IJARIIT INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 3)
Available online at: www.ijariit.com

Evaluation parameters of the infrastructure resources needed to integrate the parallel computing algorithm and the distributed file system

Amit Kumar Sharma
amit95kumarsharma@gmail.com
Bharati Vidyapeeth University, Pune, Maharashtra

Shammi Nanda
shamminanda1110@gmail.com
Bharati Vidyapeeth University, Pune, Maharashtra

ABSTRACT

Technology and the growing population in the digital world have led to a drastic explosion of data scattered across various digital components and network nodes. By other On the other hand, various technologies are being improved and innovated to maintain the processing and conversion of these raw data into useful information in various fields with proliferating data. Since the data and the application to process this data are quantitatively increasing, it is also necessary to change or to update the infrastructure to comply with the current status of the requirements. The question that arises here is how improved resources will be beneficial and in what way affect the performance of the application. This document focuses on understanding how infrastructure resources will have an impact Extreme to the final performance of the distributed computing platform and what are the parameters taken into account with high priority to deal with the performance problems in a distributed environment.

Keywords: *Performance evaluation, Active storage, Storage array, Resource utilization, Parallel and distributed systems*

1. INTRODUCTION

The buzz on the distributed computing platform is map reduction and full investigations are improving the performance of that distributed environment. There is research carried out by [1] to integrate several models of parallel computing and distributed file systems, such as the reduction of integrated maps with brightness. Luster has different benefits compared to HDFS. This integration is treated and deployed technically in the same infrastructure without also considering the necessary requirements in infrastructure.

MapReduce is a distributed computational algorithm and is widely used for works of large scale. Currently, the implementation of MapReduce is with the help of the open source framework Hadoop. By default, Hadoop uses HDFS (Hadoop Distributed File System) for the implementation of MapReduce. Instead of HDFS, we can use Hadoop in a file system distributed as Luster. Luster is a kind of file system distributed parallel which is used for large-scale clustering computing. The gloss is derived of two words, that is, Linux and cluster. Linux is the platform to implement MapReduce and Cluster is the collection of computers at a distance from each other but connected through the network. MapReduce divides the input data into a number of fragments of limited size (the size it specifies previously in the algorithm) in parallel. Then, the algorithm converts fragment data into a group of intermediate key values in a set of Map tasks. Below is the mix phase in which the values of each key are shuffled and the combined key values are processed as the output of data with the help of reducing tasks. [2]

Here, we can say that Luster is better compared to HDFS because once the data is written to HDFS, it cannot be modified while in Luster, the data is stored on object storage servers (OSS) and the metadata is stored in the metadata servers (MDS). As indicated above, Luster is designed for calculations to big scale, intensive I / O applications and performance sensitive.

The use of Luster as a backend for Hadoop's work allows flexibility when assigning homework of mapping, which means that all available nodes. They can be used for the same job, without any network problems, unlike HDFS where the number and location of the mapper tasks for a specific job distribution of the input data. HDFS thus leaves most of the cluster inactive. In Luster, the data can be moved to any of the available resources and, therefore, provide 100% utilization of the node [3][4]. Speaking of the efficiency of the network, Mapper tasks take different times and the task of Reducer can start as soon as the mapper finishes writing his data. Since the whole set of units and full network bandwidth is available with Luster to calculate the active node, the average time to write the Mapper output should decrease and with this efficiency, we mean that the final mapping task is always will complete more quickly with Luster compared to HDFS [4].

Luster uses RAID disk drives for any type of failure, which means there is less chance of disk failures when using Luster while HDFS uses data replication to make sure that data are available with any other node if the current node fails. So, here we can say that the HDFS uses more memory compared to Luster to store the same data on all the discs.

