International Journal of Hybrid Information Technology
Vol. 9, No.7 (2016), pp. 227-236
http://dx.doi.org/10.14257/ijhit.2016.9.7.21

An Interactive Job Scheduling Algorithm Based on Hadoop

Xue Tao and Yan Minglei

School of Computer Science, Xi'an Polytechnic University, Xi'an 710048, China
xt73@163.com, yml545@126.com

Abstract

Job scheduling is an important component of Hadoop. Hadoop default FIFO scheduling algorithm is simple and easy to achieve and widely used; but it lacks consideration in the characteristic of data locality, which will lead to heavy traffic during network transmission and task execution, then computing resources cannot be fully utilized and a series of other drawbacks. Meanwhile the static function of resource slots in Map and Reduce stages make such defects worse. This paper proposes a job scheduling algorithm (Interactive Scheduler Algorithm) IS based on interacting the master node and slave nodes from the data locality and tasks allocation. The algorithm improves FIFO and the usage of computing resource, realizes the dynamic conversion of map slots and reduces slots. In the end through the comparison of experiment it is proved that the IS has a great improvement in job scheduling for Hadoop.

Keywords*: Hadoop; MapReduce; interactive scheduler; slot*

1. Introduction

With the rapid development of the Internet and growth of users, data generation rate also shows explosive growth, which makes big data become a popular search term nowadays. Faced with increasingly large amounts of data, the traditional big machines process powerless. Along with the progress of software technology, the concept of distributed computing has been raised once again. This is a way of low cost and high efficiency by connecting a large number of low-cost PC and making up cluster to store and analyze massive amounts of data. Simply speaking, this is a great change in the field of data processing from "centralizing to decentralizing". Among the distributed processing platforms, Apache Hadoop [1] is the most active and widely used one and one of the open source project under the Apache Software Foundation. In Hadoop platform the realization of the bottom layer is transparent for the user. The programmers need only consider the algorithm itself to write distributed applications. Thanks for the usability, many cloud enterprises and Hadoop enthusiasts involve in Hadoop family actively, that also make Hadoop users and Hadoop attention keep raising.

In Hadoop there are three kinds of scheduling methods. FIFO Scheduler [2], Fair Scheduler [3], Capacity Scheduler [4]. The default is FIFO. Both Fair Scheduler and Capacity Scheduler is oriented multi-user and fairness, the only difference is the different realization mechanism. Capacity Scheduler is a kind of computing ability scheduling algorithm developed by Yahoo to restrict each user's resources by the memory constraint, the method of resource allocation is queue. Fair Scheduler is developed by Facebook and mainly for itself application. The algorithm has outstanding characteristics that small jobs response fast and big jobs perform high character to ensure and guarantee the fairness of each job occupation of resources. For fairness it uses resource pool to allocate.

ISSN: 1738-9968 IJHIT

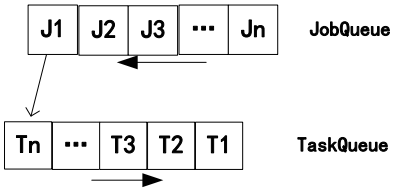

Figure 1. Model of FIFO

FIFO is a kind of batch processing scheduler. All jobs are submitted into a JobQueue based on first in first out as Figure 1 show. And each job is split into different tasks in TaskQueue. The advantage of FIFO is simple and easy to achieve and the application is very wide and less computing burden to JobTracker; but it also has many drawbacks: the task allocation for data locality just keeps as much as possible and the slot is oriented to function but job. Therefore many scholars have tried to improve from all aspects. Xiaohong Zhang [5] had been proposed scheduling algorithm considering whether allocate Non-Local tasks or not if the job scheduler cannot select Data-Local tasks. Shadi Ibrahim [6] had proposed Maestro: a scheduling algorithm for Map tasks to be selected based on the number of hosted Map tasks and on the replication scheme for its input splits. Matei Zaharia [7] proposed a scheduling algorithm aiming at solving the local problems caused by Non-Local data scheduling. Dynamic Proportional Scheduler [8] allows users to adjust the priority levels of assigning to their jobs. Kamal Kc [11] proposed a constraint based scheduler that dispatching tasks in a real time cluster. But these algorithms have failed to take into account the localization characteristic of the data and the initiative of TaskTracker nodes in task allocation. For these problems, we put forward a kind of interactive scheduling algorithm (Interactive Scheduler) IS scheduling algorithm.

