FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 41 (2016) No. 3
DE GRUYTER OPEN
DOI: 10.1515/fcds-2016-0009
ISSN 0867-6356
e-ISSN 2300-3405

PRE-PROCESSING AND MODELING TOOLS FOR BIGDATA

Hadi HASHEM[1], Daniel RANC[2]

Abstract. Modeling tools and operators help the user / developer to identify the processing field on the top of the sequence and to send into the computing module only the data related to the requested result. The remaining data is not relevant and it will slow down the processing. The biggest challenge nowadays is to get high quality processing results with a reduced computing time and costs. To do so, we must review the processing sequence, by adding several modeling tools. The existing processing models do not take in consideration this aspect and focus on getting high calculation performances which will increase the computing time and costs. In this paper we provide a study of the main modeling tools for BigData and a new model based on pre-processing.

Keywords: Data Modeling, BigData, NoSQL, MapReduce, Pre-processing.

1. Introduction

More than 2.5 Exabytes of data are created everyday on Internet, based on the automatically generated user information. Social networks, mobile devices, emails, blogs, videos, banking transactions and other consumer interaction, are now driving the successful marketing campaigns, by establishing a new digital channel between the brands and their audiences. Powerful tools are needed to store and explore this daily expending BigData, in order to submit an easy and reliable processing of user information. Expected quick and high quality results are as much important as priceless investments for marketers and industrials. Traditional processing engines face their limits in this challenge, as the information keeps growing in volume and variety, thing that can be handled only by non-relational data modeling techniques.

The challenge of BigData is to query data easily [24]. Creating data models on physical data help to manage raw data. Data Modeling provides a visual way in order to manage data resources and create data architecture. This will help user / developer creating more applications to optimize data reuse and reduce computing costs. The modeling tools to discuss in this paper help to better handle the processing of BigData.

[1] Télécom SudParis, France, hadi.hashem@telecom-sudparis.eu
[2] Télécom SudParis, France, daniel.ranc@telecom-sudparis.eu

For starting, we make a quick review on common data sources, data engines and non-relational databases. In section 2, we will discuss the use of the main modeling tools available for BigData. We will point-out by then the forces and weaknesses of every technique. In section 3, we suggest new approach of using modeling toolbox for BigData. This approach is implemented in our BigData Workbench software that we are also going to discuss. Finally, we address the conclusion and the future works in section 4.

1.1. Data sources on Internet

Consumer interaction on Internet is establishing a new digital channel between the brands and their audiences. Exabytes of data are created everyday as information based on data models that keep growing in volume and variety. Document-oriented data models assume documents encapsulate and encode data in some standard formats. Encodings in use include XML, YAML, JSON and BSON, as well as binary forms like PDF and Microsoft Office documents (old format). Whereas file types like XML allow different ways to define a schema, others like JSON have no explicit schema. Such documents have some kind of an implicitly defined schema. The main idea is to infer the schema from a sample file and provide a model to access any file which is conform to this schema.

JSON illustrated in figure 1 using Open Weather API, is the most used technique for interchanging data on the web. Based on the file content, the implicit schema can be constructed [14]. There are three main groups possible inside:

- Values, which are the lowest level of data, values can be strings, numbers, Booleans, dates or even Objects and Arrays.
- Objects, which contains a set of name / value pairs.
- Arrays, which are lists of Objects.

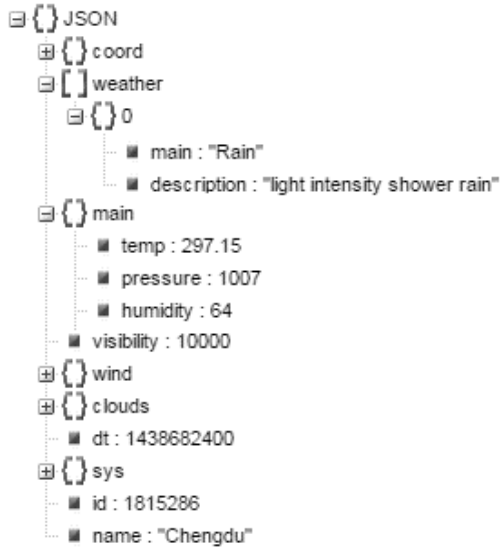

Figure 1. JSON weather data implicit schema

JSON files are language-neutral and their data structures are universally recognized. These structures are supported by almost all modern programming languages and are well known from most of the developers. For all these reasons, JSON is an ideal format for data interchange on the web. JSON is also a literal data structure in an interactive scripting language present on every computer, smartphone and tablet currently available.