2. DISTRIBUTED ENVIRONMENT

The environment distributed as shown in figure 1 hides the components as the commutator net what is responsible for the migrations of the request, data, and results through the network among the other components. The other components include Server that is the host on which the application is running. The data is stored on the disks connected to the server or set of storage arrays connected in a network and that network it connects to the application server. Taking into account the reduction of the map, the job tracker runs in a single server that is master and the task trackers run on the nodes that contain the data. These nodes can be servers or storage systems. Performance tracker task will have a direct impact on application performance. There are solutions provided within Hadoop to manage the failure of the tracker from tasks, but what about resource contention and network traffic due to data transmission and transmission of requests?

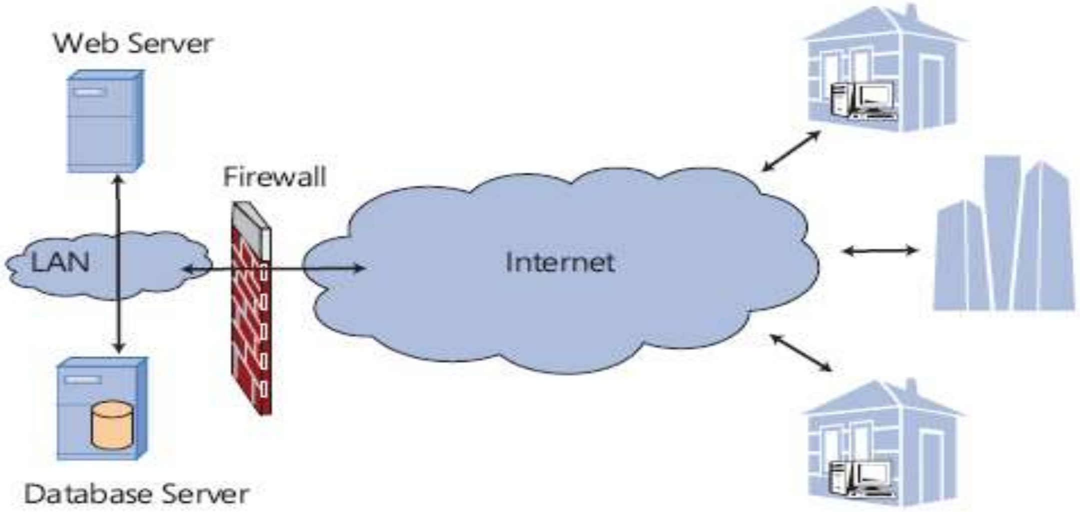

Fig. 1: Distributed environment

The storage nodes have their own computing power, memory, and disks. Map reduces is the true computer model that has realized the concept of active storage [5][6][7]. The task trackers execute the work downloaded near the data, what that will reduce the amount of traffic required to migrate between the computing and storage platform. Processing capacity, disk capacity, and memory size are very necessary to take into account when deciding the infrastructure of storage nodes. File systems on raw disks facilitate the addition of a layer for the I/O request to pass. When selecting resources and the ability to download the calculation closer to the data, it is also necessary to consider the performance of the file system. The analytical approach to file system performance [9] helps to understand the use of the disc and how well the discs can meet the requests. The investigation shows how the storage node can be evaluated from the perspective of file and disk systems.

This research paper focuses on how you can avoid the containment of resources by understanding the use of the components. A set of mathematical equations will be useful to analyze the utilization. Workloads must also consider while performance evaluation is performed. Workloads vary and utilization also varies. Workloads also affect disk file systems [10].

3. USING THE PROCESSOR

Distributed environments work primarily on hypervisors instead of metals naked. The hypervisor leads to virtualization and a single resource it can be divided and used among multiple workloads. To calculate the CPU utilization, it is necessary to consider the use of the virtual CPU because the Guest Operating System (GOS) is being installed in the virtual system. In virtualization, the bottleneck of I/O is one of the main problems, but nowadays, some automated virtual tools are present to avoid these bottlenecks. To calculate the use of the virtual CPU, we consider its use in megahertz, a number of virtual CPUs and center frequency.

