Self-learning dynamic adjustment scheduling algorithm based on Hadoop

Fucong Li [a], Zhuyu Li[b], Guohui Chen[c], Xiangxin Li[d]

School of Computer and Communication Engineering, Beijing Engineering

University of Science and Technology Beijing, Beijing 100083, China

[a] li274860615@163.com,[b]xiaoyu3696@126.com,[c]chenguohui3344@163.com

[d]lixiangxinustb@163.com

Keywords: Hadoop; Scheduling algorithm; Speculative execution; Slow tasks

Abstract. Speculative execution is the key technology to improve the execution efficiency of Hadoop cluster. In the large-scale cluster reasonable Speculative execution can effectively reduce the execution time for the job. At present, there is still a big problem in the detecting of the Hadoop's Speculative execution .Therefore this paper proposes a self-learning dynamic adjustment scheduling algorithm. The algorithm firstly study the impact of the historical information of the task execution, dynamically adjusts the time proportions of each stage of Reduce tasks And taking into account the tasks of the different types of load effect on the detection of slow tasks .Found out the real impact of the job execution time of the slow tasks to avoid performing unnecessary backup tasks. Through the experiment, the average execution time of SLDA scheduling is compared with LATE, and the average execution time of the job is significantly reduced.

Introduction

A certain scale of the Hadoop cluster is usually composed of different hardware configuration of nodes. In a cluster of heterogeneous environments, since the performance of the different nodes so that part of the task execution speed lag behind other tasks. The slow tasks will extend the completion time of the job and due to the insufficient of existing mechanisms in the detection method of the slow tasks will lead to many unnecessary tasks is judged to be slow tasks, so start the backup task. Too many unnecessary backup tasks will take up a large number of cluster resources, and then lead to the low efficiency of other tasks in the cluster. Through based LATE scheduling algorithm. To improve the insufficient of detection slow tasks this paper propose self-learning dynamic adjustment scheduling algorithm(SLDA), firstly the task is classified according to the load type of the task. Secondly according to the historical information of the task execution to dynamically adjust the proportion of each stage of the Reduce task, accurate calculation of the remaining time of the task to find out which task is the real one dealy the job's execution time. Reduce the execution time of the job, and improve the performance of the whole cluster.

LATE scheduling algorithm

Zaharia et.al in the literature [2] proposed LATE（Longest Approximate Time to End）scheduling algorithm. LATE scheduling algorithm always chose those tasks which have more remaining time than other tasks as slow tasks and launches backup tasks for those tasks. Suppose: in this phase The number of processed data is M, In this phase need to be process data is N, the task already running time T,Progress for the task is progress ,Task processing rates is ProgressRate ,remaining time is TimeToEnd. As shown below.

$$progress = M/N \quad (1)$$

$$progress = 1/3 * (K + M/N) \quad (2)$$

$$progressRate = progress/T$$

$$TimeToEnd = (1 - Progress)/ProgressRate$$

The insufficient of LATE scheduling algorithm

Estimation the remaining time of the task is not accurate. LATE still use the value of each phases of Reduce tasks(copy、sort、reduce) are1/3、1/3、1/3.For example copy、sort、reduce are 60%、20%、20%respectively In a running job. When the first stage copy finished in T seconds. IN LATE scheduling algorithm remaining timing is :67%3 (T/ 33%) = 2T.Actually the Reduce Task needs 40%3 (T/ 60%) = 0. 673 T to finish the task.

SLDA scheduling algorithm

In this paper, we propose a SLDA (Dynamic Adjustment Self-Learning) scheduling algorithm. The main shortcomings of LATE scheduling are improved. Accurate estimate of the current task progress is the way to find the real slow tasks. SLDA scheduling algorithm based on the task execution of the historical information to dynamically adjust the proportion of each stage of the Reduce tasks so It is possible to find out the real task who delay the job completion time.

Workload classification

Hadoop clusters running different types of jobs at the same time, these jobs bring different types of loads to the cluster, these loads include CPU, memory, disk, and network. Because resources are limited, Classifying these jobs by load type is very important for task scheduling. Literature [7] proposed a method for classification of jobs. The work load is divided into two categories: I/O-bound and Compute-bound. This technique through calculate the related data of Map tasks to classify load type. Map task execution can be divided into three steps

(1) Input data initialization

(2) Map task execution

(3) The output stored locally

Symbol definition as shown in table 1 [6].