1.2. Available data processing engines

Hadoop MapReduce is considered as an efficient processing technique for Big Data, since it provides better performance when dealing with complex high volumes of data. RDMS (Relational Database Management Systems) show a high performance indicator when processing small relational data, but are very limited in face of expanding data in volume and variety [23]. MPP (Massively Parallel Processing) has slowly improved the performance indicator for complex volumes of data. Still, it could not be used to process BigData in a daily expansion. RDMS are unable to handle this task for the following reasons:

- The primary constraining factor is database schema, because of the continuous changing structure of schema-less BigData.
- The complexity and the size of data, overflows the capacity of traditional RDMS to acquire, manage and process data with reasonable costs (computing time and performance).
- Relation-Entity modeling of BigData does not easily adapt with fault-tolerant and distributed systems [8].

Apache Spark which is based on Hadoop MapReduce technology provides an Ultimate Framework allowing to process different natures of data like text, graphs or real-time streaming. Spark is able to get an immediate increase in performance, using In-Memory processing feature and supports SQL queries with a dedicated command line shell.

On the other hand, Spark is currently facing some limitations in terms of memory management (maxResultSize or frameSize), small-size files processing, resource consuming GZip compression and unstable processing of real-time streaming [17]. For these reasons, Spark is not considered yet as a solution reaching the level of maturity.

NoSQL (Non-relational SQL) is increasingly chosen as viable alternative to relational databases, particularly for interactive web applications and services [9], [21]. NewSQL is a distributed storage derivative of NoSQL and created to handle high rates of transactional access using a SQL interface. It combines the scalability of NoSQL and the relational interface of standard SQL, based on a distributed In-Memory processing over the cluster. Still, the current architecture of NewSQL limits the processing to several terabytes of data only. On the other site, the In-Memory processing requires expensive hardware [18] and provides limited access to standard SQL tools.

1.3. Non-relational databases

The highly expending information nowadays contains complex and heterogeneous data types (text, images, videos, GPS data, purchase transactions…) that require a powerful data computing engine, able to easily store and process such complex structures. The four Vs definition (volume, velocity, variety, veracity) describing this expansion of data will then

lead to extract the unnamed fifth V (value) from BigData. This so-called value is sought in big amounts of enterprise data to process in the daily life [22].

Relational and non-relational data models are different. The relational model takes data and separates it into many interrelated tables that contain rows and columns. Tables reference each other through foreign keys that are stored in columns as well [7]. When querying data, the requested information will be collected from many tables, as if the user asks: what is the answer to my question?

Non-relational data models often starts from the application-specific queries as opposed to relational modeling [2]. Data modeling will be driven by application-specific access patterns. An advanced understanding of data structures and algorithms is required, so that the main design would be to know: what questions fit to my data?

Fundamental results on distributed systems like the CAP theorem apply well to non-relational systems. Since, relational models were designed to interact with the end user, the non-relational models is on permanent evolution in order to include more functionalities of the relational model, like the transactional aspect or the join operations.

2. Modeling tools and operators

There are four main families most used in non-relational database modeling:
- Key-value store, the simplest non-relational model compared to a distributed hashmap. In this model, only PUT, GET and DELETE operations are allowed. Voldemort créé par LinkedIn is an example of this model.
- Column-oriented model, a dataset that can grow to immense size with a column oriented layout, very effective to store sparse data as well as multi-value cell. Examples are Apache Cassandra, Amazon Dynamo [3], Google BigTable and Hadoop HBase.
- Document-oriented model, designed for storing, retrieving, and managing document-oriented or semi structured data, such as XML or JSON. Examples are Apache CouchDB RavenDB (for .Net platforms) and MongoDB.
- Graph data model, a schema-less non-relational database allowing the storage of information about the relationships between entries. Neo4J is a graph management solution used on a wide scale.

