BIG DATA HANDLING

Software handlers shooting for the stars

Advanced course in web development

Tord Eliasson

Web programming 2014

Faculty of Computing
Blekinge Institute of Technology
SE-371 49 Karlskrona Sweden

Summary

This paper is written about Big Data. So what is that? Big Data is simply information or "Data" in much bigger form. So big form that it takes up to days to analyse it to actually get some insights from it. This Big Data concept was born in a paper released in 2001 but the need was starting much earlier. To be able to handle these loads of information, different Big Data tools emerged.

The questions I ask in this paper is based around my own questions after working with Apache Spark at Ericsson this summer. Since I felt I didn't learn very much about Big Data itself, I took this opportunity to answer my own questions. These were much based around big data as a subject, but also around how Spark kept up with other Big Data processing tools.

This led me to three different questions. History of Big Data, how does these Big Data processing tools solve the big data problem and how the relate each other and criticism against Big Data.

The processing tools I will look into in my comparison is:

- Apache Hadoop
- HPCC Systems
- Apache Spark
- Apache Storm

Table of content

Introduction

For this summer, I've been working at Ericsson. There I was working with a proof of concept with Apache Spark. Spark is a framework for handling big data and after working with it for the whole summer, I think it's been really interesting to work with and see it as a opportunity to dive into a more specific area of software engineering.

My first thought after a few weeks there getting into it, was I really should write my thesis about this, this is interesting. But as this course is before my thesis, it gives me chance to introduce the subject and also fill in the blanks I felt I missed when working at Ericsson.

Background

Internet, this gigantic connection of different machines. Since the creation as communication between computers on different universities in USA almost 40 years ago, it has grown exponentially. It hasn't just grown in size though. You could say that it has become broader. From working between only computers, almost everything can be connected to the internet today, and it keeps growing. Internet grows with around 40% per year, or almost doubles in size every second year. By 2020, there will be as many bytes on the internet as stars in the universe.

This makes it very easy for companies to gather information. For example this can be information about how users use their system. But this will create quite a lot of data. This is simply called Big Data. Analysis of these datasets can find new correlations to business trends, diseases and so on.

As a practical example of Big Data, the election of USA 2012. Obama started gathering data around 1,5 year before the election to be able to gather enough data for this.

“Chris Wegrzyn, director of data architecture for the Democratic National Committee, described the challenges, opportunities, and path to build the analytics-driven campaign. Wegrzyn noted that the key measurements centered on the data itself, modeling, and experimentation. The core data contained the facts about the electorate and the campaign operation. Modeling was used to understand the electorate at the individual voter level. Finally, evaluating the results of experiments helped the campaign learn how its actions actually influenced people.

Of course, the key performance indicator for the campaign was the number who planned to vote for Obama, divided by those who planned to vote overall. The campaign understood there were three levers to maximize that number: registration, persuasion, and turnout. They had to encourage their target audience of voters to register, persuade the undecided to vote for

Obama, then do all they could to ensure that Obama voters would show up to vote on Election Day."[10]

Earlier different departments used advanced analytics tools independently, to get data according to their approach. This election changed this, which makes it a lot easier to see correlation. This new approach proved really effective and Obama won this election.

Big data is often said to have "3V" characteristics.

- Volume
  The quantity of the stored data has a big role, if it's too small it is hard to actually get insight from the statistics it produces.
- Velocity
  Big data is often available in real time.
- Variety
  The type and variety of the data. This helps getting efficient getting resulting insight.

But big data is quite hard to handle, often not possible with regular means, so something called big data handling has been developed to handle and analyse these datasets, which I will look in to more with this paper.

Goals

After my summer at Ericsson my understanding of Spark was quite good. But at the same time my understanding of Big Data itself was almost non-existent. Of course I understand that this is the fastest way to get results, but since I've gotten interested with big data I thought I would use this paper to answer some of my questions about Big Data.

- History of Big Data
  To give insight into Big Data. This subject will also present the different Big Data processing tools for handling Big Data.

- How does these Big Data processing tools solve the big data problem and how do they compare to each other?
  How is Big Data handled? Is there any difference between how the different processors do it?

