2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)

Data Prefetching for Heterogeneous Hadoop Cluster

D C Vinutha
Research scholar, Dept. CSE
R. N. S. Institute of Technology. Bengaluru
Associate professor, Dept. ISE
Vidyavardhaka College of Engineering, Mysuru.
Visvesraya technological University, Belagavi, Karnataka.
vinuthadc@vvce.ac.in

G T Raju
Professor, Dept. CSE
R. N.S. Institute of Technology Bengaluru
Visvesraya Technological University, Belagavi, Karnataka.

Abstract: Hadoop is an open source implementation of MapReduce. Performance of Hadoop is affected by the overhead of communication during the transmission of large datasets to the computing node. In a heterogeneous cluster if a map task wants to process the data, which is not present in the local disk then the data transmission overhead occurs. To overcome this issue, Data Prefetching for Heterogeneous Hadoop cluster for resource optimization is proposed. data prefetching is used to fetch the input data in advance from the remote node to a particular computing node. Hence transmission of data occurs in parallel with data processing and the job execution time is reduced. Different MapReduce jobs are used to conduct the experiment. The results demonstrate that the time taken to transmit the data is reduced. The job execution time is reduced by 15% for the input data size greater than or equal to 2GB and performance improvement of 25% is obtained for 64MB block size.

Keywords: Hadoop, Data transmission, MapReduce, Data Prefetching, Data Processing.

1. INTRODUCTION

Big data represents a gigantic data, a data range from terabyte to petabytes. The issue that occurs in big data is how to store, search, create, and share a huge amount of data. A solution to the problem is Hadoop[1], it is an open source implementation of the MapReduce framework, which distributes huge amount data across the nodes in the Hadoop cluster. The important components of Hadoop are MapReduce[2] and HDFS [3]. MapReduce is a programming model to process the data. MapReduce propagates the data over all the nodes present in the cluster. The input data is fragmented into number of blocks. Map task processes the block of data received, after processing a received block of data. Map task receives and processes another block of input data, this process is executed repeatedly till the map tasks completes the processing of all block of input data. In conventional Hadoop data processing and data transmission occur sequentially [4], therefore data processing phase has to wait during data transmission. Fig 1 shows Data transmission and data processing occurs in sequence. In a heterogeneous cluster if a map task wants to process the data, which is not present in the local disk then the data transmission overhead occurs [5-8]. To overcome this issue, data prefetching for heterogeneous cluster is proposed for resource optimization. It is used to fetch the input data in advance from the remote node to a particular computing node. Hence transmission of data occurs in parallel with data processing and the job execution time is reduced [9].

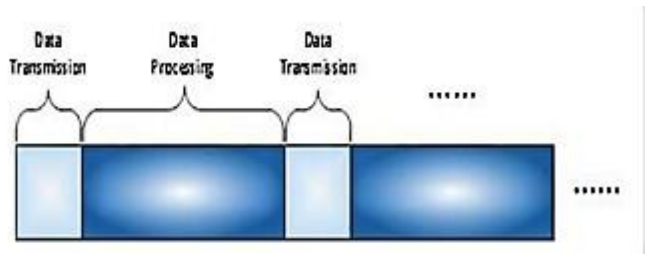

Fig 1: Sequence of Data transmission and Data processing in Hadoop MapReduce.

2. RELATED WORK

Many Researchers has carried out work to reduce the data transmission time. It affects the performance of Hadoop MapReduce. Bidirectional processing method called HPMR is proposed to ignore the modification directly from the conventional Hadoop. In this authors focused on improving the data locality by optimizing the algorithm for task scheduling,data locality is improved but the load balancing across the nodes in the cluster is not taken into account [10]. The delay scheduling algorithm is proposed by Zaharia et al.here the author considered the issues between fairness and locality of the cluster [11]. Based on the computational capability of a node, computing nodes are classified and modified the task scheduling algorithms. Data locality in a homogeneous environment is considered by Zhang et al [12]. DARE (Distributed Adaptive Data Replication Algorithms) is proposed for heterogeneous cluster and a node with greater computational capability gets more amount of data [13].

