Maximum Satisfaction Scheduling algorithm Based on Hadoop Architecture

Chen Kuan-ting[1, a], Huang Jian-hua[1, b], JIN Yi[1, c] and HE Xi[1, d]

[1]School of Information Science and Engineering, East China University of Science and Technology, Shanghai200237, China

[a]413151124@qq.com, [b]jhhuang@ecust.edu.cn, [c]jinyi@ecust.edu.cn, [d]hxhome@ecust.edu.cn

Keywords: Hadoop; MapReduce; scheduling algorithm; resources;

Abstract: Based on the MapReduce job scheduling technology for design reference, this paper has put forward the maximum satisfaction scheduling algorithm of Hadoop to effectively solve the scheduling problems in MapReduce. The algorithm has tried to modify the original algorithm of Hadoop, configure a satisfaction score for each submitted job, and obtain the maximum satisfaction score of the job under the same Hadoop system environment of hardware and software. Compared with the own scheduling algorithm of Hadoop—fair share scheduling, the experiment has eventually drawn the conclusion that the maximum satisfaction scheduling algorithm can get the outcomes that customers want, reduce a certain degree of scheduling time, and enhance the system throughput.

Introduction

Nowadays, big data technology has permeated into every corner of the world. With the development of parallel computing, distributed computing and grid computing, a new computing model based on cloud computing has emerged. Google, Microsoft, Amazon and other companies serve as the pioneers in cloud computing [1]. The birth of cloud computing does not only exert great influence on the scientific community, but also causes a stir on other industries. From 2003 to 2004, two papers published by Google respectively proposed GFS (Google File System) based on Distributed File System and MapReduce programming ideas so that a cluster of cheap, simple and efficient distributed computing framework was established [2]. It made large-scale data calculation possible. Hadoop, an open source distributed system of Google, is low in cost and high in efficiency. It is also flexible and secure [3].

MapReduce is one of the core technology of Hadoop platform. The use and selection of its job scheduling technology is closely related to the performance of Hadoop platform. At present, job scheduling algorithms all have to suffer from long task response time, inability in making full use of resources and considering the configuration parameters and failing to meet the diversity of tasks and services [4]. It can be seen that it is urgent to solve such a problem as whether job scheduling technology can effectively revitalize resources of clusters in current research.

This paper improves the weight in job scheduling techniques, and proposes a Maximum Satisfaction Scheduling (MSS) algorithm based on Hadoop Architecture. It aims at seeking the maximum scores with the existing system resources in given environment.

Related Works

Currently, there are three basic scheduling algorithms for Hadoop architecture [5], they are FIFO [6] scheduling (hereinafter referred to as FIFO), fair scheduling algorithm [7] (hereinafter referred to as Fair), and Capacity Scheduling [8] (hereinafter referred to as Capacity). The three all fall into the categories of job/task scheduling model with attachment resource in the form of queue.

As the core of job scheduling technology, JobTracker does not only need to schedule the task submitted by the client, but also makes calculation in a reasonable order. Furthermore, it should improve the system throughput and take the resource utilization into account so that Hadoop can

engage in more jobs. Besides, the efficiency must be improved so that the system can keep efficient for the sake of users. Hadoop scheduling model is shown in Fig.1.

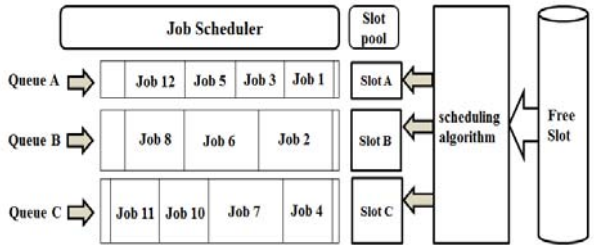

Fig.1 The architecture of Hadoop scheduling

In job scheduling, there is a pluggable module in Hadoop, so many experts and scholars at home and abroad intend to modify or redesign the scheduler according to their own needs with the view of solving the existing scheduling problems. Thus, The job scheduling technology of Hadoop is studied deeply.

