Document Identification with MapReduce Framework

Yenumula B Reddy
Department of Computer Science
Grambling State University,
Grambling, USA
email: ybreddy@gram.edu

***Abstract*—Hadoop technology made a break through to process the unformatted data and generates the results faster than ever. Before Hadoop technology, the results were produced for formatted data using SQL and other techniques. They allowed to effectively sharing memory, central processing unit, disk, and network input/output more efficiently. There was no proper system to analyze unformatted data. The paper discusses the MapReduce framework to identify a required document from a stream of documents. We proposed an algorithm called MapReduce to detect sensitive documents that identify sensitive or required document among the streams of documents. The algorithm was tested using the Hadoop package and Java program. The results conclude that the Java program is useful for small documents. The Hadoop technology helps in stream of documents and produces the results much faster than simple Java program implementation.**

Keywords-MapReduce; Hadoop Distributed File Systems; Big data; key, shuffle; Apache Zookeeper.

I. INTRODUCTION

Big data (lower case is used for *Big data* in most of the places unless it is required to stress the word *Big*) is a general term used for a large volume of data that are structured, unstructured and semi-structured data created or generated by a company. This data cannot be loaded using any database models and it is not possible to get results with available query languages. That means that this data cannot be processed using traditional tools. The data is not a specific quantity in terms of a number of bytes (terra-bytes, petabytes or exabyte). It is continuously growing in size every minute. It is real big and grows in Exabyte. The origins are from a variety of platforms. That data volume changes quickly and its growth cannot be predicted. It is expensive to analyze, organize, and preparing as useful data. Therefore, we need special effort to prepare as meaningful by using new algorithms, special hardware and software. There may not be a single tool to analyze and transform big data as meaningful data. The tools may vary to analyze data. The big data management needs a generating high quality of data to answer the business queries.

The primary goal is to discover the repeatable business patterns, uncover the hidden patterns, and understand the unknown correlations and other useful information. The business giants have access to the information, but they do not know to get the value out of this unstructured data. The traditional tools are not helpful due semi-structured storage. The big data analytics may commonly use software tools such as predictive analytics [16] and data mining [17]. The technology associated with big data analytics includes NoSQL databases [21], Hadoop [2] and MapReduce [4]. These are open source frameworks that support processing of big data. The analysis needs MapReduce to distribute the work among a large number of computers and process in real-time. Hadoop structure lowers the risk of catastrophic system failure, and even when a significant number of nodes become inoperative.

The government and information technology managers are highly motivated to turn the massive data into use. Today, Hadoop framework and MapReduce, offer new technology to process and transform Big data into meaningful data. It is required to deploy the infrastructure differently to support the distributed processing and meet the real-time demands. The Information Technology (IT) managers found that security and privacy are the major problems, particularly while using third-party cloud service providers [15].

The structured and unstructured data come from a variety of sources. Adoption of big data tools to process the big data is increasing. High priority is given to improving the big data formalizing and processing. The top data for transactions include business data documents, email, sensor data, image data, Weblogs, Internet search indexing, and attached files. The IT group found an interest in learning about technology, deploy the packages to process the data, and adopt the infrastructure for better performance and affordable cost. They are in the process of implementing Apache Hadoop frameworks, and commercial distributions of the Hadoop distributed framework and other NoSQL databases.

Currently, priority is given to processing and analyzing unstructured data resources including Weblogs, social media, e-mail, photos, and videos. Unstructured emails are given priority to analyze and process. As a first step, the IT staff is working on batch processing and move to real-time processing. The companies are concerned about the scalability, low latency, and performance in storing and analyzing the data. Further, they are worried about protection of data for third-party cloud providers. Standards are required for data privacy, security, and interoperability for data and systems.

ISBN: 978-1-61208-358-2

The big data are useful when analyze and store in a meaningful way, so that the data can be accessed immediately. The main goal is to store the data to ensure that the data are accessible for business intelligence and deepanalysis. It is required to design a tool to analyze the data and provide the answers with minimum efforts and time. The challenges include the size, structure, origin, variety, and continuous change in data. The data are real big in size (terra-bytes or exa-bytes) and unstructured. Data contains text, figures, videos, tweets, Facebook posts, website clicks, and different types of data from a variety of websites. The origins are varied and come from a variety of platforms and multiple touch points. Data change fast in terms of format, size, and types of websites. Further, specialized software is required to pull, sort, and make the data meaningful. There is no universal solution to make such data meaningful. Further, the data are produced in universally available languages. Processing such data may need separate algorithms (depending upon the data). There are many predictions in years to come about the data coming from an unknown source and unstructured in nature. The predictions include the standardization and marketing. The following predictions include the data origin, type, size and management [23][24].