2.1. Multi-model Storage

So called Multi-model storage combines scalability, fault tolerance and high performance with ACID transactions. It is based on a special concept providing the capacities of all main families described previously in one:
- Developers can store all types of data.
- Administrators easily scale and handle hardware failures.
- Business owners save money with industry-leading performance.

From a technical point of view, this model should provide 3-layer architecture as described in figure 2, allowing the application created to exchange directly with key-value stores of several servers over the cluster or intermediate stateless SQL layers, providing a

flexible architecture. On the other side, there are several aspects to take in consideration with regards to this model, reason for which it is considered as premature:
- In such structure, the system is self-dependent even if integration simplifies development, it will not be possible to upgrade one area in the system. This will make the costs too high and will slow down the ROI on long-term.
- The industry does not show any sign of going toward multi-model solutions. Currently none of the worldwide big players are using the multi-model approach.
- For now, the existing multi-model solutions are used on critical systems (health, finance, airports) but not on a wide scale.

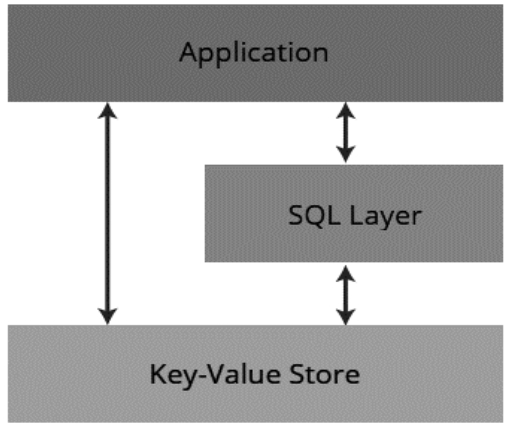

Figure 2. Multi-model storage

2.2. Common modeling operators

Modeling tools translate complex system designs into simple representations of the data flows and processes. Users often use several models to view the same data and ensure that all processes, entities, relationships and data flows have been identified. In a previous paper in this study [5], we discussed the modeling techniques listed below:
- Conceptual data modeling.
- General data modeling.
- Hierarchical data modeling.

2.2.1. Conceptual modeling tools

The aim is to identify the high-level relationships between entities using one of the following:
- De-normalization, by duplicating the same data into multiple storages (tables or documents).
- Aggregates, one of the common ways in order to guarantee some of the ACID properties.
- Joins, having significant effect on seeding queries [1].

Still, we must also discuss some existing constrains:
- By using de-normalization, the total volume of data will be increased.
- In most of the cases, Joins are not supported in Non-relational systems.

2.2.2. General modeling tools

Non-relational database engines have limited transaction support. However, one can perform transactional behaviour in some cases only, by using one of the following modeling approaches:
- The Atomic Aggregates can be applicable if the data store provides certain guaranties of atomicity, locks or test-and-set instructions.
- It is possible to map multidimensional data to a simple key-value store or to another multidimensional data by using Dimensionality Reduction operator.
- One can use Table Index operator for BigTable-style databases, as a simple or multidimensional index (using composite keys). The only thing is to create and maintain the special index table regularly or in batch-mode.
- The Enumerable Keys operator allows avoiding unordered or complex key-value records, when using multiple server clusters.

We can also consider in some cases the Inverted Search as an additional tool. Still, it is often used as a data processing pattern more than a data modeling operator.

