Hadoop Performance Tuning Based on WordCount

1st Wei Wang
BIM Computing Research Center
Shenyang Jianzhu University
Shenyang, China
519836646@qq.com

2nd Xin Liu
Human Resources Department
Shenyang Jianzhu University
Shenyang, China
6620274@qq.com

3rd Yong Shi
BIM Computing Research Center
Shenyang Jianzhu University
Shenyang, China
674365154@qq.com

4th Ning Tao
BIM Computing Research Center
Shenyang Jianzhu University
Shenyang, China
23010150@qq.com

5th Chong Xu
BIM Computing Research Center
Shenyang Jianzhu University
Shenyang, China
32500103@qq.com

***Abstract*—In order to better verify that Hadoop performance can be improved through optimization of parameters, we can use the following test methods: benchmarking, stability testing, high availability testing, scalability testing, and security testing. In this paper, the benchmark test method is used to verify the optimization of parameters and to optimize the performance of Hadoop. This article mainly focuses on the 15 parameters in Tab.1. The optimization results are shown in Tab.3. The optimization of the parameters was verified by the execution time of the WordCount algorithm in the benchmark test. During the experiment, the CPU and memory utilization rate, disk IO and network IO throughput and other indicators were collected. Fig.4-6 fully illustrates the comparison between Hadoop and WordCount algorithm after parameter default value and parameter adjustment. The experimental results show that after the Hadoop parameters are adjusted and optimized, the Hadoop performance tuning is achieved under certain conditions.**

Keywords—Hadoop, Word Count, Parameter Optimization, Performance Tuning

I. INTRODUCTION

The big data software platform is mainly composed of distributed file systems (such as HDFS), distributed computing systems (such as MapReduce), NoSQL databases (such as HBase), distributed data warehouses (such as Hive), and distributed databases (MPP)[1,2]. Provides storage, management and computing capabilities for big data. The big data software platform mainly includes open source Hadoop, Spark, etc., and is generally deployed on a general hardware platform.

TABLE I. HADOOP PARAMETER LIST

No	Category	Parameter name	Default
1	YARN	yarn.nodemanager.resource.memory-mb	5G
2	YARN	yarn.nodemanager.resource.cpu-vcores	8
3	MapRedcue	mapreduce.job.reduces	1
4	MapRedcue	mapreduce.map.memory.mb	1G
5	MapRedcue	mapreduce.map.java.opts	756M
6	MapRedcue	mapreduce.task.io.sort.mb	200M
7	YARN	yarn.scheduler.maximum-allocation-mb	2G
8	MapRedcue	mapreduce.map.sort.spill.percent	0.7
9	MapRedcue	mapreduce.map.output.compress	FALSE
10	MapRedcue	mapreduce.map.output.compress.codec	org.apache.hadoop.io.compress .DefaultCodec
11	MapRedcue	mapreduce.reduce.memory.mb	1G
12	MapRedcue	mapreduce.reduce.java.opts	756m
13	MapRedcue	mapreduce.reduce.shuffle.input.buffer.percent	0.7
14	MapRedcue	mapreduce.job.reduce.slowstart.completedmaps	0.05
15	MapRedcue	mapreduce.job.reduces	25

Because Hadoop itself contains many parameters, the relationship between parameters is also more complicated. After simple experimental verification, this paper mainly optimizes the 15 parameters in Tab.I, and the parameter optimization is to follow the order, superimposed way to achieve Hadoop performance tuning[3].

II. RESEARCH ON EXPERIMENT DESIGN AND WORDCOUNT ALGORITHM

Experiment design. In order to better verify that Hadoop performance can be improved through optimization of parameters, we can use the following test methods: benchmarking, stability testing, high availability testing, scalability testing, and security

  DOI: 10.25236/iccpb.2018.004

testing. In this paper, the benchmark test method is used to verify the optimization of parameters and to optimize the performance of Hadoop.

Benchmarking is an activity that measures and evaluates software performance metrics. At a time, benchmarks can be used to establish a known level of performance (referred to as the baseline), and a benchmark test can be performed after the hardware and software environment of the system has changed to determine the effect of those changes on performance[4].

There are a variety of test methods for benchmarking. such as: TestDFSIO, TeraSort, WordCount, MRBench, NNBench, simulator HDFS daily import/export, DistCP and so on. According to Hadoop's Map Reduce process, this paper mainly uses the WordCount algorithm to verify that the 15 parameters in Tab.1 are adjusted, and the Hadoop performance is affected and optimized under specific conditions. The Hadoop test environment is based on the pattern of 1 Name Node and 4 Date Node[5].

