International Journal of Computer Trends and Technology
ISSN: 2231-2803 / https://doi.org/10.14445/22312803/IJCTT-V68I4P135

Volume 68 Issue 4, 233-237, April 2020

Review Article

A Functional View of Big Data Ecosystem

Alrawda Abdullatif Abdulhaleem Hamid

Lecturer, Department of Information Technology, Sudan University of Science and Technology
Khartoum, Sudan

Received Date: 18 March 2020
Revised Date: 30 April 2020
Accepted Date: 01 May 2020

Abstract - *Big data analytics is a promising research area, considering its capacity to add value in decision making for both business and academia. Massive numbers of tools available in the landscape of big data analytics solutions are provided for processing data in its lifecycle, namely, ingesting, analytics, storage and visualization. Large number of such solutions and sometimes interference among functionality of constituent components are stones in the road of implementing such solutions. In response to these complexities, this work grouped similar processing components in modules and showed interdependencies among them to facilitate synthesising big data analytics systems from extant solutions.*

Keywords *- Big data, ingestion, batch analytics, real-time analytics, interactive querying, visualization, noSQL database, distributed file system, in-memory, Apache Hadoop, HDFS, MapReduce.*

I. INTRODUCTION

The unprecedented exponential growth of data in the last few decades has been beneficial and challenging for businesses at the same time. Combining analytics results from both newly emerging data sources and traditional business data sources gives insightful sight supports decision making. However, processing and reporting results of big data analytics are challenging due to the nature of the so-called big data. Big data require special handling in order to address issues emerging from a heterogeneity of data sources and data, high rate of data generation and massive amounts of produced data. So, handling big data requires combining various processing paradigms, algorithms and tools in every single stage, starting from acquiring data from data sources and ending with presenting analytics results for the end-user. Big data applications involve business intelligence, information security, meteorology, astronomy, bioinformatics, and others [1], [2]. A thorough review of the big data ecosystem has been achieved through tracking data flow through the system and investigating system modules on the way, then grouping modules that collaborate to perform a broad task in one module. Each module, then, has been examined in terms of functionality, technologies and exemplary software solutions. The objective of this work is to optimize the process of engineering big data system that suits application or business requirements by using only needed submodules. The rest of this paper is organized as follow: section two introduces the general essentials of big data, section three is composed of four subsections; each investigates a processing module in the big data ecosystem, and section four draws conclusions from the review.

II. BIG DATA

The evolving web 2.0 applications and internet of things (IoT) was accompanied by the exponential growth of data from various data sources such as social media, sensors, mail servers and e-commerce transactions [3], [4] and others, leading to the emergence of gold mines of data with new formats and opens appetite of data scientists to analyse such data. Such data is attributed to big data because it differs from data generated from traditional data sources in terms of volume, velocity and variety. Big data is growing exponentially and streams almost infinitely [4], which require distributed, parallel and scalable storage and processing systems to cope with such massive, continually changing data. Massive volume and high velocity of data also trigger the need for adopting a real-time processing model to handle data in motion and using algorithms to speed up processing such as map-reduce and direct acyclic graph (DAG). Furthermore, a considerable portion of data produced from big data sources is unstructured [4], [6] or semi-structured and not compatible with relational databases; as a result, NoSQL databases turns dominant in the area of big data. The nature of big data and business need for analysing such data were motivators for the emergence of the big data analytics trend [6]. In this paper, the researcher thoroughly highlights vital modules involved in the big data analytics process by studying their functionality, implemented technologies and data flow among the system. Figure 1 depicts modules related to functional requirements, other modules

such as security and orchestration modules are out of the scope of this review.

