SLA-based Hadoop Capacity Scheduler Algorithm

Hongsong Chen
School of Computer and Communication Engineering
University of Science & Technology Beijing
Beijing, China

Dongqin Cui
School of Computer and Communication Engineering
University of Science & Technology Beijing
Beijing, China

***Abstract*—Hadoop capacity scheduler algorithms can not meet the different requirements of different users on service quality of cloud computing. Therefore, this paper proposed the SLA-Capacity scheduler algorithm which on the basis of the existing capacity scheduler algorithm, introduce the concept of SLA, design weighted scheduler algorithm based Qos (Quality of Service) constraints. Due to each user's jobs with different QoS preference, According to the part of QoS parameters to calculate the weight value which be used to decide the job scheduler order, so as to ensure that the user's service quality, improve user satisfaction. Experiments on Hadoop show that improved algorithm can effectively reduce the waiting time for user operation, improve the overall operating efficiency.**

Keywords—cloud computing;Hadoop; capacity scheduler algorithm;SLA;Qos

I. INTRODUCTION

Cloud computing is a calculation based on the Internet, via the Internet heterogeneous, autonomous services for individuals or businesses to provide on-demand computing that is taken. The core concept is that resources are dynamically scalable and virtualized, users can share hardware and software resources and information via the Internet . Cloud development has already made a lot of achievements, the most representative achievements including Google distributed file system GFS, Amazon Elastic Compute Cloud service and IBM's "Blue Cloud" plan.

Hadoop is a distributed system infrastructure which developed by the Apache Foundation, the core framework is distributed file system HDFS (Hadoop Distributed File System) and offline computing framework MapReduce. HDFS provides storage, and the MapReduce provides calculations for vast amounts of data. In the Hadoop platform, as a pluggable component, the job scheduler has a crucial impact on the overall performance of the platform. But now, hadoop own scheduling algorithm have some of drawback that does not consider the diversity of user job, they cannot guarantee the quality of service, thereby reducing the user's satisfaction.

In this paper, we proposed the SLA-Capacity scheduler algorithm, design the weighted scheduling algorithm based Qos (Quality of Service) constraints.

II. RELATED WORK

Currently there are three kinds of hadoop main job scheduling algorithm:

A. FIFO scheduler

FIFO scheduler FIFO scheduler is Hadoop early scheduler. All jobs are submitted to the same queue and sorted in order by submitted time, the new jobs are inserted into the tail of the queue. After a job is running, the first job in the queue is the next. The advantage of this scheduling strategy is simple and easy to implement. But its disadvantages are also obvious , all jobs are equal in the sight of it, without considering the urgent job and small job.

B. Capacity scheduler

Capacity scheduler is multi-user scheduler developed by Yahoo! The scheduler take queue as a unit to divide resources, each queue can be set a percentage of the minimum guarantee and resource usage limit, and all jobs in the queue share resources. Each has strict ACL list to regulations on its users queue, each user can allows others specified to view their job's status or control those applications. Each user can also set a certain resource usage limit in order to prevent the abuse of resources, when a queue of resources remaining, they can temporarily share the remaining resources to other queue, if you need to select a job to run, the algorithm would first select a queue with the most free space, then select the appropriate job based on priority and submission time of the job. Capacity scheduler supports multiple users and multiple applications to share a cluster running at the same time, but for different users, each user does not be considered the diversity and the different quality requirements of the job.

C. Fair Scheduler

Fair Scheduler is the multi-user scheduler developed by Facebook. It is similar to the Capacity scheduler that take queue as a unit to divide resources, each queue can be set a percentage of the minimum guarantee and resource usage limit, and all jobs in the queue share resources. Different with the Capacity scheduler is that Fair Scheduler can choose to allocate resources for an application in accordance with the FIFO, Fair or DRF policy in each queue. Fair Scheduler provides load balancing mechanism based on a number of tasks, the mechanism of the system can distribute tasks to each node as evenly as possible. In addition, users can design load balancing mechanism according to their needs.

III. SERVICE LEVEL AGREEMENT

Service Level Agreement, abbreviated as the SLA, which is a signed contract between the customer and the service provider the contract defines the content type of service, service quality and customer payments, also contains when a service provider in violation of the service level agreements, service providers should make compensation for the user, etc. Between service providers and users often need to negotiate a service level agreement after negotiation, response time, price, violation fines and other QoS (quality of service) content specified in the agreement, so as to ensure their own interests and service quality.

