EVALUATIONS OF BIG DATA PROCESSING

Duygu Sinanc Terzi[1] , Umut Demirezen[2] , and Seref Sagiroglu[1]
[1]Department of Computer Engineering Gazi University, Ankara, Turkey
[2]STM Defense Technologies Engineering and Trade Inc., Ankara, Turkey
duygusinanc@gazi.edu.tr , udemirezen@stm.com , ss@gazi.edu.tr

Abstract

Big data phenomenon is a concept for large, heterogeneous and complex data sets and having many challenges in storing, preparing, analyzing and visualizing as well as techniques and technologies for making better decision and services. Uncovered hidden patterns, unknown or unpredicted relations and secret correlations are achieved via big data analytics. This might help companies and organizations to have new ideas, get richer and deeper insights, broaden their horizons, get advantages over their competitors, etc. To make big data analytics easy and efficient, a lot of big data techniques and technologies have been developed. In this article, the chronological development of batch, real-time and hybrid technologies, their advantages and disadvantages have been reviewed. A number of criticism have been focused on available processing techniques and technologies. This paper will be a roadmap for researchers who work on big data analytics.

Keywords: big data, processing, technique, technology, tools, evaluations

---

1. INTRODUCTION

The size of data starts from giga to zetta bytes and beyond. According to Fortune1000 Companies, 10% of increase in data provides $65.7 million extra income (McCafferty, 2014). Big data flows too fast, requires too many new techniques, technologies, approaches and handles with the difficulties it brings. Big data is generated from online and offline processes, logs, transactions, click streams, emails, social network interactions, videos, audios, images, posts, books, photos, search queries, health records, science data, sensors, and mobile phones including their applications and traffics. They are stored in databases or clouds and the size of them continues to grow massively. As a result, it becomes difficult to capture, store, share, analyze and visualize the data with typical tools. Big data concepts have a combination of techniques and technologies that help experts, managers, directors, investors, companies and institutions to gain deeper insights into their information assets and also to abstract new ideas, ways, approaches, values, perceptions from the analyzed data (Dumbill, 2012). To enable an efficient decision making practice, organizations need effective processes to turn high volumes of fast-moving and diverse data into meaningful outcomes. The value of big data market is 10,2 billion dollars now, and it is expected to reach 53.4 billion dollars by 2017 (McCafferty, 2014). Organizations and institutions might get benefits from big data analysis for their future developments, investments, decisions, challenges, and directions with descriptive, predictive, and prescriptive analytics like decision support systems, personalized systems, user behavior analysis, market analysis, location-based services, social analysis, healthcare systems and scientific researches.

To clarify and express the big data features, the five Vs of volume, variety, velocity, veracity and value (Dumbill, 2012), (Demchenko, Grosso, De Laat, & Membrey, 2013), (M. Chen, Mao, & Liu, 2014) are frequently used to explain or understand the nature of big data (Fig. 1). Volume is the size of data produced or generated. It is huge and its size might be in terabytes, petabytes, exabytes or more. The volume is important to distinguish the big data from others. Variety has different forms of data, covers the complexity of big data and imposes new requirements in terms of analysts, technologies and tools. Big data is connected with variety of sources in three types: structured, semi structured and unstructured. Velocity is important not only for big

data but also for all processes. The speed of generating or processing big data is crucial for further steps to meet the demands and requirements. Veracity deals with consistency and trustworthy of big data. Recent statistics have shown that 1 of 3 decision makers do not trust the information gathered from big data because of their inaccuracy (Center, 2012). Accordingly, collected or analyzed big data should be in trusted origin, protected from unauthorized access and normal format, even if this is hard to achieve. Value is the most important feature of big data and provides outputs for the demands by business requirements. Accessing and analyzing big data is very important, but it is useless if no value is derived from this process. Values should be in different forms such as having statistical reports, realizing a trend that was invisible, finding cost saving resolutions, detecting improvements or considering new thoughts for better solutions or achievements.

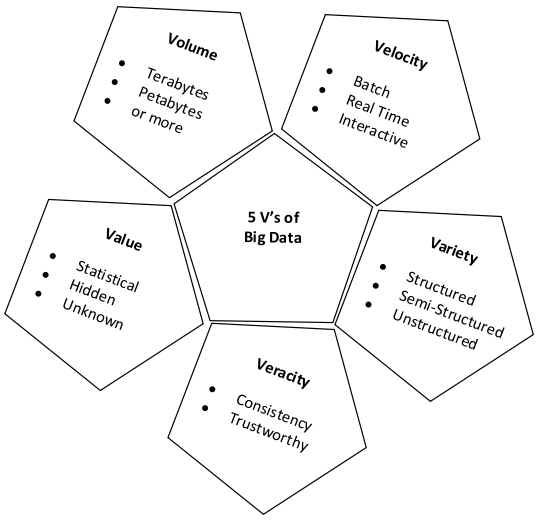

Figure 1. 5 V's of big data

Working with big data is a complex process with conceptual and technical challenges. This causes the existence of a high number of different approaches. In this paper, an overview on big data's concepts are summarized and techniques, technologies, tools and platforms for big data are generally reviewed in Section 1 and 2. In Section 3, the chronological development of big data processing is reviewed according to the technologies they cover. Finally, discussion and conclusion are outlined in Section 4.