- Massive data collection across multiple points
- Firmer grip on the data collected by different groups
- Generalization of format for Internet data and/or data generated by business media
- Entering of social media as part of big data generation

The main purpose of the data management (analyzing and processing) is to make sense of the collected data from various data collection points. Making sense of data means that the end point of the processing of data must be able to answer the business queries. The queries include data mining related predictions, business queries, and management assistance.

The Hadoop project was adopted more in the Department of Defense than in other agencies. These agencies could not use the Hadoop system design because the Hadoop design lacks reusability due to Java Application Programming Interfaces for data access in Apache. Research is required in understanding the Hadoop system design, security models, and usage in a specific application. Protocol level modifications help in improving the security at source level.

Federated systems enable collaboration of various networks, systems and organizations at different trust levels. Clients must be separated from service with authentication and authorization procedures. Existing security models protect the resources within the boundary of the organization. In federated systems, new participants join and leave continuously. They may not all be trusted. Therefore, federated systems require the security specification for each function. Depending upon the system sensitivity level the boundary constraints are incorporated. Individual protection domains are required for each entity. Therefore, the existing security domain procedures do not work in federated systems.

The federated systems are distributed. The security system in federated systems separates the client access, authentication, and authorization. Therefore, they need collaboration between networking, companies and associated systems. Due to the involvement of many entities in the federated systems, it is difficult to maintain the security among these entities. The threat may be expected (unavoidable) from a variety of sources. Hence, a high-level coarse-grained security goals need to be specified in the requirements.

In this paper, we discuss the Hadoop single node installation and introduced an algorithm to identify the sensitive documents. Section 2 discusses the current state of Hadoop distributed file system. MapReduce programming model is discussed in Section 3 and implementation in Section 4. Finally, Section 5 discusses the conclusions and future work.

II. HADOOP DISTRIBUTED FILE SYSTEMS – CURRENT STATUS

Data backup in Hadoop file systems using snapshots was discussed by Agarwal et al. [1]. The authors designed an algorithm for selective copy on appends and low memory overheads. In this work, the snapshot tree and garbage collection was managed by Name-node. The architecture of Hadoop Distributed File Systems (HDFS) and the experiences to manage 25 petabytes of Yahoo data was discussed by Shvachko et al. [2]. The fault tolerant google file system running on inexpensive commodity hardware with aggregate performance to a large number of clients was discussed by Ghemawat et al. [3]. The paper discusses many aspects of design and provides a report of measurements from both micro-benchmarks and real world use. The MapReduce programming model was explained and it’s easy to use functions were discussed by Dean and Ghemawat [4]. The document discussed the experiences and lessons learned in implanting the model. Further, it discusses the impact of slow machines in redundant execution, locality of optimization, writing single copy of the immediate data to local disc, and saving the network bandwidth. Chang et al. [5] discussed the dynamic data control over data layout and format using ‘Bitable’ a flexible solution. They claimed that many projects are successfully using this model by adding more machines for process over the time.

The federated security architecture was discussed in Windows Communication Foundation (WCF) [6]. WCF incorporates the security in federated systems to build and deploy the federated systems. The architecture of these federated systems includes the federation, domain/realm, security token service, and consists of primary security architecture. The current mobile devices were implemented with imitation of 60K tasks, but the next generation of Apache MapREduce supports 100K concurrent tasks [7]. In MapReduce, users specify the computation in terms of the map, and a reduce function. The available software modules in MapReduce automatically paralyze the computation and schedule the parallel operations with the help of network and computational facilities. These facilities help to complete the operation much faster. Every day, an average of a hundred thousand MapReduce jobs are executed on Google clusters [4]-[7].