WordCount algorithm. The principle of WordCount algorithm is as follows:

- Split the file into splits. Since the test file is small, each file is a split, and the file is split into rows to form a <key, value> pair, which will be split into <key, value> pairs. Hand over the user-defined map method to generate a new <key, value> pair, as shown in Fig 1.

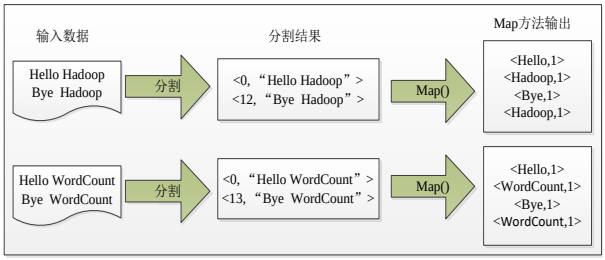

Fig 1. Segmentation process and execution of Map method

- After getting the <key, value> pairs output by the map method, the Mapper will sort them according to the key value, and execute the Combine process to accumulate the keys to the same value to get the final output of the Mapper. as shown in Fig 2.

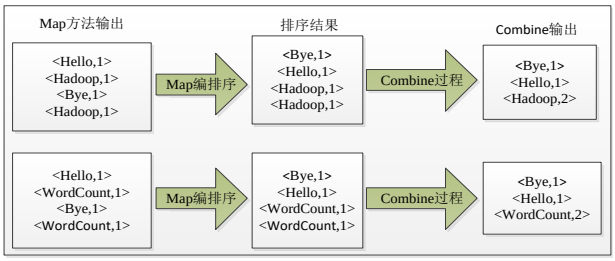

Fig 2. Map side sorting and Combine process

- The Reducer first sorts the data received from the Mapper, and then processes it by the user-defined reduce method to obtain a new <key, value> pair, and outputs the result as WordCount, as shown in Fig 3.

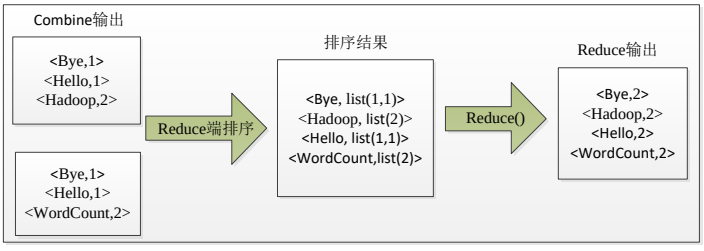

Fig 3. Reduce side sorting and output results

WordCount tests the performance of MapReduce and HDFS in Hadoop. According to the complex relationship between them, the 15 parameters in Tab.I are divided into 5 groups of parameter groups for adjustment. From the first group to the seventh group, the adjustment parameter values are accumulated in sequence, see Tab.III for details. Verified by the WordCount algorithm the correctness of parameter adjustment.

After each adjustment of the parameters, we need to execute the WordCount algorithm on Hadoop. The execution steps of the WordCount algorithm are detailed in Tab.II. During the experiment, the execution time of the statistical algorithm is referenced. The PAT (Performance Analysis Tool) tool was used to monitor the CPU, memory, disk IO and network IO of the algorithm during the execution of the algorithm.

The single file size in the experimental data is 1G, the number of files is 512, and the total file size is 512G.

TABLE II. EXPERIMENTAL PROCEDURE

Step	Content
1	hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples-2.6.0-bc1.2.2.jar randomtextwriter -D mapreduce.randomtextwriter.totalbytes=549755813888 /wordcount_input
2	echo 3 > /proc/sys/vm/drop_caches
3	hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples-2.6.0-bc1.2.2.jar wordcount -Dmapred.reduce.tasks=1 /wordcount_input /wordcount_output
4	hadoop fs –rm -r –skipTrash /wordcount_output

III. TEST AND ANALYSIS

In Tab.III, we can understand that there are mutual influences between parameters, so we try to divide them into 5 groups for testing. The test found that as the parameters are adjusted, the execution time of the WordCount algorithm is decreasing, which fully explains the parameters. With the optimization of parameters, the Hadoop performance is optimized.