2. TECHNIQUES AND TECHNOLOGIES FOR BIG DATA

Big data is a way of understanding not only the nature of data but also the relationships among data. Identifying characteristics of the data is helpful in defining its patterns. Key characteristics for big data are grouped into ten classes (Hashem et al., 2015), (Mysore & Jain, 2013), (Assunção, Calheiros, Bianchi, Netto, & Buyya, 2015) (Fig. 2).

To enable efficient decision-making, organizations need effective processes to turn high volumes of fast moving and diverse data into meaningful outcomes. Big data analytics helps boost digital economy, and provide opportunities via supporting or replacing decision-making processes with automated algorithms. In addition to that, it helps reducing the cost and predicting behaviors of groups, teams, supporters, enemies or habits from enough features of available data. Data management of big data involves processes and supporting technologies to acquire, store and prepare data, while analytics refers to techniques used in analyzing and extracting intelligence from big data (Gandomi & Haider, 2015).

The techniques for big data analytics consist of multiple disciplines including mathematics, statistics, data mining, pattern recognition, machine learning, signal processing, simulation, natural language processing, time series analysis, social network analysis, crowdsourcing, optimization methods, and visualization approaches (M. Chen et al., 2014). Big data analytics need new techniques to process huge amount of data in an efficient time manner and way to have better decisions and values.

The technologies for big data processing paradigms are chronologically transformed as batch processing, real-time processing and hybrid computation because of the big data evolution (Casado & Younas, 2015). Batch processing is a solution for volume issue, real-time processing deals with velocity issue and hybrid computation is suitable for the both issues. The techniques and technologies developed in this context are summarized in Table 1 as platforms, databases and tools (Wang & Chen, 2013), (Cattell, 2011), (Bajpeyee, Sinha, & Kumar, 2015). These tables should be helpful to companies, institutions or applicants to understand and provide new ideas, deep insights, perceptions and knowledge after analyzing big data.

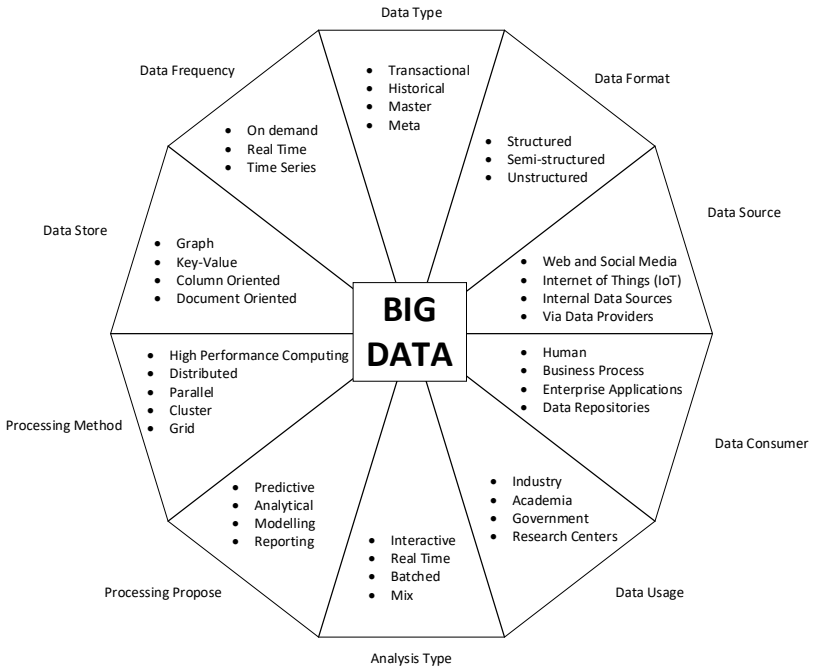

Figure 2. Big data classification

PLATFORM TYPE	TOOLS
LOCAL	Hadoop, Spark, MapR, Cloudera, Hortonworks, InfoSphere, IBM BigInsights, Asterix
CLOUD	AWS EMR, Google Compute Engine, Microsoft Azure, Pure System, LexisNexis HPCC Systems