Halevy et al. [8] discussed that a nonparametric model is needed to represent the data in large data sources. They believe that a nonparametric model holds a lot of details compared to a parametric model. The authors believe that the selection of unsupervised learning on unlabeled data generates better results than on labeled data. Halevy et al. [8] pointed out that future research includes the creation of specific data sets by automatically combining data from multiple tables. Combing data from multiple tables also includes unstructured Web pages or Web search queries. Thuraisingham [9] discussed various types of security policies including local, generic, component, export and federate. The security policy generation enforcing also included in this study.

Hadoop security was discussed in the reports [10]-[15]. Reddy [10] proposed the security model for Hadoop systems at access control and security level changes depending upon the sensitivity of the data. Authentication and encryption are the two Security levels for big data in Hadoop. Ravi [11] concludes that Kerberos files keep the intruders away from accessing the file system and Kerberos system has better protection compared to other federated systems. Chary et al. [12] discussed the current level of security in Hadoop distributed file systems that include the client access for Hadoop cluster using Kerberos Protocol and authorization to access.

O'Malley et al. [13] and Das et al. [14] discussed the security threats from different user groups in Kerberos authentication system. The research in Kerberos and MapReduce implementation details also discusses the security, role of delegation token. The research in [13] and [14] emphasizes the need for security in internal and external access level for Hadoop systems. The work describes the need for limits of access rights to specific users by application, isolation between customers, information protection and incorporation of encryption models.

Srinath [15] presented "Airavat", a MapReduce-based prototype system, provides strong security, privacy and guarantees for distributed computations of sensitive data. The model described in "Airavat" explains how to use different parameters, estimate their values, and test on several different problems. This system does not follow the software engineering methodology. Therefore, it has weak use cases and complicated processes in specifying the parameters. Since MapReduce computations are not efficient, organizations raised critical questions on privacy and trust of data during the MapReduce computations.

Preserving the privacy in big data was discussed by McSherry [16]. McSherry's Privacy integrated queries (PINQ) presents an opportunity to establish a more formal and transparent basis for privacy technology. The algorithms designed help the users in increasing the privacy-preserving and increases the portability. Partha et al. [17] presented a system that learns for data-integrated queries which use sequences of associations. The associations include foreign keys, links, schema, mappings, synonyms, and taxonomies. They create multiple ranked queries linking the matches to keywords. These queries are linked to Web-forms and users have only to fill the keywords to get answers. MapReduce application was used for integer factorization [18], Matrix computation [19], and machine learning on multicore systems [20]. None of these papers discussed the detection of sensitive data files among the streams of data files. This paper introduces an algorithm called **M**apReduce **A**lgorithm to **D**etect **SE**nsitive **D**ocuments (MADSED).

III. MAPREDUCE PROGRAMMING MODEL

One of the programming models in MapReduce is breaking the large problem into several smaller problems. The solutions to the problems are then combined using the MapReduce function to give a solution to the original problem. The functionality is similar to software engineering top-down design. There are many questions that arise in MapReduce application due to dynamic input, nodes may fail, number of smaller problems may exceed the number of nodes, dependable sub-problems, distribution of input to the smaller jobs, coordination of nodes, and synchronization of completed work. MapReduce programming model and the associated implementation can be used to solve these problems by processing and generating large data sets.

MapReduce application divides into three parts: Map, Shuffle and Sort, and Reduce. A Map part of MapReduce job splits the input data-set into independent chunks. The independent chunks are processed in a completely parallel manner using Map task. A given input pair can have zero or more output pairs. The map-outputs are merged and constructed with respect to the key values. These pairs are propagated to reduce function. The reduced function then merges these values to form a possibly smaller set of values. That is, the reduced function filters the map output and produces the results with respect to key. The total functionality includes scheduling the tasks, monitoring them, and re-executes the failed tasks.

The mapper breaks down the problem into smaller bits. These bits are processed parallel to produce a solution. The formula (1) produces new key values.

$$\forall(k,v)\rightarrow(k^{'},v^{'}) \tag{1}$$

where, k = key, v = value. These values are used for searching the keywords in another semantic domain to produce intermediate values (document identification and document) for each call. In Figure 1a, A is document identification and α is a document. The mapper then combines related keys and prepare the partitions to search in the documents. The map-outputs are sorted, merged and constructed with respect to the key values in the shuffle step. The aggregated values are filtered and then return a new set of results in the reduce phase. After completion of reducing phase, it returns all possible values with respect to a key. The process is shown in Figures 1a [21] and 1b [22].