Fischer[9] et al built a model to prove that Hadoop task allocation is the NP-complete problems. They proposed the flow-based MaxCover-BalAssign algorithm, but its time complexity is $O(m^2n)$, which was apparently impractical for some running time-critical applications. Thawari [10] et al put forward a heuristic computing scheduling algorithm Balance-Reduce so as to shorten job execution time gradually. But it has not been applied on Hadoop platform. Aiming at the shortage of MapReduce scheduler of Hadoop, Kai and Tian [11] proposed a fair scheduling algorithm boasting priority and weight. Xu Cheng, Liu Liang Tan [12], also proposed a new task monitoring program with the view of reducing the burden JobTracker node. Li Qianmu, Zhang Shengxiao and Lu Lu[13] proposed a Max-D scheduling algorithm catering to heterogeneous clusters. However, the above algorithms cannot implement the improvement in priorities. It makes users feel poor perception.

In brief, there have been a lot of domestic and foreign researchers with much interest in MapReduce job scheduling technology. They devote to improving MapReduce job scheduling technology and researching on the task priority. However, most of the studies are not comprehensive in different scenarios and need for further study and discussion.

The maximum satisfaction scheduling algorithm

Hadoop MapReduce is built on HDFS (Hadoop distributed file system) which includes a NameNode and many DataNode. As the core node of Hadoop, NameNode is mainly responsible for scheduling and management. DataNode, as a task node, takes in charge of tasks assigned by JobTracker.

Hadoop MapReduce belongs to a system programming model, including Map and Reduce functions. When a job is submitted to the Hadoop system by users, the input data will be divided into a number of equal long blocks of data (the default is 64MB), among which each block corresponds to a Map task. The numerous Map tasks can simultaneously be executed in order to process data in parallelization. After the task completion, Map data will be sorted and distributed into Reduce task for further processing.

In addition, Slot is an important concept of Hadoop platform. Slot is a logical concept and the numbers of nodes are used to represent the resource capacity of a node, thus Slot is a resource unit of Hadoop. Hadoop resorts to Slot to manage and allocate the resource at nodes. Each job application resource takes Slot as a unit and each node determines its own computing capacity and memory so as to determine the total amount of Slot. When a job starts, it is necessary to apply Slot from JobTracker first so that JobTracker can allocate a free Slot for the job. After the job is finished, the Slot will be returned.

Theoretical Derivation of Maximum Satisfaction Scheduling Algorithm

The main design system of the maximum satisfaction scheduling must follow the following rules:

(1) Obtaining Slot resource conditions. A job can be firstly configured as Slot resource as long as it has the largest processing capacity; the highest satisfaction scores and the least rest file to be processed.

(2) Job calculation rules. If a job is most likely to witness satisfaction scores after task completion, it means that this job can spend the least time to complete the calculation task and gain a satisfaction score. Then the job should be configured as Slot firstly for MapReduce calculations.

(3) To make sure of job Slot resources in the Reduce phase. After calculation in the Map phase, the process will come into the Reduce phase. If the job has not been assigned with adequate resources in Reduce, then the entire MapReduce calculation time will be slowed down.

The following will engage in the specific steps for the maximum satisfaction scheduling algorithm. The implementation steps of the maximum satisfaction scheduling algorithm are as follows.

(1) Entering satisfaction scores. A user should set a satisfaction score before each job is submitted. Here, rules and orientations of the specified satisfaction scores (Satisfaction score, hereinafter referred to as Sat) are entirely determined by the user. If the user believes that the job should be completed firstly, then he can set a relatively high satisfaction score to the job and so on. The examples about the job satisfaction are shown in Table 2:

Table 2 Satisfaction score for each job

Job_n	Satisfaction score
Job_1	$Sat_1 = 200$
Job_2	$Sat_2 = 120$
Job_3	$Sat_3 = 80$
Job_4	$Sat_4 = 90$
Job_5	$Sat_5 = 100$
Job_6	$Sat_6 = 180$

After the user enters satisfaction score, advance to step (2).

