Big Data Issues and Processing Techniques: A Comprehensive Survey

Aisha Shabbir, Kamalrulnizam Abu Bakar and Raja Zahilah Raja Mohd. Radzi
School of Computing
University Technology Malaysia, Johor, Malaysia
saisha3@graduate.utm.my, knizam@utm.my, zahilah@utm.my

Sulaiman Azeem
Department of Electronic Engineering
Capital University of Science and Technology
Islamabad, Pakistan
xunny91@gmail.com

Abstract
Big data and its analysis are in the focus of current era of big data. The main production sources of big data are social media like Facebook, twitter, emails, mobile applications and the migration of manual to automatic of almost every entity. Currently, there is a need to investigate and process complex and huge sets of information-rich data in all fields. This paper provides a survey of big data issues and the effectual and efficient platforms and technologies which are needed to deal and process the remarkable amount of data. It revolves around two important areas namely: clustering and scheduling.

Keywords
Big data issues, Big data issues, Clustering, Scheduling

1. Introduction

The landscape of science is changing dramatically. The far reaching development of digital technology in many fields i.e. social media, person to person communication and scientific analysis has contributed in the generation of huge amount of data which is collectively known as 'Big Data'. Such data is growing enormously leading to difficulties in storing, processing and analyzing by using traditional databases and conventional tools and techniques. The processing issue is beginning to see an attainable horizon, yet at the same time has space to evolve. Thus the vital issue is not to store all of the data produced but to extract meaningful information and process it efficiently on the given resources.

To address these issues, big data analytics is there to mine untapped information from huge volume and variety of data. There are some proposed platforms to cope up the issues of the Big data like Dryad, Spark, Dremel and Pregel (Agneeswaran, 2014), Storm, and Hadoop MapReduce. One of the most successful frameworks proposed was MapReduce. It was initially proposed by Google. It is especially designed for processing big data by exploiting the parallelism among a cluster of machines.

The aim of this paper is to provide a comprehensive survey about the big data issues, platforms and techniques to solve the processing issues of big data clusters. The processing techniques are discussed in detail. Furthermore, this study provides the comparison for the algorithms of the processing technique. Thus, enables the reader to choose the appropriate selection of the technique according to the available resources.

The organization of the paper is as follows: Different types of big data issues is discussed first. Next to it, platforms and techniques for big data processing will be enlightened. The platform and the big data processing techniques are given in an explanatory manner with its further types. In addition, main results and comparison are also given in form of tables. In last, conclusion and future directions will be discussed.

2. Big Data Issues

3. Data Storage Issues

The amount of information has burst out each time and thus need for invention of a new storage method. For handling huge volume storage, Big Data storage companies such as IBM, EMC Amazon utilizing the tools like Apache Drill, SAMOA, NoSQL, IKANOW, Hadoop and Horton Works (Lomotey & Deters, 2014) and (Saraladevi, Pazhaniraja, Paul, Basha, & Dhavachelvan, 2015).

3.1 Data Management Issues

The data generation sources are different and thus the data also both by means of format and in terms of collection. People contribute digital information in a way which are suitable for them like archives, illustrations, pictures, audio and video messages and so forth. However, the collected information is promptly accessible for investigation and examination. Furthermore, data and its provenance will become a serious issue. As indicated by Gartner, Big Data challenge involves more than just managing volumes of data mentioned in his article (Gartner, 2013).

3.2 Data Processing Issues

To understand the Big Data processing issue, let us consider an example for which Exabyte of data has to be processed at a time. Divide the data into the 8 blocks i.e. 1 Exabyte would be equal to 1k petabyte and the processor uses 100 instructions per block at 5 gigahertz. Thus the processing time for end-to-end will be 20 nanoseconds. So, 1K petabytes processing will need approximately 635 years' time for end-to-end processing.

3.3 Security Issues

It is challenging to manage a large data set in secure means. Further, public and private database and inefficient tools comprise many threats. The security issues occurs for distributed systems when huge measure of private information put away in a database which is not properly encoded and encrypted.

4. Techniques To Deal With Big Data

Big data and its analysis becomes the focal point of recent modern era. Though, the amount of data to be investigated increases on one side, the demand for acceptable time to yield outcomes is shrinking on the other hand. There is a need of efficient techniques and platforms to store, process and analyze the complex and gigantic sets of information-rich data in all fields nowadays. Some of the techniques are discussed below.