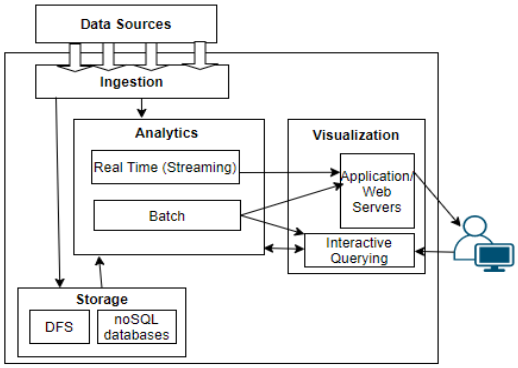

Fig. 1 Big data ecosystem modules

II. MODULES OF BIG DATA ECOSYSTEM

Big data is collected from various data sources such as data social media platforms, mail or web servers and sensors [3], [6], etc. Thereafter, it is fed into subsequent processing systems to eventually reflect analytics results to the end-user. A big data ecosystem could be viewed as a system of four interrelated modules; an ingestion module, a processing module, a storage module and a presentation module.

A. Ingestion Module

Data is collected (perhaps after crawling) from different data sources and passed through several stages before it is fed to the processing module. Typical ingestion engine is liable of data acquisition, decompression/ compression, extraction, filtering, conversion and integration [7], [9], [10]. Data connectors ask for authentication from data sources to acquire data [10]. Popular connectors could be i) database/ SQL connectors that allow connecting relational databases using application programming interfaces (APIs), vendors of DBMSs are providing such connectors, Sqoop is an example, ii) proprietary (or open-source) connectors, ii) custom connector [11] designed for particular data source through implementing APIs available by the data source. The well-known communication models adopted by connectors of real-time (stream) data sources are i) publish-subscribe messaging where interaction is taking place among a *subscriber* (consumer) subscribing to a *broker* who manages a number of topics (messaging queue) that receive messages from a *publisher* (data source) to subscribing consumers [11], [12], Apache Kafka and Amazon Kenisi are example frameworks implementing publish-subscribe messaging model [11] [12], ii) messaging queues connectors where producer pushes messages to message queues and consumer pulls them from these queues, these connectors fit the cases when the consumer pulls messages from publishers, RabbitMQ and Amazon SQS are implementing messaging queues model [11].

B. Analytics Module

Sub-modules that perform batch and real-time analytics are located in this module [8], [13], [14].

a) Batch analytics provides high throughput when processing massive data, but latency in performance could last for hours or days for completing one job [14]. Meanwhile, real-time processing is performed in applications where time matters and results are required in (near) real-time data production [15]. Map-reduce is an algorithm that allows writing programs able to partition (map) large data set among various processing units and process each individually [14], [16], [17], then combine (reduce) results of each processing step into a single result [13]. Apache's Hadoop MapReduce and Amazon's Elastic MapReduce (EMR) are example batch processing engines that are implementing map-reduce [14] for batch processing, directed acyclic graph (DAG) is another algorithm used for batch processing and implemented by Apache Spark [8].

b) Real-time analytics (also called stream) -in contrast to batch analytics, which has a start and end timings- requires timely, continuous processing of data in motion (stream data) [13], [14], [15]. Processed streams are moved to memories in cluster nodes before transforming them to disks [13], [15]. Apache Spark Streaming and Apache Storm are examples of real-time processing engines where the later is used for in-memory processing cases [8]. Interactive querying engines interacting with analytics module and having a user interface provided to facilitate querying a data set using queries of dedicated query languages [13] like Apache Spark SQL and HiveQL of Apache Spark and Hive, respectively [8], [18].

In general, all analytics operations serve two categories of analysis, particularly direct analysis and exploratory analysis, which requires a real-time response (analytics) [1], [2]. Direct analysis answers predefined questions through analytics techniques. On the other hand, exploratory analytics is required when there is no predefined question; in such cases, the analytics engine searches data to find interesting findings [1].

Data either flows from the analytics module to be visualised through the visualization module or may flow back to the analytics module for additional processing.

Data collected from data sources are stored in the storage module and forwarded to the analytics module.