3. IMPLEMENTATION OF PREFETCHING METHOD

In our proposed work data prefetching is used to make the data transmission and data processing to work in parallel. To process the map task, if data is not present in the local disk, then the data transmission phase should be hidden in our proposed method, because it takes place in parallel with data processing. Figure 2 shows the design of Hadoop with data prefetching. Node 1 assigned with a Map task M and data to be processed by the Map task M is located in Node 2, so it fetches data from Node 2 to Node 1. It creates a prefetching thread to request the input data and uses a buffer to store the fetched data temporarily. Thread fetches the data from a remote node and stores temporarily in the buffer and Map task processes the input data stored in the buffer.

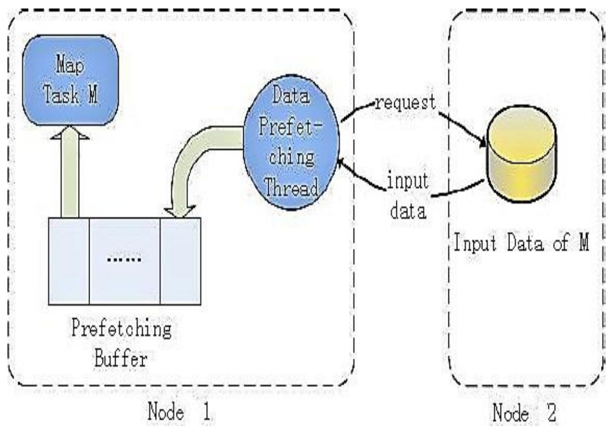

Fig2: Data prefetching method for Hadoop

Table1 lists the terminologies used in the theoretical analysis. A theoretical analysis of improvement in Modified Hadoop with prefetching mechanism by comparing it with the conventional Hadoop is given below.

Table1: Description of the terminologies used in the theoretical analysis

Symbol	Description
S_{map}	Input split size(size of input data to a map task)
S_{buf}	Buffer size for temporary storage of data
V_{map}	Data size processed per second by Map task in conventional Hadoop
V'_{map}	Data size processed per second by Map task in Modified Hadoop.
V_{trans}	Data size transferred between the computing node and remote node per second
M_{hadoop}	Average number of Map tasks Execution in a node in conventional Hadoop
$M_{improved}$	Average number of Map tasks Execution in a node in Modified Hadoop
AT_{hadoop}	Average time necessary to execute Map task in conventional Hadoop.
$AT_{improved}$	Average time necessary to execute the Map task in modified Hadoop.
t_{hadoop}	Time consumed by Map task, if the data is not present in local disk in conventional Hadoop.
$t_{improved}$	Time consumed by Map task, if the data is not available locally in modified Hadoop.
T_{hadoop}	Time taken for the job in conventional Hadoop
$T_{improved}$	Time consumed by the job, in modified Hadoop

Representation of Map task improvement for the data not present in the local disk is given in equation1

$$\Delta t = t_{hadoop} - t_{improved}$$

$$\Delta t = \left[\frac{S_{buf}}{V_{trans}} + \frac{S_{buf}}{V_{map}}\right] * \frac{S_{map}}{S_{buf}} - \left[\frac{S_{buf}}{V_{trans}} + \frac{S_{buf}}{V'_{trans}} + \left[\frac{S_{map} - S_{buf}}{\min(V'_{map}, V_{trans})}\right]\right] \quad (1)$$

An extra memory for buffer and thread is required for data prefetching to implement prefetching method. Here V'_{map} is almost equal to V_{Map}. Δt is given in equation 2.

