International Journal of Engineering & Technology, 7 (2.4) (2018) 20-22

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

A vibrant data placement approach for map reduce in diverse environments

J. Sujatha [1] *, K. Meena [2]

[1] *Research Scholar, Veltech Rangarajan Dr. Sagunthala R&D Institute Science & Technology, Chennai*
[2] *Associate Professor, Department of Computer Science & Engineering, Veltech Rangarajan Dr. Sagunthala R&D Institute Science & Technology, Chennai[3]Affiliation of the third author*
**Corresponding author E-mail: jsujathacse@gmail.com*

Abstract

Map reduce assumes that the computing capacity is same for each node in a cluster. Each node is assigned to the same load in homogeneous environment, hence it fully use the resources in the cluster. In such a cluster, there is likely to be various specifications of PCs or servers, which causes the abilities of the nodes to differ. If such a heterogeneous environment still uses the original Hadoop strategy that distributes data blocks into each node equally and the load is also evenly distributed to each node, then the overall performance of Hadoop may be reduced. The major reason is that different computing capacities between nodes cause the task execution time to differ so that the faster executionrate nodes processing local data blocks faster than other slower nodes do.The required data should be transferred from another node through the network. Because waiting for the data transmission time increases the task execution time, it causes the entire job execution time to become prolonged.

Keywords: *Map Reduce; HDFS; Dynamic Data Placement (DDP); File Systems; Data Nodes.*

1. Introduction

Map Reduce is an efficient programming tool used mainly in cloud computing and large scale data parallel applications. Hadoop is an opensource implementation of the Map Reduce [1] [2] model. It plays a vital role in data intensive applications including data mining and web indexing [3]. The current Hadoop implementation presumed that nodes in a cluster have equal computing capacity. The local tasks may increase overhead which in turn reduces the performance of Map Reduce [4] [5]. Hence, data placement algorithm is used in this paper to solve the unbalanced node workload problem.
The proposed method can dynamically adapt and balance data stored in each node based on the computing capacity of each node in a heterogeneous Hadoop cluster. The proposed method reduces data transfer time which in turn enhances the Hadoop performance. The experimental results show that the dynamic data placement policy can decrease the time of execution and improve Hadoop [6] performance in a heterogeneous cluster.
Cloud computing is a type of parallel distributed computing system that has become a frequently used computer application. Map Reduce is an effective programming model used in cloud computing and largescale dataparallel applications. Hadoop is an opensource implementation of the Map Reduce model, and is usually used for dataintensive applications such as data mining and web indexing [7] [8].
Therapid development of the Internet, network service has become one of the most frequently used computer applications. Search engine, webmail & social network services are currently indispensable data intensive applications, due to increasing usage of web services, processing a large amount of data by many people anciently can be a substantial problem. Currently, the method for processing a large amount of data involves adopting parallel computing. Since then the Google File System11 and Bitable have used Map Reduce to construct a data center that can process at least 20 pet bytes a day. The scalability, simplicity, and fault tolerance of the Map Reduce model, it is frequently used in parallel data processing in large-scale clusters.

2. Proposed method

The proposed data placement algorithm is used to resolve the unbalanced node workload problem.

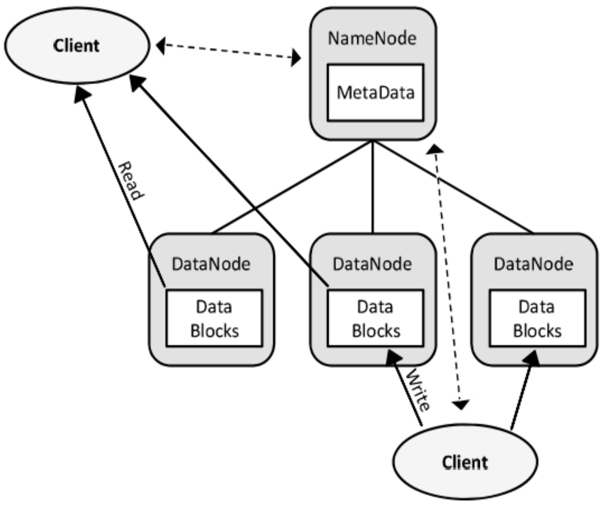

Fig. 2: The Overview of HDFS Read and Write.