C. Storage Module

Data collected from various data sources and analytics frameworks as final or temporary analytics results are stored and administered by this module, waiting for additional processing or visualization. Data stored either stored in a distributed file system

in the form of files with various file formats [10], retrieved through MapReduce jobs or stored in noSQL databases and retrieved via query languages of underlying noSQL DBMSs [13].

a) Distributed File System

A file management system for parallel processing of data in multiple nodes, such file systems are assumed to allow scalability, ability to store files of - typically – any size, and reliability, so that data availability is not affected by a node failure [19]. Hadoop distributed file system (HDFS) is a widely used choice in today's big data implementations. Input and output of map and reduce functions are read and written on the top of HDFS, HDFS is Hadoop's implementation of a distributed file system, other implementations of distributed file systems are IBM's GPFS-FPO Intel's Lustre[19].

NoSQL databases (also called stores) work as stores for temporary and final analysis results [20]. HDFS is managing file system in processing units of commodity servers (cluster) in data centres of the organization or those provided by technology giants in the form of the platform as a service (PaaS) [6], like in Amazon web services (AWS) cloud [21] and IBM's cloud [22].

b) Serving Databases

Non-relational database management systems (DBMSs) are a key storage component in the big data ecosystem [7] since they store data and analytics results for further tasks such as visualization [8]. NoSQL databases cope with the nature of big data and overcome shortcomings in relational databases [2], [23] in terms of providing requirements of databases that are distributed on a cluster(s) such as availability, scalability and fault tolerance [8], [13] and capability to handle non-structured and unstructured data. NoSQL databases are not following relational models [4]. Instead, they adopt new data models compatible with emerging data formats storage and processing needs [24]. It is worthy of mentioning that there is no standard query language for NoSQL databases since query languages are data model-dependent [25], [26]. Data models of NoSQL databases are key-value, document, graph, and column-oriented [13].

Key-Value databases store data items in tables [5] of two columns. Each item in such databases is a combination of a unique alphanumeric string key used for search operations, and a value contains data itself in the form of primitive data type or an object [24] [27], key and value relationship is specified by the programming language used to create the object; this dispenses the need for strict data model [24]. Amazon's Dynamo Riak are examples of key-value database management systems (DBMSs) [23], [27], and Redis is an in-memory DBMS [28].

Document databases are higher versions of key-value databases since they have the same data model with more complicated values [23]. Document databases use using key-value data model where the key is an alphanumeric string that could represent a path or a Uniform Resource Identifier (URI) [4], and value is a collection of semi-structured texts such as JavaScript Object Notation (JSON) and extensible markup language (XML), unstructured texts such as portable document format (PDF) and Word files documents in addition to Binary JSON (BSON) format which is used for storing images and videos and binary serializing JavaScript object notation (JSON) files, and therefore, improve processing performance [4]. In contrast to key-value databases, data in document databases could be queried either through key or value [5], [27]. MongoDB and CouchDB are examples of document DBMSs [23] and MongoDB runs partially in-memory [28].

Graph databases have been used to model graph-like data structures [29], [30], with highly interconnected data; therefore, it could be represented using graphs, particularly in the form nodes and edges where nodes represent entities and directed edges representing relationships among them, both nodes and edges have descriptive attributes [4], [25]. Although there are various mathematical graph models, property graphs are meant here. A property graph is a directed graph where both nodes and edges are labelled and can have any number of properties (attributes) and any number of edges between any two nodes. Properties represent metadata of edges or nodes in the form of key-value pairs [29]. Neo4j and Titan are examples of graph DBMSs [23], [30], and Trinity and Bitsy are running in-memory [28].

Column-oriented databases are in contrast to relational databases, where columns are defined on table level and are fixed for each row, columns in this data model are defined in row-level, this allows having various numbers of columns for various rows and adding columns whenever needed [5] which supports scalability when data is varied [15], HBase, Bigtable and Cassandra are example column-oriented DBMSs [8] where IM Column Store is running in-memory [31].

Data may need additional processing and flow back to the analytics module, or it may flow to the presentation module to be processed and presented in human-readable formats.

D. Presentation Layer

Traditional visualization systems are not fulfilling the requirements of big data visualization due to the need for dynamic visualisation [2], [32] in some use cases and the nature of big data. The task of visualizing big data is a challenging task, to overcome challenges like performance latency and massive volume of data, techniques such as parallel rendering, pre-fetching and caching relevant predicted data to speed up response time [2], [32], in addition to use of filtering, sampling and aggregation

techniques (such as clustering) to address issues of presenting massive data [2], [32].