$$\Delta t = (S_{map} - S_{buf}) * \left(\frac{1}{V_{trans}} + \frac{1}{V_{map}} - \frac{1}{\min(V_{map}, V_{trans})}\right) \quad (2)$$

$$\Delta t \approx \frac{S_{map} - S_{buf}}{\max(V_{map}, V_{trans})}$$

Equation 2 Describes, in Prefetching method transmission of data and data processing phase overlaps. Commonly processing of data is slower than the transmission of data. In this work the focus is on decreasing the data transmission cost i.e Δt and it is given in equation 3.

$$\Delta t \approx \frac{S_{map} - S_{buf}}{V_{trans}} \quad (3)$$

Performance improvement of Map task is high for larger values of S_{map}. But a very large value of S_{map} will affect the parallelism of Hadoop. As a result, the user should select the value of this parameter based on the situation. Job improvement is the collection of map task improvement in the node which executed the last map task and the results are returned.

In conventional Hadoop, Map task is allocated to a node based on the computational capability of a node. Total map tasks number is greater than the total nodes. Hence the task is distributed depending on the nodes computational capability and balanced the load across the nodes in the cluster. Average node replaces the node which executed the last map task and the results are returned. Job improvement is represented in equation 4.

$$\Delta T = T_{hadoop} - T_{improve}$$

$$\Delta T = M_{hadoop} * \Delta t_{hadoop} - M_{improve} * \Delta t_{improve} \quad (4)$$

The number of nodes and map task is same in conventional and modified Hadoop. Here Map task allocation is affected by the prefetching Mechanism. Different cases are considered to show the relationship between M_{hadoop} and $M_{improved}$.

.**Case 1:** In conventional and modified Hadoop the data is present in local disk for the task is given in equation 5.

$$M_{hadoop} = M_{improved} \quad (5)$$

Case 2: In conventional Hadoop data is stored in local disk for a task and the overhead caused by the modified Hadoop is given in equation 6.

$$M_{improved} - M_{hadoop} \approx \frac{S_{buf}}{V_{trans}} \quad (6)$$

Case 3: only in modified Hadoop the data is stored in local disk for the task and the performance improvement of modified Hadoop is given in Equation 7.

$$M_{hadoop} - M_{improved} = \frac{S_{map}}{V_{trans}} \quad (7)$$

Case 4: In both traditional and modified Hadoop the data is stored in the local disk for a task. The performance improvement by the modified Hadoop is given in equation 8.

$$M_{hadoop} - M_{improved} = \Delta t \quad (8)$$

Case 1, 3 and 4 describes the performance of modified Hadoop is better compared to conventional Hadoop and also the theoretical analysis of case 2 proves that Hadoop with prefetching method saves the time required for data transmission in both single and job consisting of various number of Map tasks.

4. EXPERIMENTAL RESULTS AND ANALYSIS

For our experiment, 8 virtual machines are used to create heterogeneous cluster. The configuration of machines used in Hadoop cluster is given in Table 2. Hadoop 1.0.4 is used to implement the prefetching method. MapReduce applications like word count and CCV Evaluator [14] is used for evaluating the performance and the configuration of various jobs are given in table 3. Word count counts the frequency of occurrence of word. CCV finds the vectors of class center which are used in algorithm of classification. The performance of prefetching mechanism will be affected by the speed of data processing and data transmission which is given in 3.2. Hence delay is introduced in experiment during data processing and transmission of data. Table 2 gives the configuration of different jobs.

Table2: Nodes configuration in Heterogeneous cluster

Node Id	N0	N1	N2	N3	N4	N5	N6	N7
No of Virtual Processors	1	1	1	1	2	2	2	2

Table 3: Jobs Configurations for experiment

Job ID	Job0	Job1	Job2	Job3	Job4	Job5
Application	Word Count	Word Count	CCV Evaluator	CCV Evaluator	CCV Evaluator	CCV Evaluator
Delay of Map Processing(Ms)	10	20	0	1	1	1
Delay of 4KB data transmission (Ms)	1	1	1	1	2	3
Total input size of job(MB)	2078.72	2078.72	1957.12	1957.12	1957.12	1957.12
Input Split Size of Map(MB)	64	64	61	61	61	61
Number of Reduce tasks	1	1	1	1	1	1

Figure 3 shows the average time spent to get the input data in both conventional and modified Hadoop. Delay of Data transmission for Job 3, Job 4, and Job 5 are varied. The time invested for transmission of data is overlapped with data processing. Hence the time spent during data transmission time will not affect the performance in real time. An experimental result in fig 3 shows 14% of decrease in time of data transmission for job5.Prefetching method works very well where the connection of network is poor.

