ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

A Survey on I/O Performance in Big Data Applications

Shraddha Shrivastava

Student, Dept. of CSE, Babulal Tarabai Institute of Research and Technology, Sagar, Madhya Pradesh, India

ABSTRACT: This is the age of exponentially increasing variedly structured data i.e. Big Data. As the volume of Big Data is increasing, the need for its quick access, analysis and storage is also increasing. If we need to use Big Data efficiently, we must have a lightning fast I/O system. However, there are several bottlenecks that serve as a hindrance in the I/O path. Data reduction and data visualization methods are used to reduce data while moving it from one place to another. An algorithm for the placement of data analysis is used to figure out an optimal location for positioning the analysis algorithm. In this paper, a review of previous approaches regarding I/O performance of HEC machines is done and potential amendments that can enhance the efficiency are described.

KEYWORDS: Big Data, HEC Machines, Hadoop, MapReduce, Data Reduction

I. INTRODUCTION

Large scale applications generate an enormous amount of data. Since this huge amount of data contains many interesting facts for scientific interesting facts for scientific interest. It becomes very crucial to rapidly move, store, analyze and visualize data[1].A quick input/output should take place in such big data applications so that the efficiency quotient of the application is maintained well. However there are severe I/O bottlenecks that prove to be hurdles in this race. Emerging scientists are being prompted to apply a different strategy to conquer this problem.

Due to growth of the volume of data, the gap between the computation power and I/O performance is increasing. This increase in gap is a result of the I/O bottlenecks that we come across in the process. A considerable amount of time is spent on writing the data back to storage after processing and vice versa. To improve the performance of input and output in-situ analysis has proved to be effective. By processing the data before placing it onto the disk, in-situ analysis can reduce I/O costs and improve its end to end performance [2]. For speeding up data processing the data analysis must be separated from simulation and executed in parallel on independent resources [3]. When data analysis and location of placing the simulation is separated, it leads to extra data movement along the input output path, a new challenge of balancing the data movement comes into play. Data Compression and Data visualization is applied for reducing the amount of data that moved between nodes. Data compression reduces repeated information so that less storage space is required and less data moves among applications. Various different lossy and lossless compression algorithms are used for the purpose. Data visualization queries include typical queries like point selection, subnetting etc. which return only a small subset of original data and significantly reduce the amount of data that need to be moved between processes or applications. Various data analytics placement strategies explore the various potential places where analytics can be placed in the scientific data analysis architecture.

This paper presents a detailed assessment of using an amalgam of data compression [4] and Data visualization [5] along with flexible placement strategies. How and where to use this data analytics along the I/O path of HEC applications so as to improve the I/O performance efficiency is the major goal of this paper.

The paper is organized as follows. Section 2 presents background information about big data, HEC machines and the data movement path. Section 3 describes the methodology required for understanding the current system i.e. data reduction and analysis placement positions. Section 4 describes the previous work done in this field including present I/O model and algorithm. This section also describes the problems being faced due to this algorithm. Section 5 presents the amendments that will potentially overcome the already mentioned flaws in the present algorithms. Section 6 concludes the paper.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

II. BACKGROUND

To study the I/O performance of Big Data applications, the detailed knowledge of Big Data, HEC Machines and the path of movement of big data in such applications is required. They are described in this section.

A. BIG DATA: Big Data is a term coined for extremely large data sets that are collected over time. Data as voluminous ranging from petabytes to Exabyte; as varied it may be structured ,partially structured or completely unstructured; as quick to accumulate that gigabytes of data may collect in just a few seconds. All these characteristics volume, variedness and velocity clearly define what big data are all about [6].This kind of data arises from numerous sources like data logs, social networking, sensors, business processes, scientific research etc. It is extremely complicated in nature hence difficult to be processed by conventional management or database handling applications [7][10].