This is the front end of the big data ecosystem and a vital component that adds value to decision-makers [3], [6], [30], [33], [18]. It allows presenting (batch, real-time) analytics results for end-user in visual form (static or dynamic) [15], [32]. This way analyst's eye could easily elicit meaningful information via relationships, trends, patterns [3], etc.

Furthermore, as mentioned in section B, it could provide an interface for user interaction through querying data set for getting analysis results via queries of dedicated query languages. The data set could be reprocessed for getting more accurate results [18]. Analytics results are visualized in traditional reports or dashboards or graphical forms [14], [3] that could be animated according to changes in data. Pygal and Seaborn are example visualization Python libraries [8], [33].

III. CONCLUSION

The field of big data pulls attention in both academia and business. Thanks to extant technologies and algorithms such as parallel processing, distributed processing, batch processing, real-time processing, noSQL databases, map-reduce, to name a few. Integrating such technologies with these used for data acquisition from sources of big data and visualizing analytics results, and choosing from the wide spectrum of available solutions in the software market requires theoretically underpinning such technologies in terms of the nature of data and processing needs.

In response to complexities attached with the development of customized solutions for big data analytics, the whole process of big data analytics had been studied, used technologies had been identified, and grouped into modules sharing the same broad objective. Based on this grouping, a graph shows interdependencies among modules have been designed. As shown in the previous sections, the whole system is heterogeneous, and modules themselves are heterogeneous in terms of used technologies in sub-modules. The modules of the system were dissected to illuminate data flow among system modules, used technologies and example solutions for different use cases. This way, with knowledge of the party's requirements, this work serves in the synthesis of big data ecosystem modules using existing technologies and tools. Modules related to non-functional requirements of the system have not been covered. Future research might extend the investigation of covered modules to embrace security and orchestration modules.

ACKNOWLEDGMENT

The author is thankful to Mr Izzeddin Elhassan, who did the proof heading for this document.

REFERENCES

[1] Mani M, Fei S. Effective Big Data Visualization. Proc. International Database Engineering & Applications Symposium'21(2017) 298.

[2] Bikakis N. Big Data Visualization Tools. arXiv preprint arXiv:1801.08336. (2018).

[3] Petrovska J, Ajdari J. Amazon's Role in the Field of Cloud Relational and NoSQL Databases: A Comparison Between Amazon Aurora and DynamoDB. Proc. ISCBE'03, 13(214) (2019).

[4] Venkatraman S, Fahd K, Kaspi S, Venkatraman R. SQL versus NoSQL Movement with Big Data Analytics, International Journal of Information Technology and Computer Science. 8(12) (2016) 59-66.

[5] Moniruzzaman, A., Hossain, S., NoSQL Database: New Era of Databases for Big Data Analytics Classification, Characteristics and Comparison, International Journal of Database Theory and Application, 6(4) (2013).

[6] Venkatraman R, Venkatraman S. Big Data Infrastructure, Data Visualisation and Challenges. Proc. International Conference on Big Data and Internet of Things'03, (2019) 13.

[7] Jagadish, V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J., Ramakrishnan, R., Shahabi, C., Big Data and Its Technical Challenges, Communications of the ACM, 57(7) (2014) 86-94.

[8] Bahga A, Madisetti V. Big Data Science & Analytics: A Hands-on Approach. VPT; (2016).

[9] Erraissi, A., Belangour, A., Tragha, A., Meta-Modeling of Data Sources and Ingestion Big Data Layers, Proc. International Conference of Smart Applications and Data Analysis for Smart Cities'02, paper 10.2139 (2018).

[10] Semlali BE, El Amrani C, Ortiz G. SAT-ETL-Integrator: an Extract-Transform-Load Software for Satellite Big Data Ingestion. Journal of Applied Remote Sensing, 14(1) (2020).

