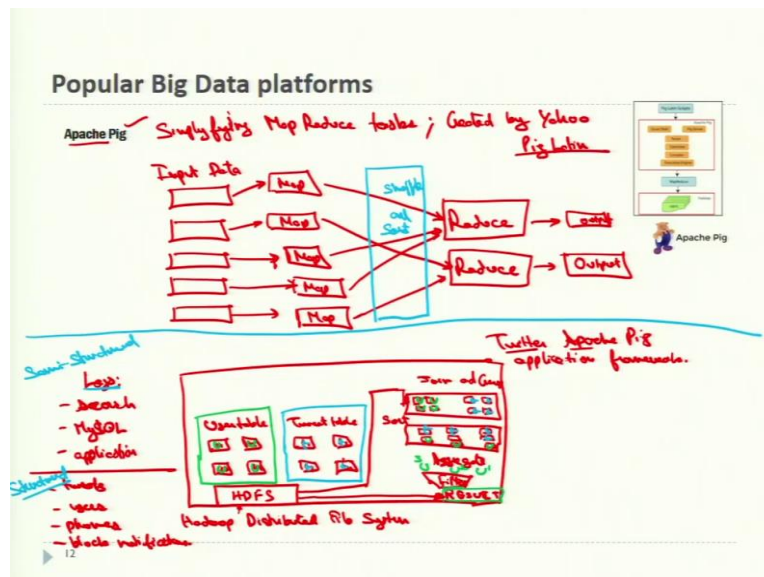


Computer Aided Decision Systems - Industrial Practices using Big Analytics
Professor Deepu Philip
Department of Industrial and Management Engineering
Indian Institute of Technology Kanpur
Professor Amandeep Singh
Imagineering Laboratory
Indian Institute of Technology Kanpur
Lecture 25
Big Data Analytics Tools and Software (Part 3 of 4)

Welcome back to the course on computer ADDC and support systems, big data analytics in practice. So, we were discussing the major platforms or the software technologies which are available to work on big data. We have discussed multiple technologies which are rising up which have matured, I will continue this.

(Refer Slide Time: 0:35)



Now, the next technology is 'Apache Pig'.

- ❖ The major purpose of the Apache Pig was to have an alternative method for simplifying the map reduce tasks. As a feedback from the internal people, I have taken as a suggestion that. Few examples I will keep on giving and also map reduce what are the few keywords which are maybe new for the people who are amateur to the big data that are also will try to explain. 'Map' reduce means we map the data first then try to reduce the data. It is nothing but when we say Extract Load Transform or Extract Transform Load. So, in that map is when we try to extract the data and try to map what is the data that we have extracted then we try to reduce it. Reduce means the data that is required is taken from there, that is when we try to transform and load it.

- ❖ So, map reduce is something like this, we have the input data in the various forms, it could be in an Oracle database, it could be a Cassandra, it could be any system that could be used any raw data in the textual form or in the structured form. We try to map the data to our requirements. Here, we have a map. Then we try to do a processing over it and try to reduce the data. So, we have crunchies of the data here. Maybe we might have connected a cluster analysis, maybe classification analysis in the metric data. If the data is in the interval or in the ratio scale, we might have conducted the factor analysis and we try to reduce crunches of these many packages of the data into two.

So, we try to reduce that if a few of them come here then it might have a connection here, this might have a connection here. We try to reduce the data. So, map reduce program first inputs the key as rows and sends the table information or the data element information to the mapper function, then the mapper function will identify the user ID and associative unit values. For instance, that value could be 0 or 1, for every user ID, the value would be different, then you try to shuffle it and try to sort some of the user IDs together and try to reduce function and then that reduce function will add all the number of the inputs or the data taken from the mapped stage together, which are belonging to the same user. So, this kind of map is reduced, examples are there. Then, we try to have an output of it.

- ❖ So, majorly to simplify the map reduce tasks, Yahoo created Apache Pig, (created by Yahoo) so that Hadoop reduce programs could be better used. So, Pig lets programmers use Pig Latin, which is a scripting language but this is a scripting language which is created for the Pig framework and executed by the Pig runtime. 'Pig Latin' is a language which is similar to SQL that the compiler converts in the background into a map reduce program. In order to process large amounts of data, it converts Pig Latin into a map reduce program, so major features of Apache Pig I would like to jot it down. So, before that let us take an example of how Twitter used Apache Pig for analysis and whatever the data sets and how do we group them.