The valued knowledge mined from big data helps in numerous areas. Main areas of application of big data [8] may include:

1. Understanding and engaging with the customers, their behavior and preferences effectively.
2. Optimizing business decisions based on the insights from big data. Concrete decisions about storing stock of goods, supply chains or delivery route optimization.
3. Improving scientific research, healthcare [9], sports performances and weather forecasts using patterns found from big data.
4. Training data sets for self-learning machines or devices and advancing the security mechanisms against cyber-attacks.
5. As individuals, the analysis of data generated from smart watches or other wearable devices which counts our calorie consumption or sleeping patterns will benefit us.

B. HIGH END COMPUTING (HEC) MACHINES: Big data is the trendiest buzzword in the IT industry. The processing capability of machines required for handling big data is far higher than that of conventional database systems. New technologies and frameworks are being introduced in the industry to conquer the challenges of increasing volume, variety and velocity of big data. Machines with high capability and high capacity computing capabilities called High End Computing Machines (HEC). Such applications consist of concurrent programs designed using multi-process as well as multi-threaded models. The applications consist of various parallel constructs like threads, distributed processes, local processes etc. with different degrees of parallelism. The High End Computing (H.E.C.) machines are capable enough to capture, manage, store and analyze the big data to find implicit and valuable information for various businesses.

C. DATA MOVEMENT PATH: The path along which the big data moves inside the complicated HEC machines systems the machine is called the data movement path. It has three major types of nodes namely the compute nodes, Analytics nodes (staging nodes) and storage (I/O nodes) [11].

- The nodes responsible for simulation and analytics are called computing nodes.
- Another type of nodes that are specifically dedicated to ease the process of analytics through staging is called the analytics nodes. Staging nodes are a part of compute nodes that are responsible for reducing any I/O overheads on applications. The data generated from simulators moves to the storage directly or via the staging nodes.
- The third type of nodes responsible for storing data is called storage nodes.

A diagrammatic representation of this data movement path is shown in Figure 1. Any of the algorithms that measure the performance of applications are placed in the potential locations lying along this path.

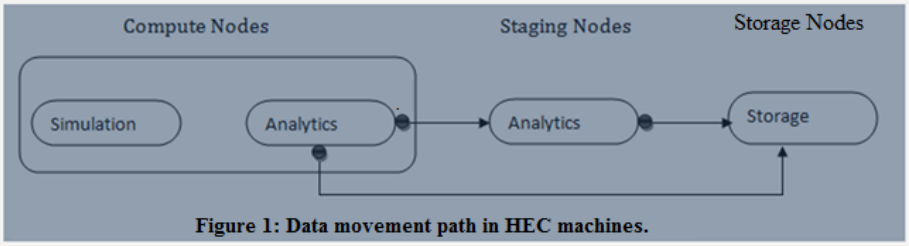

Figure 1: Data movement path in HEC machines.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

III. METHODOLOGY

The main requisites of creating this system are knowledge and use of proper data reduction algorithms, apt strategies for placing analysis and the knowledge for probable places where analytics should be placed.

A. DATA REDUCTION : Data reduction or Data compression [12] is the process of minimizing the size of data that needs to be processed by removing redundant information from it. Data reduction boosts the efficiency of the application and substantially reduces the cost of handling data. It is categorized into lossy and lossless compression.

Lossless data compression makes use of data compression algorithms that reconstruct the exact data from the compressed data. Some main types of files that can use lossy compression are text, executables, images, and sound.

A lossy data compression method is one where compressing data and then decompressing it retrieves data that may be slightly different from the original, but is similar to be useful for the same purpose. Lossy data compression is used mainly on the Internet, streaming media and telephony applications. Three data compression algorithms lossy, bzip2 and gzip (both lossless) are used here.

bzip2 is a free and open-source high-quality data compressor that uses the Burrows–Wheeler algorithm. It compresses data in blocks of size between 100 and 900 kB and converts frequently-recurring character sequences into strings of identical letters. It typically compresses files to within 10% to 15% of the best available techniques being around twice as fast at compression and six times faster at decompression [13].

gzip (GNU zip) is a compression utility designed to be a replacement for barely compression. Its main advantages over compression are much better compression and freedom from patented algorithms. gzip is based on the DEFLATE algorithm, which is a combination of Huffman coding and LZ77 algorithm[14].

