Research on the load balancing strategy for original pages based on cloud storage

Kunpeng Kang

School of Computer and Information Technology, Shangqiu Normal University, Shangqiu, People’s Republic of China

ABSTRACT
Load balance plays an important role in the information acquisition system’s performance. Especially in the state of cloud storage, the load balance is well done, which is conducive to the full utilization of computing resources and reduces the response time of distributed operations. The internal mechanism of original page load balance is given based on the analysis of the two recent commonly used dynamic load balance methods. Five original page oriented load balancing strategies are compared from the experimental and theoretical perspectives on the premise of calculating load index. Finally, the conclusion is drawn that date channel storage calculation sensitive partition is the most optimal load partition strategy.

ARTICLE HISTORY
Received 23 February 2019
Accepted 15 May 2019

KEYWORDS
Cloud storage; original page; load balance; polling

1. Introduction
Cloud storage is a new type of distributed storage system, which combines a large number of network and different types of storage devices to provide data storage services by using various software applications based on large computer application clusters and distributed file systems to work together. The query of massive data in cloud storage needs a distributed search engine, and load balancing should be considered to avoid the appearance of hot spots. The load balancing in the storage system of the distributed search engine can be divided into two aspects from the machine resources: hard disk storage load balancing, including the original page, page content and index files in different nodes of the hard disk storage. Load balance of computation refers to the load balancing of the calculation among the storage nodes (Wang, Li, Xiong, & Pan, 2012). Each node needs to carry out many computing tasks, including the original page writing disk, the original page content extraction, content page update, index update, index query.

From the point of view of the application, the load balancing of these computing and storage resources may be divided into two stages. The first stage is related to the original page, with the crawler node writing the original page drive on the system. For the storage, it is the three data volume increasing, and for the calculation it is writing disk caused by the original page writing disk, content extraction, index and content page update; the second stage is relevant to the index query, between storage nodes equilibrium index query load, to make the system have a faster response time of the distributed index query. The load balancing of the first stage is critical for equilibrium and full use of the storage capacity, as well as for the improvement of the writing response time of the grasping subsystem. The load balancing of the second stage can optimize the index query response time (Bonvin, Papaioannou, & Aberer, 2010).

The storage facing the original page and computational load balancing is mainly studied in this paper. The second section analyses the research status of the original page load balance; the third section puts forward the original page load balance system; the fourth quarter puts forward five strategies of the original page load balancing; the fifth section carries out the test and performance analysis for the load balancing strategies; and the last section is the conclusion.

2. Relevant task
In cloud storage, when the crawler node of the distributed collection subsystem writes pages to the storage system, it is necessary to ask the storage system management node before the crawler writes pages to the storage node of the storage system to get the storage node to which the page should be stored. The load balancing strategy is implemented at the time when the management node
writes requests to the crawler to allocate the storage node. Load balancing strategy implementation begins at the moment when the management node writes the request to the crawler to distribute the storage nodes. The main task of the balance algorithm is to decide how to choose the next node, then transmit the new service request to it. A good load balance algorithm is not omnipotent; generally speaking it is closely linked with the application scene (Lakshman & Malik, 2010). So we should consider the equalization algorithm comprehensively according to the characteristics of the process and make use of different algorithms and techniques. At present, there are two commonly used dynamic load balance methods.

2.1. Polling method

In a task queue node, every member has the same status. The polling method simply makes cycle selection in turn in this group. In a load balancing environment, the algorithm will distribute new request next node in this node queue, so continuous, round and round. Each node is chosen in turn in equal status. Polling method activity is predictable, that is, each node's chance to be chosen is $1/N$ (assuming there are $N$ nodes). The polling method is the most simple and most easy implementation approach. This method does not consider the machine isomerism.

2.2. Load index method

The load index algorithm calculates nodal Load Index $LI$ (Load Index) based on the node current Load condition. The load index constitutes the Load priority queue of a storage node, it takes the team node from priority queues and forwards service request every time when the service request arrives (Putrycz & Bernard, 2003). The load index is a dynamic estimate. The disadvantage of this method is that the cost of the dynamic monitoring of the load index and the calculation is too high.

3. The original page load balance system

On the cloud platform, the computation load of this part is the first two stages of three processing phases of a storage node, namely the original page analysis storage and the content page processing. The content page processing needs to update the content page and index; the storage load refers to the hard disk storage of three data, of which the storage of the original page accounts for more than 80% of the storage; the weight of the original page load balance is a maximum.

The first target of the load balance is that the storage capacity of the machine can be used fully and balanced, make full use of the system’s hard disk storage space; secondly, the computing power is equalization, avoid calculation of overloading of the node and make the whole storage system complete analysing the received original page as soon as possible.