2. Related Definition of IS

In order to describe the IS scheduling algorithm, it need to be given some related definition. The concept of IS as follow.

2.1. Transmission Time

The transmission time is that data nodes copy the input split of the task to the nodes requesting tasks. t_1, s, v respective represent the transmission time, the size of block, transmission speed of switch. So,

$$t_1 = \frac{s}{v} \qquad (1)$$

2.2. Waiting Time

The waiting time is the remaining execution time of the task in execution status in the TaskTracker nodes. t_2, w respective represent the waiting time and the percentage of task completed. Accounting the difference between the different tasks, we evaluate $MAX\{T_1, T_2, \cdots T_N\}$ represent the maximum execution time in the queue of the TaskTracker node. So,

$$t_2 = MAX\{T_1, T_2, \cdots T_N\}(1-w) \qquad (2)$$

Slot [9] is the basic unit of execution tasks in MapReduce. Each node in the default configures 2 map slots and 1 reduce slot, And map slot is just for map task and reduce slot just for reduce task. The slot number can represent clusters' calculation scale in a certain extent.

2.3. Effective Slot

The slot number of the map or reduce task in execution status. Suppose cluster A has n nodes, so the quantity of effective map slot is $Q_m \in [0,2n]$ and reduce slot is $Q_r \in [0,n]$ in default.

2.4. Effective Slot Rate

The ratio of the slot number of the map or reduce task in execution status and the corresponding total slots. So,

$$l_m = \frac{Q_m}{Q} * 100\% \qquad (3)$$

$$l_r = \frac{Q_r}{Q} * 100\% \qquad (4)$$

In the equation l_m, l_r, Q represent effective map slot rate, effective reduce slot rate, the total number of Map slots or Reduce slots in cluster.

3. IS Scheduling Algorithm

According to the definition of part 2, it explain the IS from the goal, the data flow and the control flow and the implementation process in this chapter.

3.1. The Goal of IS

The default FIFO is scheduling according to the order of arrival in job queue as show in Figure 1. Being scheduled job is cut into task splits and assigned to the TaskTracker nodes for executing. The next scheduling waits until the whole work is done. The whole process is a serial. So there are the following drawbacks:

1) The tasks assigned to the TaskTracker nodes for execution cannot ensure the data locality. And the executions for Non-Local tasks will heavy the network transmission.

2) As a result of machine processing speed difference, the uncertainty of the task, the static allocation of resources and a series of reasons, parts of the machines are in a waiting state after map task. For computing resources it is a huge waste.

Therefore, this paper propose IS scheduling algorithm is to give full play to the initiative of the TaskTracker nodes that does not have the data locality; it chooses the most suitable task assigned to the current TaskTracker node through interaction with the JobTracker node. Secondly, in the initial cluster the initialization of slot all set for Map and Reduce slot value set to 0. In the process of job executing, the Map slot dynamically converted to Reduce slot. So it can effectively reduce the amount of data transmission in the network and improve the utilization rate of resources and the effective slot rate of the platform.

3.2. The Combination of HDFS and IS

Because HDFS has fully taken into account the reliability and fault tolerance of the data. Therefore, we should make full use of this characteristic to reach the data locality maximum in task scheduling in IS algorithm. Note that we will strictly ensure the HDFS blocks are consistent with the size of the Map task splits for avoiding the task of data distribute in two or more data block.