- Criticism against Big Data.
  Everything about Big Data can't be positive…

Methodology

Since my paper is written mostly around the internet, I have been using the internet to find the information. Mostly I've been using google first to find out information. To find out about history of big data, I simply searched "History of Big Data". There I found a great article about it, and my whole history bit is built on it.

For my comparison of processing handlers, I had to find out which processing handlers there was, before doing any research. I knew earlier about Hadoop and Spark, but since the Hadoop ecosystem, which Spark is included in, almost owns the whole market, I needed something to compare it to. At this point I wasn't really sure what to search for so it took a while to find what I actually was searching for. The search "Big Data processing tools" gave me some of the results I wanted, and after working through a few sites, I found a good site.

For the critic against big data, I checked Wikipedia for links to papers, which gave some interesting insights. Also I looked around for big data related failures and found out about the Google Flu Trends.

Results

The Big Data market has been growing rapidly since its specification in a paper in 2001. But to actually fully understand and get insight into Big Data, we have to look back further...

History

Ancient History of data

Around 2400 BC the abacus, or the counting frame, started to come into use in Babylon. Libraries are also starting to appear around this time.

While mentioning libraries, I feel I can't do it without mentioning the library of Alexandria. It was built in Egypt and had as purpose to collect all the world's knowledge, to show off the wealth of Egypt. This made it the intellectual centre of the Ancient world. The library kept somewhere between 400 000 to 700 000 papyrus scrolls, at most, which is quite a collection even today.

The Antikythera Mechanism were made sometime between 100 and 200 AD. Probably by Greek scientists. It is a very early analogue computer, made up of 30 different interlocking bronze gears. Its purpose was to keep track of different celestial bodies and also when the Olympic Games were held.

The appearance of statistics

John Graunt released in 1662 *Natural and Political Observations Made upon the Bills of Mortality*, where he used analysis of the mortality rolls of early modern London, to try to create a system to warn of the bubonic plague. His system was never finished, but this resulted in the first statistically based estimation of the population of London.

In 1880 the US census bureau is starting to experience troubles. They estimates that the 1880 census will take 8 years to crunch, and the next in 1890 will take over 10 years. Therefore making it outdated already when the data is collected next time. But in 1881 a young employee at the bureau, Herman Hollerith creates what will be known as Hollerith Tabulating Machine. Using punch cards he manages to take the census crunch from around 10 years, down to three months which gives him the title as father of the modern automated computation. The company he creates will later be known as IBM.

The early days of modern data storage

Fritz Pfleumer a German-Austrian engineer, invents a method to store data magnetically on tape in 1928. The same principles he used is still in use in the modern hard drives.

In one of the first attempts to speculate around how much data is created, Fremont Rider, a librarian releases a paper called The Scholar and the Future of the Research Library, in 1944. He observed that to contain all the popular and academic works, libraries had to double in size every 16th year. This led him to speculate around the size of Yale library, by 2040 will contain 200 million books spread over 6000 miles of shelves.

The beginnings of Business Intelligence

In 1958 IBM researcher Hans Peter Luhn defines the phrase Business Intelligence as “the ability to apprehend the interrelationships of presented facts in such a way as to guide action towards a desired goal.”

Also in 1962, System Development Corporation of California is first to use the term data-base, in a specific technical term. This marks the start of databases, as different databases soon emerged.

The beginning of big data centres

The US government is planning for the world’s first data centre in 1965. This to keep 742 million tax returns and 175 millions of fingerprints.

Unhappy with one of the earlier approaches on databases called CODASYL approach, Edgar Cobb, an IBM Mathematician, wrote in 1970 a number of papers which resulted in A Relational Model of Data for Large Shared Data Banks. There he wrote about how databases could work with “tables” instead of using linked lists as before. This approach differentiated quite much from the older ones, since with this approach, anyone who knew what they was searching for could find that. Earlier you often needed an expert. Many databases are still relational today.

The standard is set in 1974 for relational databases. It's called standardized query language or SQL. This to date, one of the biggest language for handling databases.
Material Requirement Planning (MRP) Systems are getting more common in the business world around 1975. This represent one of the first commercial uses of computers to speed up everyday process and make it more efficient.