So, what was the framework it decided? So, on Twitter, we had a user table that showed how many users there are, maybe User 1 will say, U1, U2, U3 and U4. We try to ingest the data into Hadoop Distributed File System (HDFS). So, then the user table is there then also we have a tweet table, that is whatever tweets which were made, tweet 1, tweet 2, tweet 3, tweet 4. Now, joining the data sets, grouping them, sorting them and retrieving the data to make it easy to understand or make it simple, this image represents or this framework represents how they try

to use the HDFS data while joining and grouping in which the users and the tweets were put and various groups were made out of them, these are users 4, 1, these are tweets. I will put tweets in a blue color, these are tweets, users in a green color, these were grouped and then sorted. And we have a user and a tweet, then we have two tweets and a user, then we might have two other tweets and a user. So, here these are users down here and up we have tweets.

So, what kinds of tweets a specific user is making that could be taken from here. So, Twitter had bought semi-structured data like Twitter Apache logs, Twitter search logs, Twitter MySQL query logs, application logs.

❖ So, this is a 'Twitter Apache Pig application framework'. So, in the logs which were there, it had

- 1- search logs,
- 2- MySQL,
- 3- application logs.

Then also structured data was also available which is the specific tweets that I am trying to put here,

- 1- tweets,
- 2- users,
- 3- phones
- 4- block notifications.

So, Twitter developed a system using the Apache Pig where they were able to better have a sentiment analysis of what the user is feeling while reading some tweet or when they are blocking someone or what specific tweets a user does when.

So, this help them to develop a machinery algorithm which they could use for the future design of their system design, of their interface, design of their back end of the software or the application that they were providing. So, there are two tables here, they majorly have one is a user table, another is tweet table, that means Twitter first had all these kinds of the data which I showed the semi structured data, the logs or so. This is semi structured here and we have structured. Here also in the map reduce that is given above, the reduce does not come directly, here also sorting happens, so, shuffling and sorting, that also happens in general. So, this is connected here from the HDFS database in which we have got the dumps of the archive data which has user data, tweets data, user data contains information about the user may be like

username the followers, the followings of the user, the number of the tweets that user has made then while the tweet data contains the tweets, the owner of the tweets, the number of the time some content is retweeted or number of likes for the tweet, this is all the tweet data. So, tweet data has all that information. So, this is all dumped into HDFS first.

Now, the data is extracted from HDFS and put into a processor, where it joins and groups. So, we are now trying to transform the data. Once we transform the data then the groups were made, sorting were made, then aggregate groups were made. Here, I would say aggregate data in which it says user 1, 2 tweets or 3 tweets, user 2, 3 tweets and so on. It filters and tries to do other Pig operations, that is we try to get a result. So, here we have user, user, user, maybe the number of tweets of the user 3 this is 2 this is 1 or so. Then, it is filtered further to add more pre-programs and we get a result which is a program that could be taken. This result could be taken directly from the HDFS system.

Now, it is coming through an 'Apache Pig' program, in which we join groups, sort, aggregate the data, try to filter the data, try to add more data from the previous or other Pig operations and we try to get an output of it.

Now, when we try to analyze this data. It could be, if I take an example, the user table might be having usernames maybe username maybe again maybe Amandeep, Deepu, Prabal, Ramchandra, Mukesh or so, usernames are there. Then the tweet table would have the types of the tweet that they have made: tweet 1 tweet 2 by Amandeep, tweet 1, 2, 3 by Deepu or so. So, these would come. Then, the number of times a tweet is retweeted, that could also come.

Then, the name and the count of the data, then, we try to make groups. What kind of similar to which one person is making regarding a specific topic, maybe this course is now floated Computer Aided Decision Support Systems. How many people are tweeting about this course, how many students in Operations Management, students from the IIT, students from the Management Sciences, students from the Computer Sciences, those groups could be made. So, then some of those groups how many tweets to specific users from this group is making.