For example, group the words of same length in the statement "*The Big data is useful if we analyze and stored in a meaningful way so that the data can be accessed quickly*". The MapReduce framework groups all of the values by key (each word). The Table I shows the output of key (the number of letters in a key and key word) called value pairs. The same procedure will be applied for each document (one document or multiple documents). Each document will be treated separately at the time of process in MapReduce function.

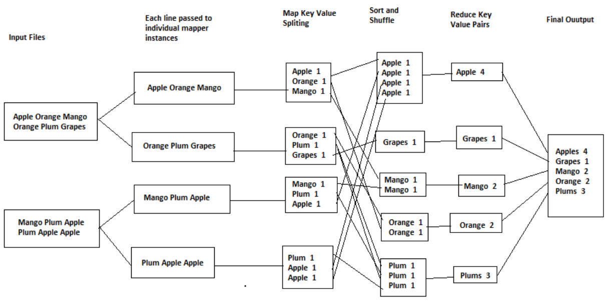

Figure 1a. Map, Shuffle, and Reduce phase example.

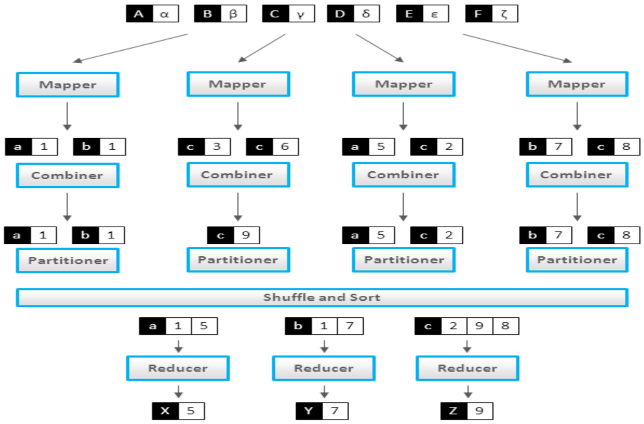

Figure 1b. Map, Shuffle, and Reduce phase example.

TABLE I. KEY AND WORD

3: The 3: big 4: data 2: is 6: useful 2: if 2: we	7: analyze 3: and 6: stored 2: in 1: a 10: meaningful 3: way	2: so 4: that 3: the 4: data 3: can 2: be 8: accessed 7: quickly

In Table I, the keys are grouped according to the number of letters in a word (1 letter word or 2 letter word, etc.) and figure out number of items in each key. The number of items in each key (1 letter word or 2 letter word etc.) is shown in Table II. The reduce function counts the number of items with key size of the list (Table III). The reduction can be done in parallel using Graphics Processing Units (GPUs).

TABLE II. WORDS RELATED TO EACH KEY

1: a
2: is, if, we, in, so, be
3: the, big, , way, the, can
4: data, and, that, data
6: useful, stored
7: analyze, quickly
8: accessed
10: meaningful

TABLE III. KEY WITH THE SIZE APPEARS IN THE LIST

1: 1
2: 6
3: 5
4: 4
6: 2
7: 2
8: 1
10: 1

The algorithms for Map, Shuffle, and reduce are given below. The MapReduce algorithm to generate Table I, II and III for all documents has three phases namely, mapper, combiner, and reducer.

```
class Mapper
  method Map (Document-id id, document d)
       for each term t in document d
              store the term and its size

class Combiner
   method combine (term t, [c1, c2, …])
      sum =0
      for each term t in list [c1, c2, …]
              append to list of same size terms
      complete for different size terms

class Reducer
  method Reduce (term t, counts [c1, c2, ….])
    for count c in [c1, c2, …]
    count the same size terms and store number of
    occurrences of each term in the document
repeat this for all terms
```

The algorithm generates the tables similar to Table I, II, an III by using this algorithm from the given document or documents. This algorithm is enough to find the number of occurrences of each term. We need the extension of the algorithm to detect the importance or sensitivity of the document. The algorithm for detecting the required documents among the documents with the help of keys and their weights is called MapReduce Algorithm to Detect Sensitive Documents (MADSED).

