
- Physical sensors in buildings (such as energy monitoring device offered by Nest/Google[4]) and in transportation systems (from cars to trains)
- Social media streams such as those available from the Twitter fire-hose, enterprise applications like Salesforce or aggregator API, including DataSift[5]

Stream processing is increasingly common with MapR customers. Some high-level use case descriptions of stream processing include:

- In the **manufacturing industry** (for example, car makers) sensors are used to realize pro-active maintenance. For this to happen, a large number of independent sensors need to be taken into account, at scale. The goal here is to inform the owner of a vehicle of ahead of time of potential issues.
- **Retailers** benefit from sensors introduced in supply chain management: being able to track delivery routes more accurately to optimize storage and logistics management
- Another huge application area of stream processing is for **predictive online analysis**[6], be it for churn predictions of mobile phone or online-magazine subscribers or real-time customized ads for credit card owners
- In the context of **online alerting**, MapR customers use stream processing to minimize idle transports, be it for trucks or vessels alike. More details about this use case can be found later in this document

3 http://mashable.com/category/wearables/

4 http://www.forbes.com/sites/parmyolson/2014/01/13/nest-gives-google-its-next-big-data-play-energy/

5 http://datasift.com/

6 http://www.mapr.com/blog/real-time-learning-quick-without-dirty

The MapR Offering

Reliability and high performance across the Lambda Architecture stack

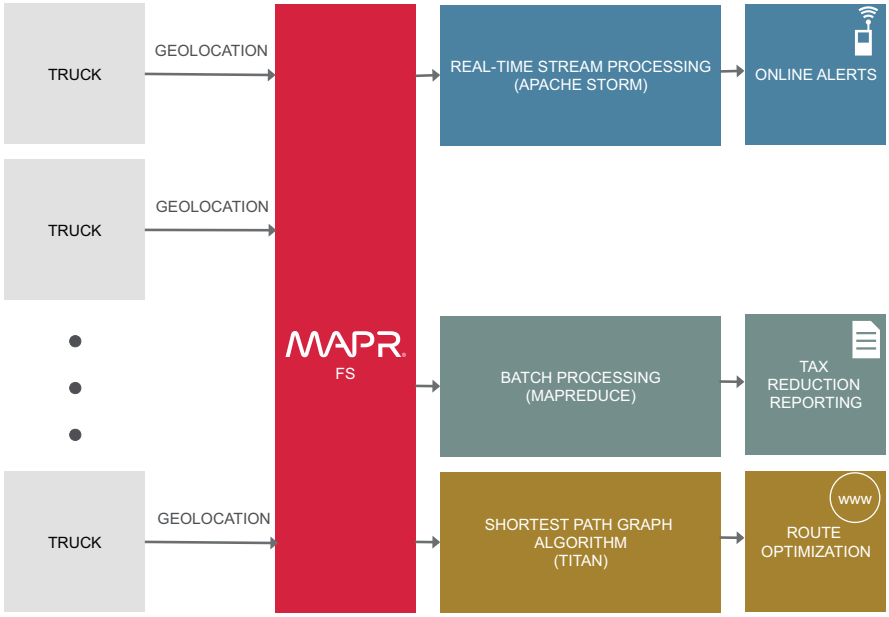
MapR provides a dramatically simplified architecture for real-time, stream processing engines. Streaming data can be written directly to the MapR file system for long-term storage and MapReduce processing (establishing the batch layer of the Lambda Architecture). Because MapR enables data streams to be written directly to the cluster, applications typically don't need queuing systems such as Apache Kafka, reducing the numbers of moving parts. Further, such a model enables publish-subscribe models within the data platform. Storm can 'tail' a file to which it wishes to subscribe, and as soon as new data hits the file system, it is injected into the Storm topology. This allows for strong Storm/Hadoop interoperability, and a unification and simplification of technologies onto one enterprise Hadoop platform.

Another important innovation and improvement in the context of the Lambda Architecture is the MapR M7 Enterprise Database Edition[7], an enterprise-grade, Apache HBase-API compliant NoSQL database, which is a perfect fit for the unified serving/speed layer database.

With a highly optimized architecture for big data analytics, MapR provides world-record performance for MapReduce and an order-of-magnitude performance differential for NoSQL databases providing the best ROI for Hadoop deployments.

Online alerting: a quick example use case from a MapR customer

One MapR customer uses stream processing to capture in real-time the geo-spatial location of its truck fleet which directly goes into MapR.



Next, a Storm topology is used to detect and filter out trucks that are idle. Based on this detection, alerts such as text messages or mails are generated online. This allows the operator to optimize the overall utilization, minimizing operational costs and the environmental footprint.

7 http://www.mapr.com/products/m7

Conclusion

Big data stream processing is an increasingly common use case across several industries. MapR driven Enterpriser Data Hub provides a dramatically simplified architecture to process data streams on Hadoop that obviates the need for separate message queuing infrastructure. MapR reliability, performance and interoperability provide the fastest time to market, easiest maintenance and 24x7 uptime for stream-processing applications on Hadoop.