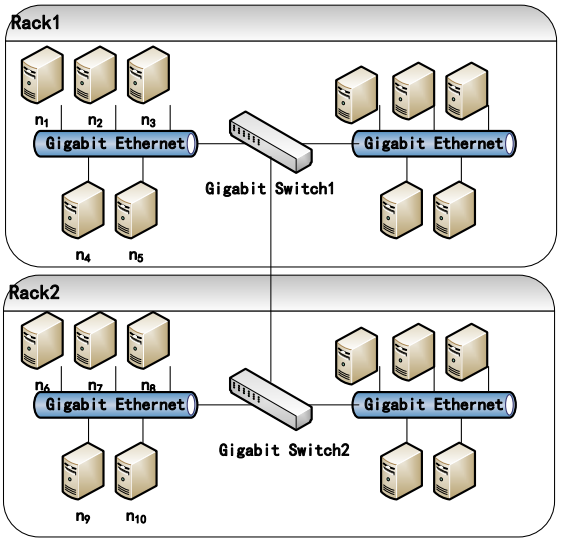

Figure 2. Topology of the Cluster

As show in Figure 2 the topology of the cluster. Rack1 and Rack2 interconnect with each other through switch 1 and switch 2. The data block is well-distributed according to the HDFS default storage strategy. With the unexpected failure downtime and load balancing into account, the storage layout will change dynamically. In the IS algorithm, when the data does not have localization features, it will consider the weight of the transmission time and waiting time on the "near" node with data to balance. Here have a measurement on the definition of the "near" node. In the precondition of without loss of generality, it follows the default storage rules to choose any three nodes. Now suppose that it choose n_1, n_5, n_{10} three nodes. The data of D in these three nodes have a backup. The letter d represents the distance between points. Thus

$$f(d) = \begin{cases} d(n_1, n_1) = 0 \\ d(n_1, n_5) = 1 \\ d(n_1, n_{10}) = 2 \end{cases} \quad (5)$$

Where 0 represents the distance between processes in the same node and 1 represents the distance between nodes in the same rack and 2 represents the distance between nodes in different racks. JobTracker will select a node according distance from small to large after weighting on neighboring nodes with distance.

3.3. The Data Flow of IS

The data go through 4 processes from Input to Output.

step1. From Input phase to Map phase that exist the problem of data locality. If data split in the current TaskTracker where a backup have, it is a kind of optimal situation; Otherwise, turn to the next step.

step2. If the task is assigned to the other node without backup data that need to consider the weight of wait time and the cost of transmission time in the "near" node. When the waiting time of the node is smaller than transmission time, this data block is reserved for the "near" node and waiting for the completion of the current task for the next call. Otherwise the data block transmission to the current assigned task node.

step3. All of above are not satisfied then it believe the TaskTracker enter to Reduce phase. Map slots will change to Reduce slots dynamically and Reduce slots will process the intermediate data during that time.

step4. Eventually all nodes enter into the Reduce phase, handle the Reduce tasks and

output operations completely.

The overall flow Figure of data as follows:

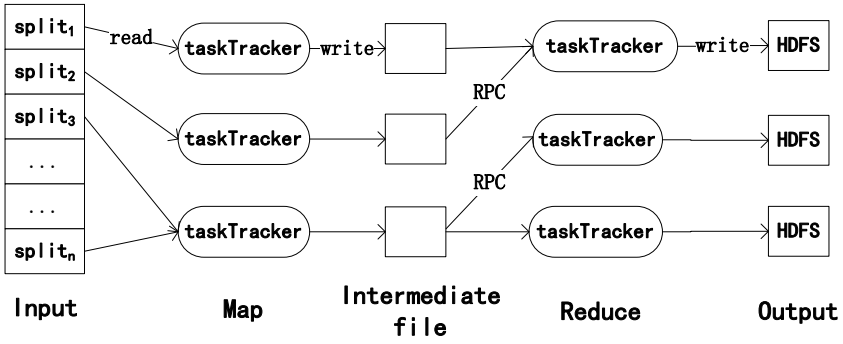

Figure 3. Data Flow of MapReduce

3.4. The Implementation Process of IS

For IS it takes a coordinated approach to task allocation between JobTracker and TaskTracker based on the data locality. First exploration of sequence allocate in FIFO as show in Figure 1. Go back to the JobTracker node adjusts redistribution when the TaskTracker node does not meet the data locality rather than one request according to the order of tasks.