2.2.3. Hierarchical modeling tools

Hierarchical modeling consists of organizing the data into a tree-like structure which allows representing the data records using parent / child relationships, where each parent can have many children, but each child has only one parent. There are several implementations of hierarchical modeling existing:
- Tree Aggregation, allows combining trees and graphs into a single record or document.
- Adjacency Lists, allows searching for nodes by their parents or children identifiers.
- Path Enumeration, considered as a variant of de-normalization and allows storing the chain of ancestors in each node.
- Nested Sets, consists of storing leafs of the tree in an array and to map each non-leaf node to a range of leafs using start and end indexes.

As for related constraints, we also noticed the following:
- While using Tree Aggregation, search and arbitrary access to the entries might be problematic.
- In Adjacency Lists approach, while doing one hop per query, it is inefficient to get an entire sub-tree for a given node, for deep or wide traversals.

2.3. Graph processing

Batch graph processing technique related to graph databases can be done using MapReduce routines [4], in order to explore the neighborhood of a given node or relationships between

two or a few nodes. This approach makes key-value stores, document databases and BigTable-style databases suitable for processing large graphs [13].

Adjacency list representation can be used in graph processing. Graphs are serialized into key-value pairs using the identifier of the vertex as the key and the record comprising the vertex's structure as the value. In MapReduce process represented in figure 3, the shuffle and sort phase can be exploited to propagate information between vertices using a form of distributed message passing. In the reduce phase, all messages that have the same key arrive together and another computation is performed. And so, the state of one node rapidly propagates across the cluster. All nodes that were updated by this state start to update all their neighbors

Combiners in MapReduce are responsible for performing local aggregation which reduces the amount of data to be shuffled across the cluster. They are only effective if there are multiple key-value pairs with the same key, computed on the same machine that can be aggregated [12].

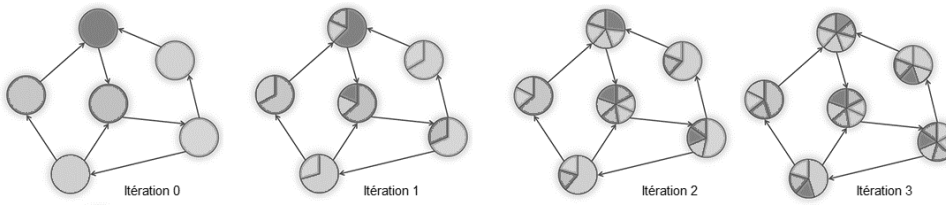

Figure 3. Graph processing with MapReduce

3. Dynamic use of modeling tools

The modeling tools combined together can bring-up a high value-added to the processing chain. It can be implemented in a pre-processing phase [6], by adding several modeling operators to identify salient data between all collected data from different sources and platforms [19]. Finally, only relevant data will be sent to the calculation server.

3.1. Pre-processing advantages

Nowadays, companies can collect heterogeneous data with different types. The collected data can be either structured (contractual or voluntary data collected from consumers through additional services) or unstructured that is essentially present on the web (Social Media content) or CRM systems (consumer profile data) [11].

The data becomes more valuable as much as it is more personalized and up-to-date. Companies must convince consumers that they would get better products and services by providing feedback information [20]. In this case, the companies will be well informed about consumer expectations and brands will be able to improve the consumer satisfaction. In order to extract valuable data needed for calculation, one or more data modeling tools can be used. As discussed previously, there are three main data modeling categories, in which one can pick-up his selection of modeling operators:

- Conceptual techniques able to identify the highest-level relationships between different entities (duplication, aggregates, joins).
- General techniques (dimension reduction, index table, enumerable keys).
- Hierarchical techniques that organize the data into a tree-like structure, allowing representing information using parent / child relationships (tree aggregation, adjacency lists, path enumeration).

These classic data modeling tools must be coupled with an efficient pre-processing algorithm for better performances, as illustrated in figure 4.