With the development of cloud computing technology, more and more enterprises to migrate applications and services to the cloud, but cloud computing security problems have also been more and more people's attention. In order to safeguard the enterprise and cloud service providers their own interests, SLA need to be signed between enterprises and cloud providers. In the service level agreement, the type of service and service parameter index need to be specified for ensuring that the interests of users better.

IV. SLA-Capacity Scheduler Algorithm

A. Algorithm thought

In the Capacity scheduler, YARN uses three levels resource allocation policies, Figure 4-1 shows when there are free resources on one node, It will in turn select a queue, applications, and container (request) to use the resource , The following will introduce three resource allocation policies so:

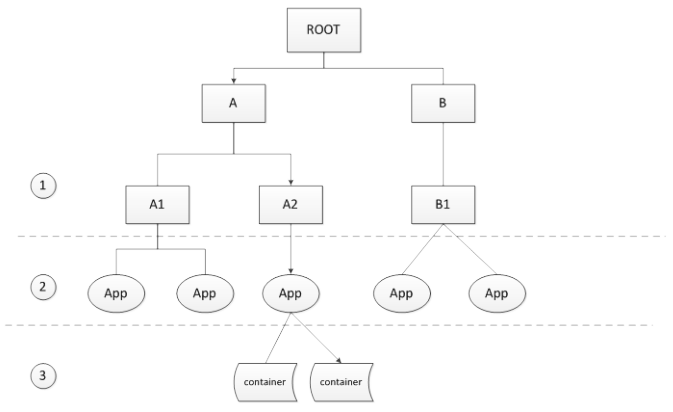

Fig.1. Capacity Scheduler resource allocation process

1)Step 1: Select the queue. YARN queue using a hierarchical organization, convert this structure to a tree, and then select the queue based on the priority use of depth-first traversal methods. As follows: starting from the root queue, according to subqueue resource usage (equivalent priority) ascending to traverse each sub-queue until you find the highest priority of the leaf nodes.

2)Step 2: Select the application. Select a leaf node in step one after, Capacity scheduler sort application according to the Submitted time of the application(In accordance with the Application ID is actually sort, because the earlier submission, Application ID is smaller), Sequentially traversing sorted applications.

3)Step 3: Select container (task request). For the same application, Container it requests may be diverse, with different priorities, nodes, resources, and quantity. When selected an application, Capacity scheduler attempts to give priority to high priority Container.

Capacity scheduler priority scheduling to application, according to the application submitted time, without taking into account the characteristics of the user's job diversity and quality of service requirements, Algorithm of this paper is mainly aimed at the second step to choose the application, Application re-sorted by SLA weight value calculated, priority scheduling weight large applications, thus ensuring the quality of the user's service, improve customer satisfaction.

B. Qos parameters

When signed service level agreement ,Cloud service providers and users must be agreed for the definition of Qos parameters, in designing the SLA-based weight calculation function, this article will focus on the following several parameters index:

1) Response time: It indicates the response time of cloud service providers of various services requested by the user. Cloud service providers must respond to user requests within a good time , otherwise the cloud service provider in violation behavior, need to pay compensation for violation to the user.

2) Price: Cloud service providers require prices for a variety of business, issued by the cloud service provider. Users must pay the price requested cloud service providers to use a cloud service.

3) Penalty: When out of the user and the service provider in the SLA negotiated various service performance, need to pay liquidated damages.

C. Algorithm Design and Implementation

According to the selected Qos property, job's weight value is related to response time, response smaller, the weight should be greater, inverse relationship. And weight value has a close relationship with price and penalty, when the price of cloud services or penalty higher, the corresponding weight value should be greater, it is proportional relationship. Thus, we can draw the following calculation functions:

$$sla = \beta * (price + penalty) / (\alpha * response) \qquad (1)$$

α, β is a coefficient, and $\alpha > \beta$, $\alpha + \beta = 1$.

