How to Store and Process Big Data: Are Today's Databases Sufficient?

Jaroslav Pokorný

Department of Software Engineering, Faculty of Mathematics and Physics
Charles University, Prague, Czech Republic
pokorny@ksi.mff.cuni.cz

Abstract. The development and extensive use of highly distributed and scalable systems to process Big Data is widely considered. New data management architectures, e.g. distributed file systems and NoSQL databases, are used in this context. On the other hand, features of Big Data like their complexity and data analytics demands indicate that these tools solve Big Data problems only partially. A development of so called NewSQL databases is highly relevant and even special category of Big Data Management Systems is considered. In this work we will shortly discuss these trends and evaluate some current approaches to Big Data management and processing, identify the current challenges, and suggest possible research directions.

Keywords: Big Data, NoSQL, NewSQL, Hadoop, SQL-on-Hadoop, ACID.

1 Introduction

Without doubts Big Data is a fashionable topic used by many people in different contexts and without precise semantics. In interview with Roberto V. Zicari[1], Jochen L. Leidner from R&D at Thompson Reuters says that buzzwords like "Big Data" do not by themselves solve any problem – they are not magic bullets. He offers some advice: to solve any problem, look at the input data, specify the desired output data, and think hard about whether and how you can compute the desired result – nothing but "good old" computer science. An excellent confirmation of this idea is offered in the book [8] describing methods, algorithms including their complexities, in context of very large amounts of data. Using today's other buzzwords the book provides a guide how to program so called Big Analytics.

Obviously, any efficient data processing system requires not only effective algorithms but also tools for storing and processing large datasets. We can use:

- traditional relational parallel database systems,
- distributed file systems and Hadoop technologies,

[1] `http://www.odbms.org/blog/2013/11/big-data-analytics-at-thomson-reuters-interview-with-jochen-l-leidner/` (accessed 20 July 2014)

- NoSQL databases,
- new database architectures (e.g., Big Data Management Systems, NewSQL databases, NoSQL databases with ACID transactions).

The paper focuses on issues and challenges coming with use of these tools in context of processing Big Data. Its goal is also to show some alternatives in this area.

In Section 2 we mention shortly some basics concerning Big Data and Big Analytics. Section 3 presents an overview of technologies, platforms, and tools for Big Data storage and management. Finally, Section 4 summarizes the paper, tries to answer the question in its title, and offers some challenges.

2 Big Data and Big Analytics

There are many different (pseudo)definitions of Big Data. An attempt to compare them is offered in [12]. Usually we talk about the Big Data when the dataset size is beyond the ability of the current database tools to collect, process, retrieve, manage, and analyze the dataset.

Big Data are most often characterized by several *V*'s which also pose problems for their storage and processing: *Volume*, *Velocity*, *Variety*, and *Veracity*. We can distinguish three areas concerning data management and processing of Big Data:

- storage and low-level file processing with simple database features,
- more sophisticated database processing with high-level query languages,
- so called Big Analytics working with big amounts of transaction data as extension of methods used usually in technology of data warehouses (DW).

What is Big Data analytics? Big Analytics is the process of examining large amounts of different data types, or Big Data, to uncover hidden patterns, unknown correlations and other useful information. In a Web context Big Analytics ca be charactrized in this way [1]: The crucial difference between Web search and Web data mining is that in the first case we know what we are looking for, while in the second we try to find something unusual that will be the answer to a (yet) unknown question.

The above rather simplified and vague Big Data characterization does not consider explicitly *complexity* occurring, e.g., in graph datasets and heterogeneous environments used for Big Analytics. Complexity together with big volume mostly requires a scalability of computer systems and algorithms used. In fact, just some well-known algorithms used to analyze data in DW may not scale and some techniques of their parallelization and distribution are needed. Big Analytics requires not only new database architectures but also new approaches to methods for data analysis. The latter means either reformulation of old data mining methods, or their new implementation, or even a development of completely new methods. A recent overview of Big Data issues is presented in [7].

3 Big Data Storage and Management

For storage and processing Big Data two features are preferred: scalability and high-speed access to massive volumes of information. We present some today's options in terms of their suitability for Big Analytics.

3.1 Relational DBMSs

Traditional relational DBMS both centralized or distributed is based on usage of SQL language and transactional properties guaranteeing ACID properties. ACID stands for atomicity, consistency, isolation and durability and is fundamental to database transaction processing. A significant part of the relational database technology usable for Big Data is called *Massively Parallel Analytic Databases* (MPAD). Unlike traditional DW, these DBMSs are capable of quickly proceed large amounts of mainly structured data with minimal data modeling required and can scale-out to accommodate multiple terabytes and sometimes petabytes of data. Interactive query capabilities are possible in MPAD. Possibilities of near real-time results to complex SQL queries are also at disposal.