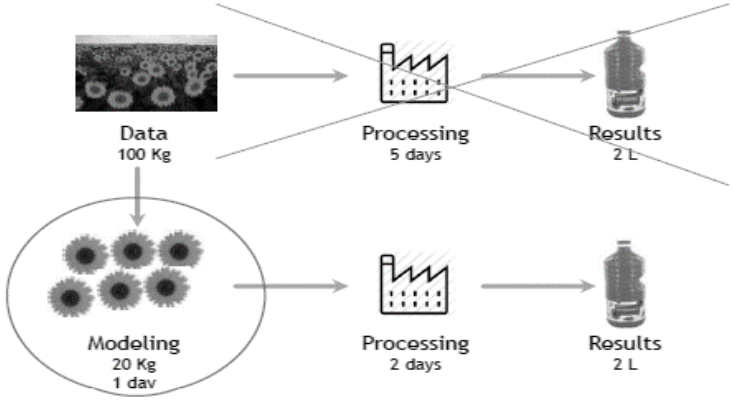

Figure 4. Pre-processing advantages

3.2. BigData workbench

There is a high impact on the use of the modeling tools on non-relational data processing. In order to implement a new abstraction based on model driven architecture, we thought about creating new automatic programming software allowing the users / developers, based on drag-and-drop features, to do the following:
- Add one or more components from available data sources (data files, social networks, web services).
- Apply one or more of non-relational data modeling tools by connecting the components together.
- Apply predefined analysis on sample data in order to dynamically define the structure of the files / messages.
- Select a Hadoop processing engine available on a local or distant network.

In the example of figure 5, we capture the data from several available data sources (Facebook, Twitter, Open Weather API, Here Traffic API) all related to keyword *Storm*.

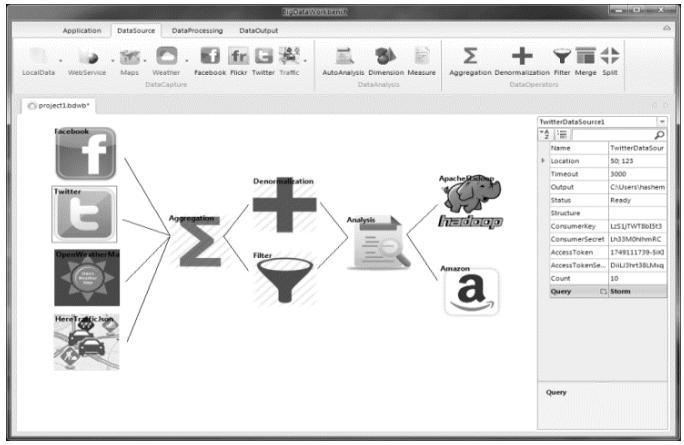

Figure 5. BigData workbench

3.3. Experiment scenarios

We conducted an experiment in order to measure the frequency of the keyword in a time period. The experiment is based on 3-level criteria:
- Scenario 1: Processing bulk data imported from the different data sources without using any modeling tool. 25 GB of data is sent to a processing server.
- Scenario 2: We introduced the Aggregation modeling tool between the data source and the processing server. The goal is to process all data in one shot instead of different processing levels depending on the data source. This modeling tool is based on one data file format, independently from related data source.
- Scenario 3: Finally, we also introduced the De-normalization and the Filter modeling tools in order to localize and isolate the salient data from the others (using Filter component) and also, to duplicate this data for better processing performances (using De-normalization component).

In all processing scenarios, the data is sent to a local (Hadoop) and distant (Amazon) processing server in order to compare computing duration. Both servers were based on the following hardware specifications:
- 1 TB hard disc drive.
- 2.5 GHz CPU.
- 8 GB RAM.
- 10 GBit Ethernet connection.

3.4. Experiment results

We noticed the following result information of our experiment (the figures shown depend on the content of the captured data). Based on the results in table 1, we built-up the graph in figure 6 below. Using the modeling tools, the data processing provides better performance on the calculation server:

- It required less than *1h* of pre-processing.
- The server calculation provides almost similar results locally and on the Cloud.
- In total, about *5h* were needed to get the required results, which still much better than sending the data to the calculation server without using the data modeling components.

