Holistic Approach to Big Data #5: Technical Details of key Big Data Components

1

In this video we will look at the technical details of some of the key BigData components.

We will look at Streams-type processing and text analytics in particular detail before moving on to look at some futures of Big Data and Cloud.

2

Business sentiment can be garnered from free-form text documents such as:

- Social Media
- Machine Log
- Call Center Logs
- Email
- Financial Services documents

3

An AQL extractor consists of a collection of views, for example Name, Title, and Organization. Out of these, some output views — for example, we may be interested in a composite NameTitleOrganizationView as the final view of interest for the business problem.

Dictionaries are used to hold titles and organizations. Personal names are recognized by their structure of FirstName and LastName with possible middle initials or prefixes/postfixes on the LastName.

The particular program in this case could be used to extract and match these three fields from a relatively free-form text document and provide the data for insertion into, for instance, a database table, or, at least, provide it in table format.

4

The diagram shows the AQL execution model details.

The picture shows the architecture diagram of a Text Analytics component. The example used here is based on the Annotation Query Language (AQL), which is modeled after the SQL query language.

The AQL developer writes the business logic in a .AQL file. The AQL program is declarative — that is, expresses what needs to be done rather than how to do it.

The AQL program is next compiled by the Text Analytics optimizer into an execution plan in the form of annotation operator graph file (with suffix .AOG). The optimizer is highly efficient and picks the best performing and optimized plan for execution. This .AOG file, or compile plan, is an executable that can be incorporated into a runtime environment to handle either streaming data (with InfoSphere Streams) or data-at-rest (with InfoSphere BigInsights).

The Text Analytics runtime component is a **document-at-a-time** execution model. The runtime component executes the .AOG plans against the input document and populates the output views with the extracted results.

5

Similarly we could run a text extractor to find sentiment. Here is analysis of comments on the movie Ra.One. You can notice that the adjectives that we picked up are an indication of positive, negative or neutral sentiments expressed by reviewers. Most of the reviewers have used negative adjectives which explain why the overall rating for the movie is low.

The center column, outlined in red, shows sentiment; the left- and right-columns show the context of the comments.

This context would have been used in the previous example dealing with Names, Titles, and Organizations to associate a name with a title and an organization based on proximity: within so many characters of the name.

6

This diagram shows how IBM InfoSphere Streams works [click-by-click (3 and 4 are timed—no click needed)]:

1. The overall application is composed of many individual operators (or, maybe, Processing Elements [PEs] consisting of one or more operators each); those are the bubbles in this graph. The data packets (sometimes called messages; we call them tuples) generated by one bubble travel to the next downstream bubble, and so on. The technical name for the bubbles in the diagram is Operator. Sometimes the output from one Operator becomes input to multiple Operator; conversely, sometimes multiple output streams merge as input to one Operator. But the tuples keep flowing.

   Note that we are doing continuous ingestion and continuous analysis on a stream of data, in the form of tuples.

*

2. Having all these separate processes working together gives us flexibility and scalability. If we need more power than a single computer can provide, you can simply distribute the Operators over multiple boxes (hosts).

   You can achieve scale:

   — By partitioning applications into software components

   — By distributing across stream-connected hardware hosts

*

3. Some (compute-intensive) Operators get their very own box; other Operators are best placed in the same box, because they must exchange a lot of data and going across the network would slow that down. The Streams runtime services can help distribute the load. In the picture, the streams flowing within a host (interprocess communication) turn rad, indicating that they are faster than the blue ones that go across the network between hosts.
4. Finally, we label some of the boxes with some of the processing stages typically found in many Streams solutions: Filter/sample to reduce the amount of data flowing through the system; transform to change the data format, compute new quantities, and drop unneeded attributes; correlate to join or otherwise combine data from multiple sources; classify is based on models generated from historical or training data; annotate is used to enrich the data by looking up static information based on keys present in the data.

7

IBM InfoSphere Streams is one of a class of Streaming-Data Processing Systems. It is the oldest and the most advanced, owing its origins to a large contract with the US Government Department of Defence. Development is based on System S software (“S” for Streams) that goes back to soon after 9/11. The first non-government customers were added in 2009.

The other streaming-data processing systems are much newer and often have very small development teams. Thus, for instance, Nathan Marz was the sole developer of Storm until its acquisition by Twitter.

Storm programming is done in Java. InfoSphere Streams programming is done in SPL (Streams Processing Language), a high level language that is compiled down to C++ and then run-time code is generated from the C++ derived code.

Some of these packages are now open source and will become important in the future. In the meantime, package features, user experience, and a strong customer base are important in selecting which software package to use.

8

The conceptual flow shown above is driven using the BigInsights app framework. The BigInsights app framework is leveraged by the Data Scientist to run the analytic components of the SDA.

The Social Data Analytics (SDA) accelerator provides code assist to facilitate analysis data in real-time for extracting insights (buzz, sentiment, intent) using sophisticated Text Extractors. In addition, social media analytics can also be used to build to build consumer profiles based on context accumulated over time. Realtime insights are derived by running analytics on the streaming engine. Realtime analytics,, as well as consumer profile information, can be made available on InfoSphere BigInsights, IBM’s Hadoop distribution, for more comprehensive analysis, as part of a comprehensive BigData platform.

9

Let’s look briefly as some of the futures for BigData processing and for Cloud processing.

10

An important area for future processing capability is SQL for Hadoop support improvements. This is part of the NoSQL movement (“Not Only SQL,” but perhaps better described as “Not Only Relational Databases” since SQL is going to be used, but the underlying data will be held in formats and containers other than a relational database management system [RDMS]). This is heading slowly towards full ANSI support.

The candidates here are

- Hive
- Impala (Cloudera)
- BigSQL (IBM)
- Stinger (Hortonworks)
- Drill (MapR)
- HAWQ (Pivotal)
- SQL-H (Teradata)

BigSQL from IBM is already available with full ANSI-92 SQL support, with a comprehensive optimizer, etc. Remember, IBM developed the original relational databases and the first implementations of SQL anyhow.

Other areas for the future are:

- Improvements in Multimedia Analytics
- Growth in usage and adoption of R programming language

IBM has just released BigR integrated with Streams and Hadoop processing.

For the Cloud we can expect:

- Bare metal support helping with Hadoop workloads
- Private network
- Full support with APIs

11

Here are some of the features of BigSQL from IBM.

BigSQL fully integrates with SQL applications and existing business intelligence (BI) / datawarehouse tooling with benefits that include:

- Existing queries run with no or few modifications (it is ANSI-92 compliant)
- Existing JDBC and ODBC compliant tools can be leveraged
- Applications do not have to compensate for constraints of Hive QL which otherwise require:
  - more statements
  - potentially moving more data over the network to the application

—-

You have now completed this video.