4.1 Big Data Clustering

Big Data clustering is an emerging technique for big data analytics and facilitating big scale data management, exploration and processing of huge of big data (Thomas & Leiponen, 2016). The clustering process consists of dividing the un-labeled data entities in different sets. This technique have been exploited in many fields such as machine learning, data mining, bioinformatics and biochemistry (Venkataraman, Panda, Ananthanarayanan, Franklin, & Stoica, 2014). Depending on the nature of the data or the purpose of the cluster, various types of cluster algorithms have been developed as shown in Figure 1.

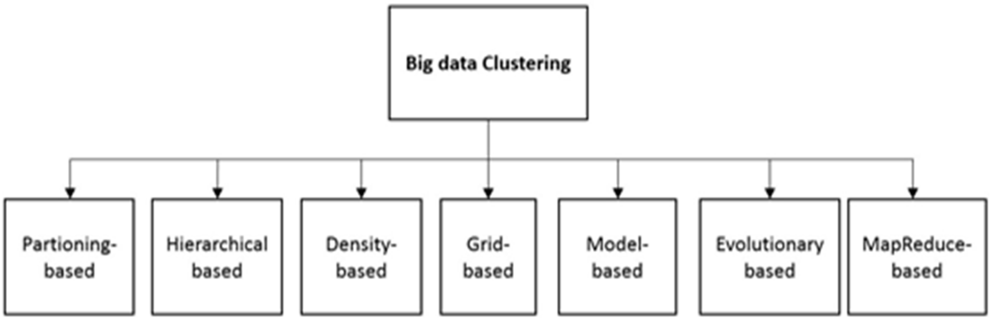

Figure. 1 – Big data clustering types

4.2 Scheduling

Scheduling is one of the processing scheme which deals with the ordering, prioritization and assignment of tasks to the relevant machines for execution. Scheduling is done by assigning the tasks to suitable processors in a specific order. To find an optimal schedule for homogenous or heterogeneous environments is termed as an NP-complete problem (Tang, Liu, Ammar, Li, & Li, 2016).

4.2.1 Scheduling Algorithms for homogeneous systems

Many researchers proposed scheduling algorithms by incorporating various issues as first in first out (FIFO) scheduler was presented to take a shot at the possibility of first start things out serve premise. In this scheduler, all employments are given to a solitary queue, when a Task Tracker's pulse arrives, the scheduler just gets a job from the head of the queue, and tries to appoint the undertakings of the activity. The upside of the FIFO scheduler is that the algorithm is very simple, the heap of the Job Tracker is generally low. So it can bring about starvation when the little jobs come after the extensive jobs (Hadoop, 2009) .

Fair scheduler was presented to ensure the fair distribution of cluster resources such that all of the jobs ought to hold equal cluster mutual resources with the progress of time. It accomplish jobs and allocate the jobs to every pool in a way to maintain fairness of the percentage distribution of the sources. The benefits of this scheduler are that different kinds of jobs will have distinct sources assigned by the cluster, thus, it is going to enhance the high-quality of the service. The negative end is that it does not keep in mind the actual loading state of every nodes, so the load stability in every node may not that precise (Hadoop, 2009).

Next, capacity scheduler stepped forward the feature of HOD (Hadoop On Demand), and encountered the drawbacks of HOD. Capacity scheduling algorithm makes use of numerous queues, each queue get its sources in keeping with the computing potential, and the rest sources will be to the ones queues that have not met its usage limits, and allocate them base on the load of its calculation, which is greater affordable. In case that all the assets of the queue are occupied via one job, the scheduler restricts the assets of each user within the queue. The advantage of the scheduler provisions job priority and can run in parallel to assign sources dynamically and thus improves the efficiency. Its shortcomings raised by the requirement of extra records of all the jobs thus making capacity scheduling algorithm more complex and costly (Zaharia et al., 2009).

Next, a scheme named as Delay algorithm was proposed in (Zaharia et al., 2010). It was proposed to overcome the drawbacks of FIFO scheduler. The mechanism of the scheduler relies on the idea of fair distribution of resources in a cluster. For this, it allocates a specific time to each job so that it get equal resources. Thus in this way the proposed idea creates a long delay for certain type of jobs. Another problem of the scheduler was that if it does not find a local job in predefined time, then it launches a non-local job in the multiple slots.

Further, a Learning scheduler was proposed in (Loshchilov, Schoenauer, & Sebag, 2013). This scheduler examines the jobs and recognize it as great or awful. Plan classifier organizes the job employments. According to the asset use, steady jobs will consider for additionally taking care of. A good job does not make any over-weight to the Task Trackers. Awful jobs will be disallowed. The scheduler considers CPU utilization, memory use, IO utilization and network usage. If more than one great job find in the activity queue, job will be picked by the new expected utility limit.