3.2 Distributed File Systems and Hadoop Technologies

Distributed file systems considered here distinguish from traditional network file systems (e.g., in UNIX). They use, e.g., file partitioning and replications. The most famous is *Hadoop Distributed File System* (HDFS) [11].

Hadoop is a batch processing system based on the framework MapReduce [5] and HDFS. On the analytics side, MapReduce (M/R) emerged as the platform for all analytics needs of the enterprise. Because Hadoop clusters can scale to petabytes and even exabytes of data, enterprises no longer must rely on sample data sets but can process and analyze all relevant data.

Hadoop is the main part of well-known software architecture called *Hadoop stack*. A special attention belongs to the SQL-like language HiveQL. It is originally a part of infrastructure (DW application) Hive[2], which the first *SQL-on-Hadoop* solution is providing an SQL-like interface with the underlying MapReduce.

On the other hand, MapReduce is still very simple technique compared to those used in the area of distributed databases. Users require more complex, multi-stage applications (e.g. iterative graph algorithms and machine learning) and more interactive ad-hoc queries. Then faster data sharing across parallel jobs is needed.

Recently, there are other software tools providing a lot machine learning methods. They include, e.g., the engine Spark[3], distributed in-memory parallel computing framework. Spark is targeted for iterative and interactive algorithms, where Hadoop does not perform well. Its authors claim that it runs programs up to 100× faster than Hadoop MapReduce in memory or 10× faster on disk.

[2] `http://hive.apache.org/` (accessed 20 July 2014)
[3] `https://spark.apache.org/` (accessed 20 July 2014)

A lot of NoSQL databases are built on top of the Hadoop core, i.e. their performance depends on M/R jobs. For example, Big Analytics requires often iteration that is hardly achieved in NoSQL based on MapReduce. Consequently, some modifications like HaLoop framework [3] occur now which support iteration.

3.3 NoSQL Databases

NoSQL databases are a relatively new type of databases which were initiated by Web companies in early 2000s. Although some popular lists of them include all non-relational datastores such as XML databases, etc., mostly various key-value stores and graph databases represent this category now (see, e.g., [9, 6]). Beside their typical features like simplified data model, rather query driven database design, no support of integrity constraints, no standard query language, especially weakening ACID semantics is most relevant in context of Big Data processing.

NoSQL provide little or no support for OLTP as it is required for most enterprise applications. CAP theorem [5] has shown that a distributed computer system can only choose at most two out of three properties: *C*onsistency, *A*vailability and tolerance to *P*artitions. Then, considering *P* in a network, NoSQL databases support *A* or *C*. In practice, mostly *A* is preferred and the strict consistency is mostly relaxed to so-called *eventual consistency*. Eventual consistency guarantees only that, given a sufficient period of time during which there are no writes, all previous writes will propagate through the system so that all replicated copies of the data will be consistent. Eventual consistency has been widely adopted, becoming something of a default for NoSQL databases. However, there are some examples, where the consistency is tunable (e.g., in Cassandra[4]) or configurable (e.g., *CP* or *AP* in Oracle NoSQL database).

3.4 Big Data Management Systems

Some of NoSQL databases are a part of a more complex software architectures, e.g. Hadoop stack, or even so called *Big Data Management Systems* (BDMS). Unlike relational databases, where the user sees only SQL in the outermost DBMS layer for manipulating data, these systems allow to access the data through various means at different layers. Often the Hadoop stack is presented as the first generation of BDMS. ASTERIX [13, 2] uses different technologies than Hadoop stack: a special Hyracks data platform, an algebraic level Algebricks, HiveQL, but also a compatibility with Hadoop MapReduce. Typically, BDMS use a lot of other special high-level manipulations languages. Oracle sees BDMS as an architecture that seamlessly incorporates Hadoop, NoSQL and relational DW.

BDMS are often equipped by advance methods for data analytics. For example, Myria[5] is appropriate for actual statistical analysis. Some methods supporting similarity, preference, and uncertainty in data processing are becoming a part of basic algorithms used directly in DBMSs, e.g., fuzzy joins in ASTERIX. Today's BDMSs are evolving rather to completely remove the MapReduce layer.