Figure 1 is the load balancing schematic diagram. First of all, grab the subsystem's crawler inquiry storage unit; then the load balancing module chooses the storage node of which the current load index is the smallest of the storage destination of this unit. Finally, modify Figure 2 files in a directory information sheet, record the unit storage information for subsequent query, and return the unit storage node information to the crawler.

4. The original page oriented load balancing strategy

The isomerism of the machine is not considered in the Simple polling method; on the other hand, even if the isomerism is considered, there are differences in the size of the cells partitioned by task, in order to ensure the instantaneousity of the information collection, we cannot make the big unit write request after the cache. There is great uncertainty as long as the storage unit is big enough.

The premise of the load index method is to designate the distributed system to a load index which can correctly reflect the current load condition of the system. The definition of the load index is critical (Chen, Li, & Chen, 1997). The literature suggest using resource utilization rather than the resources queue length as the load index. Besides, in the distributed applications, processing memory, hard disk, CPU, I/O, etc. will affect the overall speed. It is more practical to define the composite load index comprehensively according to the nature of the task (Nong, Jinfeng, & Yutong, 1998).

To balance the relevant storage and computational load of the original page, we considered the overall effect of the size of the load cell, load balance method, load information acquisition and load balancing algorithm for the storage system, and designed a variety of load balance methods. In each method, the load index is defined (the polling method after considering machine isomerism transmitting into the load index method, just without dynamic monitoring load information).

4.1. Page round partition

The Page Round Partition P-RP (Page Round Partition) method regards the original page as the load partition unit, proportionally distributes load according to the initial storage capacity of each storage node. It does not take
Load index calculation is shown as follows:

$$LI_i(t) = \frac{\text{pages}_i(t)}{S_i}.$$  \hspace{1cm} (1)

In the formulation, pages$_i(t)$ means at moment $t$, the page numbers that the storage node $i$ has received, $S_i$ refers to the hard disk storage capacity of storage nodes $i$. This method hypothesizes that the page size is equal, due to the more number of pages and pages being small files. The equalization of the number of the page represents the equalization of the storage capacity so it can balance capacity load very well. But this strategy brings much metadata traffic and yuan data storage to the management node. Because the number of the page in the system memory is large, the management node cannot stand these loads.

### 4.2. Channel round partition

The channel round partition C-RP (Channel Round Partition) method regards Channel as the load partition unit, proportionally distributes load according to the initial storage capacity of each storage node, and not take the balance of computing power into consideration. Its load index calculation is as follows:

$$LI_i(t) = \frac{\text{channels}_i(t)}{S_i}.$$  \hspace{1cm} (2)

In the formulation, channels$_i(t)$ means at the moment, the channel numbers that the storage node $i$ has received, $S_i$ refers to the hard disk storage capacity of the storage nodes $i$. This method hypothesizes the channel size is equal, and its load is uneven because the channel number is not large and the sizes are different; on the other hand, its load difference will become worse with the passage of time because once a channel is assigned to a node,
the data that the channel later grabs will be stored in this node, then the original load difference will become greater with time. But the related operations of page content of this method can only be carried out in the machine, making the whole system simple in design.

4.3. Channel date round partition

The Channel Date Round Partition CD-RP (Channel Date Round Partition) methods regard the day Channel as the load partition unit, proportionally distributes load according to the initial storage capacity of each storage node. It does not take the balance of computing power into consideration. Its load index calculation is as follows:

\[ L_i(t) = \frac{\text{DateChannels}_i(t)}{S_i}. \]  

In the formulation the \( \text{DateChannels}_i(t) \) refers to the day channel number stored by the storage node \( i \) at the moment \( t \), \( S_i \) refers to the hard disk storage capacity of storage node \( i \). This method hypothesizes that the size of day channel is equal, though there is a great difference between the channel sizes, the day channel number is equal, though there is a great difference between the channel sizes, the day channel number is equal, though there is a great difference between the channel sizes.

4.4. Channel date storage-sensitive partition

The Channel Date Storage-Sensitive Partition CD-SSP (Channel Date-Storage-Compute-Sensitive Partition) method regards Channel date as the load partition unit, real-time measuring the storage nodes disk and computational load, calculating the comprehensive load index, and assigning the load unit to its lower index. Its load index calculation is as follows:

\[
\begin{align*}
L_i(t) &= K & \rho_i^S(t) \geq \rho_i^C(t) \geq \rho_i^C(t) \\
L_i(t) &= u_1 \rho_i^S(t) + u_2 \rho_i^C(t) & \rho_i^S(t) < \rho_i^C(t) \\
L_i(t) &= u_1 \rho_i^S(t) & \text{others,}
\end{align*}
\]

In the formulation, \( \text{DateChannels}_i(t) \) refers to the data quantity stored by the storage node \( i \) at the moment \( t \), \( S_i \) refers to the storage capacity of the hard disk of storage nodes \( i \). This method distributes load according to the storage occupation rate on the basis of CD-RP, which reflects the storage load condition of nodes. But in addition to the shortcoming CD-RP, the management node needs to real-time monitoring disk’s occupation quantity of the storage nodes to increase the load balance mechanism expense.