[11] Bahga, A., Madisetti, V.K., Madisetti, R.K. and Dugenske, A. Software-Defined Things in Manufacturing Networks. Journal of Software Engineering and Applications, 9 (2016) 425-438.

[12] Ta VD, Liu CM, Nkabinde GW. Big Data Stream Computing In Healthcare Real-Time Analytics. Proc. IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), (2016) 37.

[13] Lipic T, Skala K, Afgan E. Deciphering Big Data Stacks: an Overview of Big Data Tools. Proc. International Workshop on Big Data Analytics: Challenges, and Opportunities, 14 (2014).

[14] Ta-Shma P, Akbar A, Gerson-Golan G, Hadash G, Carrez F, Moessner K. An Ingestion And Analytics Architecture For Iot Applied To Smart City Use Cases. IEEE Internet of Things Journal, 5(2) (2017) 765-74.

[15] Hurwitz, J., Nugent, A., Halper, F., and Kaufman, M., Big Data for Dummies, John Wiley & Sons, Inc., New Jersey, USA, (2013).

[16] Stolpe M. The Internet of Things: Opportunities and Challenges for Distributed Data Analysis. ACM SIGKDD Explorations Newsletter, 18 (1) (2016) 15-34.

[17] Merla P, Liang Y. Data Analysis Using Hadoop MapReduce Environment. Proc. IEEE International Conference on Big Data (Big Data), (2017) 4783.

[18] Cho W, Lim Y, Lee H, Varma MK, Lee M, Choi E. Big Data Analysis with Interactive Visualization Using R Packages. Proc. International Conference on Big Data Science and Computing, (2014) 1.

[19] Mazumder S, Dhar S. Hadoop Ecosystem As Enterprise Big Data Platform: Perspectives and Practices. International Journal of Information Technology and Management, 17(4) (2018) 334-48.

[20] Ranjan R. Streaming Big Data Processing In Datacenter Clouds. IEEE Cloud Computing, 1(1) (2014) 78-83.

[21] The Amazon AWS website. [Online]. Available: aws.amazon.com

[22] The IBM website. [Online]. Available: https://www.ibm.com/cloud/blog/implementing-big-data-platform-cloud

[23] Gupta A, Tyagi S, Panwar N, Sachdeva S, Saxena U. NoSQL Databases: Critical Analysis and Comparison. Proc. International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN), (2017) 293.
[24] Seeger M, Ultra-Large-Sites S. Key-Value Stores: a Practical Overview. Computer Science and Media, Stuttgart, (2009) 21.
[25] Kaur K, Rani R. Modeling and Querying Data in NoSQL Databases. Proc. IEEE International Conference on Big Data, (2013) 1.
[26] Phiri, H., Kunda, D., A Comparative Study of NoSQL and Relational Database, Zambia Information Communication Technology (ICT) Journal, 1(1) (2017).
[27] Bhuvan, N., Elayidom, M., A Technical Insight on the New Generation Databases: NoSQL, International Journal of Computer Application, 121(7) (2015).
[28] Zhang H, Chen G, Ooi BC, Tan KL, Zhang M. In-memory Big Data Management and Processing: A Survey. IEEE Transactions on Knowledge and Data Engineering. Vol. 27(7) (2015) 1920-48.
[29] Angles R, Gutierrez C. An Introduction to Graph Data Management. Graph Data Management, (2018) 1-32.
[30] Angles R. The Property Graph Database Model. Proc. AMW, (2018).
[31] The oracle-base website. [Online]. Available: https://oracle-base.com/articles/12c/in-memory-column-store-12cr1
[32] Agrawal R, Kadadi A, Dai X, Andres F. Challenges and Opportunities with Big Data Visualization. Proc. International Conference on Management of Computational and Collective Intelligence in Digital Ecosystems,7 (2015) 169.
[33] Caldarola EG, Rinaldi AM. Big Data Visualization Tools: A Survey. Proc. International Conference on Data Science, Technology and Applications, 6 (2017) 296.
[34] Lu, J., Data Analytics Research-Informed Teaching in a Digital Technologies Curriculum, Informs Transactions on Education, 20(2) (2020) 57–72.