The rise of the internet

In 1991 is the birth of, what will be known as, the internet. This is announced by computer scientist Tim Berners-Lee in a post in the Usenet group alt.hypertext. Here he sets specifications for a worldwide, interconnected web of data, accessible from all over the world.

According to R J T Morris and B J Truskowski in their book The Evolution of Storage Systems which released in 2003, 1996 was the year that digital storage became more cost effective than paper.

Michael Lesk publishes his paper How Much Information is there in the World? in 1997. In this he speculates around the size of the internet, and saying “12000 petabytes is perhaps not a unreasonable guess”. But he also speculates around the early development makes the web grow much faster than it will later on, since it grows 10-fold every year. He also points out, much of this data will never will be seen and give no insight.

In 1998 the NoSQL term is first used to name Carlo Strozzi's Strozzi NoSQL open-source relational database.

Early ideas around big data

Visually Exploring Gigabyte Datasets in Real Time is published by the Association for Computing Machinery in 1999. The paper highlights the tendency for storing large amounts of data without any way to adequately analysing it. Also the paper quotes computing pioneer Richard W Hamming with ”The purpose of computing is insight, not numbers”.

In 2001 Doug Laney, analyst at Gartner defines the commonly known characteristics of big data in his paper 3D Data Management: Controlling Data Volume, Velocity and Variety [2].

Google publishes Google File System paper [4] late 2003. This spawns another paper from Google, MapReduce: Simplified Data Processing on Large Clusters [6]. A web crawler project called Apache Nutch picks up these ideas to handle big data.

In 2006 Apache Nutch’s big data handling sub project spins off as its own project. It’s called Hadoop. An open source framework for storage and analysing of big data. Its flexibility makes it particularly useful for handling unstructured data, which we now generate and gather in loads of.

Today's use of Big Data is established

Wired brings the concept of Big Data to the masses in 2007 with their article “The End of Theory: The Data Deluge Makes the Scientific Model Obsolete”.[3]

NoSQL is brought up again 2009 in a last.fm event to discuss open source distributed, non-relational databases. The name was used to label non-relational databases, which started to increase. NoSQL databases today is often used together with big data since they often are faster and more effective then relational databases.

Eric Schmidt makes an announcement at a conference in 2010, that the same amount of data created between the beginnings of civilization until 2003, is created every two days.

HPCC Systems is open sourced, as a competitor to Hadoop. According to LexisNexis, their creator, they have been working on it since 1999.

Storm is released. It is a stream processing framework, made for Big Data. Created by Nathan Marz and his team on Backtype in 2011.

The rise of the mobile machine was dated in 2014. It meant that for the first time, people are using more mobile devices than office and home computers to access digital data.

In May 2014 was Apache Spark released. This after Matei Zaharia tried to solve machine learning problems using Hadoop. This led to him creating Spark instead together with Benjamin Hindman and other colleges.

Different Big Data processing tools

Now when we actually know a bit about Big Data, the discussion can be started. Here are four different Big Data Processing tools. We will make a deeper comparison in the conclusion, but this will give you an understanding of the specific programs.

Apache Hadoop

When talking about Big Data you can't start with anything else then talking about Hadoop. Hadoop is an open source framework for handling distributed storage and processing of big data on clusters, first released in 2006. Mostly Hadoop is written in Java but also some C and a little bit of shell script.

The Hadoop framework contains four different modules:

- Hadoop Common – Libraries and utilities used by the other modules
- Hadoop Distributed File System (HDFS) – A distributed file system that built on commodity hardware. Unlike other distributed file systems, it is very fault tolerant and designed using low cost hardware.
- Hadoop YARN - A platform made for managing computing resources in clusters.
- Hadoop MapReduce – An implementation of MapReduce model, for processing big data.

Over the years, different packages built around Hadoop. This have created a whole ecosystem around Hadoop and the name Hadoop has also come to represent this whole ecosystem.

Packages included are for example Apache Pig, Apache Hive, Apache Hbase and many more.

As stated earlier, Hadoop utilizes the programming model MapReduce. This model is built on two functions in functional programming, called Map and Reduce. Map performs filtering and sorting, while Reduce performs a summery operation. Hadoop gathers the data it should handle. Send it out to different parts of the cluster, the map function is a run on the cluster and then reduce operation is run to put the information back together. To work with Hadoop, you need to know Java.