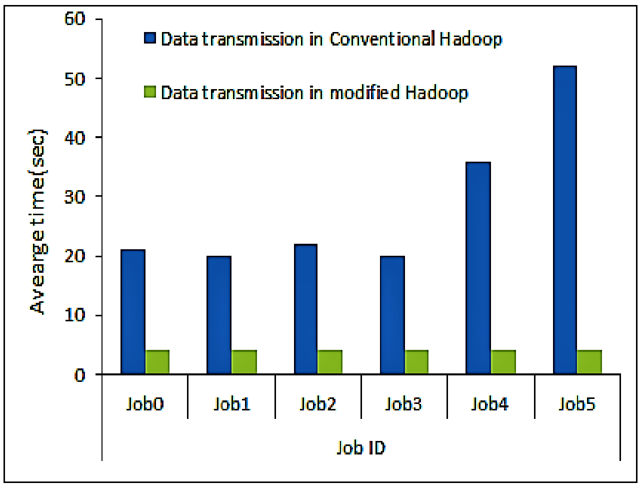

Fig 3: Average Time taken for data transmission by different jobs

The parameters like block size and the total size of the data are varied for job2, Results shown in figure 4 and 5 illustrates the impact of these 2 parameters. Figure 4 shows on average 15% of job Execution time is decreased for the input data size greater than or equal to 2 GB. For 1 GB of input data the improvement in job execution time is less.

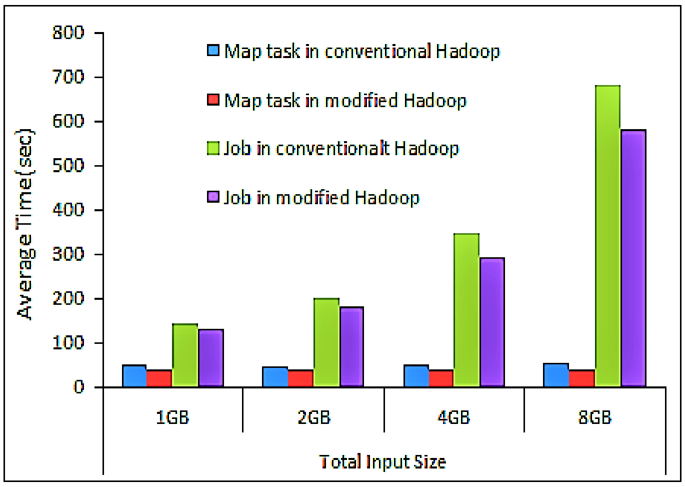

Fig 4: Performance of Hadoop for different input size

Figure 5 shows the performance improvement increases with an increase of block size. The performance improvement of 25% is obtained for the block size of 64 MB.

