Big Data: Behind The Scenes

Kaustubh Vaghmare*

Persistent Systems Ltd., Pune, Maharashtra, India

Abstract

The phrase "big data" has become a highly common phrase in recent times. However the phrase 'big data', while it may have been coined to indeed vaguely refer to large volumes of data, has now evolved to mean something much more specific. The phrase 'big data' actually refers to a family of technologies, platforms, software and techniques aimed at solving a variety of problems otherwise untenable through conventional or traditional solutions. The present article summarizes a traditional approach to a data management problem and presents limitations of this approach. The three 'Vs' used to characterize and define 'big data' are then elucidated upon. Finally, the article summarizes commonly used tools and technologies to solve the 'big data' problem.

Keywords: *big data, databases, hadoop, mapreduce, spark, nosql, rdbms*

1. Introduction

The phrase 'big data' is often used very informally. It maybe that there was a time when this word indeed was a vague term used to refer to very large volumes of data being generated by modern telescopes (or the Internet / Internet of Things / Social media etc.) but today the phrase has evolved to mean something more specific. The author's personal favorite definition of the phrase 'big data' is *a phrase used to describe a characteristic(s) of data which makes the use of traditional solutions infeasible.*

However, a major problem with this definition of 'big data' is that it is not self contained. To complete the definition, one needs to spell out what is meant by 'traditional solutions' and also spell out under what circumstances or scenarios do those 'traditional solutions' become infeasible. It is the endeavour in the rest of the article to spell out both these aspects.

The article is organized as follows - in Section 2, I summarize what is meant by a traditional solution and explore the circumstances under which this solution becomes infeasible; in Section 3, I describe the hardware and software requirements of handling big data; in Section 4, I briefly summarize the ecosystem of commonly used software technologies available for tackling big data processing challenges; in Section 5, I briefly explore the concepts related to database technologies needed to solve big data storage and organization; and Section 6 offers a brief summary of the article.

2. Traditional Solutions

A traditional solution can be decomposed into hardware and software components. The typical / traditional hardware can be characterized by

- A single monolithic unit of hardware.
- Presence of single or multiple CPUs with multiple cores per CPU
- Storage in the form of a single disk or a collection of disks arranged in a RAID (Redundant Array of Inexpensive Disks) configuration.

*kaustubh.vaghmare@gmail.com

DOI: 10.52526/25792776-2020.67.2-264

- Primary memory in the form of a collection of RAM chips

And the software ecosystem comprises of

- A conventional Operation System
- With processing implemented using standard software or programming languages.
- Unstructured data handled using simple file storage, and
- Structured data handled using relational database management systems (RDBMS) (such as MySQL, Postgres etc.)

There are however three main conditions under which these traditional solutions become infeasible - these are the 3Vs often invoked to characterize big data!

- high volume,
- high velocity, and
- variety

However, it is important to emphasize a subtle point here. It is not enough that the total volume, variety or velocity of data is high - what is important is that volume, variety or velocity of *single coherent unit of data* should be high. For example, if the total data volume is 100+ terabytes, but it is composed of images with each *independent* image being $\sim 1MB$, then each image can be processed using a traditional solution. As a contrary example, if the total volume of the data is $\sim 500GB$, but *all of it needs to be processed in one go*, then despite lower total data volume, a traditional solution is not feasible.

Traditional solutions are 'vertically scalable'. If a traditional solution becomes infeasible, it is quite hard to upgrade the existing hardware. And replacement of existing hardware with a new hardware of higher specifications, is non trivial.

3. Big Data Hardware and Software

To tackle the situations where traditional solutions no longer work, a different approach to hardware and software is needed. In terms of hardware, we need a cluster of machines interconnected to each other. However, it is important to emphasize that not all clusters are suitable for big data processing. A conventional HPC (High Performance Computing) setup is composed of many independent machines that work as a single whole but rely on a network mounted storage for bulk of their data storage. This results in a large amount of data flowing across the network and is not desirable for big data processing.

Big data clusters on the other hand, are composed of machines where each machine has its own substantial self storage. And the tools that are used to orchestrate big data processing work in a manner so as to allow each machine to work on data already stored on it. In other words, the emphasis is on *data locality.* Thus the total data transferred across a network is minimized. Another important aspect of big data clusters is that they are made of *commodity hardware.* This is important because it allows easy expansion of the capacity of a cluster.

On the software side, there are two main components required for big data processing.

- A distributed file system which presents a unified logical view of the total storage across all machines. Since commodity hardware is used, such a filesystem has to guard against fault tolerance.
- A scheduler which coordinates the data processing across all computers, typically designed to minimize network data transfer.

Big data clusters are *horizontally scaleable.* The augmentation of the capacity can be done by just adding more machines to the cluster. The software is designed to seamlessly adjust to the changing capacity.

4. The Big Data Software Ecosystem

Arguably, the most famous big data software is *Hadoop*. Hadoop can be thought of an Operating System for Big Data. It primarily comprises of two parts

- The Hadoop Distributed Filesystem (HDFS)
- The YARN (Yet Another Resource Negotiator), a distributed computing scheduler.

In a typical Hadoop data processing job, one uses the *MapReduce* framework. Under this framework, all data processing jobs must be decomposed as a series of map and reduce functions, with the map function responsible for a state independent local data process and the reduce function responsible for assembling the final results of individual map functions. While the MapReduce framework is excellent for immense data volume processing jobs, it is limited by a) the lack of flexibility of the framework itself and b) not being amenable to interactive analyses.

Spark solves this problem through graph based *in-memory* analytics. Spark is currently a very popular solution providing rich APIs which allow a user to solve several problems related to interactive data processing.

5. Big Data Databases

The most popular databases are the *relational databases* which organize data in the form of related or linked tables. These databases conform to the ACID properties (Atomicity, Consistency, Isolation and Durability). Because of this, it is often very hard to operate these databases successfully in a distributed environment. Also, it is very difficult to use these databases in the face of a large variety because of the emphasis on a rigid structure or schema. This has led to a whole family of databases popularly called *NoSQL* (Not only SQL) databases. Popular examples include HBase, MongoDB, Redis, Cassandra etc. One of the reasons for the existence of so many different NoSQL databases is the CAP theorem.

The CAP theorem says that it not possible for any database to achieve all three properties of Consistency, Availability and Partition-Tolerance simultaneously. Here, consistency refers to the condition that all clients see the same data at any given time, availability refers to the condition where system continues to respond even in the failure of one of the nodes in a cluster, and partition-tolerance refers to the condition that the system continues to operate even though data partitions can no longer communicate with each other. A given NoSQL database takes a different approach to balancing these three properties.

6. Conclusions and Summary

Big Data is not a term to be thrown about casually. It specifically refers to an ecosystem of hardware and software solutions designed to handle data processing workloads not feasible using a traditional solution. Big data hardware is a cluster of machines but there is a difference between an ‘HPC’ and a big data cluster. Relational databases are generally not capable of high volume distributed data storage and this has led to the rise of multiple databases known as NoSQL databses.

Acknowledgements

I would like to acknowledge the BAO for the opportunity to present the contents of this article as an invited speaker. *I also declare that views presented in this article are my own and should not be construed as the views of my present organization viz. Persistent Systems Ltd.*