[4] `http://cassandra.apache.org/` (accessed 20 July 2014)
[5] `http://myria.cs.washington.edu//` (accessed 20 July 2014)

3.5 NewSQL Databases

In last years the development of data management shows that there are applications that want strong consistency. NewSQL is a subcategory of RDBMSs preserving SQL language and ACID properties, and moreover, the performance and scalability issues posed by traditional OLTP RDBMs. For example ClustrixDB[6], F1 [10], VoltDB[7], and MemSQL[8] belong here. These DBMSs achieve high performance and scalability by offering architectural redesigns that take better advantage of modern hardware platforms such as shared-nothing clusters of many-core machines with large or non-volatile in-memory storage. They allow real-time analytics and transaction processing at the same time.

3.6 NoSQL with ACID Transactions

Also many NoSQL designers are exploring a return to transactions with ACID properties as the preferred means of managing concurrency for a broad range of applications. Some observations show that CAP concerns only a part of the design space for distribution and scalability. When network partitions are rare it is not necessary to decide firmly between *C* and *A*, because it is usual adopting eventual consistency. Google's Spanner [4] belongs to this category. It is not a pure relational system implemented on top of a key-value store. Further, e.g., the FoundationDB Key-Value Store[9] is a distributed database with true ACID transactions, scalability, and fault tolerance. Such data stores offer also the SQL layer. Both examples also show that the data storage technology is decoupled from its data model.

4 Challenges and Conclusions

We can observe that there is a lot of possibilities how to store and process Big Data. Their use depends strongly on application, the data volume the application will access, the complexity of mining algorithms used, etc. A new feature is, that some of these systems have more different components that enable access and process data stored in various ways. Obviously, it requires an additional optimization and more complex decision making in the selection of the individual components.

A rather difficult challenge concerns role Big Analytics. So far, the mining process was controlled by the analyst or the data scientist. Dependent on the application scenario he/she determines the portion of data where/from the useful patterns can be extracted. A better approach would be the automatic mining process and to extract approximate, synthetic information on both the structure and the contents of large datasets. This seems to be the biggest challenge in Big Data.

[6] `http://www.clustrix.com/` (accessed 20 July 2014)
[7] `http://voltdb.com/` (accessed 20 July 2014)
[8] `http://www.memsql.com/` (accessed 20 July 2014)
[9] `https://foundationdb.com/` (accessed 20 July 2014)

Acknowledgments. This paper was supported by Czech Science Foundation (the grant No. P103/13/08195S).

References

1. Baeza-Yates, R.: Big Data or Right Data? In: Proc. of the 7th Alberto Mendelzon Int. Workshop on Foundations of Data Management, Puebla/Cholula, Mexico, May 21-23. CEUR-WS.org (2013)
2. Behm, A., Borkar, V.R., Carey, R.M., Grover, J., et al.: ASTERIX: Towards a Scalable, Semistructured Data Platform for Evolving-world Models. Distributed and Parallel Databases 29(3), 185–216 (2011)
3. Bu, Y., Howe, Y., Balazinska, M., Ernstm, M.D.: The HaLoop approach to large-scale iterative data analysis. The VLDB Journal 21(2), 169–190 (2012)
4. Corbett, J.C., Dean, J.C., Epstein, M., et al.: Spanner: Google's Globally-Distributed Database. In: Proc. of 10th USENIX Symposium on Operation Systems Design and Implementation (OSDI 2012), pp. 261–264 (2012)
5. Dean, D., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clus-ters. Communications the ACM 51(1), 107–113 (2008)
6. Hecht, R., Jablonski, S.: NoSQL evaluation: A use case oriented survey. In: CSC 2011 Proceedings of the 2011 International Conference on Cloud and Service Computing, pp. 336–341 (2011)
7. Kelly, J.: Big Data: Hadoop, Business Analytics and Beyond, Wikibon (2014), `http://wikibon.org/wiki/v/Big_Data:_Hadoop,_Business_Analytics_and_Beyond` (accessed July 20, 2014)
8. Leskovec, J., Rajaman, A., Ullman, J.D.: Mining of Massive Datasets. Cambridge University Press, Cambridge (2011)
9. Pokorny, J.: NoSQL Databases: a step to databases scalability in Web environ-ment. International Journal of Web Information Systems 9(1), 69–82 (2013)
10. Shute, J., Vingralek, R., Samwel, B., et al.: F1: A Distributed SQL Database That Scales. PVLDB 6(11), 1068–1079 (2013)
11. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The Hadoop Distributed File System. In: Proceedings of MSST2010, pp. 1–10. IEEE Press (2010)
12. Stuart, J., Barker, A.: Undefined By Data: A Survey of Big Data Definitions. CoRR, arXiv:1309.5821 (2013)
13. Vinayak, R., Borkar, V., Carey, M.-J., Li, C.: Big data platforms: what's next? ACM Cross Road 1, 44–49 (2012)