(2) calculation of Map computing performance. After getting satisfaction scores, according to the existing resources of the calculation conditions, and in the set time, the system can complete a certain amount of data of the Map calculation. Map processing performance refers to the amount of data that can be processed by each Slot per second. It can speculate the possible time in Map phase in accordance with file size and time processed by the job. The purpose of calculating the Map computing performance is to calculate the satisfaction scores of the user input, and to achieve the following Job scheduling and Slot resource allocation in Reduce phase

For example: Job_1 spends X_1 seconds in completing 30 percent of Map task, then it can be predicted that the Map performance of the job is MAP_1. Therefore, it is necessary to use the JobState class to record the file size and processing time for each job that is processed in the Map phase. Its parameter record sample is shown in Table 3.

Table 3 the sample of parameter record of Map phase.

Job	Job_1	Job_2	Job_3	Job_4	Job_5	Job_6
Sat	200	120	80	90	100	180
Map efficiency（MB/Slot.sec）	10	5	10	15	20	5
Residual file（MB）	4000	5000	1000	2000	3000	1700

After Map calculation is completed in step (1) and step (2), there will be the Table 3 for the parameter records so as to go into step (3) for further calculations.

(3) Configuration rules of Slot. After the completion of step (1) and step (2) and a list of parameter data records in Table 3, the rest file will be based on for calculation to get the data on job end time.

After the end time is speculated, the parameter data needed for Slot is obtained according to the parameter data obtained from table 3. The examples are shown in Table 4.

Table 4 End time data speculation

Job	Job_1	Job_2	Job_3	Job_4	Job_5	Job_6
Sat	200	120	80	90	100	180
end Time（sec/Slot）	400	240	100	133.3	150	340
Slot needed	2	1	1	1	1	2

Hence, Slot configuration rules will be gained in the next system evaluation.

(4) Job calculation sorting at Map phase can be gained. Once the Slot configuration rules are established, job satisfaction units can be calculated. Furthermore, the sorting of them can indicate a fact that a job with higher satisfaction in a unit will be worthy of priority, which complies with the rules of the maximum satisfaction scheduling. Derivation unit satisfaction results are shown in Table 5.

Table 5 MSS Scheduling algorithm derivation results

Job	Job_1	Job_2	Job_3	Job_4	Job_5	Job_6
Sat	200	120	80	90	100	180
Map efficiency（MB/Slot.sec）	10	5	10	15	20	5
Slot needed	2	1	1	1	1	2
Sat/Sn（min/Slot）	100	120	80	90	100	90

As shown in Table 5, the data are sorted from highest to lowest with unit satisfaction as the basis and in accordance with (Job, needed Slot, Sat) format. Thereby a set of priority queues for Job calculation are drawn: (Job_2, 1, 120), (Job_1, 2, 100), (Job_5, 1, 100), (Job_4, 1, 90), (Job_6, 2, 90), (Job_3, 1, 80). After getting sorting result, the operation must follow the following conditions.

(1) Score level of unit satisfaction.

(2) Slot numbers in need decide the calculation priority. The unit satisfaction sorting should be modified so that the fewer should calculated in priority. If there is the same unit of satisfaction scores, the smaller Slot number required in ending will be listed in the front. The sequence is re-arranged as: (Job_2,120,1), (Job_5,1,100), (Job_1 , 2,100), (Job_4,1,90), (Job_6,2,90), (Job_3,1,80).

(3) Reduce calculations.

By now, the theory and specific examples for the greatest satisfaction scheduling algorithm derivation is completed.

This paper sets the same computing power of each calculator, so each unit Slot has the same computing power. A certain proportion of Slot at the beginning of calculation is retained so that parameter *mapred.reduce.slowstart.completed* can set the percentage parameter of Reduce Task start as 0.45 [8], namely *Slot_Reserved* = 45%. The system parameter setting starts scheduling interval, that is, *MSS_EvaluateTime* = 3 seconds.

The maximum satisfaction scheduling **algorithm** focuses on each parameter data calculated by a job so as for maximizing throughput and overall satisfaction scores based on the derived calculation of Map computing performance and the corresponding satisfaction scores. By this way, the maximum satisfaction scheduling choices can be done and Slot resource attachment for the Job / Task by virtue of the same reasoning at the Reduce phase can be gained. The job with higher satisfaction scores at the Reduce phase will have an access to more resources than that in Map phase in order to ultimately achieve maximum satisfaction scheduling **algorithm.**

Experimental analysis

Experimental hardware and software environment. Two physical servers with Intel (R) Xeon E5620 CPU and 16GB memory are used in our experimental environment. We set up four nodes, including a master node and three slave nodes. The master node uses a single server, and the slave nodes are located in another server. Virtual machines are built with VMware Workstation 9.0 version. The operating system is Ubuntu desktop 12.10. Hadoop' version is 1.2.1. JDK is version 1.6 and OpenSSH version is 6.0.