In order to more accurately calculate the weight of job, dynamically adjusting the weight value is necessary according to the actual situation of job , especially considering the priority of the job and waiting time in queue. Thus we have:

$$weight = sla * (currentTime - startTime) * priority \qquad (2)$$

In which, currentTime representing the current time, startTime indicates when the job was submitted, priority indicates the priority of the job.

When the user submits a job, according to the SLA agreement signed by user and provider to obtain negotiated Qos parameter values, then pass the parameter value to the cluster. After the cluster receiving parameter value ,calculating sla value by formula (1), on this basis, we can dynamically calculate weight value by formula (2). Finally, according to the calculated value of the weight , cluster sort to the application in queue, the greater the weight, the higher the priority, job priority scheduling.

last but not least, replace the original Capacity scheduler with improved SLA-Capacity scheduler to schedule job in the queue.

The key part of the algorithm of this paper is to modify the applications comparator of original queue, then reorder applications. Comparator pseudo-code as follows:

```
Input:Qos Parameters,job1,job2
Output:int
//get Qos parameters
sla = β * (price +penalty) / (α *response)
// get priority value
weight = sla * (currentTime - startTime) * priority
if(job1. Weight < job2. Weight)
  return -1;
else if((job1. Weight > job2. Weight)
  Return 1;
Else if(job1. Weight > job2. Weight)
  Return job1.startTime < job2.startTime ? -1 :1;//if equal,compare
time
```

V. Experimental Analysis

A. Experimental environment

In this study, three virtual machines, operating systems is CentOS6.5, jdk version is 1.6.0_45, hadoop version is 2.4.0. One virtual machine is master, the remaining two as slave.

B. Experimental results

In order to compare the performance of our algorithm with traditional scheduling algorithm, we selected four different user, including hadoop, hadoop1, hadoop2, hadoop3, and selected commonly used text word count jobs – word count, designed a different text size for each user.

1): Using traditional capacity scheduler algorithm Four users uploaded word count jobs to the cluster at the same time. Several experiments, each experiment recorded job run time of each user's job and calculated average run time ,The results shown in Figure 2

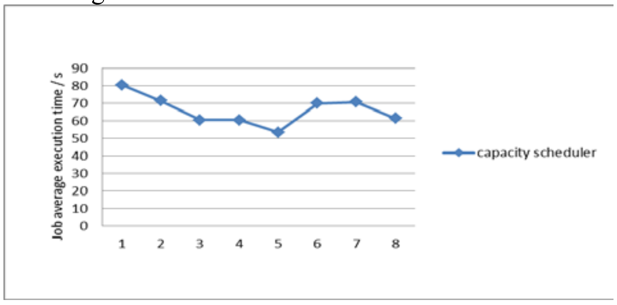

Fig.2. capacity scheduler algorithm

2): Redesign word count job to form a new wordcountsla job by the QoS parameter values are passed as parameters to the job, then the four users uploaded word count jobs to the cluster at the same time. Several experiments, each experiment recorded job run time of each user's job and calculated average run time.

Calculation of the weight depends on the priority of a job and job submission time, which corresponds to the value of the priority of a job as table Ⅰ.

TABLE I JOB PRIORITY VALUE DESIGN

Priority	Value
VERY_LOW	1
LOW	2
NORMAL	3
HIGH	4
VERY_HIGH	5

For more information about the job selected, such as table Ⅱ.

TABLE II JOB INFORMATION

user	Text size/K	Qos parameter(response/s，price/w，panalty/w)
Hadoop	301	3, 10, 1
Hadoop1	251	3, 10, 2
Hadoop2	326	4, 10, 1
Hadoop3	201	3, 15, 1

Using SLA-Capacity scheduler proposed in this paper, record the average time job of each experiment, show as figure 3.

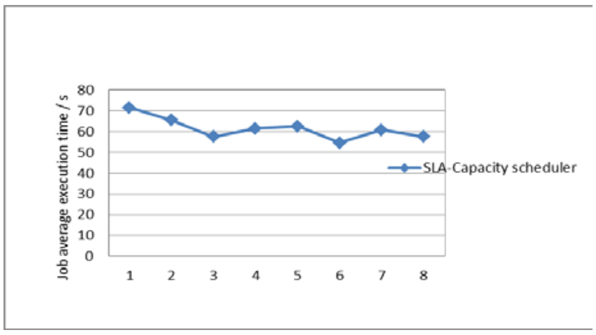

Fig.3. SLA-Capacity scheduler algorithm

By Figures 5-1 and 5-2, we can see that the volatility of average time for each experiment using capacity scheduler algorithm is greater than the volatility using SLA-Capacity scheduler algorithm, In comparison, the latter has better stability, it does not occur that reducing the user's satisfaction because of excessive job execution time. According to the above results, we can draw the overall average time job of the two kinds of algorithms, obtain a conclusion as Figure 4.

