Module 7
7.1 Introduction to big data

By **Francisco Javier García Castellano**

Associate Professor at the Department of Computer Science and Artificial Intelligence (DECSAI), University of Granada.

---

1. WHAT IS BIG DATA?

'Big data', also referred to as 'massive data', is a buzzword which even appears in traditional mass media but is often used without proper knowledge of its true meaning. Big data refers to the problem that arises when it is impossible to work with a data set using the traditional data science tools described so far in this course. I like to use the example that a big data set may be so large that it cannot be saved onto a personal computer.

Of course, the data sets considered big data today may not still be considered so in 5 or 10 years' time. Thus, as computer capacities increase, we may even be able to work with big data sets on laptops. It will also depend on the means at our disposal. For me, a data set of about 300 GB is big data right now. However, in the cloud, we can hire a server with 96 processors with 384 GB of RAM and 3 TB of SSD storage. Consequently, someone who can hire such a server may not consider it a big data problem.

Although here we have focused on the size of the data, the problem caused by big data is not only related to storage; the problem is far more complex. We may be able to store data with no space issues, but it might take years to process it. Alternatively, the size of the data might double every couple of months, or it could be generated so fast that we may not even have time to store it.

The '3Vs' of big data, which allow us to better understand this concept, were first defined in 2001 and are as follows:

- Volume: The amount of data can be a problem, for instance, because it is difficult to store or because it grows exponentially.
- Velocity: We may not be able to process the data in an acceptable time or perhaps we cannot process it at the rate at which it is generated.
- Variety: The data can be of different types, such as text, images, or videos.

In addition, a fourth 'V' was added in 2013:

- Veracity: Uncertainty might be associated with the data because of inconsistent, incomplete, ambiguous, or erroneous data.

Indeed, some other authors use up to 10 Vs and even, the '42 Vs of big data' have been humorously defined.

2. BIG DATA AND BIOINFORMATICS

Omics sciences such as genomics, proteomics, transcriptomics, metabolomics, or epigenomics are quickly generating an enormous volume of data in a wide range of formats, especially text strings (e.g., nucleotides) and images. In fact, the bottlenecks laboratories currently face are related to data processing and analysis. Different technologies recently introduced to the omics sciences allow the analysis of the many molecules present in a single sample. For example, high-throughput genetic sequencing (next-generation sequencing) simultaneously sequences billions of DNA fragments, generating an enormous volume of data in a very short time. This technology has been used to sequence the genomes of diverse organisms including, in March 2020, 253 different genomes for the SARS-CoV-2 coronavirus responsible for COVID-19.

Sequencing of a single human genome generates about 180 GB of data and so studies involving several individuals can easily create in excess of a terabyte of information. Thus, the amount of data generated in more ambitious projects can be overwhelming; for example, in 2012, the 1,000 genomes project generated 260 terabytes of data. Moreover, estimates suggest that the data from high-throughput sequencing will treble every year. In other words, if we reached the first exabyte of genomic sequencing data in 2018, we may reach a zettabyte by 2024. Other estimates calculate that by 2025, we will 'only' have to deal with 40 exabytes of genetic sequencing data (equivalent to ≈ 40,000 petabytes or 40,000,000,000 terabytes). Thus, compared to other sources of big data such as YouTube videos or Twitter messages, genomics alone represents a huge big data problem, without even considering other omics sciences.

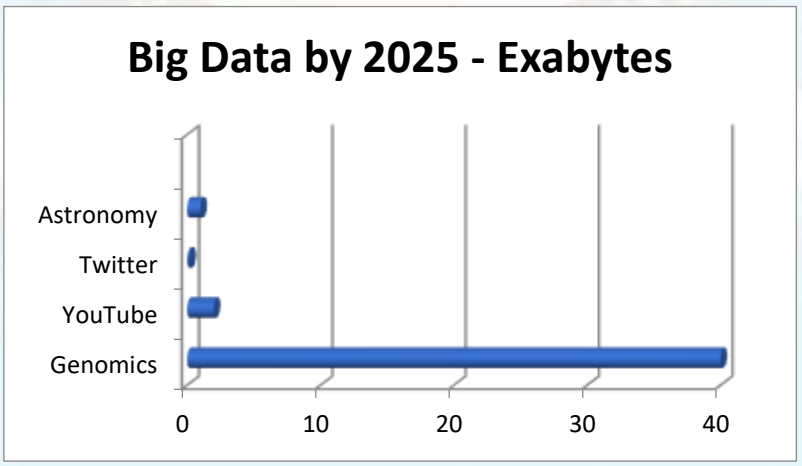

Figure 1. Estimated data storage requirements by 2025 for different areas of big data generation.