TABLE III. WORDCOUNT ALGORITHM EXECUTION TIME COMPARISON

No	Parameter name	Default	Optimization	Time
Ⅰ	yarn.nodemanager.resource.memory-mb	5G	100G	432s
	yarn.nodemanager.resource.cpu-vcores	8	25	
	mapreduce.job.reduces	1	25	
Ⅱ	mapreduce.map.memory.mb	1G	4G	406s
	mapreduce.map.java.opts	756M	3G	
	mapreduce.task.io.sort.mb	200M	2047M	
	yarn.scheduler.maximum-allocation-mb	2G	4G	
Ⅲ	mapreduce.map.sort.spill.percent	0.7	0.9	385s
Ⅳ	mapreduce.map.output.compress	FALSE	TRUE	352s
	mapreduce.map.output.compress.codec	org.apache.hadoop.io.compress.DefaultCodec	Lz4Codec	
Ⅴ	mapreduce.reduce.memory.mb	1G	4G	297s
	mapreduce.reduce.java.opts	756m	3G	
	mapreduce.reduce.shuffle.input.buffer.percent	0.7	0.9	
	mapreduce.job.reduce.slowstart.completedmaps	0.05	0.8	
	mapreduce.job.reduces	25	95	

According to Tab.III, we can understand that with the adjustment of the default value of 15 parameters, the execution time of the algorithm is obviously shortened. In the case that 15 parameters are all at the default value, the execution time of the WordCount algorithm is 9867 seconds.

Fig.4 shows the comparison between the default value of the parameters and the execution time of the WordCount algorithm after parameter adjustment. After the parameters are optimized, the execution time of the algorithm is greatly improved.

Fig 4. Execution time

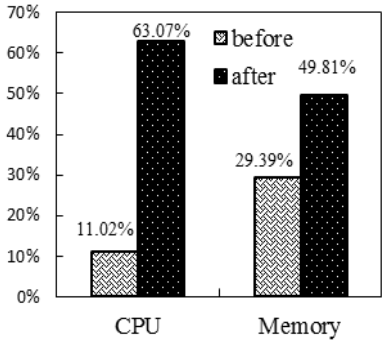

Fig 5. Occupancy

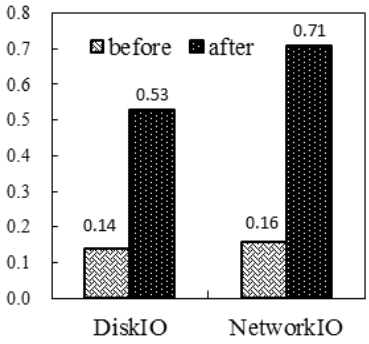

Fig 6. Throughput

Fig.5 shows clearly parameters default values and adjusted during the execution of the WordCount algorithm, Hadoop compares the cpu and memory usage rate of the server, the occupancy rate of the cpu is obviously increased, and the memory usage rate is also increased.

Fig.6 clearly shows the implementation of the WordCount algorithm after parameter default value and parameter adjustment. Hadoop significantly increased the server's DiskIO and NetworkIO throughput.

They are verified that they can, to some extent, improve the performance of Hadoop.

IV. SUMMARY

This paper mainly validates the parameters of Hadoop through the running time of the WordCount algorithm in the benchmark test, and then optimizes the performance of Hadoop. Hadoop performance tuning involves not only the performance tuning of Hadoop itself, but also the tuning of systems, such as lower-level hardware, operating systems, and Java virtual machines. In order to make better use of Hadoop, we should optimize the Hadoop parameters according to the actual application scenarios and requirements so that the Hadoop performance can be optimized under certain conditions.

REFERENCES

[1] Apache. Hadoop. [2013-07-05]. http:// Hadoop . Apache. org. index. Html.

[2] Ashlesha S, R. M. A Review of Hadoop Ecosystem for BigData[J]. International Journal of Computer Applications, 2018, 180(14):35-40.

[3] Trivedi M, Nambiar R. Lessons Learned: Performance Tuning for Hadoop Systems[M]// Performance Evaluation and Benchmarking. Traditional - Big Data - Interest of Things. 2017.

[4] Lammel R. Google's MapReduce programming model—revisited[J].Science of Computer Programming, 2008, 70(6):1-30.

[5] Besard T. Leenknegtz T. Vanhecke S. Adapting Hadoop task sizes to TaskTracker capabilities [EB/OL]. [2012-03-07]. http:// maleadt. be. 8080 /downloads/ods.