DATABASE TYPE		TOOLS
SQL		Greenplum, Aster Data, Vertica, SpliceMachine
NOSQL	**Column**	HBase, HadoopDB, Cassandra, Hypertable, BigTable, PNUTS, Cloudera, MonetDB, Accumulo, BangDB
	Key-value	Redis, Flare, Sclaris, MemcacheDB, Hypertable, Valdemort, Hibari, Riak, BerkeleyDB, DynamoDB, Tokyo Cabinet, HamsterDB
	Document	SimpleDB, RavenDB, ArangoDB MongoDB, Terrastore, CouchDB, Solr, Apache Jackrabbit, BaseX, OrientDB, FatDB, DjonDB
	Graph	Neo4J, InfoGrid, Infinite Graph, OpenLink, FlockDB, Meronymy, AllegroGraph, WhiteDB, TITAN, Trinity
IN-MEMORY		SAP HANA

TOOL FUNCTIONS	TOOLS
DATA PROCESSING	MapReduce, Dryad, YARN, Storm, S4, BigQuery, Pig, Impala, Hive, Flink, Spark, Samza, Heron
DATA WAREHOUSE	Hive, HadoopDB, Hadapt
DATA AGGREGATION & TRANSFER	Sqoop, Flume, Chukwa, Kafka, ActiveMQ
SEARCH	Lucene, Solr, ElasticSearch
QUERY LANGUAGE	Pig Latin, HiveQL, DryadLINQ, MRQL, SCOPE, ECL, Impala
STATISTICS & MACHINE LEARNING	Mahout, Weka, R, SAS, SPSS, Pyhton, Pig, RapidMiner, Orange, BigML, Skytree, SAMOA, Spark MLLib, H2O,
BUSINESS INTELLIGENCE	Talend, Jaspersoft, Pentaho, KNIME
VISUALIZATION	Google Charts, Fusion Charts, Tableau Software, QlikView

SOCIAL MEDIA	Radian6, Clarabridge

Table 1. Big data tools in different perspectives

3. BIG DATA PROCESSING

3.1 BATCH PROCESSING

Big data batch processing was started with Google File System which is a distributed file system and MapReduce programming framework for distributed computing (Casado & Younas, 2015). MapReduce splits a complex problem into sub-problems implemented by Map and Reduce steps. Complex big data problems are solved in parallel ways then combined the solution of original problem.

Apache Hadoop is well-known big data platform consisting of Hadoop kernel, MapReduce and HDFS (Hadoop Distributed File System) besides a number of related projects, including Cassandra, Hive, HBase, Mahout, Pig and so on (C. P. Chen & Zhang, 2014). The framework aims for distributed storage and processing of big data sets in clusters (P. Almeida, 2015). Microsoft Dryad is another programming model for implementing parallel and distributed programs that can scale up capability. Dryad executes operations on the vertexes in clusters and use channels for transmission of data. Dryad is not only more complex and powerful than Map/Reduce and the relational algebra but also support any amount of input and output data unlike MapReduce (M. Chen et al., 2014). HPCC (High Performance Computing Cluster) Systems are distributed data intensive open source computing platform and provide big data workflow management services. Unlike Hadoop, HPCC's data model defined by user. The key to complex problems can be stated easily with high level ECL (Enterprise Control Language) basis. HPCC ensure that ECL is executed at the maximum elapsed time and nodes are processed in parallel. Furthermore, HPCC Platform does not require third party tools like GreenPlum, Oozie, Cassandra, RDBMS, etc. ("Why HPCC Systems is a superior alternative to Hadoop,").

3.2 REAL-TIME PROCESSING

Big Data Applications based on write once - analyze multiple times data management architectures are unable to scale for real-time data operations. After some years from using MapReduce, big data analytic applications shifted to use Stream Processing paradigm (Tatbul, 2010). Hadoop-based programming models and frameworks are unable to offer the combination of latency and throughput requirements for real-time applications in industries such as real-time analytics, Internet of Things, fraud detection, system monitoring and cybersecurity. Stream processing programming model mainly depends on the freshness of the data in motion. When any type of data is generated at its source, processing the data from travelling from its source to its destination is very challenging and also effective way. Potential of this approach is very important because eliminating the latency for gaining value from the data has outstanding advantages. The data is analyzed to obtain results at once. The data travels from its source to destination as continuous or discrete streams and this time velocity property of the Big Data has to be handled with this approach.

Using stream processing techniques in Big Data requires special frameworks to analyze and obtain value from the data. Because the data stream is fast and has a gigantic volume, solely a small part of the stream can be stored in bounded memory. In contrast to the batch data processing model where data is first stored, indexed and then processed by queries, stream processing gets the inbound data while it is in motion, as it streams through its target. Stream processing also connects to external data sources, providing applications to integrate selected data into the application flow, or to update a destination system with processed information.