The proposed technique, can dynamically adapts and balance data which is stored in each node mainly depends on the computing capacity of each node. The proposed method can reduce data transfer time to achieve improved Hadoop performance.
The experimental results show that, dynamic data placement policy is used to decrease the time of execution hence it enhances the Hadoop performance in a heterogeneous cluster. HDFS is defined as clustered file management system that aims to carry numerous datasets and provides better throughput and faster access to information.

3. Map reduce

Map Reduce exhibits several advantages that differ from those of traditional parallel computing systems. First, regarding scalability, even when new machines are added to a cluster, the system still works well without reconstruction or much modification.
Second, regarding fault tolerance, the MapReduce model can automatically manage failures and mitigate complexity of fault tolerance mechanisms. When a machine fails, MapReduce moves the task that was run on the failedmachine to be rerun on another machine. Third, regarding simplicity, programmers can use the MapReduce model without needing to understand thoroughly the details of parallel distributed programming.

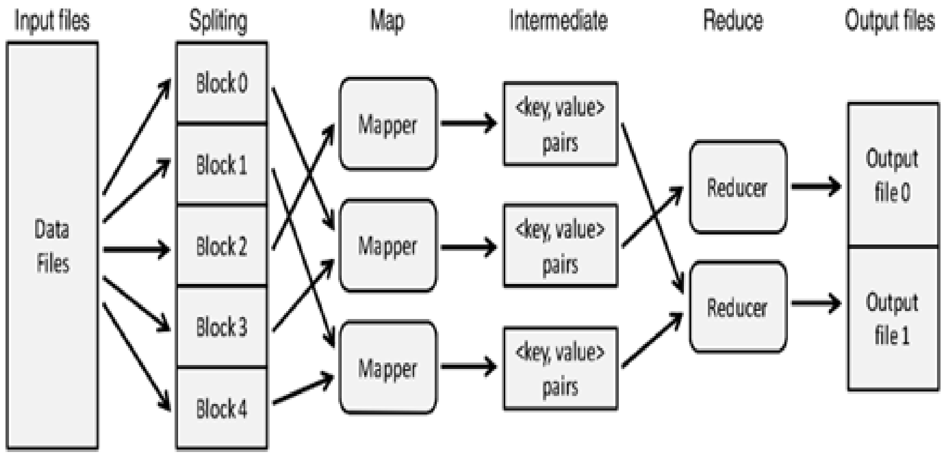

Fig. 3: Map Reduce Model.

HDFS Architecture (Read & Write)
1) Name Node and Data Nodes.
- An HDFS cluster is made up of a single Name Node.
- Server is acting as a master managing the file access and name space regulations.

2) The File System Namespace.
- HDFS supports a file structure.
- Directories can be created by user or an application.
- Files are stored inside those directories.

3) Data Replication.
- HDFS is programmed to manage last file.
- It is used to store large clusters of data mines structures while ensuring reliability.

Heterogeneous cluster having the property as the computing capacity for each node is unequal. Moreover, for different types of job, the computing capacity ratio of nodes are also not the same. Therefore, a Dynamic Data Placement (DDP) Approach is presented according to the types of jobs for adjusting the distribution of data blocks. The proposed DDP algorithm consists of two phases: the first phase is performed when the input data are written into the HDFS, and the second phase is performed when a job is processed.