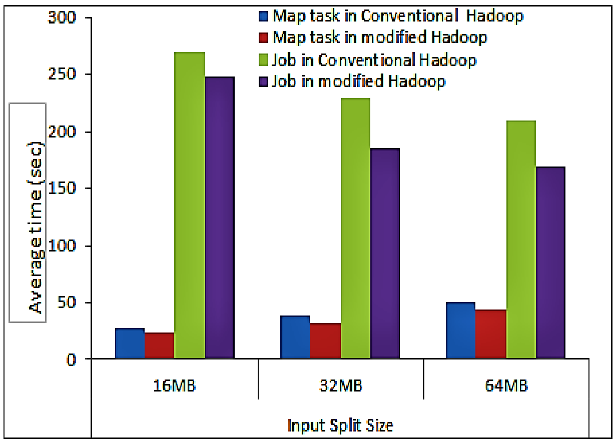

Fig 5: Performance of Hadoop for varied Input Split Size

5. CONCLUSION

In conventional Hadoop the implementation of data transmission and processing of data is sequence to process the Map tasks. If the data is not present in the local disk, then data processing phase is delayed till the data transmission occurs, Hence in this paper we have proposed and presented Data Prefetching for Heterogeneous Hadoop cluster for resource optimization. In this work the data is fetched from remote node to the prefetching buffer in advance, then the computing node launches the task by accessing the data from prefetching buffer. Here it overlaps the data transmission with the processing of data phase. Hence the time taken for transmission of data is decreased and job execution time is decreased by 15% for the input data size greater than or equal to 2GB.

Reference

[1]. J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters”, Communications of the ACM, vol. 51, no. 1, (2008).

[2]. Apache Hadoop. [Online]. Available: http://hadoop. apache.org/ (2014).

[3]. K. Shvachko, H. Kuang, S. Radia and R. Chansler, “The Hadoop Distributed File System”, IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), (2010) May 3-7: Incline Village, USA.

[4]. Improving MapReduce Performance Using Data Prefetching mechanism in heterogeneous or Shared Environments Tao gu, Chuang Zuo,Qun Liao , Yulu Yang and Tao Li, International Journal of grid and distributed computing (2013).

[5]. Sun-Yuan Hsieh, Chi-Ting Chen,Chi-Hao Chen,Tzu-Hsiang Yen,Hung-Chang Hsiao,and Rajkumar Buyya “Novel Scheduling Algorithms for Efficient Deployment of MapReduce Applications in Heterogeneous Computing Environments”,ieee transactions on cloud computing,vol 6,n0.4,October-december 2018.

[6]. Hadoop Capacity Scheduler (2016).[online].Available:https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoopyarn-site/CapacityScheduler.html.

[7]. M.Zaharia(2015) "The fairscheduler "[online]. Available:https://hadoop.apache.org/docs/r2.7.1/hadoop-yarn/hadoop-yarn-site/FairScheduler.html.

[8]. Kamble Ashwini, Kanawade Bhavana,"A Brief of MapReduce Performance," International Journal of Innovative Research in Advanced Engineering (IJIRAE) Volume 1 Issue 1(April 2014).

[9]. S. Khalil, S. A. Salem, S. Nassar and E.M. Saad, "MapReduce Performance in Heterogeneous Environments: A Review", International Journal of Scientific & Engineering Research,Vol.4,no 4,(2013).

[10]. S. Seo, I. Jang, K. Woo, I. Kim, J.S. Kim and S. Maeng, "HPMR: Prefetching and Pre-shuffling in Shared MapReduce Computation Environment", IEEE International Conference on Cluster Computing and Workshop,(2009) August 31-September 4:New Orleans, USA.

[11]. M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker and I. Stoica, "Delay Scheduling: A Simple Technique for Achieving Locality and Fairness in Cluster Scheduling", Proceedings of the 5th European conference on Computer systems,(2010)April 13-16:Paris,France.

[12]. X. Zhang, Z. Zhong, S. Feng and B. TU ,"Improving Data Locality of MapReduce by Scheduling in Homogeneous Computing Environments", IEEE 9th International Symposium on parallel and Distributed Processing with Applications (ISPA),(2011) May 26-28:Busan,Korea.

[13]. C. Abad, Y. Lu and R.Campbell," DARE: Adaptive Data Replication for Efficient Cluster Scheduling", IEEE International Conference on Cluster Computing (CLUSTER), (2011) September 26-30: Austin, USA.

[14]. X. Zhang," Using Class-Center Vectors to Build Support Vector Machines", IEEE Signal Processing Society Workshop Neural Networks for Signal Processing IX,(1999) August 23-25:Madison,USA.