Of course, these huge amounts of data are usually generated worldwide but at the laboratory or research center level, a study involving genome sequencing of several individuals can easily generate several terabytes of information. If storing this data represents a problem, its analysis is another because it will not always be possible to use traditional data science techniques; big data techniques must be used instead.

3. BIG DATA TOOLS: APACHE *HADOOP* AND APACHE *SPARK*

Several big data tools are available that allow us, for example, to reliably process unbounded data streams; these include Apache *Storm* and NoSQL (Not only SQL) databases. Others are very scalable and widely distributed, such as Apache *Cassandra* or *MongoDB*. However, here we will focus on standard tools such as Apache *Hadoop* and the one that interests us most for data analysis, Apache *Spark*.

Apache *Hadoop* is very scalable and is the standard framework used to manage and/or process massive data. It allows us to store data using the *Hadoop Distributed File System* (*HDFS*) and also allows us to perform distributed data processing through its *Map-Reduce* methodology. *Hadoop* works in a distributed environment with several nodes in a cluster, in other words, as a group of interconnected computers. In this context, distributed means that the data storage and processing are performed among several computers over a network. *Hadoop* is highly scalable and so it adapts easily to increased workloads involving more nodes. It is also fault-tolerant, meaning that it will continue operating properly in the event of failure of one or more nodes.

Figure 2. The Apache *Hadoop* logo.

Hadoop is a broad collection of software utilities and so it is sometimes referred to as the *Hadoop* ecosystem. Among these utilities, the following modules are worth highlighting:

- *Hadoop Common*: Libraries used by the other *Hadoop* modules.
- *Hadoop Distributed File System* (*HDFS*): A distributed file system that allows us to physically store data in different nodes, but when we access them they appear to be on the local computer.
- *Hadoop YARN*: A distributed resource management and job scheduling utility that runs transparently on multiple networked machines.
- *Hadoop MapReduce*: A working methodology for parallel processing of large volumes of data.

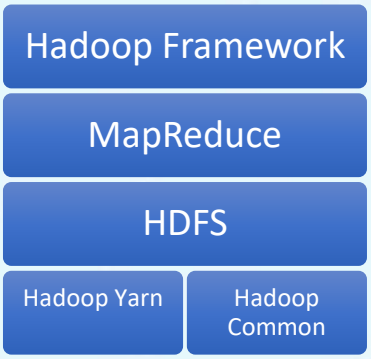

Figure 3. The Apache *Hadoop* framework.

Apache *Spark* is the successor of *Hadoop MapReduce* for data processing, although they are often used together. It is very common to use *HDFS* to store data and *Spark* to process it. Apache *Spark* runs applications up to 100× faster in memory and 10× faster on disk than *Hadoop*.

Figure 4. The Apache *Spark* logo.

Apache *Spark* comprises the following components:

- *Spark SQL*: This allows us to use SQL to access structured data.
- *Spark Streaming*: Allows us to work with data streams, making it easier to build scalable, fault-tolerant streaming applications.
- *GraphX*: Lets us perform parallel calculations using graphs.
- *MLlib*: (machine learning library): These are scalable machine learning libraries and data processing utilities and will be studied in greater depth in this course.

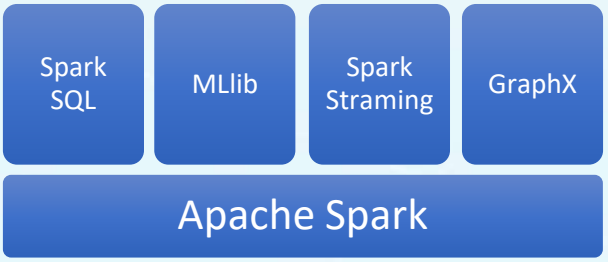

Figure 5. The Apache *Spark* stack.