Table 1. Experiment results

	Data pre-processing	Local Apache Hadoop server	Amazon
Scenario 1		>24h	>24h
Scenario 2	+23m	11h17m	10h42m
Scenario 3	+39m	5h21m	4h54m

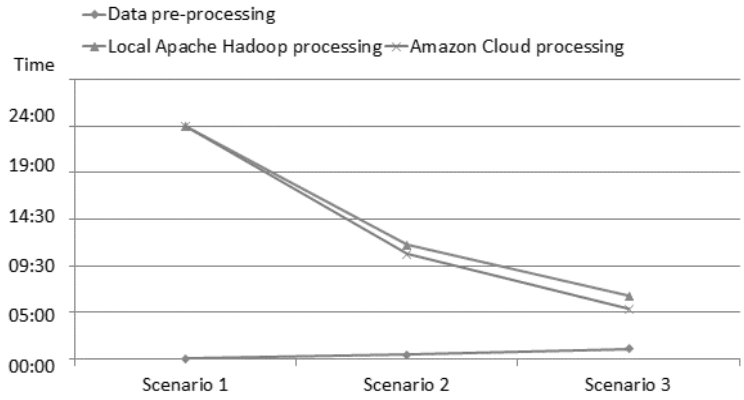

Figure 6. Experiment results

3.5. Business impact

Time consuming calculations, in finance, business intelligence or any other activity fields, can now be processed in the Cloud at a new level of speed. Intuitive platforms are available so that everyone could run time consuming calculations on clusters with high number of CPUs. Only large enterprises and universities were getting access so far, to High Performance Computing. This fact was leading to important competitive disadvantages for small enterprises. Using today's provider solutions, High Performance Computing is accessible to everyone [15], [16].

Since users only pay for what they consume, the cluster scalable servers lead to cost savings and minimal out of work times. In order to reduce these costs, it is important to process on the distant server only salient data in terms of relation with final results. We believe that such technical approach will help to make this preparation step and reduce data processing costs by:
- Making own design of the processing chain.
- Dispatching the processing on several computing engines.
- Reducing the volume of data to compute.

4. Conclusion

In this paper, we discussed the available modeling tools and techniques for BigData. We mentioned the advantages and disadvantages, so that we provide a state of the art able to fill the gap in the domain of BigData processing approaches. We discussed some concepts which are not used on a wide scale, such as Multi-model Storage. We also discussed common concepts such as Graph Processing.

In the last section, we shared an interesting experience combining several modeling tools, using BigData Workbench solution on which our study in next level will be oriented. Our goal is to provide the users in general and the scientific community in specific, a new technique of data processing for BigData, with better performance in terms of data calculation and business costs [10]. The IT technology is moving forward very fast, especially on this field. We believe that we are close to deliver such solution

References

[1] Afrati F. N., Ullman J. D., Optimizing joins in a map-reduce environment, International Conference on Extending Database Technology, 2010, Print ISBN 978-1-60558-945-9.

[2] Chalkiopoulos A., Programming MapReduce with Scalding, Packt Publishing, 2014, Print ISBN 978-1783287017.

[3] DeCandia G., Hastorun D., Jampani M., Kakulapati G., Lakshman A., Pilchin A., Sivasubramanian S., Vosshall P., Vogels W., Dynamo: Amazon's highly available key-value store, ACM Symposium on Operating Systems Principles, 2007, DOI 10.1145/1323293.1294281.

[4] Ghemawat S., Gobioff H., Leung S.K., The Google File System, SOSP '03 Proceedings of the nineteenth ACM symposium on Operating systems principles, 2003, Print ISBN 1-58113-757-5.

[5] Hashem H., Ranc D., An integrative Modeling of BigData Processing, International Journal of Computer Science and Applications, 2014, Print ISSN 0972-9038.

[6] Hashem H., Ranc D., Predicate-based Cloud Computing, 8th International Conference on Next Generation Mobile Apps, Services and Technologies, 2014, Print ISBN 978-1-4799-5072-0.

[7] Kaur, K., Rani, R., Modeling and querying data in NoSQL databases, Big Data, 2013 IEEE International Conference, INSPEC Accession Number 13999217.