4. USE OF MEMORY

For the use of memory, there is no such equation, but consider the reference location, also known as the locality principle, which is a phenomenon that describes the same value or storage locations related to which is accessed frequently. Basically, it does refer to two types of reference locations. These are temporary and spatial locations.

The temporary location refers to the reuse of data or resources for a specific time or for a duration of weather relatively little, that is, the location will be referenced again. In the temporary location, it is necessary to store a copy of the data referenced in the spatial memory storage while, spatial locality refers to the reuse of data within relatively close storage locations. The spatial locality it can be extended to your special case, known as sequential locality. With the sequential name, it is clear that the spatial locality will take place only when the data elements be willing and accessed linearly.

Locality is used most of the time when a substantial part of the references is added in clusters, and if the shape of this cluster system can be predicted well, it can be used to optimize the speed. Again here, in the reference location, the reduction in time takes place due to the area of stored memory for temporary and spatial use.

5. BANDWIDTH IOPS CALCULATION

Now, we have to calculate the bandwidth. The bandwidth can you describe as the bit rate that is available or the information capacity consumed that is expressed in metric multiples of bits per second.

Here, we consider the bandwidth for the system. The bandwidth it is measured in hertz, which means that it is equal to the frequency. For the calculation of the bandwidth, we consider:

$$\text{Bandwidth} = \text{IOPS} * \text{Size of I / O}$$

Where the I/O size is different w.r.t. workloads.

We have to consider the reading/writing pattern for the system of distributed files due to the nodes since all the nodes are referred to in a distributed environment. The I/O size is larger, for example, in the system of Luster files than in HDFS and, therefore, we can say that Luster is better compared to HDFS (which is why all nodes in Luster are referenced above).

For the bandwidth, it is necessary to calculate the IOPS. So, going to look for the IOPS, we have to consider:

1) Percentage of cache hits
2) Disk IOPS
3) Impact RAID on disk
4) IOPS

Here the percentage of hits from the cache it is (1% missing cache).

$$\text{Disk IOPS} = 1/(\text{RPM} + \text{disk transfer speed})$$

Where, the RPM is the rotations per minute, that is, the total I/O in a single rotation of the disk. Therefore, we can clearly see that the IOPS disk and RPM are inversely related, which means that with the increase in RPM, the IOPS disk decreases and vice versa. The "cache hit" is based in the cache and what data is stored in the cache and what is the size of the data present there.

With this, we can now calculate the application performance distributed file system and platform. The performance of the application can be obtained through the sum of all the data given above with respect to CPU utilization (virtual included), memory utilization (reference location), bandwidth consumption and IOPS.

The time required for the application tasks to be completed with the storage integrated computation it can be calculated using this set of equations.

6. CONCLUSION

The Distributed File System is based fundamental design factors and, therefore, these designs can result in scalability and performance limitations. Therefore, the integration of the computing platform and the file systems approach must also include the performance factors together with the design factors and problems, since the underlying file system compute clusters in HPC environments (high computing) performance) can generate significant improvements in system performance and cluster efficiency, or can also offer a lower overall system cost. In this document, we have considered small cluster supports. However, comparative tests must be done on a much larger scale that should be undertaken in the future. To improve the current implementations, we can consider the Performance Platform model, since it fits well with the experimental data collected. The performance model of the platform characterizes the execution of the phase as a function of the processed data and to achieve this, we need to find relationships between the amount of data processed and the durations of different phases of execution using the set of collected measurements.

7.REFERENCES

[1] Archana, R. C., Naveenkumar, J., & Patil, S. H. (2011). Iris Image Pre-Processing And Minutiae Points Extraction. *International Journal of Computer Science and Information Security*, *9*(6), 171.

[2] Bhore, P. R., Joshi, S. D., & Jayakumar, N. (2016). A Survey on the Anomalies in System Design: A Novel Approach. *International Journal of Control Theory and Applications*, *9*(44), 443–455.