Despite not supporting stream processing, MapReduce can partially handle streams using micro-batching technique. The idea is to process the stream as a sequence of small data chunks. The incoming g stream is grouped to form a chunk of data and is sent to batch processing system to be processed in short intervals. Some MapReduce implementations especially real-time ones like Spark Streaming (Zaharia, Chowdhury, Franklin, Shenker, & Stoica, 2010) support this technique. However, this technique is not adequate for demands of a low-latency stream processing system. In addition, the MapReduce model is not suitable for stream processing.

The streaming processing paradigm is used for real time applications, generally at the second or even

millisecond level. Typical open source stream processing frameworks Samza (Feng, Zhuang, Pan, & Ramachandra, 2015), Storm (Toshniwal et al., 2014), S4 (Neumeyer, Robbins, Nair, & Kesari, 2010), and Flink (Renner, Thamsen, & Kao, 2015). They all are low-latency, distributed, scalable, fault-tolerant and also provide simple APIs to abstract the complexity of the underlying implementations. Their main approach is to process data streams through parallel tasks distributed across a computing cluster machines with fail-over capabilities.

There are three general categories of delivery patterns for stream processing. These categories are: at-most-once, at-least-once and exactly-once. At most once delivery means that for each message handed to the next processing unit in a topology that message is delivered zero or one time, so there is a probability that messages may be lost during delivery. At least once delivery means that for each message handed to a processing unit in a topology potentially multiple endeavors are made for delivery, such that at least one time this operation is succeeded. Messages may be sent multiple times and duplication may occur but messages are not lost. Exactly once delivery means that for each message handed to a processing unit in a topology exactly once delivery is made to the receiver unit, so it prevents message lost and duplication.

Another important point for stream processing frameworks is state management operations. There are different known strategies to store state. Spark Streaming writes state information into a storage. Samza uses an embedded key-value store. State management has to be handled either implementing at application level separately or using a higher-level abstraction, which is called Trident in Apache Storm. As for Flink, state management based on consistent global snapshots inspired Chandy-Lamport algorithm (Chandy & Lamport, 1985). It provides low runtime overhead and stateful exactly-once semantics. Because of the latency requirements for an application, a stream processing frameworks have to be chosen carefully depending on the application domain. Storm and Samza support sub-second latency with at least once delivery semantics while Spark Streaming supports second(s)-level latency with exactly one delivery semantics depending on the batch size. In addition to this, Flink supports sub-second latency with at exactly once delivery semantics and check-pointing based fault tolerance. If large-scale state management is more important, Samza may be used for that type of application. Storm can be used as a micro-batch processing by using its Trident abstraction and in this case, the framework supports medium-level latency with exactly one delivery semantics.

Scalable Message Oriented Middleware (MOM) plays very important role in distributed and stream processing application development. The integration of different data sources and databases is critical for a successful stream processing. MOM is used to help building scalable distributed stream processing applications across multiple platforms, gathering data from different sources, creating a seamless integration. There are different types of commercial and open source MOM's available in this area. Every MOM has its own unique advantages and disadvantages based on its architecture and programming model. In a simple manner, MOM delivers messages from a sender source to a receiving target. It uses queue-based techniques for sending/receiving messages; for instance, a sender application that needs to deliver a message will put the message in a queue. After that, MOM system gets the message from the queue and sends it to the particular target queue.

One of the most well-known messaging system is Apache Kafka (Kreps, Narkhede, & Rao, 2011). Kafka is a distributed publish-subscribe messaging system and a fast, scalable, distributed in nature by its design. It also supports partitioned and replicated commit log service. A stream of particular type of messages is defined by a topic in Kafka system. A producer client can publish messages to a topic and so the published messages are stored at a cluster of servers called brokers. A consumer can subscribe to one or more topics from the brokers. It can consume the subscribed messages by pulling data from the brokers. Kafka is very much a general-purpose system. Many producers and consumers can share multiple topics.

In contrast, Flume (Han & Ahn, 2014) is a special-purpose framework designed to send data to HDFS and HBase (C. P. Chen & Zhang, 2014). It has various optimizations for HDFS. Flume can process data in-motion using its interceptors. These can be very useful for ETL operations or filtering. Kafka requires an external stream processing system to execute this type of work. Flume does not support event replication in contrast to Kafka. Consequently, even when using the reliable file channel, if one of the Flume agent in a node goes down, the events in the channel cannot be accessed until a full recovery.