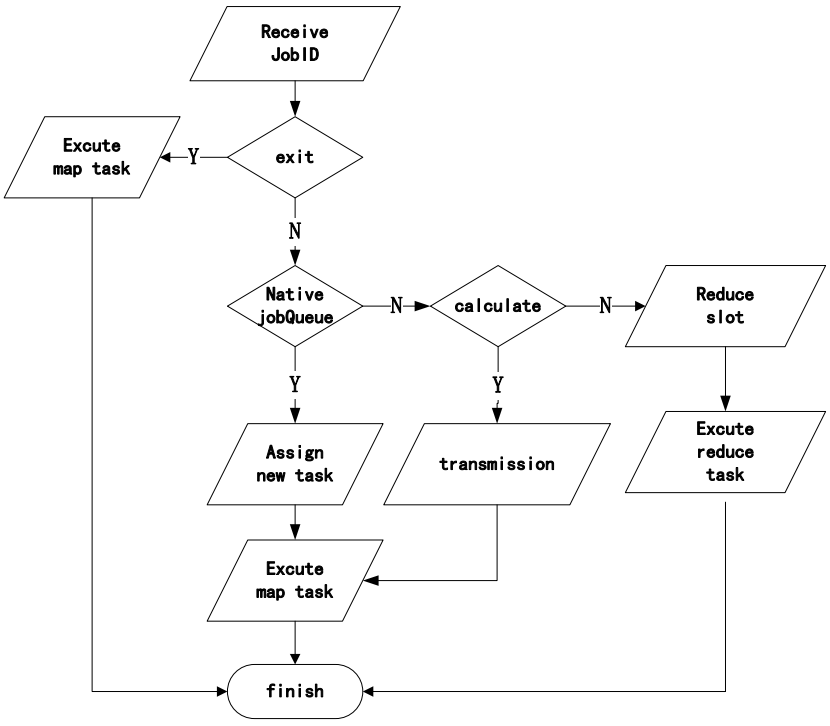

Figure 4. The Process of IS

The step of execution as follows.

step1. When receiving a task, first of all TaskTracker node judge whether the data exist or not. If do, execute directly. Otherwise, turn to the next step.

step2. TaskTracker nodes will feedback the information to the JobTracker node through heart beat. The JobTracker node will reselection task according to data locality in

JobQueue. If there exist then allocate and execute directly, Otherwise the next step.

step3. JobTracker node will select the most neighboring nodes after calculating the waiting time and transmission time according to the distance. The data will transfer and the node will perform a task if transmission time is less than the waiting time, or turn to the next step.

step4. All of these conditions are not satisfied that indicates the task is being carried on or will be executed, on the whole the Job is about to complete the map stage. The idle map slots change to reduce slots for reduce task.

step5. It will wait for the arrival of the next job for rescheduling after all the tasks completed.

4. Verification

For fairness it uses the available slot and available slot rate predicate pros and cons between FIFO and IS algorithm first. The effective map slot for FIFO is related with the cluster scale and is an inverse relationship from experience show; but in IS it is based on the FIFO focus on data locality, so slot rate will change in a high level. And in the Map and Reduce phase it has certain concurrency; to a certain extent it will shorten the execution time of tasks.

In order to demonstrate the effectiveness of the algorithm, it validates the performance on the Hadoop1.1.2 platform, environment shown in the table below.

Table 1. The Verification Environment

Version	Hadoop1.1.2
File System	HDFS
Benchmark program	sort
Scheduler	IS/FIFO
Data	RandomWriter

Table 2. The Configuration of Master Node

CPU	Intel®Xeon(R) CPU E5-2620 @2.00GHZ×2
OS	Ubuntu12.04
Kernel	Linux 3.2.0-60-generic
Memory size	4G

Table 3. The Configuration of Slave Nodes

CPU	Intel®Xeon(R) CPU E5-2620 @2.00GHZ×2
OS	Ubuntu12.04
Kernel	Linux 3.2.0-60-generic
Memory size	2G

The Hadoop cluster consists of 9 nodes for the experimental environment. The master node and four slave nodes install in a single virtual host, the other 4 nodes in another virtual host. Two hosts are connected through two Gigabit switches.