Table 1: Example: Ratio Table

Content	Node A	Node B	Node C
Word Count	3	15	1
Grep	25	15	1

Data Flow Diagram:

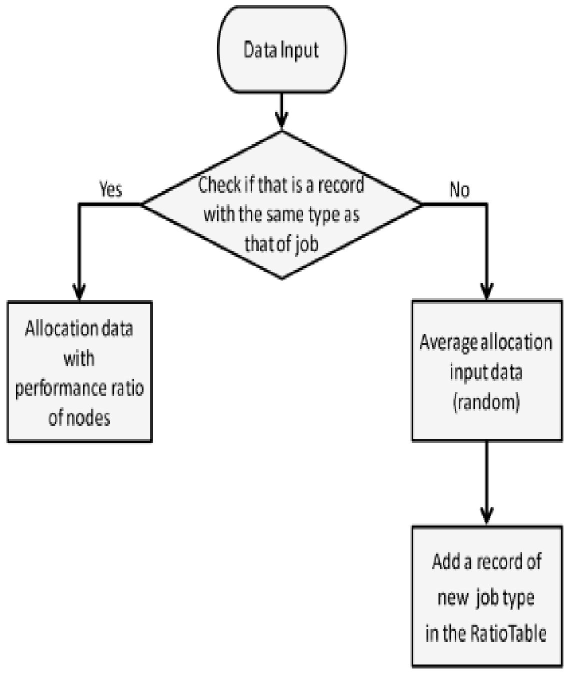
Data Input
Check if that is a record with the same type as that of job
Yes
No
Allocation data with performance ratio of nodes
Average allocation input data (random)
Add a record of new job type in the RatioTable

Fig. 4: Data Flow Chart – Phase 1.

Algorithm 1– Initial Data Allocation
Step 1: For each fragment, initialize the counter values equal to zero (i.e. set Mij = 0, where i = 1, 2, n and j = 1, 2, m).
Step 2: Process an access request for the stored fragment.
Step 3: Increase the corresponding access counter of the accessing node by one for the stored fragment and also store the time of corresponding access.

Step 4: If the accessing node is the current owner, go to Step 2. (i. e. Local access, otherwise it is remote access).
Step 5: If the counter of the remote node is greater than the threshold value (t) and all last “t+1” accesses are made in a specified time (T) then reset corresponding fragment’s counter to zero for all the node and transfer the fragment to the node who’s counter value was greater than threshold value (t). Step 6: Go to step 2.
Data Flow Chart – Phase 2

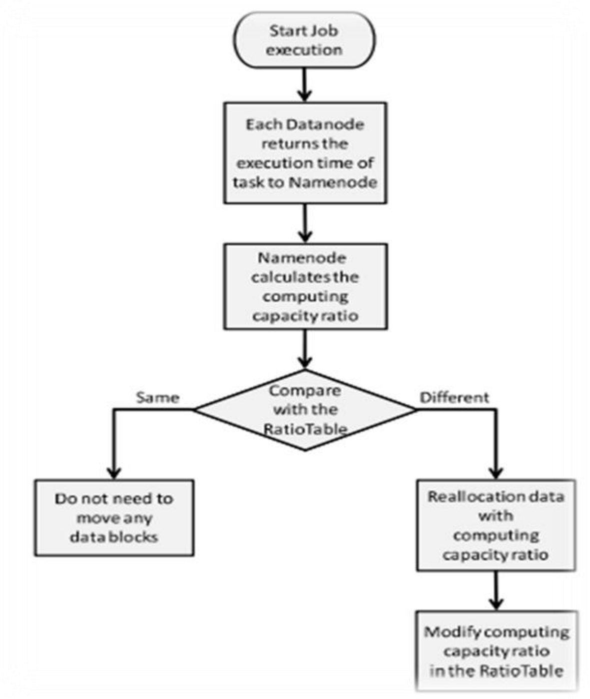
Start Job execution
Each Datanode returns the execution time of task to Namenode
Namenode calculates the computing capacity ratio
Compare with the RatioTable
Same
Different
Do not need to move any data blocks
Reallocation data with computing capacity ratio
Modify computing capacity ratio in the RatioTable

Fig. 5: Data Flow Chart – Phase 2.