Flume and Kafka can be used together very well. Streaming the data from Kafka to Hadoop is required, using a Flume agent with Kafka producers to read the

data has some advantages. For instance, Flume's integration with HDFS and HBase is natural, not only even adding an interceptor, doing some stream processing during delivery is easily possible as well. For this reason, using Kafka if the data will be consumed by multiple sinks and Flume if the data is designated for Hadoop are best practices for this type of work. Flume has many built-in sources and sinks that can be used in various architectures and designs. However, Kafka, has a quite smaller producer and consumer ecosystem.

There are also other Message Oriented Middlewares and selection for an application depends on specific requirements. Simple Queue Service (SQS), is a message-queue-as-a-service offering from Amazon Web Services (Yoon, Gavrilovska, Schwan, & Donahue, 2012). It supports only useful and simple messaging operations, quite lighter from the complexity of e.g. AMQP (Advanced Message Queuing Protocol), SQS provides at-least-once delivery. It also guarantees that after a successful send operation, the message is replicated to multiple nodes. It has good performance and no setup required. RabbitMQ is one of the leading open-source messaging systems. It is developed in Erlang and very popular for messaging. It implements AMQP and supports both message persistence and replication with partitioning. If high persistence is required, RabbitMQ guarantees replication across the cluster and on disk for message sending operations. Apache ActiveMQ is one of the most popular message brokers. It is widely used as messaging broker with good performance and wide protocol support. HornetQ is multi-protocol, embeddable, very high performance, clustered, asynchronous messaging system and implements JMS, developed by JBoss and is part of the JBossAS. It supports over-the-network replication using live-backup pairs. It has great performance with a very plenty messaging interface and routing options.

3.3 HYBRID PROCESSING

Many big data applications include batch and real-time operations. This problem can be achieved with hybrid solutions. Hybrid computation in big data started with the introduction of Lambda Architecture (LA) (Casado & Younas, 2015). LA provides to optimize costs by understanding parts of data having batch or real-time processing. Besides, the architecture allows to execute various calculation scripts on partitioned datasets (Kiran, Murphy, Monga, Dugan, & Baveja, 2015).

Basically, a LA compromises of three distinct layers for processing both data in motion (DiM) and data at rest (DaR) at the same time (Marz & Warren, 2015). Every layer of the LA is related to a certain task for processing different type of data, combines the processed results from these different layers together and serves these merged data sets for querying purposes. Speed Layer is mainly responsible for processing the streaming data (DiM) and very vulnerable to delaying and recurring data situations. Batch Layer is basically used for processing the offline data (DaR) and correction of the errors that sometimes occur for data arrival to the speed layer. Serving layer is in charge of ingestion of data from batch and speed layer, indexing and combining result data sets from the queries from the applications. This layer has a special need for importing both streaming data real time as it comes and also batch data in huge size. Because of this special requirement, usable technology for this layer is currently limited but not a few. It has to be emphasized that LAs are eventually consistent systems for data processing applications and can be used for dealing with CAP theorem (Twardowski & Ryzko, 2014).

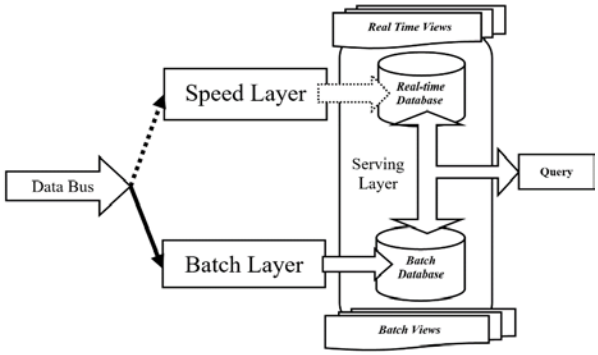

Figure 3. Big data classification

A conceptual explanation of LA is shown in Fig. 3. Incoming data from data bus is sent to both speed and batch layers and then generated views for the layers hosted on the serving layer. There are different technologies can be used in all three layers to form a LA. According to polyglot persistence paradigm, each technology is used for the special capability to process data. It is indispensable using big data technologies for IoT devices. LA can be used for IoT based smart home applications (Villari, Celesti, Fazio, & Puliafito, 2014). Data from different IoT sensors can be collected and processed both real time and offline with Lambda Architecture. With this three-layered architecture, Apache Storm is used for real

time processing, and MongoDB was used for both batch layer and serving layer respectively.