It uses effective slots of slave node number, total execution time and CPU utilization three indexes comparing the two algorithms. In a certain cluster, the effective rate of slot is determined by effective slot number. Therefore, we only take effective slot number one parameter for comparing.

Each TaskTracker running on a slave node in the cluster monitor the state of the CPU through relevant “/proc/stat” command [10] in the present time. “/proc/stat” displays all the activity information of all CPUs and every CPU. It takes the I/O waiting time

information iowait and collects sample points to CPU usage to calculate. For different data splits, it set different threshold. When the I/O waiting time exceeds a specified threshold, it believes the data is in transmission at this point; otherwise it is an effective slot.

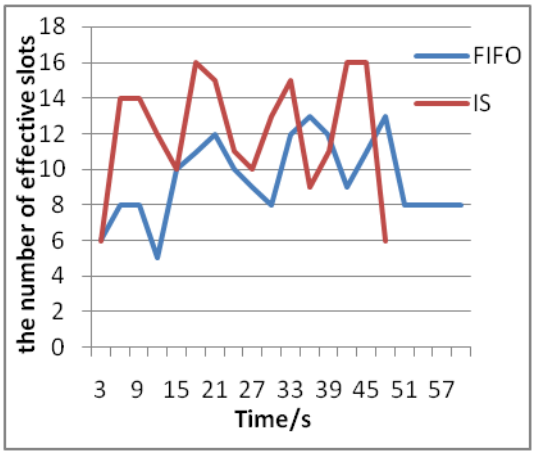

Figure 5. The Comparison of Effective Slots

Figure 5 shows the number of effective slots of two algorithms in dealing with the same task. IS scheduler algorithm is more than FIFO overall, but the gap is a little difference. This is because the number of nodes is relatively small. Even if allocation in order will have a large probability selected data of the local node. But in the cluster with more nodes, with the decrease of the probability, the IS algorithm will reflect the advantage better. The gaps between the two can be widened. Another IS scheduler is ahead of FIFO in deadline. This also reflects the superiority of algorithm.

Figure 6 is the same amount of data processing time of the normalized representation. First of all, on the execution time IS scheduler is obviously faster than FIFO. On the other hand, with the increase of the amount of data, the gap of the execution time is increasing that also illustrated the advantage of IS on the large amount of data.

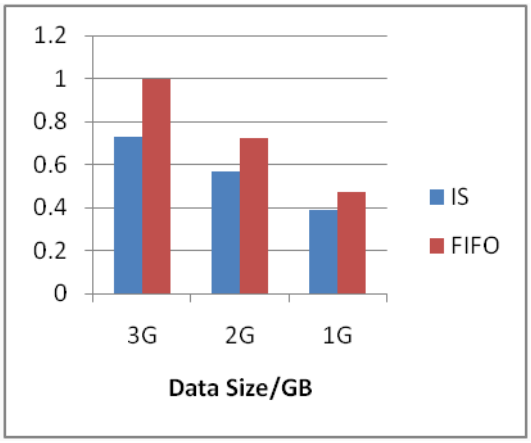

Figure 6. The Comparison of Execution Time of Different Data Size

Figure 7 is a comparison chart by collecting samples calculated CPU usage. IS on the ratio of resource basically is higher than FIFO. The initial IS utilization rate lower than the FIFO because the node is not assigned to the data localization task, which is a dynamic process of adjustment influence the CPU usage. But the IS has some improvement in the utilization of computing resources after all.