Moreover, in (Nita, Pop, Voicu, Dobre, & Xhafa, 2015) a scheduling scheme is proposed via providing some flexibility to the clients. The algorithm allows the customers that have greater than one requests to adjust the priority. Thus the allocation of the assets would be granted in accordance with the requirements and the priority demands. In addition, it facilitates the clients to cut back the tasks when the demand is greater. But the proposed mechanism became expansive and works nicely for the homogeneous surroundings. The comparison of the discussed algorithms are shown in Table 1.

Table 1 -Comparsion of Big Data Scheduling Algorithms

Author	**Algorithm Name**	**Advantages**	**Mode**
Apache Hadoop, (2009)	FIFO	Easy to understand and equally easy to program	Non Preemptive
Apache Hadoop, (2009)	Fair	Fair distribution of resources.	Preemptive
(Zaharia et al., 2009)	Capacity	Effective utilization of the resources by utilizing the idle assets of cluster.	Non Preemptive
(Zaharia et al., 2010)	Delay	Simplicity of scheduling	Preemptive
Loshchilov et al.(2013)	Learning	Effective CPU and memory utilization	Preemptive
(Nita et al., 2015)	MOMTH	Flexibility to clients for priority setting	Preemptive

4.2.2 Scheduling Algorithms Handling Straggling Issue

It has been seen generally, that the performance of a big data computing cluster is degraded sometimes due to the in-completion of one or a number of tasks. These slow tasks are termed as stragglers and the phenomenon of this delay is called straggling. The initial Google MapReduce framework just starts to dispatch standby tasks when a job is near to finishing point. It has been demonstrated that speculative execution can diminish the activity service time by approximately 44%. After that, Longest Approximate Time to End (LATE) scheme was proposed in (Zaharia, Konwinski, Joseph, Katz, & Stoica, 2008) which measures the progress rate for the tasks. The Late scheduler was intended to address this unusual terminal of the task. The proposed algorithm measure the progress rate by the completion time of task and providing the backups of some of the tasks according to fractional ratio of the running phase. Major flaw of this set of rules became that it really works for the slow tasks and become unable to break the one kind of phases of MapReduce at some stage in its progression.

Furthermore, the MANTRI was proposed in (Ananthanarayanan et al., 2010) by developing a speculative scheme which monitors the finish time of each task and detects a slow task but re-allocates the detected task only if completion time will decrease. MANTRI lacks in encountering the tradeoffs between the average job flow time and the cost. Next to it, in (Ananthanarayanan, Ghodsi, Shenker, & Stoica, 2013) a cloning scheme named as DOLLY has been put forward to tackle the stragglers in a proactive manner. Rather than cutting the stragglers it launches multiple clones. Thus this scheme results into a drawback due to a considerable overhead. Further, a smart speculative execution technique is presented in (Qi Chen, Liu, & Xiao, 2014) which has scheme two main themes. First one was to use the exponential weightage of moving average for the prediction of stragglers and the other is to back-up the slower task by encountering the cluster load. It has a limitation that the scheme considers only the optimization of task level rather than job level performance. Then, in (Q. Chen et al., 2014) the extended scheme of (Ananthanarayanan et al., 2010) has been produced and named as Grass for handling the jobs by encountering the normal jobs rather than small. But, still is lacking in encountering the long jobs.

A straggler handling scheme was proposed in (S. Chen et al., 2014) which perform the redundant requesting of storage codes and then analyzing it while the service time for as task shows an exponential growth. Furthermore, in (Ren, Ananthanarayanan, Wierman, & Yu, 2015) a speculative based scheduling algorithm named as Hopper is proposed. The scheduling algorithm i.e. Hopper makes the virtual size of the each task bigger than its actual size. It allocates the resources on the bases of the virtual size to detect the stragglers. Upon the detection, it re-schedules the whole virtual size data to the other nodes. The downside in its works is the wastage of the resources not only by allocating the resources more than the needed but also on the point of resolving the issue. Next, Self-Adaptive scheduling scheme was proposed in (Quan Chen, Zhang, Guo, Deng, & Guo, 2010). It distinguishes the slower and faster tasks merely and does not provide any effective solution to handle the stragglers.