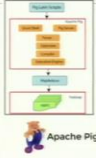
This kind of small processing could maybe let us have an output or let us have an inference that students from mathematics background with a little information of data science are more trying to tweet or more trying to like this program. So, these kinds of output could come. This is actually used by Twitter, so, the Twitter Apache Pig application framework.

(Refer Slide Time: 13:42)

Popular Big Data platforms

Apache Pig

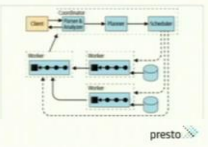
- Users can design their own functions.
- complicated use cases are handled best
- Can manage all type of data



Presto (Created by Facebook (300 PB); 2012)

Data from various sources could be combined

- Pure memory-based architecture
- Installing and debugging is simple
- Adaptable and straight-forward
- User-defined functions are required



13

Now, let us come to the features of the Pig program.

- ❖ In Apache Pig, the users can design their own functions. So, these functions are designed by their own that is a tailored system, which they use to carry out processing for a particular purpose. They can set a target, they can set a hypothesis and based upon that, they can define their own functions and try to use them.
- ❖ It works best for handling complicated use cases are handled best. When I say use cases, this means how a product or a software is being used at what is the reaction or what is the feedback of the people who are actually using the Twitter or any of the softwares. So, usage scenario of a piece of software often it is used in a plural to suggest situations where a piece of software may be useful. So, use cases is a usage scenario, in which how a piece of software (and it is generally used in use cases) and is generally used in plural to suggest a situation where a piece of software may be useful. Apache Pig is more used when use cases are complicated and it can try to manage all of them.
- ❖ So, it can manage all types of data. When I say all types of data that means a structured and unstructured data or semi structured data could also be handled.

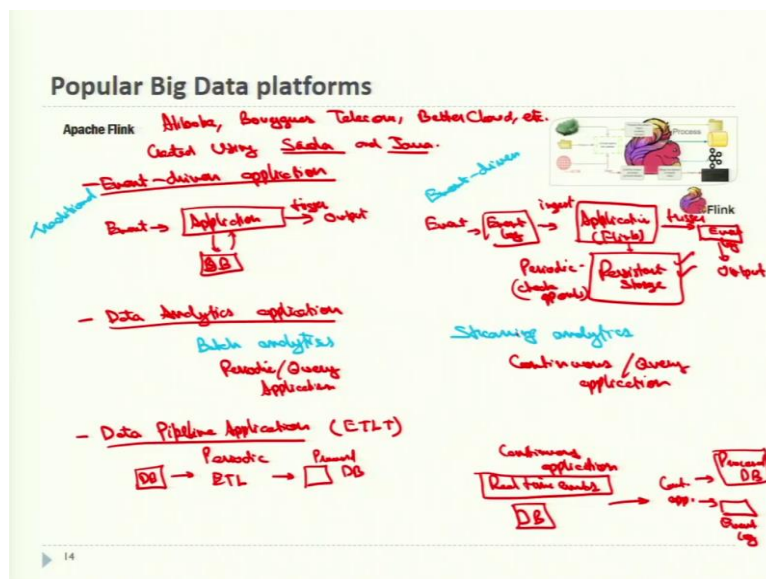
Next comes 'Presto'. Presto is a software that was created by Facebook. Presto query engine was created which is an SQL on Hadoop system to run interactive analytic queries against big amounts of data that means the petabytes of data. It enabled querying of data which was stored in relational databases, Cassandra, Hive or other proprietary data stores.

- ❖ The data from various sources could be combined in a single Presto query. So, Presto is independent of Hadoop map reduce techniques and has a very quick data retrieval time. The data retrieval time may be from a few microseconds to a few minutes only. So, it is very fast and the specific applications that Presto is being used are in the user defined functions wherever those are required. So, Facebook developed the Presto software.
- ❖ The major USP of the Presto platform is that, it can query the data wherever it is stored without the need of moving the data into a separate structured system, like that happens in a relational database or maybe a data warehouse. So, wherever the query execution runs, that runs in parallel to a very scalable pure memory-based architecture. So, it is a pure memory-based architecture.
- ❖ So, it was developed by Facebook when Facebook wanted to handle their 300 Petabytes of data. This warehouse where the data was structured as a Hadoop cluster implemented this in 2013. The data sources for the Presto as I just said could be catalog, a schema or finally a table so, it can come from the various systems, maybe from Cassandra, Hive or many other property softwares.