4. Experimental results

Machine	Specification		VM Amount	
Hadoop Version	HP G6 Server	16 CPU, 20GB Memory 500 GB HDD	3	1 master 2 slaves
Virtual machine Management	HP G6 Servers	16 CPU, 20GB Memory 500GB HDD	2	2 Slaves
Hadoop Version	Hadoop – 0.20.205.0 (stable Version)			
Virtual machine management	Oracle Virtual			

5. Results

Table 2: Node Specification

Node	CPU	Memory	Disk
Node A	2	2 GB	50 GB
Node B	4	4 GB	50 GB
Node C	2	2 GB	50 GB
Node D	1	1 GB	50 GB
Node E	1	1 GB	50 GB

6. Conclusion

The Map Reduce data placement approach is assumed to be applied in a homogeneous environment. In a homogeneous cluster, the Map Reduce approach makes use of full resources of each node. But, a heterogeneous environment can produce load imbalance hence, it creates the necessity to spend additional overhead. The proposed method considered different computing capacities of nodes to allocate data blocks, hence improves data locality and reduces the additional overhead. Finally in the experiment, for two types of applications, Word Count and Grep, the execution time of the DDP compared with the Hadoop default policy was improved. Regarding WordCount, the DDP can improve by up to 24.7%, with an average improvement of 14.5%. Regarding Grep, the DDP can improve by up to 32.1%, with an average improvement of 23.5%.

References

[1] AmazonElasticMapReduce, http:// aws. amazon. Com /elasticmapreduce.
[2] Apache, http://httpd.apache.org/.
[3] Hadoop, http://hadoop.apache.org/.
[4] Hadoop Distributed File System, http:// hadoop. apache.org/ docs/stable/hdfs_design.html.
[5] HadoopMapReduce,http://hadoop.apache .org /docs /stable /mapred_tutorial. html.
[6] HadoopYahoo, http: // www. ithome. com.tw /itadm/article.php
[7] D.Borthakur, K.Muthukkaruppan, K.Ranganathan, S.Rash, J.-S. Sarma, N.Spiegelberg, D.Molkov, R.Schmidt, J.Gray,H.Kuang,A.Menon,A. Aiyer, Apache Hadoop goes realtime at Facebook, in: SIGMOD ’11, Athens, Greece, June 12–16, 2011. https://doi.org/10.1145/1989323.1989438.
[8] F. Chang, J. Dean, S. Ghemawat, W.-C. Hsieh, D.A. Wallach, M. Burrows, T. Chan- dra, A. Fiker, R.E. Gruber, BigTable: a distributed storage system for structured data, in: 7th USENIX Symposium on Operating Systems Design and Implemen- tation, OSDI’06, 2006, pp. 205–218.
[9] Q. Chen, D. Zhang, M. Guo, Q. Deng, S. Guo, SAMR: a self-adaptive MapRe- duce scheduling algorithm in heterogeneous environment, in: 2010 IEEE 10th International Conference on Computer and Information Technology (CIT), IEEE, 2010, pp. 2736–2743.
[10] J. Dean, S. Ghemawat, MapReduce: simplified data processing on large clusters, in: OSDI ’04, Dec. 2004, pp. 137–150.
[11] S. Ghemawat, H. Gobioff, S.-T. Leung, “The Google file system, in: Proc. SOSP 2003, pp. 29–43. https://doi.org/10.1145/945445.945450.
[12] B. He, W. Fang, Q. Luo, N. Govindaraju, T. Wang, Mars: MapReduce framework on graphics processors, in: ACM 2008, 2008, pp. 260–269. https://doi.org/10.1145/1454115.1454152.
[13] G. Lee, B.G. Chun, R.H.Katz, Heterogeneity-aware resource allocation and scheduling in the cloud, in: Proceedings of the 3rd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud, vol.11, 2011.
[14] C. Tian, H. Zhou, Y. He, L. Zha, A dynamic MapReducescheduler forheteroge- neous workloads, in: Eighth International Conference on Grid and Cooperative Computing, GCC’09, IEEE, 2009.
[15] M. Zaharia, A. Konwinski, A.D. Joseph, R. Katz, I. Stoica, Improving MapReduce performance in heterogeneous environments, in: Proc. OSDI, San Diego, CA, De- cember 2008, pp. 29–42.