Further, a scheme named as Failure-Aware, Retrospective and Multiplicative Speculation (FARMS) is proposed in (Fu, Chen, Zhu, & Yu, 2017). FARMS measures the responsiveness of each node. In case of detection of the unresponsive node, it copies the whole tasks running on that node to the other node. The main flaw of the FARMS is that it replicates the whole tasks while the cause of the un-responsiveness might be one or more tasks not the whole number of tasks running on the given node. Therefore, it produces more replication overhead with the wastage of resources. Recently, an optimization for speculative execution scheme to mitigate stragglers is presented in (Xu & Lau, 2017). However the presented scheme degrades the system performance badly by posing a limit on the dimensions of the facts and a ratio of stragglers just cut down underneath the threshold edge. In addition, along with the above scheme it also lacks in encountering the data-dependency among the tasks. The comparison of the straggling handling algorithms is shown in Table 2.

Table 2 -Comparsion of Straggling Handling Algorithms

Author	**Algorithm Name**	**Method**	**Replication Overhead**	**Execution Time**
Google, (2008)	GMRS	-	High	No
Zaharia et al. (2008)	LATE	Reactive	Moderate	No
Ananthanarayanan et al. (2013)	MANTRI	Reactive	Moderate	Yes
Chen et al., (2010)	SMAR	Reactive	High	No
Sun et al., (2012)	ESAMR	Reactive	Low	Yes
Ananthanarayanan et al. (2013)	DOLLY	Proactive	High	No
Ananthanarayanan et al. (2014)	GRASS	-	Moderate	Yes
Ren et al. (2015)	HOPPER	Reactive	High	No
Fu et al., (2017)	FARMS	Proactive	Highest	-

5. Conclusion

Big data and its analytics is still in its initial stage but evolving at a great speed. The vast sums of data are a consequence of jobs from different variety, which generally stored within big data clusters. Big data clustering and scheduling are appearing one of the active research areas which plays a vital role in the completion of big data processing and effectively utilizing the cluster resources. To conclude, viable processing of Exabyte's of information would require broad parallel handling with a specific end goal to give convenient and significant data. MapReduce based clusters has been appearing as the most effective big data solutions. It empowers parallelism along with several low-end computing nodes. In addition, there is no common task scheduling algorithm to accommodate and handle different type of jobs at the same time. For the future research direction, the big data platform should be able to handle different type of jobs for the processing of big data.

Acknowledgement

We are thankful to University Teknologi Malaysia (UTM), for providing research environment, tools and technical support to accomplish this work.

References

Agneeswaran, V. S. (2014). Big data analytics beyond hadoop: real-time applications with storm, spark, and more hadoop alternatives: FT Press.

Ananthanarayanan, G., Kandula, S., Greenberg, A. G., Stoica, I., Lu, Y., Saha, B., & Harris, E. (2010). Reining in the Outliers in Map-Reduce Clusters using Mantri. Paper presented at the Osdi.

Ananthanarayanan, G., Ghodsi, A., Shenker, S., & Stoica, I. (2013). Effective Straggler Mitigation: Attack of the Clones. Paper presented at the NSDI.

Ananthanarayanan, Ganesh, Michael Chien-Chun Hung, Xiaoqi Ren, Ion Stoica, Adam Wierman, and Minlan Yu.(2014) "Grass: Trimming Stragglers in Approximation Analytics."

Chen, Q., Liu, C., & Xiao, Z. (2014). Improving MapReduce performance using smart speculative execution strategy. IEEE Transactions on Computers, 63(4), 954-967.

Chen, S., Sun, Y., Kozat, U. C., Huang, L., Sinha, P., Liang, G., Shroff, N. B. (2014). When queueing meets coding: Optimal-latency data retrieving scheme in storage clouds. arXiv preprint arXiv:1404.6687.

Chen, Q., Zhang, D., Guo, M., Deng, Q., & Guo, S. (2010). Samr: A self-adaptive mapreduce scheduling algorithm in heterogeneous environment. Paper presented at the Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on.

Fu, H., Chen, H., Zhu, Y., & Yu, W. (2017). FARMS: Efficient mapreduce speculation for failure recovery in short jobs. Parallel Computing, 61, 68-82.

Gartner. “Gartner Survey Reveals That 73 Percent of Organizations Have Invested or Plan to Invest in Big Data in the Next Two Years”. (2013) STAMFORD, Conn

Hadoop, A. (2009). Hadoop. In./Fifo.

Hadoop, A. (2009). Hadoop. In./Fair.