So, the major features that I would like to jot down here is

- ❖ Installing and debugging is simple.
- ❖ It is adaptable and straightforward.
- ❖ Here also user defined functions are required.

(Refer Slide Time: 18:54)



Next comes 'Apache Flink'. Flink is one of the preferred business Big Data platforms used by Alibaba, Bouygues Telecom or Better Cloud, etc. So, Flink was created using the programs Scala and Java. Flink or Apache Flink is an excellent choice to develop and run many different types of applications.

So, due to its great sets of features, the features include that it supports the stream or the batch processing both, sophisticated state management, event time processing semantics and maybe it can deal with

- ❖ Event-driven applications
- ❖ It can work on data analytics applications
- ❖ Data pipeline applications

Event driven means stateful application where we ingest the events from one or more event streams and we react to the incoming events by triggering computations or state updates or external actions. So, in general, the traditional applications are we have an event, we try to go to its application, then we have to run the application, what we try to do, we try to get the data or read or write the data from the general database, we have a tester database here, we have a database here and we try to read or write the data from here. Then, we trigger and try to have an output or the action but this is a traditional application. In event driven applications what do we have? We have events and we try to have an event log as well, from where we ingest the data into an application, which could be a Flink and we periodically write a synchronous checkpoints and persistent storage is there. This is ingested, then we try to trigger the data and again we have a modified event log and then we try to go to another application or output, that is a bigger data size could be handled.

So, instead of querying from the remote database when driven applications access the data locally wherever it is, which yields a better performance both in the terms of throughput or latency, periodic checkpoints could be made. So, these are persistent data storage so we have periodic checkpoints.

So, this could be for example maybe fraud detection, a normal detection in a database that we are getting. So, then business process monitoring how is the business process going on, how is the business process events. For example, in manufacturing if we try to say first, we procure the raw material, then we try to put the raw material into a machine, we try to transfer raw material from one place to another. What it is a time for procuring, what is the time for material

transfer, what is a time on actual machining on processing, then after the machining or processing, then we have data or the product storage, that is the final product, that is being stored, being stored and packaging for all of them the time of the event could be taken. For these events times go through the application in general. It tries to tell that overall throughput for the day would be this.

In a Flink based application, when we have the past data as well that how many numbers of privacies were produced, what was the times taken and these times that we are trying to talk about, what was the distribution this time following, maybe for the machining itself. Is it following a normal distribution of the times, if we plot for the last one year?

Then we are trying to keep on monitoring that when we have persistent storage here whatever the data is being developed, it goes through persistent storage here. In periodic checkpoints whether the program that we have made that event will take this time is being actually followed or not, these all things keep on going. So, event log output in an event data driven application is used.

So, the first one is a traditional one, and second is event driven. Data analytics application is the second use of the Flink Apache software. So, generally streaming analytics could also be taken. So, batch analytics and streaming analytics. So, in batch analytics, I could say periodic application is run or query, but in streaming analytics, we run continuous query replication. Here, the Flink could be used, when a continuous real time based or real-time fashion system is to be run. So, advantages of a continuous streaming analytics compared to a general batch analytics are not only limited to the latency or the time that it takes from the events. To insert into the elimination of the periodic import or query execution.

Streaming queries do not have to deal with artificial boundaries, that is, it has an open system where the sandbox could be expanded as and when required. So, the input data is taken at any time and continuous improvement, that is the periodic imports are not taken, continuous imports are taken.

So, another aspect is simple application architecture, because application architecture is more developed in the Apache Flink system. It can help us to have applications maybe in the quality monitoring of new networks or maybe in ad hoc analysis of the live consumer data, consumer technology or maybe product updates, mobile applications, how the people are behaving

towards different applications in the Android systems or so, that could be used in continuous streaming analytics.