MADSED Algorithm

Let us assume that the key sizes are: 3, 5, 6, 7 letters

- Divide the document into N sub-documents
- Filter each sub-document by leaving only words of sizes 3, 5, 6, and 7
- Count # of times each key word appears in the sub document
- Shuffle the sub document results into single output with number of times the keyword appears
- Calculate value of each key word by multiply each keyword by its weight and times appear
- Add each keyword output values as result
- If the result (output) is greater than threshold value, the document is important; if it is boarder on threshold, it is for consideration; otherwise reject
- End of algorithm

The Algorithm was implemented using Hadoop technology and Java program implementation.

IV. IMPLEMENTATION

The Hadoop 1.2.1 from the Apache Website was installed on Linux operating system [25]. The JavaTM 1.7.x was installed as part of the installation [26]. Initially, we tested a single node installation. It has login and password protection. We used a text file from the Hadoop folder by using the commend (2).

```
bin/hadoop dfs -copyFromLocal
/home/csadmin/Downloads/gutenberg /user/csadmin/Gutenberg.
                                                        (2)
```

The commands and their usage are available at http://hadoop.apache.org/docs/r2.3.0/hadoop-project-dist/hadoop-common/FileSystemShell.html

For more information http://www.apache.org/

HDFS as a general DFS for applications are available at http://www.opensourceforu.com/2013/12/peek-hadoop-distributed-file-system/

To make sure the file is copied successfully use the following command

```
bin/hadoop dfs -ls /user/csadmin
```

The command to run the MapReduce is

```
bin/hadoop jar hadoop-examples-1.2.1.jar wordcount
/user/csadmin/gutenberg /user/csadmin/gutenberg-output
```

Finally, the output generated by MapReduce is checked. It generates all words and number of times each word repeated. Table IV shows the sample output of first few words.

TABLE IV: SAMPLE OUTPUT OF MAPREDUCE

"(Lo)cra"	1
"1490	1
"1498," 1	
"35"	1
"40,"	1
"A	2
"AS-IS".1	
"A_	1
"Absoluti	1
"Alack!	1
"Alack!"1	
"Alla	1

As a next step, we used the keywords and stored only those keywords and number of times each keyword repeated from the generated output. Then, we multiplied each keyword with its weight and number of times repeated and adds all the resulted values. For example, the keywords are Oil and Alaska. If the weight for oil = 0.02 and Alaska is 0.1, the oil repeats 15 times in the document and Alaska repeats 20 times in the document. The total value is 0.02*15 + 0.1*20= 0.3+2.0=2.3. If the total value greater than the threshold value, then the document is retrieved (*the threshold value is established by the programmer depending upon the expected minimum value required to select the document*). The algorithm was implemented using Java program and generated the results. The results produced through Java program in the current case (to select a required document) are compatible (means document is selected or rejected) to Hadoop results. The Java program is good for small files and takes more time in case of large files. Hadoop technology produces the results much faster and recommended for large files. The algorithm is useful for detecting the sensitive data or files whenever a stream of files is entering in the organization. The algorithm separates the sensitive files from a large set of files. Further, the process helps to retrieve required files among thousands of files.

V. CONCLUSIONS AND FUTURE RESEARCH

The paper discussed the MapReduce framework to identify an essential document from a stream of documents. The research discusses the current state of Hadoop distributed File Systems, MapReduce programming model, and presents an algorithm to identify sensitive or required document among the streams of documents. The algorithm was tested using the Hadoop package and Java program. The results conclude that the Java program is useful for small documents, whereas Hadoop helps in large and a stream of documents. Further, Hadoop produces the results must faster than simple Java program implementation.

The future research involves testing the documents with Hadoop multimode (installed and tested) and Hadoop with Graphics Processing Unit (GPU) based technology. The work is underway and results will be available soon. Big data security is another problem that is not discussed in this paper.

ACKNOWLEDGEMENTS

The research work was supported by the AFRL Collaboration Program – Sensors Research, Air Force Contract FA8650-13-C-5800, through subcontract number GRAM 13-S7700-02-C2.

REFERENCES

[1] S. Agarwal, D. Borthakur, and I. Stoica, "Snapshots in Hadoop Distributed File System", UC Berkeley Technical Report UCB/EECS, 2011.

[2] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop Distributed File System", 26th IEEE (MSST2010) Symposium on Massive Storage Systems and Technologies, May, 2010, pp. 1-10.

[3] S. Ghemawat, H. Gobioff, and S. Leung, "The Google File System", SOSP'03, October 19–22, 2003, pp. 29-43.