Table1 Symbol definition table

Symbol	Instruction
MID	Map Input Data
MOD	Map Out Data
λ	Parameters proportion
MTCT	Map Task Completion Time
DIOR	Disk I/O Rate
n	Map Task Nums

The Map output data and the Map input data proportion (λ) depends on the type of workload

$$MOD = \lambda \times MID \quad \lambda > 0 \tag{3}$$

Each node may be assigned the Map task of n that share system disk I / O bandwidth. Therefore, if the MID and MOD data transmission time is greater than the Map task completion time, this kind of work is I/O – bound [6].

$$\frac{n \times (MID + MOD)}{DIOR} > MTCT \tag{4}$$

By (3) and (4) available

$$\frac{n \times (1 + \lambda) MID}{DIOR} > MTCT$$

Compute-bound is different from I/O-bound. MID and MOD data transfer time than the Map task completion time is short.

$$\frac{n \times (MID + MOD)}{DIOR} \le MTCT \tag{5}$$

By (3) and (5) available

$$\frac{n \times (1+\lambda) \mathrm{MID}}{DIOR} \le MTCT$$

In a real environment, the value of MTCT may be affected by the current load of TaskTracker. The work load on the node may cause the MTCT to be larger than the ideal value. This means that according to (4), (5). If a job is defined to be I/O-bound, it must be I/O-bound. But if a job is defined as Compute-bound, it may not be a Compute -bound type. In [7] the proposed solution is to run a Benchmark at the same time to test the work, If the execution time of the Benchmark is not longer, the test must be Compute -bound.

Self-learning dynamic adjustment

Reduce task is divided into three phases(copy、sort、reduce) The execution time Proportion of each phase is R1, R2, R3. At the same time must meet R1 + R2 + R3 = 1 Using records of historical information of all nodes dynamically obtained the proportion from the average historical information to adjust the proportion in different phases of the value.

Suppose：Calculate the current progress of the task in the following ways。The number of processed data is M, In this phase need to be process data is N Current phase of the number of K（0、1、2） .The progress in the phase is subProgress The progress of the task is Progress

The progress in the phase is:

sub Progress =M /N

Progress of Reduce tasks is：

If K=0

Progress =R1 * sub Progress

If K=1

Progress =R1 + R2 * sub Progress

If k=2

Progress = R1 + R2 + R3 * sub Progress

According to the Progress use to accurate calculation the remaining time of the task. Choose the longest remaining time task as the slow task. SLDA scheduling algorithm to define three thresholds

(1) SpeculativeCap: The largest number of backup tasks performed at the same time in the system.

(2) SlowNodeThreshold: is used to classify TaskTrackers into fast TaskTrackers and slow TaskTrackers. The backup task is placed in the fast node to process.

(3) SlowTaskThreshold :is used to classify tasks into fast and slow tasks, If the taskRate and the AveTaskRate of the task can be satisfied: TaskRate <(1- SlowTaskThreshold)*AveTaskRate.

Then the task is slow task and execute backup task for this task. To avoid waste cluster resources start the backup task for a non slow tasks on a free node. If a TaskTracker sends a request to the JobTracker for a new task and the current number of backup task being executed less than SpeculativeCap then

(1)Analyzing request task node speed if less than SlowNodeThreshold then this request will be ignored, otherwise continue.

(2)Sort the tasks by the TimeToEnd .

(3)Choose the longest remaining time task and the speed is less than TaskThreshold, Start backup tasks for this task

Experimental results and performance analysis

In this paper, we build a heterogeneous test environment through virtual machine. All virtual machines using VMware Workstation9.0.1, Install Ubuntu 12.04, JDK version 1.7. 0.25, using Hadoop version 0.22. The cluster consists of 1 Master and 3 Slave(Slave1.Slave2,Slave3). Each node hardware Configuration as shown in table 2

Table 2 The cluster environment configuration

nodes	Memory/GB	Cpu core	Hard disk/GB
Master	4	2	60
Slave1	2	2	60
Slave2	2	2	60
Slave3	1	2	40

Choose TeraSort, GrepCount, WordCount as test program..TeraSort belong to I / O-Bound-type,Grep-Count belong Compute -bound type, WordCount belong to Class Sway type. Experiments set three best parameter values have demonstrated in the literature[2].Itis SpeculationCap 20%, SlowTaskThreshold 25%, SlowNodeThreshold 25%. Test data volume is 1.4GB. Hadoop cluster was tested by LATE scheduling and SLDA scheduling respectively. Each scheduling algorithm runs 5 times, and calculate the average execution time is shown in Figure 3.

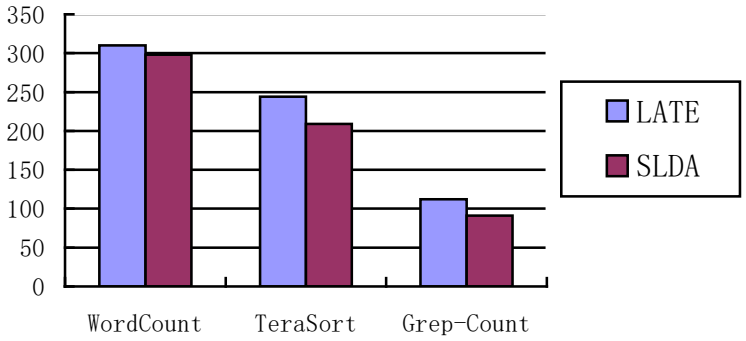

Fig 1 Comparison of different algorithms execution time

SLDA scheduling algorithm performs TeraSort 、GrepCount execution time less 16% and 18% than the LATE scheduling algorithm, When performing wordCount, Two kinds of scheduling algorithm s execution time is not big. Because SLDA scheduling algorithm improvements mainly for IO-bound and Compute -bound type work and WordCount‘s load type between in IO-bound and Compute -bound, the CPU load and IO load of WordCount are both very large, So the advantages of the SLDA scheduling algorithm is not obvious.