Next comes 'Data Pipeline'. A data pipeline is a concept where we actually conduct the ETLT. In general, the ETL system where we extract, transform and load is a common approach, which converts and moves data between different storage systems. So, these are often periodically triggered to copy data from a transaction database system to maybe an analytic database system or to a data warehouse. Data pipelines serve a similar purpose of ETL jobs, but they transform and enrich the data which can move from one storage system to another and it can approach in a continuous streaming mode.

Again, I would say, data pipeline application in general we have periodic ETL. So, this is a continuous application, where in the periodic ETL we only have the transactional database which goes to the periodic application and it tries to only write the data and we get the file system or database that is the processed database. In continuous application, we have real time events and the database, from where we ingest the data to a continuous application and we try to develop.

- 1- Simple processed database.
- 2- You also try to have an event log.


So, the advantages of the pipeline system is that it reduces latency in moving the data to its destination. Data pipelines are more versatile and can be employed for more use cases because they are able to continuously consume and emit the data. So, how does Flink support the data pipelines many general data transformation or enrichments activities which are there that can be addressed by Flink's SQL interface or table API and it supports for user driven functions. Data pipelines with more advanced requirements can be realized using the data stream API, which is more generic.

(Refer Slide Time: 30:14)

Popular Big Data platforms

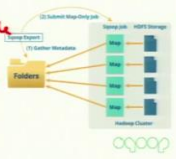
Apache Flink

- It provides a rich set of connectors.
- Different storage systems
- Low latency and high throughput



Apache Sqoop **RDBMS**

- Connectors to all major RDB are available
- Allows to load whole DB in a single command
- Incremental load load the updated parts of the data table.



15

- ❖ Flink provides a rich set of connectors, Connectors to what, different storage systems. Storage systems could be that we discussed Kafka, elastic search that we will discuss further, then some of the database systems.
- ❖ It also features continuous sources for file systems that monitor directories and things that write files in a time bound manner. So, real-time data search indexing buildings in e-commerce or continuous ETL in e-commerce could be major applications of Flink.
- ❖ Flink offers low latency and high throughput. Applications for the data driven data analytics data pipelines, all use Flink for streaming processing.

Next is 'Apache Sqoop', an Apache Sqoop is a top-level project that is open source. It is a tool which was developed for moving massive amounts of data between the structured data stores and Apache Hadoop relational databases like MySQL, Oracle and others. So, all structured data stores were taken as a base. Sqoop finds an application in importing the data from the relational databases into the HDFS or exporting the data from HDFS into the relational databases. So, both kinds of transactions could happen.

So, it extracts operational data from Relational Database Management System (RDBMS). So, it extracts and processes in Hadoop and optionally we can send the end result to a RDBMS once again. Let us say an example of a billing data which needs to be run every week. In this case, one can process the billing in a batch in a Hadoop, take the advantage of parallel processing and then send the summarized billing data to an RDBMS. So, this billing data could be processed using Apache Sqoop.

So, Sqoop helps in copying the data from RDBMS to Hadoop and vice versa. It sends the process data back to RDBMS. So, if I put the features of Sqoop, I would say:

- ❖ Connectors to all major relational databases are available.
- ❖ It allows us to load the entire database into a table or in a single command.
- ❖ There is a feature known as incremental load. This allows Apache Sqoop to load the updated portions of the data table. For instance, if the billing system is there, we have conducted each week. We have to summarize what are the bills that we have gotten. If the billing system, there are some regular buyers in the store which comes each weekend or which comes each evening, those billing systems number of pieces they have produced are also same, those would not be changed, only new systems of the new builds that have come that would only be changed. It would only highlight in the table that these are the new updated parts of the data table which are processed. So, this is usable in identifying. What are the changes? It is kind of like a track change like a system.

So, with this I will just have like to have a small pause here and we will be keep discussing other major software like Rapid Miner, KNIME, Elastic search and we cannot miss the biggest or the most sort of the software or the platform that is R and Microsoft can also be not be missed. So, Microsoft Azure will also be discussed in the next lecture. Let us meet in the next lecture. Thank you.