Apache *Spark*'s increased speed over *Hadoop*'s *MapReduce* methodology is the result of two features:

- ***Directed Acyclic Graph (DAG)*:** This allows us to use a directed graph without cycles to control the data flow. In *Hadoop*'s *MapReduce* methodology, an acyclic directed graph is created with two pre-defined states ("Map" and "Reduce"). However, in *Spark*, we can create an acyclic directed graph with any number of states. When using multiple *MapReduce* stages in *Hadoop*, intermediate results between stages must be written to disk while *Spark* is more memory-based, which improves its performance.
- ***Resilient Distributed Dataset (RDD)***: This is an immutable distributed collection of fault-tolerant objects. Or to put it more clearly, it is the way *Spark* stores a dataset in the computer memory distributed among different nodes. These nodes perform 'lazy evaluations'. In other words, the different operations are not applied until there is no other choice. These operations are stored in an acyclic directed network and their execution is optimized because some operations are more expensive than others. For example, suppose we want to filter two combined tables; it is faster to filter the tables and then join them than to combine the tables and then filter them.

4. *MAP REDUCE* METHODOLOGY

Big data processing tools are usually based on the *MapReduce* methodology which was created by Google and initially implemented by Yahoo in the free software *Hadoop*. Its code was donated to the Apache Foundation which currently manages its development. *MapReduce* is a programming technique for parallel processing large amounts of data

in a group of computers. The processing is based on two functions: mapping ("Map") and reduction ("Reduce").

The mapping function receives key-value pairs and returns a list of pairs in a different domain. The reduction function joins pair sets with the same key. However, this process is somewhat more complicated than just two steps because we must split the data, map, shuffle/sort pairs with equal keys, reduce the data, and generate the output.

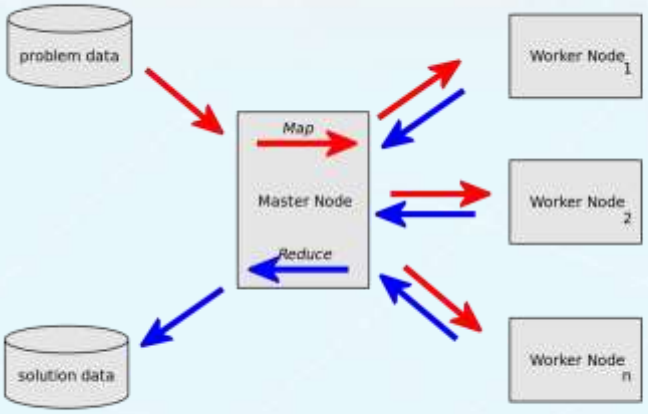

Figure 6. Work distribution when applying the *MapReduce* methodology.

Let's look at an example. Suppose we have genes that are either expressed (1) or not (0) and patients that may or may not have lymphoma. Also suppose that we want to know the probability that, if a given gene is activated, the patient will have lymphoma. In the mapping stage, we will retain the activated genes regardless of the presence of lymphoma. In the reduction, we will calculate the probability of each gene being linked to lymphoma.