The experiment selects five files in the same format, with an average size of about 100MB. The test files will be accessed through preprocessing so as to upload the test data to the HDFS. The relevant algorithms and other procedures can be added into Hadoop custom scheduler for further modification according to the experimental needs. The experimental test steps are as follows.

Job sorting verification experiment. While selecting the amount of test files, it is considered to be divided into two tasks for calculation in order to reflect the maximum satisfaction scheduling algorithm and core. Hadoop custom Block is 64MB, so a test file of 100MB size is selected in that it can at least be divided into two tasks to get more real experimental data and better reflect the advantage of the maximum satisfaction scheduling algorithm. If the job with a higher score does the experiment in a fast manner, then the maximum satisfaction will be gained.

(1) First, fair scheduling is introduced for MapReduce calculation without introducing the maximum satisfaction scheduling algorithm. The experimental data are shown in Table 6. Job will succeed in Map calculation in case of no the maximum satisfaction scheduling algorithm. As shown in Table 6, the completion time at Map phase is displayed and the completion time sequence of 5 jobs is listed as Job01, Job02, Job03, Job04, and Job05.

Table 6 Fair Scheduling algorithm experimental results of job sorting

Job	Complete Map time	time	sequence
Job01	14:21:46	46s	1
Job02	14:22:31	45s	2
Job03	14:23:15	44s	3
Job04	14:23:59	44s	4
Job05	14:24:45	46s	5

(2) After an introduction of the maximum satisfaction scheduling algorithm, the experimental data of MapReduce calculation are shown in Table 7. It can be seen that the completion time of 5 jobs at Map phase is Job02, Job05, Job01, Job04, and Job03 in a sequence.

Table 7 MSS Scheduling algorithm experimental results of Job sorting

Job	Sat	Complete Map time	time	sequence
Job01	100	15:41:23	40s	3
Job02	200	15:40:02	40s	1
Job03	80	15:43:25	41s	5
Job04	90	15:42:44	41s	4
Job05	120	15:40:43	41s	2

It can be seen that the maximum satisfaction scheduling algorithm changes the job orders in the Map phase so that the satisfaction scores can change the calculation orders of job.

The experiment indicates that the maximum satisfaction scheduling algorithm through the satisfaction scores entered by users is applied into MapReduce computation as a parameter. The job calculation sequence is changed in accordance with the satisfaction scores. It helps users to set a higher score for job with high-priority at their own will. In this way, the priority flexibility is improved and users can obtain the MapReduce results they expect.

Comparative and analysis between MSS algorithm and Fair in an experiment. In this experiment, the maximum satisfaction Scheduling algorithm and Fair scheduling algorithm are cast into a comparison for analyzing in time-consuming. The experimental data source is still consistent with the data source in section 3.2, the experiment equipment is also the same as one in 3.2 and the set parameters are as the same as the previous section as well. The experimental data are shown in Table 6 and Table 7. It can be seen that the time-consuming of MapReduce is shorter than that of Fair Scheduling. The time-consuming of each job is shown in Figure 2.

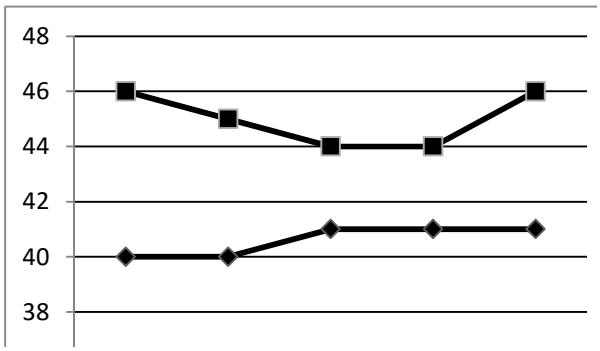

Fig.2 MapReduce calculation time

Figure 2 indicates that compared with fair scheduling algorithm in the MapReduce computation phase, computing time of each job of the maximum satisfaction scheduling algorithm reduces by 4 to 5 seconds. The calculation time of each job decreases by 8% to 10%.