It is very common that Apache Storm is used for speed layer of LA, and MongoDB is also used for batch and serving layer for LA applications. Using two different technologies for speed and batch layer result in development of two different software and processing applications for these layers. Maintaining at least two different software for LA applications are not easy in big data domain. Debugging and deployment of different software on large hardware clusters for big data applications require extra effort, attention, knowledge and work. This sometimes may be painful job to do. To overcome this problem, using the same data processing technology for different layers is an approach (Demirezen, Küçükayan, & Yılmaz, 2015). It is shown that by combining the batch and serving layers and also using the high speed real time data ingestion capabilities of MongoDB helps accomplishing this task but not enough. The same data processing technology for speed and batch layers has to be selected. Multi-agent based big data processing approach was implemented for using collaborative filtering to build a recommendation engine by using LA (Twardowski & Ryzko, 2014). Apache Spark/Spark Streaming, Apache Hadoop YARN and Apache Cassandra technologies were used for real time, batch and serving layers respectively for this application. Agent based serving, batch and speed layers were implemented to build both real time and batch views and querying the aggregated data. Apache Hadoop and Storm are mature technologies to implement LA with different technologies.

A different approach for using the speed and batch layers of LA was implemented (Kroß, Brunnert, Prehofer, Runkler, & Krcmar, 2015). In cases of time constraints are not applicable in minutes, running speed layer in a continuous manner is not required. Using stream processing only when batch processing time exceeds the response time of the system is a method to utilize the cluster resources efficiently. Running a speed layer at the right time requires predicting the finishing time of the batch layer data processing operation. Using performance models for software systems to predict performance metrics of the system and cluster resources. Then running the speed layer is an application specific approach. This has to be investigated in design time.

Data bus is formed to ingest high volume of real time data for the Lambda Architecture. One of the most widely used framework for data bus is Apache Kafka and it is very mature and good for this purpose. It supports high throughput data transfer and is a scalable, fault tolerant framework for data bus operations. For the speed layer Apache Samza, Apache Storm and Apache Spark (Streaming) are very good choices. Using Apache Hadoop, Apache Spark is very common for the batch layer operations. Apache Cassandra, Redis, Apache HBase, and MongoDB might be used as speed layer database. These databases support not only high speed real time data ingestion but also random read and write as well. MongoDB, CouchbaseDB, SploutSQL and VoldemortDB can be used as a batch layer database. These databases can be import bulk data to form and serve batch views. Generally using NoSQL databases for LA is very common instead of relational databases. Scalable and advanced capabilities for data ingestion are main reasons to be used at serving layer.

Programming in distributed frameworks may be complex and debugging and may be even harder. This has to be done twice in LA for batch and speed layers. The most important disadvantage of Lambda Architecture is that sometimes it is not practical to write the same algorithm twice with different frameworks for the developers. Same business logic might be used both layers and it requires implementing same algorithm for both layers. Maintaining and debugging the code might be very challenging process in distributed computing. Using Apache Spark and Spark Streaming together provides reuse of the same code for batch and online processing, join data streams against historical data. As for the Lambda Architecture, Spark Streaming and Spark can be used for developing speed layer and batch layer applications. However, one problem remains that serving layer has to be integrated with both layers for data processing and has to provide data ingestion for both layers. Speed and Batch layers require different data ingestion capabilities and operations. Therefore, Serving Layer has to be formed according to this design challenges. Generally serving layer is formed with using different database technologies in LA. Querying both databases, merging the results and sending as a response, is very hard work to do, especially in Big Data analytics applications. Instead of using this approach, serving layer can be formed by using special database technology that supports the both requirements in LA.

4. CONCLUSIONS

Big data approaches provide new challenges and opportunities to the users, customers or researchers, if

there have been available data and sources. Although available big data systems provide new solutions, they are still complex and require more system resources, tools, techniques and technologies. For this reason, it is necessary to develop cheaper, better and faster solutions.

Big data solutions are specific in three forms: software-only, as an appliance and cloud-based (Dumbill, 2012). These solutions are preferred according to the applications, requirements and availability of data. There are a large number of big data products that have Hadoop environment with a combination of infrastructure and analysis capabilities, while some of the big data products are developed for specific framework or topics. Big data infrastructures and techniques should trigger the development of novel tools and advanced algorithms with the help of distributed systems, granular computing, parallel processing, cloud computing, bio-inspired systems, hybrid-systems and quantum computing technologies (C. P. Chen & Zhang, 2014). For example, cloud computing is served to big data for the purposes of being flexible and effective on infrastructure. Storage and management issues of heterogeneous data are handled via distributed file systems and NoSQL databases. Dividing big problems into smaller pieces can make them easier and faster so, granular computing and parallel processing are good choices. Simulating intelligence or social behaviors of living creature are connected the development of machine learning and artificial intelligence fields with the help of bio-inspired systems. In addition to all, for software innovations, hardware innovations in processor and storage technology or network architecture have played a major role (Kambatla, Kollias, Kumar, & Grama, 2014).