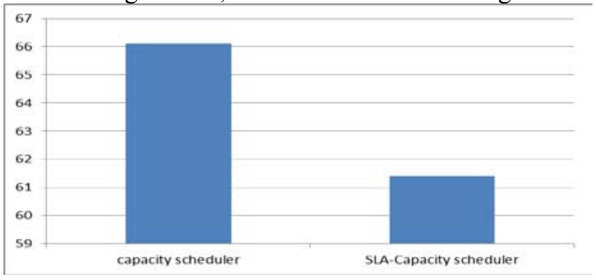

Fig. 4. overall average time

Figure 5-3 shows that the improved average total execution time of job is significantly lower than the average total execution time before the improvement, so this paper proposed SLA-based algorithm has higher efficiency on the overall level, and improve the satisfaction of users.

VI. CONCLUSION

This paper presents the SLA-Capacity scheduler based on user service level agreement (SLA), which design capacity weighted scheduling algorithm based Qos (Quality of Service) Constraints. Because the job of each user submitted has different QoS preferences, we can computing job weights by it that determine priority scheduling jobs, which ensure a relatively important user jobs can receive priority scheduling, improve customer service quality and customer satisfaction. Experiments on hadoop show that improved algorithm can effectively reduce the waiting time for user operation, improve the overall operating efficiency.

VII. ACKNOWLEDGEMENTS

This work was supported by Beijing Natural Science Foundation (4142034), Beijing Science and Technique Plan Project (No.D141100003414002), China Scholarship Council Foundation, Beijing Higher Education Young Elite Teacher Project (YETP0380), Fundamental Research Funds for the Central Universities (FRF-TP-14-042A2). Chinese National 863 research project (No.2013AA01A209).

References

[1] V. Venkatesa Kumar and K. Dinesh. Job Scheduling Using Fuzzy Neural Network Algorithm in Cloud Environment. Bonfring International Journal of Man Machine Interface[J], 2012,2(1):1-6.

[2] Linlin Wu, Saurabh Kumar Garg, Steve Versteeg and Rajkumar Buyya.SLA-based Resource Provisioning for Hosted Software as a Service Applications in Cloud Computing Environments[J]. IEEE Transactions on Sevices Computing.2013.

[3] Eyraud-Dubois L,Larcheveque H. Optimizing Resource allocation while handling SLA violations in Cloud Computing platforms[C].2013 IEEE 27th International Symposium on Parallel & Distributed Processing.2013: 79 - 87.

[4] Zhuo Tang, Junqing Zhou, Kenli Li, Ruixuan Li. MTSD: A task scheduling algorithm for MapReduce base on deadline constraints[C]. 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum. 2012: 2012 - 2018.

[5] Perret Q,Genie Electr,Charlemagne G,Sotiriadis S,Bessis N. A Deadline Scheduler for Jobs in Distributed Systems[C]. 2013 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA).2013: 757 - 764.

[6] Yongcai Tao ,Qing Zhang ,Lei Shi ,Pinhua Chen. Job Scheduling Optimization for Multi-user MapReduce Clusters[C]. 2011 Fourth International Symposium on Parallel Architectures, Algorithms and Programming (PAAP).2011: 213 - 217.

[7] Manjaly J.S,Chooralil V.S, TaskTracker Aware Scheduling for Hadoop MapReduce[C]. 2013 Third International Conference on Advances in Computing and Communications (ICACC).2013: 278 - 281.

[8] Raj A, Kaur K, Dutta U,Sandeep V.V, Rao S. Enhancement of Hadoop Clusters with Virtualization Using the Capacity Scheduler[C]. 2012 Third International Conference on Services in Emerging Markets (ICSEM).2012:50-57