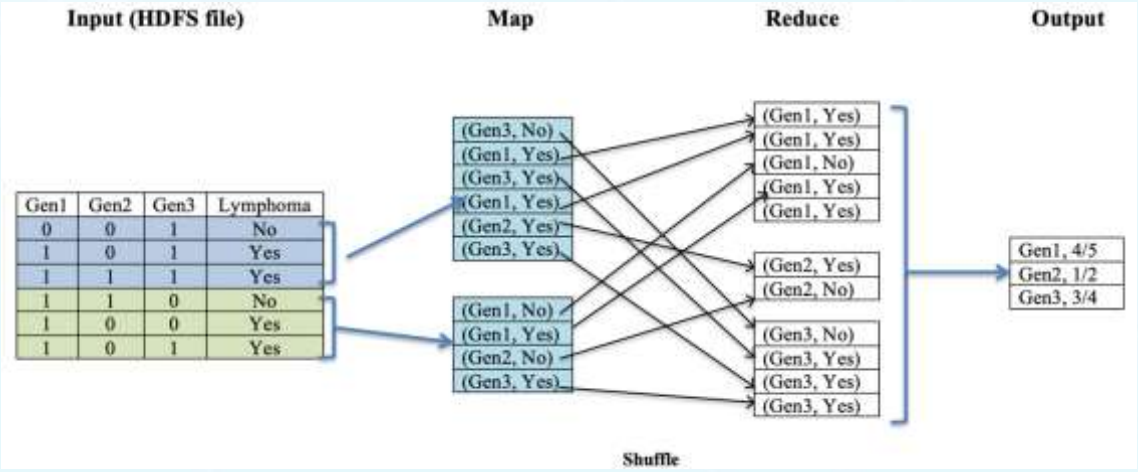

Figure 7. Example of calculations performed using the *MapReduce* methodology.

Not all problems can be addressed with *MapReduce*. However, if the algorithm can be parallelized with *MapReduce*, this methodology is highly scalable to process data on multiple nodes. Thus, *MapReduce* offers great performance in distributed and fault-

tolerant environments. The interesting thing about this technology is that to use it to process a big data problem, we need only to define the mapping and reduction functions. Node management, processing, error management, and other issues are handled by *Hadoop* or *Spark*. However, we need not worry about this, because here we will use the algorithms already programmed in the *MLlib* library.

BIBLIOGRAPHIC REFERENCES

- Laney, D. "3D Data Management: Controlling Data Volume, Velocity, and Variety". META group Inc., (2001). [Accessed 29 May 2020]. Available from: https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf .

- IBM. "The Four V's of Big Data". (2013). [Accessed 29 May 2020]. Available from: https://www.ibmbigdatahub.com/infographic/four-vs-big-data .

- Stephens ZD, Lee SY, Faghri F, Campbell RH, Zhai C, Efron MJ, et al. (2015) Big Data: Astronomical or Genomical?. PLoS Biol 13(7):e1002195. doi:10.1371/journal.pbio.1002195.

- Guo, R., Zhao, Y., Zou, Q., Fang, X. y Peng, S. (2018). Bioinformatics applications on Apache Spark. *GigaScience*, *7*(8), giy098.

- Apache Software Foundation. "Apache Hadoop Project" [Accessed 3 June 2020]. Available from: https://hadoop.apache.org/ .

- Apache Software Foundation. "Apache Spark™ - Unified Analytics Engine for Big Data" [Accessed 3 June 2020]. Available from: https://spark.apache.org/.

Additional bibliographic references

- Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., Mccauley, M., Franklin, M.J., Shenker, S. y Stoica, I. (2012). Fast and interactive analytics over Hadoop data with Spark. *Usenix Login*, *37*(4), 45-51.

- Tipos de instancias de Amazon EC2. (2020). [Accessed 28 May 2020]. Available from: https://aws.amazon.com/es/ec2/instance-types/

- Tom Shafer. "The 42 V's of Big Data and Data Science". (2017). [Accessed 29 May 2020]. Available from: https://www.kdnuggets.com/2017/04/42-vs-big-data-data-science.html .

- Forster, P., Forster, L., Renfrew, C. y Forster, M. (2020). Phylogenetic network analysis of SARS-CoV-2 genomes. *Proceedings of the National Academy of Sciences*, *117*(17), 9241-9243.

- Clarke, L., Zheng-Bradley, X., Smith, R., Kulesha, E., Xiao, C., Toneva, I., Vaughan, B., Preuss, D., Leinonen, R., Shumway, M., Sherry, S., Flicek, P. The 1000 Genomes Project Consortium (2012). The 1000 Genomes Project: data management and community access. *Nature methods*, *9*(5), 459-462.

- Mario Vega. "Procesamiento con MapReduce Part 1". (2017). MOOC Big Data. Universidad Politécnica de Madrid. [Accessed 8 June 2020]. Available from: https://www.youtube.com/watch?v=kYsaCCmuYIg

- Santos, P. "Apache Spark VS Hadoop Map Reduce". (2019). [Accessed 29 May 2020]. Available from: https://openwebinars.net/blog/apache-spark-vs-hadoop-map-reduce/