IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

Phase Aware Job Scheduling for MapReduce in Hadoop

Snehal Kapde[1], J.V. Shinde[2], N.R. Wankhade[3]

Student, Comp Dept, Late G.N. Sapkal C.O.E., Nashik, India [1]

Asst. Professor, Comp Dept, Late G.N. Sapkal C.O.E., Nashik, India [2]

Assoc. Professor, Comp Dept, Late G.N.Sapkal C.O.E., Nashik, India [3]

Abstract: MapReduce is a framework using which we can write applications to process huge amounts of data, inparallel, on large clusters of commodity hardware in a reliable manner. MapReduce is a model for the data-intensivecomputation. However, despite recent efforts towards designing efficient scheduler to perform MapReduce, the available solutions focuses on scheduling at the task-level offers suboptimal performance in executing jobs. This isbecause tasks can have highly varying resource requirements during their lifetime, which makes it difficult forschedulers to effectively utilize the available resources to reduce job execution time. PRISM-a fine grained phaseand resource aware scheduler was introduce which mainly focuses on designing task level scheduler. The phasebased resource aware scheduler offers high resource utilization and provides improvement in job running time.In my proposed system I have used the virtual memory to overcome the disadvantage of PRISM and using thepausing phase. In this the resource will able to use virtual resource. Instead of going into paused phase it is possibleto use resource virtually till the resource is available. This will surely help the job running significantly reducing the delay.

Keywords: Cloud computing, MapReduce, Hadoop, scheduling, resource allocation.

I. INTRODUCTION

Previous to our development of mapreduce, the authors and many others at google applied hundreds of special-purpose computations thatmethod massive amounts of raw facts, such as crawled files, netrequest logs, and many others., to compute diverse kinds of derived records, which includesinverted indices, numerous representations of the graph structure of webdocuments, summaries of the number of pages crawled according to host, andthe set of most common queries in a given day. Maximum such computations are conceptually straightforward. However, the enter facts is use massive and the computations need to be disbursed across massesor heaps of machines in order to finish in an affordable quantity oftime. The troubles of the way to parallelize the computation, distribute therecords, and cope with disasters conspire to difficult to understand the unique easy computation with huge quantities of complicated code to cope with those troubles. The major contributions of this work are a simple and powerful interface that enables automatic parallelization and distribution of large-scale computations, combined with an implementation of this interface that achieves high performance on large clusters of com- modity PCs. The programming model can also be used to parallelize computations across multiple cores of the same machine.The phase-level scheduling algorithm improves execution time and resource utilization without introducing any stragglers with the help of Virtual Space.The main objective of the phase-level scheduling alongwith the Virtual Space is to reduce the delay that occursduring the "pause" stage of the node manager and also toincrease the job running time and the resource utilizationwhen compared with the task-level scheduling. Phase basedscheduling algorithm. In order to overcome the issue ofdelay during the job execution due to the pause: stage, thevirtual space is allocated to the node manager. Thus theaverage job running time will be improved when comparedwith the phase based scheduling algorithm without virtualspace.

II. OBJECTIVE

- To improve the job running time and resource utilization.
- To improve the data locality.

III. LITERATURE SURVEY

1] H. Herodotou, H. Lim, G. Luo, N. Borisov, L. Dong, F. Cetin, and S. Babu, "Starfish: A self-tuning system for big data analytics,"in Proc. Conf. Innovative Data Syst. Res., 2011, pp. 261–272.

Starfish can provide accurate resource information that can be used by the scheduler so that it can take effective scheduling decisions and reduce the job execution time. By using such job profiler integrated within the scheduler, the performance of the scheduler is effective than existing task level scheduler. It collects the past executed jobs profile information at a fine granularity for job estimation and automatic optimization. However, collecting detailed

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

job profile information with a large set of metrics generates an extra overhead, especially for CPU-intensive applications.As a result, Starfish overestimate the execution time of a Hadoop job.

2]J. Polo, C. Castillo, D. Carrera, Y. Becerra, I. Whalley, M. Steinder,J. Torres, and E. Ayguade, "Resource-aware adaptive scheduling for MapReduce clusters," in Proc. ACM/IFIP/USENIX Int. Conf.Middleware, 2011, pp. 187–207.

-In the slot-based resource allocation scheme, the physical resources on each machine are captured by the number of identical slots that can be assigned to tasks. The problem with that slot-based resource allocation is that the run-time resource consumption varies from task to task and from job to job.

3] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," Commun. ACM, vol. 51, no. 1, pp. 107–113, 2008.