The two experiments and calculated data prove that the maximum satisfaction scheduling algorithm has the following results:

(1) Change the output order of job through the job satisfaction score, improve the flexibility of job priority.

(2) The efficiency of Hadoop scheduling algorithm is improved, the computational time of MapReduce calculations is reduced. As a result, it is proved that the maximum satisfaction scheduling algorithm can resort to job priority scheduling to tune and improve processing time of scheduling algorithm which is of great significance of priority research in Hadoop scheduling algorithm.

Conclusion

This paper has presented the maximum satisfaction scheduling algorithm based on Hadoop platform which can effectively solve the poor flexibility of job priority in MapReduce. The scheduling algorithm makes some improvement in Hadoop by means of designing a new MapReduce algorithm scheduler. Experiments show that the maximum satisfaction scheduling algorithm can improve the flexibility of job scheduling, reduce calculation time of job in MapReduce, and increase throughput. In addition, the maximum satisfaction scheduling algorithm leaves users good impression and effectively improve the MapReduce computational efficiency.

References

[1] Liu Peng. Cloud computing [M]. Beijing: Electronic Industry Press, 2011.

[2] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters [J]. Communications of the ACM, 2008, 51(1): 107-113.

[3] Ralf L. Google's MapReduce programming model-revisited [J]. Science of Computer Programming, 2008, 70(1): 1-30.

[4] Chen Yan-jin. The Improvement and Implementation of job scheduling algorithm of Hadoop in MapReduce [D] Guangzhou: South China University of Technology, 2011.

[5] Information on http://Hadoop.apache.org.

[6] Labrinidis Alexandra, Roussopoulos Nick. Update propagation strategies for improving the quality of data on the Web [English Conference] 2001.

[7] Information on http://hadoop.apache.org/docs/r1.1.2/fair_scheduler.html.

[8] Information on http://hadoop.apache.org/docs/stahle/capacity_scheduler.html.

[9] FISCHER M J, SU Xue-yuan and YIN Yi-tong. Assigning tasks for efficiency in Hadoop: extended abstract [J].Proc of the 22nd ACMSymposium on Parallelism in Algorithms and Architectures. New York: ACM Press, 2010: 30-39.

[10] THAWARI V W, BABAR S D and DHAWAS N A, An efficient data locality driven task scheduling algorithm for cloud computing [J]. International Journal in Multidisciplinary and Academic Research, 2012, 1(3): 151-158.

[11] Kai Hua-dong, Tian Qi. Design and implementation priority based weighted fair Queue of based on MapReduce cluster [J]. Computer knowledge and technology. 2011: 2129-2132.

[12] Xu Cheng, Liu Hong and Tan Liang. New mechanism of monitoring on Hadoop cloud platform [J] Computer Science, 2013, 40 (1): 112-117.

[13] Li Qian-mu, Zhang Sheng Xiao and Lu Lu et al. A job scheduling algorithm and hybrid scheduling method on Hadoop platform [J]. Computer Research and Development, 2013, 50 (z1): 361-368.

[14] Zhu TS. Web usage mining for Internet recommendation [D]. Canada: University of Alberta Edmonton, 2001.

[15] Catledge. L. D, J. E. Pitkow. Characterizing browsing strategies in the Word-Wide Web [M]. Computer Networks and ISDN systems, 1995.27(6): p.1065-1073.

[16] Huang Yi-hua. Miao Kai-xiang. In-depth understanding of Big Data [M] Beijing: China Film Press, 2014: 91-122.

[17] Tanter E, Figueroa I and Tabareau N. Execution levels for aspect-oriented programming: design, semantics, implementations and applications [J]. Science of Computer Programming, 2014, 80(2): 311-342.

[18] Bao Dong-xing, Li Xiao-ming . A study on LRU cache scheme [J] Computer Engineering 2007, 33 (9): 272-274.