Lomotey, R. K., & Deters, R. (2014). Towards knowledge discovery in big data. Paper presented at the 2014 IEEE 8th International Symposium on Service Oriented System Engineering (SOSE).

Loshchilov, I., Schoenauer, M., & Sebag, M. (2013). Kl-based control of the learning schedule for surrogate black-box optimization. arXiv preprint arXiv:1308.2655.

Nita, M.-C., Pop, F., Voicu, C., Dobre, C., & Xhafa, F. (2015). MOMTH: multi-objective scheduling algorithm of many tasks in Hadoop. Cluster computing, 18(3), 1011-1024.

Ren, X., Ananthanarayanan, G., Wierman, A., & Yu, M. (2015). Hopper: Decentralized speculation-aware cluster scheduling at scale. Paper presented at the ACM SIGCOMM Computer Communication Review.

Saraladevi, B., Pazhaniraja, N., Paul, P. V., Basha, M. S., & Dhavachelvan, P. (2015). Big Data and Hadoop-A study in security perspective. Procedia computer science, 50, 596-601.

Tang, Z., Liu, M., Ammar, A., Li, K., & Li, K. (2016). An optimized MapReduce workflow scheduling algorithm for heterogeneous computing. The Journal of Supercomputing, 72(6), 2059-2079.

Xu, H., & Lau, W. C. (2017). Optimization for speculative execution in big data processing clusters. IEEE Transactions on Parallel and Distributed Systems, 28(2), 530-545. Mettam, G. R., & Adams, L. B. (1999). How to prepare an electronic version of your article. In B. S. Jones & R. Z. Smith (Eds.), Introduction to the electronic age (pp. 281–304). New York: E-Publishing Inc.

Zaharia, M., Konwinski, A., Joseph, A. D., Katz, R. H., & Stoica, I. (2008). Improving MapReduce performance in heterogeneous environments. Paper presented at the Osdi.

Zaharia, M., Borthakur, D., Sarma, J. S., Elmeleegy, K., Shenker, S., & Stoica, I. (2009). Job scheduling for multi-user mapreduce clusters. Retrieved from

Zaharia, M., Borthakur, D., Sen Sarma, J., Elmeleegy, K., Shenker, S., & Stoica, I. (2010). Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. Paper presented at the Proceedings of the 5th European conference on Computer systems.

Biographies

Aisha Shabbir received her B.Sc. degree in Math Applied, Math Pure & Physics from the Hazara University, Pakistan, in 2008, M.Sc. degree in Electronics, from Quaid-i-Azam University, Pakistan, in 2010. She did MS in Electronic Engineering from Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Pakistan in 2013. Currently, she is doing PhD in Computer Science at Universiti Teknologi Malaysia (UTM), Malaysia, She is Student member IEEE and member of Pervasive Computing Research Group (PCRG). Her research interest includes data aggregation in networks, partioning and scheduling algorithms for big data and cloud computing.

Kamalrulnizam Abu Bakar received his B.Sc. degree in Computer Science from the Universiti Teknologi Malaysia, Malaysia, in 1996, M.Sc. degree in Computer Communications and Networks, from Leeds Metropolitan University, United Kingdom, in 1998, and his PhD. degree in Computer Science from Aston University, United Kingdom, in 2004. He is a Professor in Computer Science at Universiti Teknologi Malaysia, Malaysia, and member PCRG. His research interest includes mobile and wireless computing, ad hoc and sensor networks, big data and cloud computing. He is a member of the ACM, Internet Society (ISOC) and International Association of Engineering (IAENG). He involves in many research projects and is a referee for several scientific journals and conferences.

Raja Zahilah Raja Mohd. Radzi received her Dr. Eng in Electrical & Information System, from Osaka Prefecture University, Japan on 2012. She obtained M.Eng. in Electronics and Telecommunications and a B.Eng. in Computer, both from Faculty of Electrical Engineering, Universiti Teknologi Malaysia. Currently, she is a Senior Lecturer at Faculty of Computing, UTM, Malaysia. She is a member of the PCRG, IEEE Society (Communication Society & Photonics Society) and an overseas member of The Institute of Electronics, Information and Communication Engineers, Japan. Her research interests include IoT, Big Data and Cloud Computing, Software Define Networking and Embedded System.

Sulaiman Azeem received his Bachelors' Degree in Electronic Engineering, from Capital University of Science Technology, CUST (MAJU) Pakistan in 2016. Currently, he is working as maintenance Engineer in Elevator Company in Erbil, Iraq. His research interests include IoT, Big Data and Cloud Computing, Wireless networks.