The input data is usually large and the computations have to be distributed across hundreds or thousands of machines in order to finish in a reasonable amount of time. The issues of how to parallelize the computation, distribute the data, and handle failures conspire to obscure the original simple computation with large amounts of complex code to deal with these issues.

As a reaction to this complexity, a new abstraction that allows us to express the simple computations we were trying to perform but hides the messy details of parallelization, fault tolerance, data distribution and load balancing in a library was introduced. The use of a functional model with user-specified map and reduce operations allows us to parallelize large computations easily and to use re execution as the primary mechanism for fault tolerance.

The automatic parallelization and distribution of large-scale computations, combined with an implementation of this interface achieves high performance on large clusters of commodity PCs. Still the operation is applicable at task level only that can be further divided into phases and the further parallelization in phases can cause the overhead problem.

IV. RELATED WORK

The maximum famous implementation of mapreduce is apache hadoop mapreduce [1]. A hadoop cluster consists of a huge quantity of commodity machines with onenode serving as the master and the others appearing as slaves. The master node runs a useful resource supervisor (also referred to as a jobtracker) that is liable for scheduling task on slave nodes. Each slave node runs a neighborhood node manager (also referred to as a task tracker) this is chargeable for launching and allocating sources for each task. To accomplish that, task tracker launches a Java Virtual Machine (JVM) that executes the corresponding map or reduce task. As a hadoop cluster is mostly a multi-consumer machine, many customers can concurrently publish jobs to the cluster. The job scheduling is accomplished by using the useful resource manager inside the master node, which continues a listing of jobs inside the system.

Each slave node monitors the progress of each running task and available resources at the node, and periodically (usually among 1-3 seconds) transmit a heartbeat message to deliver this facts to the master node. The resource scheduler will use the furnished data to make scheduling decisions. Current hadoop activity schedulers carry out task-level scheduling where tasks are taken into consideration as the greatest granularity for scheduling. In the map section, a mapper fetches input data block from the hadoopDistributed File System [4] and applies the consumer-described map characteristic on every record. The map characteristic generates facts which might be serialized and amassed right into a buffer.

V. PROPOSED SYSTEM

We present a prism, such that it allocates a fine-grained assets at the phase-level to perform process scheduling. Prism specially consists of 3 additives: first one is the phase based totally scheduler at master node, local node manager at segment transaction with scheduler and job development monitor to capture phase -level information.

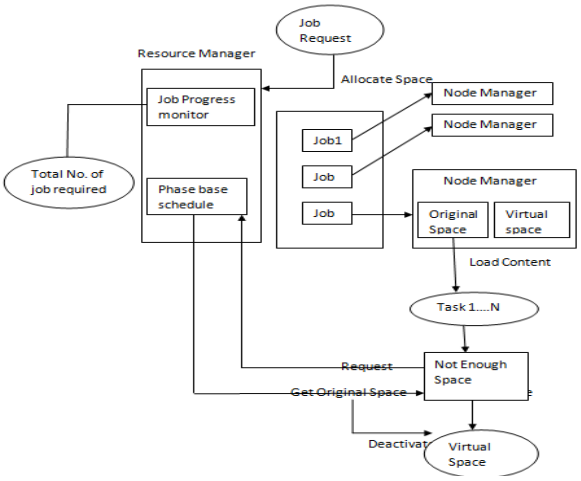
To achieve those three stages, will carry out a phase-level scheduling mechanism. When the task needs to scheduled from node manager, scheduler replies with assignment scheduling request. Then node manager launches a task. After final touch of its execution of phase, alternatively subsequent task will launches. Even as proceeding these phases, it will pause for a while to remove the useful resource battle. At the same time as proceeding a task, it has many run-time resources within its lifetime.

While scheduling the process, PRISM offers higher degree of parallelism than present day hadoop cluster. It refers on the phase-level to improve aid utilization and performance. The primary objective of the phase-stage scheduling alongside the digital space is to lessen the delay that occurs for the duration of the "pause" level of the node manager and additionally to boom the process going for walks time and the aid usage when in comparison with the mission-level scheduling.

Segment primarily based scheduling algorithm. So as to overcome the issue of put off at some point of the job execution because of the pause: stage, the digital space is allocated to the node manager. As a consequence the average activity running time could be improved while as compared with the segment primarily based scheduling algorithm without virtual area.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

VI. ARCHITECTURE DIAGRAM



VII. MODULES

There are 3 modules to be used. They are as follows

• HadoopMapReduce

It is simple slot-based allocation scheme. It will not take any run time sources whilst enforcing task. To begin with it has one hadoop cluster, consisting of 1 large machine as master node and its miles related to many slave nodes. The responsibility of master node is to scheduling process to all slave nodes. On this module simple mapper and reducer features could be cope with by using the task. Here hadoop allotted file system offer statistics blocks to all map and reducetasks.