[3] Bhore, P. R., Joshi, S. D., & Jayakumar, N. (2017a). A Stochastic Software Development Process Improvement Model To Identify And Resolve The Anomalies In System Design. *Institute of Integrative Omics and Applied Biotechnology Journal*, *8*(2), 154–161.

[4] Bhore, P. R., Joshi, S. D., & Jayakumar, N. (2017b). Handling Anomalies in the System Design: A Unique Methodology and Solution. *International Journal of Computer Science Trends and Technology*, *5*(2), 409–413.

[5] Desai, P. R., & Jayakumar, N. K. (2017). A Survey on Mobile Agents. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, *5*(XI), 2915–2918.

[6] Gawade, M. S. S., & Kumar, N. (2016). Three Effective Frameworks for semi-supervised feature selection. *International Journal of Research in Management & Technology*, *6*(2), 107–110.

[7] GAWADE, S., & JAYKUMAR, N. (2017). ILLUSTRATION OF SEMI-SUPERVISED FEATURE SELECTION USING EFFECTIVE FRAMEWORKS. *Journal of Theoretical & Applied Information Technology*, *95*(20).

[8] Jaiswal, U., Pandey, R., Rana, R., Thakore, D. M., & JayaKumar, N. (2017). Direct Assessment Automator for Outcome-Based System. *International Journal of Computer Science Trends and Technology (IJCS T)*, *5*(2), 337–340.

[9] Jayakumar, D. T., & Naveenkumar, R. (2012). SDjoshi, *International Journal of Advanced Research in Computer Science*

and Software Engineering, *Int. J*, *2*(9), 62–70.

[10] Jayakumar, M. N., Zaeimfar, M. F., Joshi, M. M., & Joshi, S. D. (2014). INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY (IJCET). *Journal Impact Factor*, *5*(1), 46–51.

[11] Jayakumar, N. (2014). Reducts and Discretization Concepts, tools for Predicting Student's Performance. *International Journal of Engineering Science and Innovative Technology (IJESIT)*, *3*(2), 7–15.

[12] Jayakumar, N. (2015). Active storage framework leveraging processing capabilities of the embedded storage array.

[13] Jayakumar, N., Bhardwaj, T., Pant, K., Joshi, S. D., & Patil, S. H. (n.d.). A Holistic Approach for Performance Analysis of Embedded Storage Array.

[14] Jayakumar, N., Iyer, M. S., Joshi, S. D., & Patil, S. H. (2016). A Mathematical Model in Support of Efficient offloading for Active Storage Architectures. In *International Conference on Electronics, Electrical Engineering, Computer Science (EEECS) : Innovation and Convergence* (Vol. 2, p. 103).

[15] Jayakumar, N., & Kulkarni, A. M. (2017). A Simple Measuring Model for Evaluating the Performance of Small Block Size Accesses in Lustre File System. *Engineering, Technology & Applied Science Research*, *7*(6), 2313–2318.

[16] Jayakumar, N., Singh, S., Patil, S. H., & Joshi, S. D. (n.d.). Evaluation Parameters of Infrastructure Resources Required for Integrating Parallel Computing Algorithm and Distributed File System.

[17] KAKAMANSHADI, M. G., NAVEENKUMAR, M. J., & PATIL, S. H. (2011). A METHOD TO FIND SHORTEST RELIABLE PATH BY HARDWARE TESTING AND SOFTWARE IMPLEMENTATION. *International Journal of Engineering Science*.

[18] Kulkarnia, A., & Jayakumar, N. (2016). A Survey on IN-SITU Metadata Processing in Big Data Environment. *International Journal of Control Theory and Applications*, *9*(44), 325–330.

[19] Kumar, N., Angral, S., & Sharma, R. (2014). Integrating Intrusion Detection System with Network Monitoring. *International Journal of Scientific and Research Publications*, *4*, 1–4.