To achieve more successful big data management and better results for applications, it is necessary to select appropriate programming models, tools and technologies (Lei, Jiang, Wu, Du, & Zhu, 2015). Even after providing and qualifying the technical infrastructure, there is a need for big data experts to select appropriate data management model and analysis process, to organize data priorities and to suggest creative ideas on big data problems for scientific developments or capital investments (M. Chen et al., 2014), (Rajan et al., 2013). For qualified people to achieve professional results, it is necessary to supply training and learning opportunities via providing big datasets in public domains to be used in research and development. Opening new courses and programs at universities might help to increase number of experts to handle problems easily and effectively.

It is also expected that big data will not only provide opportunities for improving operational efficiency, informing better strategic targets, providing better customer services, identifying and producing new tools, products and services, distinguishing customer and users, but also prevent threats and privacy violation and provide better security. Generally accepted that the traditional protection techniques are not suitable for big data security and privacy. However, open source or new big data technologies may host unknown drawbacks if they are not well understood. For this reason, confidentiality, integrity and availability of information and computer architecture must be discussed from every angle in big data analysis. The development of big data systems and applications is led to abolish the individual control about collection and usage of personally identifiable information like to know new and secret facts about people or to add value organizations with collected data from unaware people. As indicated in (Wong, Fu, Wang, Yu, & Pei, 2011), anonymization techniques such as k-anonymity, l-diversity, and t-closeness may be solutions to prevent the situation. Therefore, the law and the regulations must be enforced with clarified boundaries in terms of unauthorized access, data sharing, misuse, and reproduction of personal information.

The evaluations have shown that big data is a real challenge not only for official organizations but also companies, universities and research centers having big data to profound influences in their future developments, plans, decisions, actions and imaginations. Even if enough tools, techniques and technologies are available in the literature, it can be concluded that there are still many points to be considered, discussed, improved, developed and analyzed about big data and its technology. It is hoped that this article would help to understand the big data and its ecosystem more and to be developed better solutions not only for today but also for future

5. REFERENCES

Assunção, M. D., Calheiros, R. N., Bianchi, S., Netto, M. A., & Buyya, R. (2015). Big Data computing and clouds: Trends and future directions. Journal of Parallel and Distributed Computing, 79, 3-15.

Bajpeyee, R., Sinha, S. P., & Kumar, V. (2015). Big Data: A Brief Investigation on NoSQL Databases,". International Journal of Innovations & Advancement in Computer Science, 4(1), 28-35.

Casado, R., & Younas, M. (2015). Emerging trends and technologies in big data processing. Concurrency and Computation: Practice and Experience, 27(8), 2078-2091.

Cattell, R. (2011). Scalable SQL and NoSQL data stores. Acm Sigmod Record, 39(4), 12-27.

Center, I. I. (2012). Peer Research: Big Data Analytics. Intel's IT Manager Survey on How Organizations Are Using Big Data.

Chandy, K. M., & Lamport, L. (1985). Distributed snapshots: Determining global states of distributed systems. ACM Transactions on Computer Systems (TOCS), 3(1), 63-75.

Chen, C. P., & Zhang, C.-Y. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. Information Sciences, 275, 314-347.

Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. Mobile Networks and Applications, 19(2), 171-209.

Demchenko, Y., Grosso, P., De Laat, C., & Membrey, P. (2013). Addressing big data issues in scientific data infrastructure. Paper presented at the Collaboration Technologies and Systems (CTS), 2013 International Conference on.

Demirezen, M. U., Küçükayan, Y. G., & Yılmaz, D. B. (2015). Developing Big Data Realtime Stream And Batch Data Processing Platform. Paper presented at the Ulusal Savunma Uygulamalari Modelleme ve Simülasyon Konferansı,, Ankara.

Dumbill, E. (2012). Big Data Now: Current Perspectives: O'Reilly Radar Team, O'Reilly Media, USA.

Feng, T., Zhuang, Z., Pan, Y., & Ramachandra, H. (2015). A memory capacity model for high performing data-filtering applications in Samza framework. Paper presented at the Big Data (Big Data), 2015 IEEE International Conference on.

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. International Journal of Information Management, 35(2), 137-144.

Han, U., & Ahn, J. (2014). Dynamic load balancing method for apache flume log processing. Advanced Science and Technology Letters, 79, 83-86.

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. (2015). The rise of "big data" on cloud computing: Review and open research issues. Information Systems, 47, 98-115.

Kambatla, K., Kollias, G., Kumar, V., & Grama, A. (2014). Trends in big data analytics. Journal of Parallel and Distributed Computing, 74(7), 2561-2573.

Kiran, M., Murphy, P., Monga, I., Dugan, J., & Baveja, S. S. (2015). Lambda architecture for cost-effective batch and speed big data processing. Paper presented at the Big Data (Big Data), 2015 IEEE International Conference on.