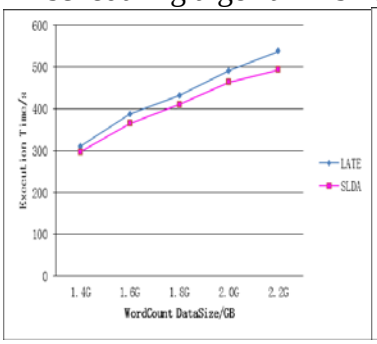

Fig2 WordCount Execution time

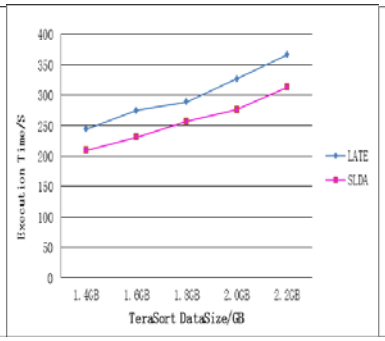

Fig3 TeraSort Execution time

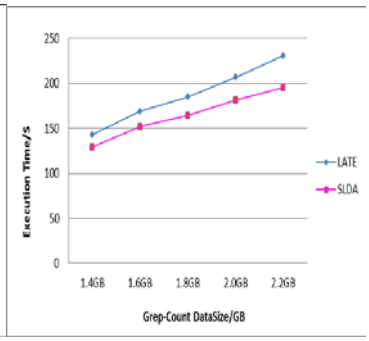

Fig4 TGrepCount Execution time

Fig 2, Fig3, Fig 4 shows the WordCount, TeraSort and GrepCount three jobs to test the input data for 1.4GB, 1.6GB, 1.8GB, 2.0GB, 2.2GB， in turn with the LATE scheduling algorithm and the improvement of the SLDA scheduling algorithm execution time. With the increasing amount of input data, SLDA scheduling algorithm’s advantage began to gradually clear。The reason is that with the increase of the amount of input data, the load of the cluster increases, and the probability of the slow tasks becomes larger. SLDA scheduling algorithm can more accurately detect the slow tasks, reducing load on the cluster and shorten the completion time of the job. It is proved that the SLDA scheduling algorithm of this paper has significantly improved the performance of the cluster.

Conclusion

Through comparative analysis of existing LATE scheduling algorithms, fully considered characteristics of the Hadoop platform proposed SLDA scheduling algorithm .To solve the problem of inaccurate detection of slow tasks. The improved algorithm was test on local Hadoop cluster. Experimental data show that the algorithm can choose the appropriate tasks as slow tasks to avoid the waste of resources, reduce the job completion time ,Improve the efficiency of the Hadoop platform.

References

[1] Boutaba R, Cheng L, Zhang Q. On cloud computational models and the heterogeneity challenge[J]. Journal of Int Services and Applications, 2012,

[2]Zahafia M，Konwinski A，Joseph A.Improving MapReduce performance in heterogeneous environments[C]//Proc of the 8th Usenix Symp on Operating Systems Design and Implementation,2008：29- 42.

[3]HADOOP-3759：Provide ability to run memory intensive jobs without affecting other running tasks on the nodes

[4]Hadoop Capacity Scheduler http://hadoop.apache.org/docs/r0.19.0/capacity_scheduler.html

[5]Hadoop Fair Scheduler http://hadoop.apache.org/docs/r1.2.1/fair_scheduler.html

[6] HU Dan, YU Jiong, YING Changtian, et al. Improved LATE scheduling algorithm on Hadoop paltform. Computer Engineering and Applications, 2014, 50（4）:86- 89.

[7] Chao T ，Zhou H ，He Y，et al.A dynamic MapReduce scheduler for hetergeneous workloads[C]//Proc of the 8th International Conference on Grid and Cooperative Computing，China，2009：2 18- 224 .

[8] Quan Chen, Daqiang Zhang, Minyi Guo, Qianni Deng, Song Guo, “SAMR: A Self-adaptive MapReduce Scheduling Algorithm in Heterogeneous Environment,” Computer and Information Technology (CIT), 2010 IEEE 10th International Conference

[9] Dean J, Ghemawat S. MapReduce: Simplified Data Processing on LargeClusters[J].Communications of the ACMS2008,51 (1) : 107-113

[10] Changhang Lin, Wenzhong Guo et al. Self-Learning MapReduce Scheduler in Multi-job Environment , 2013 International Conference on Cloud Computing and Big Data 2013

[11] Huanle Xu, Wing Cheong Lau et al Resource Optimization for Speculative Execution in a MapReduce Cluster 978-1-4799-1270-4/13 c!2013 IEEE