HPCC Systems

HPCC or High Performance Computing Cluster is an open source data intensive system developed by LexisNexis Risk Solutions. It was released as an open source project in 2011, but according to LexisNexis, they had been working on it since 1999.

HPCC is built on two clusters:

- Thor – which handles the data refinery. It was given the name of the Norse god for with a large hammer crushing data into useful information.
- Roxie – handles data delivery using indexed files.

To handle these LexisNexis created a programming language called Enterprise Control Language or ECL. This is a high level programming language for parallel data processing.

The platform is built in an architecture implemented on commodity computing clusters [7], to give high performance and parallel processing to applications built for handling big data. Commodity computing can be described as the use of large numbers of already-available computing components for parallel computing, to get the greatest amount of useful computation at low cost.

In September 2011 HPCC systems made a test against Hadoop and Apache Pig. They took PigMix, which is a set of 17 programs meant to measure The Pig programming language against standard Hadoop MapReduce. Against Apache Pig, ECL scored an average of 4.45 times faster than its opponent and against Hadoop MapReduce it scored an average of 3.23 times faster.

Apache Spark

Apache Spark is fast and general open source engine for large scale data processing. It was built by Matei Zaharia and was built as a response to MapReduce's cluster limitations. It was created in 2009 and was open sourced in 2010. In 2013 Spark was donated to the Apache foundation and became a top level Apache project in February 2014. Spark is written in Scala.

Spark includes five different libraries.

- Spark Core – contains all basic functionality, including cluster handling and Resilient Distributed Dataset (RDD). RDDs represent immutable, partitioned collection of elements that can be run in a parallel. This is the basic functionality Spark is built upon.
- Spark SQL – contains the functionality to use SQL in Spark and also two other APIs for handling big data, in addition to RDDs. These APIs are called DataFrames and Datasets and are more effective than RDDs.
- Spark Streaming – enables stream processing of live data streams.
- Mlib – is a scalable Machine learning library.
- Graph X – handles Sparks API for graphs and graph-parallel computation.

So to explain how spark works, I will explain the RDD since the other models are working in the same way, just better. First, Spark is lazy, which means it doesn't run anything it absolutely has to. This is achieved by using something called transformations and actions. Actions are functions returning the data to the master node of the cluster, for example 'show()'. Transformations instead change the data, and is run on the cluster, but transformations always wait until an action is run to actually run, otherwise the information on what it is that should run is all that is saved. If you then stack many transformations upon each other, the cluster only computes what it absolutely has to when an action is run. Notable is also that Spark tries to keep everything in memory and not save data on file, which gives it even more speed.

When explaining the RDD, you can say it is an array, without any possibility to change it once you have created it. Instead you create new ones. This array contains references to partition objects on the cluster, where the computation is run, whenever an action is run.

Spark require a cluster manager and a distributed file system. For example Hadoop YARN and HDFS, but also many more. According to Spark’s website, they are around 100 times faster than Hadoop MapReduce while keeping the data in memory and ten times faster when working on disk. As a programmer you can choose language to write your Spark job in. Either Scala, Java, Python or R.

Apache Storm

Storm is a stream computation framework. Originally created by Nathan Marz and his team on Backtype in 2011, when acquired by twitter in 2014, it was open sourced. It is written mostly in Conjure, but also some Java.

Storm is built around what they call Sprouts and Bolts. A Sprout is a source of streams, while bolt consumes streams, and does processing on it. If the stream transformation is complex, the stream can be run through many bolts creating a directed acyclic graph. Storm can process over a million tuples per second, per node. So it is really fast.

Critique against Big Data

Big data have been a thing for around 15 years, counting from the specification of big data in 2001. Nothing can last that long without getting critique, which I will take a look on.