Kreps, J., Narkhede, N., & Rao, J. (2011). Kafka: A distributed messaging system for log processing. Paper presented at the Proceedings of the NetDB.

Kroß, J., Brunnert, A., Prehofer, C., Runkler, T. A., & Krcmar, H. (2015). Stream processing on demand for lambda architectures. Paper presented at the European Workshop on Performance Engineering.

Lei, J., Jiang, T., Wu, K., Du, H., & Zhu, L. (2015). Robust local outlier detection with statistical parameter for big data. COMPUTER SYSTEMS SCIENCE AND ENGINEERING, 30(5), 411-419.

Marz, N., & Warren, J. (2015). Big Data: Principles and best practices of scalable realtime data systems: Manning Publications Co.

McCafferty, D. (2014). Surprising Statistics About Big Data. Surprising Statistics About Big Data. February 18, 2014. http://www. baselinemag. com/analytics-big-data/slideshows/surprising-statistics-about-big-data. html.

Mysore, S. D., & Jain, S. (2013). Big Data Architecture and Patterns, Part 1: Introduction to Big Data Classification and Architecture. IBM Corp.

Neumeyer, L., Robbins, B., Nair, A., & Kesari, A. (2010). S4: Distributed stream computing platform. Paper presented at the Data Mining Workshops (ICDMW), 2010 IEEE International Conference on.

P. Almeida, J. B. (2015). A comprehensive overview of open source big data platforms and frameworks. International Journal of Big Data (IJBD), 2(3), 15-33.

Rajan, S., van Ginkel, W., Sundaresan, N., Bardhan, A., Chen, Y., Fuchs, A., Manadhata, P. (2013). Expanded top ten big data security and privacy challenges. Cloud Security Alliance, available at https://cloudsecurityalliance. org/research/big-data/, viewed on, 12.

Renner, T., Thamsen, L., & Kao, O. (2015). Network-aware resource management for scalable data analytics frameworks. Paper presented at the Big Data (Big Data), 2015 IEEE International Conference on.

Tatbul, N. (2010). Streaming data integration: Challenges and opportunities. Paper presented at the Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on.

Toshniwal, A., Taneja, S., Shukla, A., Ramasamy, K., Patel, J. M., Kulkarni, S., Donham, J. (2014). Storm@ twitter. Paper presented at the Proceedings of the 2014 ACM SIGMOD international conference on Management of data.

Twardowski, B., & Ryzko, D. (2014). Multi-agent architecture for real-time big data processing. Paper presented at the Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on.

Villari, M., Celesti, A., Fazio, M., & Puliafito, A. (2014). Alljoyn lambda: An architecture for the management of smart environments in iot. Paper presented at the Smart Computing Workshops (SMARTCOMP Workshops), 2014 International Conference on.

Wang, E., & Chen, G. (2013). An Overview of Big Data Mining: Methods and Tools. Paper presented at the International Symposium on Signal Processing, Biomedical Engineering and Informatics, China.

Why HPCC Systems is a superior alternative to Hadoop. 2017, from https://hpccsystems.com/why-hpcc-systems/hpcc-hadoop-comparison/superior-to-hadoop

Wong, R. C.-W., Fu, A. W.-C., Wang, K., Yu, P. S., & Pei, J. (2011). Can the utility of anonymized data be used for privacy breaches? ACM Transactions on Knowledge Discovery from Data (TKDD), 5(3), 16.

Yoon, H., Gavrilovska, A., Schwan, K., & Donahue, J. (2012). Interactive use of cloud services: Amazon sqs and s3. Paper presented at the Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: Cluster Computing with Working Sets. HotCloud, 10(10-10), 95.

Authors

Duygu SINANC is research assistant of Gazi University Graduate School of Natural and Applied Science. She received her M.Sc. degree from Gazi University Department of Computer Engineering and continues her Ph.D. at the same department. Her research interests are big data analytics, data mining and information security.

Umut DEMIREZEN is Cyber Security and Big Data Research and Development Group Manager at STM Defense Technologies Engineering and Trade Inc. He received his Ph.D. degree from Gazi University Department of Electrical and Electronics Engineering. His research interests are data science, big data and machine learning.

Seref SAGIROGLU is professor Department of Computer Engineering at Gazi University. His research interests are intelligent system identification, recognition and modeling, and control; artificial intelligence; heuristic algorithms; industrial robots; analysis and design of smart antenna; information systems and applications; software engineering; information and computer security; biometry, electronic signature and public-key structure, malware and spyware software. Published over 50 papers in international journals indexed by SCI, published over 50 papers in national journals, over

100 national and international conferences and symposium notice and close to 100 notice are offered in national symposium and workshops. He has three patents and 5 pieces published book. He edited four books. He carried out of a many national and international projects. He hold the many conferences and continues academic studies.