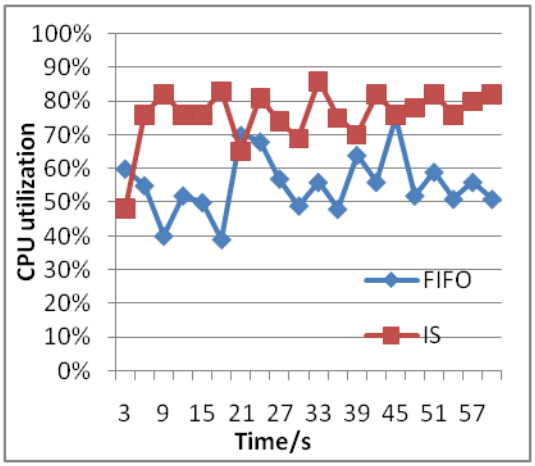

Figure 7. The Comparison of CPU Utilization

5. Conclusion

This paper puts forward a kind of interactive job scheduling algorithm (IS), the algorithm maximize the data locality in an interactive mechanism between master node and slave nodes and reduce the amount of transmission data in the network to speed up the task completion. It is an improvement to the FIFO algorithm optimization, but there are still some flaws in the algorithm, several parameters calculation formula on accuracy is difficult to grasp that will affect the allocation of tasks. At the same time, too much calculation also increases the workload of the master node. Therefore, this is the focus of the next step study. On the whole the task execution time has certain ascend.

Acknowledgments

This work is supposed by the National Development and Reform Commission high-tech industrialization of Shanxi Province of China (No [2009]1365) and the Doctoral Scientific Research Foundation of Xi'an Polytechnic University (BS0752)

References

[1] A. Hadoop, Apr. http://hadoop.apache.org, **(2014)**.
[2] C. He, Y. Lu and D.Swanson, "Matchmaking: A new mapreduce scheduling technique", 2011 IEEE Third International Conference in Cloud Computing Technology and Science (CloudCom) on IEEE, **(2011)**, pp. 40-47.
[3] M. Zaharia, D. Borthakur, J. S. Sarma, K. Elmeleegy, S. Shenker and I. Stoica, "Job scheduling for multi-user mapreduce clusters", EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-55.
[4] A. Raj, K. Kaur, U. Dutta, V. V. Sandeep and S. Rao, "Enhancement of Hadoop Clusters with Virtualization Using the Capacity Scheduler", Proceedings in Services in Emerging Markets (ICSEM), 2012 Third International Conference on IEEE. **(2012)**, pp. 50-57.
[5] X. Zhang, Y. Feng, S. Feng, J. Fan and Z. Ming, "An effective data locality aware task scheduling method for MapReduce framework in heterogeneous environments", In Cloud and Service Computing (CSC), 2011 International Conference on IEEE, **(2011)**, pp. 235-242.
[6] S. Ibrahim, H. Jin, L. Lu, B. He, G. Antoniu and S. Wu, "Maestro: Replica-aware map scheduling for mapreduce", In Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on IEEE, **(2012)**, pp. 435-442.
[7] M. Zaharia, D. Borthakur, J. S. Sarma, K. Elmeleegy, S. Shenker and I. Stoica, "Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling", In Proceedings of the 5th European conference on Computer systems. ACM, **(2010)**, pp. 265-278.
[8] T. Sandholm and K. Lai, "Dynamic proportional share scheduling in hadoop", In Job scheduling strategies for parallel processing in Springer Berlin Heidelberg, **(2010)**, pp. 110-131.
[9] S. Kurazumi, T. Tsumura, S. Saito and H. Matsuo, "Dynamic Processing Slots Scheduling for I/O

Intensive Jobs of Hadoop MapReduce", In ICNC, **(2012)**, pp. 288-292.
[10] /proc/stat, http://blog.sina.com.cn/s/blog_691c5f870100mmqx.html, **(2014)**.
[11] K. Kc and K. Anyanwu, "Scheduling hadoop jobs to meet deadlines", In Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on IEEE, **(2010)**, pp. 388-392.

Authors

Xue Tao, he is an Associate Professor in the Computer Science Dept at Xi'an Polytechnic University. He received his Ph.D. in Computer Science from Xi'an Jiaotong University. He obtained his M.S. degree in Computer Science from the Northwestern University of China. His current research interests include cloud computing, big data and content-based networking.

Yan Minglei, he received his Bachelors of Science in Mathematics and Computer Science from Shanxi Datong University in 2011. Now he is a Graduate student in the Computer Science Dept at Xi'an Polytechnic University. His current research interests include cloud computing, big data, Hadoop platform.