The big question with big data is of course the privacy question around all this data. Or as danah boyd and Kate Crawford describes it:

"On one hand, Big Data is seen as a powerful tool to address various societal ills, offering the potential of new insights into areas as diverse as cancer research, terrorism, and climate change. On the other, Big Data is seen as a troubling manifestation of Big Brother, enabling invasions of privacy, decreased civil freedoms, and increased state and corporate control." [8]

Secondly, we have the poster child for failure of Big Data. In 2008 Google started a project called Google Flu Trends. It was built around the Google search engine and around the idea that people search google for remedies and information when they feel sick. If it was anything relevant to flu they gathered the data. Off this data, Google thought that they could faster and better predict flu outbreaks, than the U.S. Centers for Disease Control and Prevention (CDC). Together with CDC's expertise around the disease, this created a massive opportunity.

Sadly Google Flu Trends had big problems to succeed over the years, and never had the accuracy of CDC's predictions. After failing to predict the 2013 flu outbreak, Google quietly scrapped the project.

Also according to Prof. Dr. Michael Berthold, "there is tendency for analyses to become a lot more shallow when more data is available". [9]

Conclusion and Discussion

Comparison between processing tools

So if we start with the different tools for handling Big Data, Hadoop was the first. For many people Hadoop is almost synonymous with Big Data and it has grown into its own world of Big Data handlers rather than being its own program. When I started working on this I found a problem. It actually took me some time to find a tool that at all didn't utilize any function of Hadoop. It's that big.

But on to the direct comparison.

	Hadoop	HPCC	Spark	Storm
Language	Java	ECL	Scala,Java,Python,R	Java
Functions	Write yourself	Pre-defined	Pre-defined	Pre-defined
Speed	1x	3.23x	100x/10x	1million tuple/sec
Type	Batch	Batch	Batch	Stream

Specification of what is compared.

- Language
  What programming languages the programmer can use to handle this tool.
- Functions
  How does the programmer handle the data, write yourself exactly how to, or is it pre-defined by the program, ready to be called by the programmer.
- Speed
  Comparison in Speed, with Hadoop MapReduce as base, if comparable.

Based solely on the information in this table, at least of the batch type tools, Spark looks like the winner.

But at the same time, we can see that Storm and the rest of the tools aren't comparable. While the other tools handle saved data, Storm can handle a stream of data, which hasn't even been saved yet. It can crush data into a format which the database can handle, delete unnecessary data before it's even saved and so on. So it works really well in other situations.

But you might think, "But Spark has a Streaming library, you said so in the results". But Sparks streaming library differs from Storm, it just uses smaller batches to give partial results faster, which gives Storm different usage.

Conclusion around critique

Gathered in the critique is three different examples of what can go wrong when working with Big Data. First we have the privacy aspect, is it OK for me to gather this data? Secondly, how will we gather this data? Will it yield results in the way we want? Thirdly, have we analysed this data enough? Since we already gathered it can we use it in another way?

Hopefully if companies gather data like this, carefully about privacy, the privacy challanges that is bound to come, isn't so bad.[12]

References

1. https://www.linkedin.com/pulse/brief-history-big-data-everyone-should-read-bernard-marr
2. http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf
3. The End of Theory: The Data Deluge Makes the Scientific Model Obsolete.
4. http://research.google.com/archive/gfs.html
5. http://www.datamation.com/data-center/slideshows/5-open-source-big-data-analysis-platforms-and-tools.html
6. http://static.googleusercontent.com/media/research.google.com/es/us/archive/mapreduce-osdi04.pdf
7. spacejournal.ohio.edu/pdf/Dorband.pdf
8. http://www.tandfonline.com/doi/full/10.1080/1369118X.2012.678878?scroll=top&needAccess=true
9. http://www.kdnuggets.com/2014/08/interview-michael-berthold-knime-research-big-data-privacy-part2.html
10. http://www.infoworld.com/article/2613587/big-data/the-real-story-of-how-big-data-analytics-helped-obama-win.html
11. http://cacm.acm.org/magazines/2016/6/202655-what-happens-when-big-data-blunders/fulltext
12. http://www.forbes.com/sites/bernardmarr/2016/03/15/17-predictions-about-the-future-of-big-data-everyone-should-read/#63324423157c

Also much information about how the different tools work, was found at their respective website:

http://hadoop.apache.org/

https://hpccsystems.com/

http://spark.apache.org/

http://storm.apache.org/