4.5. Channel date storage-compute-sensitive partition

The Channel Date Storage-Compute-Sensitive Partition CD-SCSP (Channel Date-Storage-Compute-Sensitive Partition) method regards Channel date as the load partition unit, real-time measuring the storage nodes disk and computational load, calculating the comprehensive load index, and assigning the load unit to its lower index. Its load index calculation is as follows:

\[
\begin{align*}
L_i(t) &= K & \rho_i^S(t) \geq \rho_i^C(t) \geq \rho_i^C(t) \\
L_i(t) &= u_1 \rho_i^S(t) + u_2 \rho_i^C(t) & \rho_i^S(t) < \rho_i^C(t) < \rho_i^C(t) \geq \rho_i^C(t)/2, \\
L_i(t) &= u_1 \rho_i^S(t) & \text{others,}
\end{align*}
\]

\( \rho_i^S(t) = (S_i^{\text{used}}(t) / S_i) \) is the hard disk occupancy rate of storage nodes \( i \) at the moment of \( t \); \( \rho_i^C(t) = (C_i^{\text{used}}(t) / C_i) \) refers to the calculation occupancy rate of storage node \( i \) at the moment \( t \); \( u_1 \) and \( u_2 \) refers to storage and the calculating adjustment factor, respectively, equally large; \( \rho_i^C(t) \) is storage and computing occupancy rate threshold, respectively, being less than and close to 1; \( K \) is a very large number, on behalf of overload. When the node calculates or memory occupancy rate is more than the threshold, the load index indicates overload; in the premise of not more than the threshold value, if the node calculation load is up to more than the half calculation threshold value, it will add the calculation load to the load index, otherwise, it only takes the capacity load as the load index. This method adds the calculation load regulation factor based on the CD-SSP method, gives full consideration to the influence on the system load of storage and calculation. The shortcoming is load information acquisition quantity increases.

4.6. Dynamic load information acquisition

The acquisition of load information is the premise of calculating the load index. The fast and simple load information acquisition mechanism can ensure the load index be timely and effective. The first three kinds of methods do not need to acquire the dynamic load information because the management node has a distribution unit of each node (Bozyigit & McIhi, 1997). The latter two strategies need to obtain dynamic load information: storage load and computational load.

The storage load is the hard disk occupancy rate; each node obtains the disk occupation through the system call and then takes the information real-time feedback to the management node. The content page analysis is the most onerous link of storage node calculation. The computational load takes the original page cache data quantity to be
analysed of the storage nodes as the measurement, and takes the buffer occupancy rate as the calculating load to be sent to the management node at a true time (Harchol Balter & Downey, 1997).

5. Experiment

Since the Page Round Partition method is infeasible in actual deployment, we do not make a further experimental analysis. This paper makes a further experimental measurement with the latter four methods about their equilibrium effect. Two experiments are carried out, and each experiment takes six days. In each experiment, the storage node is 8; the storage node operating system is Red Hat Enterprise Linux AS release 4. The former one is isomorphism storage nodes, the machine’s processing speed is the same between hard disk and CPU; the latter one is heterogeneous storage nodes, the machine’s processing speed is different between the hard disk and CPU. No matter which group of machine, the network card is the MB Etheric card, the size of the cache area of content analysis is 1 G. At the end of the first day of each experiment, the load is recorded, and at the end of the sixth day, the load is recorded again. Coefficient of variation is a statistical mathematical concept, its value is the mean square error divided by average of the statistic data, representing the distribution of the data. If the coefficient of variation is smaller, the difference between the data is smaller, and the distribution is more average; on the contrary, if the difference is bigger, the distribution is uneven. In this paper, the load balance is measured, calculation or storage load of each machine is quantized, and then the coefficient of variation of these values is calculated. By using load coefficient of variation we indicate storage system load balance. The smaller the load coefficient of variation, the load is well distributed. Statistics are made with three kinds of load Coefficient of Variation.

- Storage load Coefficient of Variation (S-CV): Coefficient of Variation of hard disk space occupancy of all storage nodes, this value represents storage load distribution.
- Computation load Coefficient of Variation (C-CV): Coefficient of Variation of content analysis buffer occupancy of all storage nodes, this value represents computation load distribution.
- Total load Coefficient of Variation (T-CV): Average of S-CV and C-CV values, indicating total load distribution of the system.