[20] Kumar, N., Kumar, J., Salunkhe, R. B., & Kadam, A. D. (2016). A Scalable Record Retrieval Methodology Using Relational Keyword Search System. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies* (p. 32).

[21] kumar Singha, A., Patilb, S. H., & Jayakumar, N. (2017). A Treatment for I/O Latency in I/O Stack. *Http://Www.Ijcstjournal.Org/Volume-5/Issue-2/IJCST-V5I2P83.Pdf*.

[22] Namdeo, J., & Jayakumar, N. (2014). Predicting Students Performance Using Data Mining Technique with Rough Set Theory Concepts. *International Journal of Advanced Research in Computer Science and Management Studies*, *2*(2).

[23] Naveenkumar, J. (2012). Keyword Extraction through Applying Rules of Association and Threshold Values. *International Journal of Advanced Research in Computer and Communication Engineering*, *1*(5), 295–297. Retrieved from http://www.ijarcce.com/upload/july/3-Keyword Extraction.pdf

[24] Naveenkumar, J., & Joshi, S. D. (2015). Evaluation of Active Storage System Realized Through Hadoop. *International Journal of Computer Science and Mobile Computing*, *4*(12), 67–73.

[25] Naveenkumar, J., Makwana, R., Joshi, S. D., & Thakore, D. M. (2015a). OFFLOADING COMPRESSION AND DECOMPRESSION LOGIC CLOSER TO VIDEO FILES USING REMOTE PROCEDURE CALL. *Journal Impact Factor*, *6*(3), 37–45.

[26] Naveenkumar, J., Makwana, R., Joshi, S. D., & Thakore, D. M. (2015b). Performance Impact Analysis of Application Implemented on Active Storage Framework. *International Journal*, *5*(2).

[27] Naveenkumar, J., & Raval, K. S. (2011). Clouds Explained Using Use-Case Scenarios. In *INDIACom-2011 Computing For Nation Development* (pp. 1–5).

[28] Naveenkumar J, P. D. S. D. J. (2015). Evaluation of Active Storage System Realized through MobilityRPC. *International Journal of Innovative Research in Computer and Communication Engineering*, *3*(11), 11329–11335.

[29] NAVEENKUMAR, M. J., Bhor, M. P., & JOSHI, D. R. S. D. (2011). A Self Process Improvement For Achieving High Software Quality. *International Journal of Engineering Science*, *3*.

[30] RAVAL, K. S., SURYAWANSHI, R. S., NAVEENKUMAR, J., & THAKORE, D. M. (2011). The Anatomy of a Small-Scale Document Search Engine Tool: Incorporating a new Ranking Algorithm.

[31] Rishikesh Salunkhe, N. J. (2016). Query Bound Application Offloading: Approach Towards Increase Performance of Big Data Computing. *Journal of Emerging Technologies and Innovative Research*, *3*(6), 188–191.

[32] Salunkhe, R., Kadam, A. D., Jayakumar, N., & Joshi, S. (n.d.). Luster A Scalable Architecture File System: A Research Implementation on Active Storage Array Framework with Luster file System.

[33] Salunkhe, R., Kadam, A. D., Jayakumar, N., & Thakore, D. (n.d.). In Search of a Scalable File System State-of-the-art File Systems Review and Map view of the new Scalable File system. In i*nternational Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT) - 2016* (pp. 1–8).

[34] Sawant, Y., Jayakumar, N., & Pawar, S. S. (2016). Scalable Telemonitoring Model in Cloud for Health Care Analysis. In *International Conference on Advanced Material Technologies (ICAMT)* (Vol. 2016).

[35] Singh, A. K., Pati, S. H., & Jayakumar, N. (2017). A Treatment for I/O Latency in I/O Stack. *International Journal of Computer Science Trends and Technology (IJCS T)*, *5*(2), 424–427.

[36] Zaeimfar, S. D. J. N. J. F. (2014). Workload Characteristics Impacts on file System Benchmarking. *Int. J. Adv*, 39–44.