• Prism

At the same time as allocating the resources, sometimes resources can be idle or resouces are run -time useful resource. If they're idle, resource allocation must be wasted. So run-time resources stimulate to develop fine-grained resources on the phase level to acquire unique volumes of data in single device such that it improve useful resource usage in comparison to the opposite tasks. The key difficulty is that once one task has finished in phase level, next phase of task is not scheduled without delay. It'll “pause” for some time time to remove resource conflict then proceed next phase.

• Design Rationale

The responsibility of MapRduce is to assigning task with consideration of efficiency and fairness. It must maintain high resource utilization in cluster and job running time implies job execution.

VIII. CONCLUSION

Mapreduce is programming model for cluster to perform a data intensive computing. In this paper we especially exhibit that, if the resources attention on task level, execution of each task may divided into many levels. Even as executing those phases, many breaking- down of map and reduce responsibilities will takes location and execute them in a parallel throughout a massive wide variety of system, so that you can reduce running time of of data-intensive job.so they may perform resources allocation at the phase-level. We can introduce prism on the phase-level. Prism show that, how run-time resources may be used and the way it varies over the long existence time. Prism improves process execution set of rules and overall performance of sources without introducing stragglers.

REFERENCES

[1] HadoopMapReduce distribution [Online]. Available: http://hadoop.apache.org, 2015.
[2] Hadoop Capacity Scheduler [Online]. Available: http://hadoop.apache.org/docs/stable/capacity_scheduler.html/, 2015.
[3] Hadoop Fair Scheduler [Online]. Available: http://hadoop.apache.org/docs/r0.20.2/fair_scheduler.html, 2015.
[4] Hadoop Distributed File System [Online]. Available: hadoop.apache.org/docs/hdfs/current/, 2015.
[5] GridMix benchmark for Hadoop clusters [Online]. Available:http://hadoop.apache.org/docs/mapreduce/current/gridmix.html, 2015.
[6] PUMA benchmarks [Online]. Available: http://web.ics.purdue.edu/fahmad/benchmarks/datasets.htm, 2015.
[7] The Next Generation of Apache HadoopMapReduce[Online]. Available: http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html, 2015.
[8] R. Boutaba, L. Cheng, and Q. Zhang, “On cloud computationalmodels and the heterogeneity challenge,” J. Internet Serv. Appl.,vol. 3, no. 1, pp. 1–10, 2012.
[9] T. Condie, N. Conway, P. Alvaro, J. Hellerstein, K. Elmeleegy, andR. Sears, “MapReduce online,” in Proc. USENIX Symp. Netw. Syst.Des. Implementation, 2010, p. 21.
[10] J. Dean and S. Ghemawat, “Mapreduce: Simplified data processingon large clusters,” Commun. ACM, vol. 51, no. 1, pp. 107–113, 2008.
[11] Savitri D.H, Narayana H.M, "PRISM: Allocation of Resources in Phase-level using MapReduce in Hadoop," in International Journal ofResearch in Science and Engineering Vol. 1, Special Issue: 2.
[12] T. Condie, N. Conway, P. Alvaro, J. Hellerstein, K. Elmeleegy, and R. Sears, "MapReduce online," in Proc. USENIX Symp. Netw. Syst.Des. Implementation, 2010, p.21.
[13] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, and K. Talwar,“Quincy: Fair scheduling for distributed computing clusters,”in Proc. ACMSIGOPS Symp. Oper. Syst. Principles, 2009, pp. 261–276.
[14] Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker,and I. Stoica, “Dominant resource fairness: Fair allocation of multiple resource types,” in Proc. USENIX Symp. Netw. Syst. Des. Implementation, 2011, pp. 323– 336.
[15] Cloudera: 7 tips for Improving MapReduce Performance. http://www.cloudera.com/blog/2009/12/ 7-tips-for-improving-mapreduce-performance.
[16] Hadoop distributed file system, http://hadoop.apache.org/hdfs/
[17] Zaharia M, Konwinski A, Joseph AD, Katz R, Stoica I (2008) ImprovingMapReduce performance in heterogeneous environments. In: Proc. OSDI
[18] Isard M, Budiu M, Yu Y, Birrell A, Fetterly D (2007) Dryad: distributed data-parallel programs from sequential building blocks. In: Proc. Eurosys, March 2007, pp 59–72
[19] Y. Chen, A. Ganapathi, R. Griffith, and R. Katz, “The case for evaluating mapreduce performance using workload suites,” in IEEE MASCOTS 2011, pp. 390–399.