This paper made statistics on the distribution of the data channel partition unit. Figure 3 is data channel data distribution, a comparison is made between its distribution and random data distribution.

In random data, median and mean are nearly equal; the coefficient of variation of all values is 0.5, well-distributed.

In the actual data channel data, small channels are many, the greater the channel, the sparser the distribution; the coefficient of variation of all the values is large, up to 2.9; the median value is much smaller than the mean, indicating the data tends to the small. The difference between the maximum and minimum of the actual data channel data is great; the biggest reach 2G, and the minimum is 100 k.

Figures 4 and 5 are the result of the first group of experiments. With the C-RP method, storage load deteriorates, the effect of long-term operation is the worst; with the CD-RP method, although the load coefficient is larger in the first day, it declined in six days, which is the adjustment by the Date Channel Partition; C-RP and CD-RP are two methods based on the storage round, S-CV is much greater compared with the non-round method; the CD-SSP method and CD-SCSP method are the most load balanced, they are almost the same. The CD-SCSP method is a little better in the calculation of loading. With the CD-SSP method, the main reason for calculating the coefficient of variation is the isomorphism of machines, the result of storage balance is that the calculation is nearly equal for each machine (calculation is written by the original page drive).

Figures 6 and 7 are the results of the second set of experiment. Compared with the first set of experiments, the C-CV of this group of data of the former three methods is particularly big, because with the former three methods it only carries out storage load balancing, without considering computation load, to make the machine calculation overload, which will, as a result, increase the crawler writing response time. The CD-SCSP method still has the least load coefficient of variation, although its S-CV is bigger than CD-SSP, its C-CV is much smaller, and in the CD-SCSP method, S-CV and C-CV both are about 0.3. The system can accept this uneven load.
A comparison of two groups of the experiments shows that the CD-SCSP method has the most equilibrium computing and storage load distribution in both isomorphism and heterogeneous fleet. The CD-SSP method takes the second place, and in the heterogeneous network, it could form serious uneven distribution of calculation. The round method performs the worst.

6. Conclusion
In cloud storage, load balancing during data access plays a key role in the performance of an information acquisition system. Only when load balance is well done, can we make full use of system memory and computing resources to reduce the response time of the distributed operation. This paper deals with balance processing to storage and calculation related load brought by the original page write system, and puts forward a variety of load balancing strategies, and further measures the load balance effect of each scheme through experiments. Finally, we find that data channel storage calculation sensitive partition is the most optimal load partition strategy. It efficiently allocates access to multiple access units, avoiding the formation of access hotspots, and provides an effective and transparent way to expand the bandwidth of network equipment and services, increase throughput, enhance network data processing capabilities, and improve network flexibility and availability.

Disclosure statement
No potential conflict of interest was reported by the author.

Funding
This paper is supported by the Scientific and Technological Project of Henan Province (grant number 182102210486), the
Key Scientific Research Project of University in Henan Province (grant number 18A520008).

**ORCID**

Kunpeng Kang © http://orcid.org/0000-0002-0310-9592

**References**

Bonvin, N., Papaioannou, T. G., & Aberer, K. (2010). A self-organized, fault-tolerant and scalable replication scheme for cloud storage. *Proceedings of the 1st ACM symposium on cloud computing* (pp. 205–216). New York: ACM.

Bozyigit, M., & Mclhi, M. (1997). Load balancing framework for distributed system. *Computer System Science & Engineering, 12*(5), 287–293.

Chen, H., Li, J., & Chen, G. (1997). Task scheduling in parallel distributed computing (Part 2). *Journal of Computer Science, 24*(2), 23–27.

Harchol Balter, M., & Downey, A. B. (1997). Exploiting process lifetime distributions for dynamic load balancing. *ACM Transactions on Computer Systems, 15*(3), 253–285.

Lakshman, A., & Malik, P. (2010). Cassandra: A decentralized structured storage system. *ACM SIGOPS Operating Systems Review, 44*(2), 35–44.

Nong, X., Jinfeng, H., & Yutong, L. (1998). Dynamic load balancing strategy of network parallel computing. *Computer Engineering and Science, 20*(3), 13–17.

Putrycz, E., & Bernard, G. (2003). Connecting frameworks: Case study with middleware-based load balancing. Proceeding of the 23rd ICDCSW03 (pp. 89–92), Providence, Rhode Island, USA.

Wang, Z., Li, T., Xiong, N., & Pan, Y. (2012). A novel dynamic network data replication scheme based on historical access record and proactive deletion. *The Journal of Supercomputing, 62*(1), 227–250.