[8] Kim S., Jung W., Kim H. S., A location inference algorithm based-on smart phone user data modelling, International Conference on Advanced Communication Technology, 2014, Print ISBN 978-89-968650-2-5.

[9] Li Y., Manoharan S., A performance comparison of SQL and NoSQL databases, Communications, Computers and Signal Processing, 2013 IEEE Pacific Rim Conference, Print ISSN 1555-5798.

[10] Lin J., Bahety A., Konda S., Mahindrakar S., Lowlatency, high-throughput access to static global resources within the Hadoop framework, Technical Report - University of Maryland, 2009, HCIL-2009-01.

[11] Lin J., Dyer C., Data-Intensive Text Processing with MapReduce (Synthesis Lectures on Human Language Technologies), Morgan and Claypool Publishers, 2010, Print ISBN 978-1608453429.
[12] Lin J., Schatz M., Design Patterns for Efficient Graph Algorithms in MapReduce, University of Maryland, College Park, 2010, Print ISBN 978-1-4503-0214-2.
[13] Malewicz G., Austern M. H., Bik A. J. C., Dehnert J. C., Horn I., Leiser N., Czajkowski G., Pregel: A system for largescale graph processing, ACM Conference on Management of Data, 2010, Print ISBN 978-1-4503-0032-2.
[14] Mesiti M., Valtolina S., Towards a User-Friendly Loading System for the Analysis of Big Data in the Internet of Things, Computer Software and Applications Conference Workshops, 2014 IEEE 38th International Conference, Print ISBN 978-1-4799-3578-9.
[15] Miner D., Shook A., MapReduce Design Patterns: Building Effective Algorithms and Analytics for Hadoop and Other Systems, O'Reilly Media, 2012, Print ISBN 978-1449327170.
[16] Mohanty S., Jagadeesh M., Srivatsa H., Big Data Imperatives: Enterprise Big Data Warehouse, BI Implementations and Analytics, APress, 2013, Print ISBN 978-1430248729.
[17] Petrovic S., Osborne M., Lavrenko V., Streaming first story detection with application to Twitter, International Conference of the North American Chapter of the Association for Computational Linguistics, 2010, Print ISBN 1-932432-65-5.
[18] Schroeder B., Pinheiro E., Weber W. D., DRAM errors in the wild: A large-scale field study, International Joint Conference on Measurement and Modeling of Computer Systems, 2009, Print ISBN 978-1-60558-511-6.
[19] Shaikh M. A., Jiaxin W., Investigative Data Mining: Identifying Key Nodes in Terrorist Networks, IEEE International Conference on Multitopic Conference, 2006, Print ISBN 1-4244-0795-8.
[20] Suraworachet W., Premsiri S., Cooharojananone N., The Study on the Effect of Facebook's Social Network Features toward Intention to Buy on F-commerce in Thailand, IEEE/IPSJ International Symposium on Applications and the Internet, 2012, Print ISBN 978-1-4673-2001-6.
[21] Tudorica B.G., Bucur C., A comparison between several NoSQL databases with comments and notes, Roedunet International Conference (RoEduNet), 2011, Print ISBN 978-1-4577-1233-3.
[22] Wang G., Tang J., The NoSQL Principles and Basic Application of Cassandra Model, Computer Science & Service System, 2012 International Conference, Print ISBN 978-1-4673-0721-5.
[23] Yang H. C., Dasdan A., Hasiao R. L., Parker D. S., MapReduce-Merge: Simplified relational data processing on large clusters, ACM Conference on Management of Data, 2007, Print ISBN 978-1-59593-686-8.
[24] Zheng Z., Zhu J., Lyu M. R., Service-Generated Big Data and Big Data-as-a-Service: An Overview, IEEE International Congress on Big Data, 2013, Print ISBN 978-0-7695-5006-0.

This is an extended version of the paper presented at the International Conference on Big Data Intelligence and Computing (DataCom 2015), Chengdu, China, December 19-21, 2015