[4] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Proc. 6th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2004, San Francisco, USA, Dec. 2004, pp. 107-113.

[5] F. Chang et al., "Bigtable : A Distributed Storage System For Structured Data", ACM Transactions on Computer Systems (TOCS), vol. 26, Issue 2, June 2008, pp. 204-218.

[6] Authentication, Microsoft Patterns & Practices, http://msdn.microsoft.com/en-us/, 2012 [Retrieved: July 27, 2014].

[7] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters", CACM 50th anniversary issue, vol. 51, issue 1, Jan 2008, pp. 107-113.

[8] A. Halevy, P. Norvig, and F. Pereira, "The Unreasonable Effectiveness of Data", IEEE Intelligent Syst., 2009, pp. 8-12.

[9] B. Thuraisingham, "Security issues for federated database systems", Computers & Security, 13 (1994), pp. 509-525.

[10] Y. B. Reddy, "Access Control for Sensitive Data in Hadoop Distributed File Systems", Third International Conference on Advanced Communications and Computation, INFOCOMP 2013, November 17 - 22, 2013 - Lisbon, Portugal, pp. 72-78.

[11] P. Ravi, "Security for Big data in Hadoop", http://ravistechblog.wordpress.com/tag/Hadoop-security/, April 15, 2013 [Retrieved: July 27, 2014] .

[12] N. Chary, K. M. Siddalinga, and Rahman, "Security Implementation in Hadoop", http://search.iiit.ac.in/cloud [retrieved: January 2013].

[13] O. O'Malle, K. Zhang, S. Radia., R. Marti, and C. Harrell., "Hadoop Security Design", http://techcat.org/wp-content/uploads/2013/04/Hadoop-security-design.pdf, 2009, [Retrieved: July 27, 2014].

[14] D. Das, O. O'Malley, S. Radia, and K. Zhang, "Adding Security to Apache Hadoop", Hortonworks Technical Report 1, http://www.Hortonworks.com, 12 pages.

[15] I. Roy Srinath, T.V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel, "Airavat: Security and Privacy for MapReduce", 7th USENIX conference on Networked systems design and implementation (NSDI'10), 2010, Berkeley, CA, pp. 1-16.

[16] F. McSherry, "Privacy Integrated Queries: An Extensible Platform for Privacy-Preserving Data Analysis", Proceedings of SIGMOD, 2009, pp. 19-30.

[17] P. P. Talukdar et al. "Learning to Create Data-Integrating Queries", VLDB, 2008, pp. 785-796.

[18] J. Tordable, "MapReduce for Integer Factorization", arXiv, arXiv:1001.0421v1 [cs.DC], January 4, 2010. [Online]. http://arxiv.org/abs/1001.0421v1

[19] S. Seo, E. J. Yoon, J. Kim, S. Jim, J. Kim, and S. Maeng, "HAMA: An Efficient Matrix Computation with the MapReduce Framework", IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), 2010, pp. 721-726

[20] C. Chu et al., "Map-Reduce for Machine Learning on Multicore", Advances in Neural Information Processing Systems 19 (NIPS 2006) pp. 281-288.

[21] Highly Scalable Blog, Articles on Big data, NoSQL, and Highly Scalable Software Engineering, MapReduce Patterns, Algorithms, and Use Cases, http://highlyscalable.wordpress.com/, Posted on February 1, 2012 .

[22] K. S. Bejoys, "Word Count - Hadoop Map Reduce Example", http://kickstarthadoop.blogspot.com/, April 29, 2011 [Retrived: July 27, 2014] [Retrieved: July 27, 2014].

[23] E. Namey, G. Guest, L. Thairu, and L. Johnson, "Data Reduction Techniques for Large Qualitative Data Sets", Handbook for team-based qualitative research, pp. 137-161.

[24] J. Karlsrud, "Peacekeeping 4.0: Harnessing the Potential of Big Data, Social Media and Cyber Technology", in Kremer, J. F. & Müller, B. (eds.) Cyber Space and International Relations. Theory, Prospects and Challenges. Berlin: Springer,2013, pp. 141-160.

[25] http://hadoop.apache.org/docs/r1.2.1/ 2011 [Retrieved: July 27, 2014].

[26] http://en.softonic.com/s/java-1.7 [Retrieved: July 27, 2014].