3.2 ANALYTICS PLACEMENT POSITIONS: Let us now briefly discuss the potential places where data reduction analytics can be placed along the data movement path already described before. Any scientific application includes two stages, the simulation stage and analytics stage. Simulation is done on the compute nodes of the HEC Machine. The analytics can be placed at different locations. The potential places are marked in Figure 1.

i. If analytics is placed at point ‘A’ it will reduce the output dumped in storage. Although placing the analytics at staging node will result in multiple data movement processes between compute node and staging nodes.

ii. Another potential position is point ‘B’ which will reduce the size of data moving from compute nodes to staging nodes.

iii. Yet another position can be point ‘C’ which may be used to reduce data movement cost between the staging and I/O nodes.

The final decision is made by the data analytics placement algorithm on the basis of the profiling and monitoring metrics.

IV. PREVIOUS WORK AND ITS ANALYSIS

A. I/O PERFORMANCE MODEL: To analyze the I/O performance of HEC applications, a reduction based compression model is used here. The principle of this compression based model is that if the data is compressed before moving it from one location to the another , a smaller amount of data will be moving which is our target. In this model the data that is moved to the storage (or between applications) is reduced using various reduction algorithms. The cost and performance parameters that are used in this model are described in **Table 1**. The reduction algorithm with the minimum value of total end to end transfer latency is selected for compression.The calculation formulae for various parameters are given in **Table 2**.

B. ANALYTICS ALGORITHM: The algorithm aims at placing the analysis at the most efficient location so as to reduce the cost of data transfer subject to the availability of resources and performance penalties. The algorithm takes as input the potential locations, available resources at those locations and the performance profiles. The step by step process of the algorithm is given below.

Step 1: When the data is generated by the simulation, an analytics is selected to reduce the data on the idle node, thereby profiling it.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

Table 1: Parameters Used in Model	
ρ , ρ_c ,ρ_d	ρ :Available processor cycles Ratio ρ_c ,ρ_d : Available cycles Ratio on compression ,decompression sideρ=1 (when special core is used for compression), <1(when a processor is shared by processes) , >1 (when parallel execution is done with multiple processors
BW	Bandwidth of Data transfer
B_d	Size of data Block
T_c , T_d	Compression/Decompression Throughput
δ	Compression Ratio
$\mathcal{E}$	Impact factor of cache (here $\mathcal{E}$ =1)
t_o	Total end to end latency to original data transfer
t_{comp} ,t_{decomp}	Time taken to compress decompress data
$t_{transfer}$	Transfer time
t_{Total}	Total end to end transfer latency

Table 2: Calculation of different parameters	
t_o	B_d/BW
t_{comp}	$B_d/(T_c * \mathcal{E} * \rho)$
$t_{transfer}$	$(B_d *)/BW$
t_{decomp}	$B_d/(T_d * \mathcal{E} * \rho)$
t_{Total}	$t_{comp}+t_{transfer}+t_{decomp}$

Table3: Profiling and monitoring metrics	
Profiling metrics (profiled on idle nodes)	
t_m	Data transfer time per iteration
t_c	Query execution time per iteration
c_p	Processor usage (percentage) for the analytics
m_p	Memory usage (percentage) for the analytics
Monitoring metrics (periodically collected on the candidate place)	
T_c	Data transfer time per iteration
C_p	Query execution time per iteration
M_p	Processor usage percentage for the analytics

Step 2: The algorithm checks the potential places and collects the monitoring information and profiling metrics at each of them. These are given in **Table 3.**

Step 3: A placement decision diverging from the query's initial placement onto an idle node involves locating the analysis. The analysis execution time is rechecked at the new place.

Step 4: Placement decisions are re-evaluated at regular time intervals. The placement decision is enforced by deploying the analytics code to the best place among the data path.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

C. PROBLEMS WITH PRESENT ALGORITHM: The current algorithm checks the profiling and monitoring metrics at all potential locations and records them appropriately. This process consumes a substantial number of processor cycles. Every time the algorithm is run the same metrics are recorded and compared again and again. Also the order in which the algorithm searches the potential locations in not organized. To save valuable milliseconds of the algorithm and give it an organized path to search the potential locations to find the most efficient location, some amendments must be made in the present algorithm.

V. PROPOSED CHANGES AND ITS IMPACT

A. PROPOSED ALGORITHM: The concept of priority indexing can be used in the present algorithm on selected parameters so that repeated calculations are avoided. Results of the algorithm can be collected in a table and priority indexing can be applied on its tuples to select an optimal location to place the algorithm.

B. ADVANTAGES OF PROPOSED ALGORITHM: The location priority index allotted to every potential location will serve as an excellent index to order the sequence of locations in the analysis placement algorithm. The location priority index can become the base when the analysis placement algorithm searches the potential locations to find the most efficient location to place the analytics. It will increase the efficiency of the algorithm as the most efficient location would be found within no time.

VI. CONCLUSION AND FUTURE WORK

This work deals with the I/O performance of Big Data Applications. An efficient analytics algorithm checks the potential places lying along the data movement path and places the reduction analytics algorithm at the most eligible position. With the indexed performance table we can guide the analytics algorithm through various candidate locations in a specifically indexed order so that the process of finding the optimal location becomes easier and quicker.

REFERENCES

1. FlexAnalytics: A Flexible Data Analytics Framework for Big Data Applications with I/O Performance Improvement , HongboZoua,∗, YongenYub, WeiTangc, Hsuan-Wei MichelleChend
2. F. Zheng, H. Zou, J. Cao, J. Dayal, T. Nugye, G. Eisenhauer, S. Klasky, FlexIO: location-flexible execution of in-situ data analytics for large scale scientificapplications, in: Proc. IEEE International Parallel and Distributed Processing Symp. (IPDPS'13), 2013, pp.320–331.
3. A. Gerndt, B. Hentschel, M. Wolter, T. Kuhlen, C. Bischof, VIRACOCHA: an ef-ficient parallelization framework for large-scale CFD post-processing in virtual environments, in: Proc. ACM/IEEE Conference on Supercomputing (SC04), 2004, pp.50–61.
4. K. Sayood, Introduction to Data Compression, 3rd edition, Morgan Kaufmann, 2005.
5. K. Stockinger, et al., Query-Driven Visualization of Large Data Sets, 2005.
6. Bakshi, K. ; Cisco Syst. Inc., Herndon, VA, USA ,Considerations for big data: Architecture and approach, Aerospace Conference, 2012 IEEE
7. Big Data, http://hmchen.shidler.hawaii.edu/Chen_big_data_MISQ_2012.pdf
8. The Concept,Characteristics and Application of Big Data,Ma Jian-guang,JIANG Wei
9. The Inevitable Application of Big Data to Health Care , Travis B. Murdoch, MD, MSc; Allan S. Detsky, MD, PhD, JAMA. 2013;309(13):1351-1352. doi:10.1001/jama.2013.393.
10. Big Data: A Survey,Min Chen, Shiwen Mao, Yunhao Liu; Springer, April 2014, Volume 19, Issue 2, pp 171-209
11. F. Zheng, H. Zou, J. Cao, J. Dayal, T. Nugye, G. Eisenhauer, S. Klasky, FlexIO: location-flexible execution of in-situ data analytics for large scale scientific applications, in: Proc. IEEE International Parallel and Distributed Processing Symp. (IPDPS'13), 2013, pp.320–331.
12. D.A. Lelewer, D.S. Hirschberg, Data compression, in: Proc. ACM Computing Sur-veys(CSUR), 1987, pp.261–296.
13. High-quality data compressor, bzip2, http://www.bzip.org.
14. Compression utility, gzip, http://www.gzip.org.

BIOGRAPHY

Shraddha Shrivastavais a student of Masters of Technology in the Computer Science and Engineering Department, B. T. Institute of Research and Technology, affiliated to Rajiv Gandhi ProdyogikiVishwavidhyalaya, Bhopal. She received Bachelors of Engineering (BE) degree in 2012 from RGPV Bhopal, India. Her research interests are Big Data, Data Mining,and Database Systems etc.