Advances in Intelligent Systems Research, volume 133
2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE2016)

OLP Scheme on Backup Log and HBase

Chao Feng[1] and Baoan Li[1, 2, *]
[1]Computer School, Beijing Information Science and Technology University, Beijing 100101, China
[2]Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, Beijing Information Science and Technology University, Beijing 100101, China
*Corresponding author

***Abstract*—As a non-relational database, HBase (Hadoop Data Base) is an open source data storage system based on column cluster and is applied wisely. The HBase will write the log file before the memory caches data, so the cache size and the writing speed of the log file have become two important factors affecting the system performance. Based on backup log, this paper proposes a persistence and availability scheme OLP (Other Log Process), which makes HBase can achieve better performance in different cache sizes. Experiments show that under the premise of ensuring the persistence and availability of data, OLP can obtain stable performance under different cache sizes, and obviously improve the time performance of write operation when the cache does not exceed the default** settings.

Keywords-component; HBase; persistence; availability; OLP; write operation efficiency

I. INTRODUCTION

For a data storage system, the persistence and availability of data are the basic principles that which must be guaranteed [1]. In HBase, there are two ways to ensure the persistence and availability of data:

- Data needs to be written to the disk eventually. Disk is a non-volatile storage medium, and it is generally believed that the data on the disk can be guaranteed for persistence and availability [2];
- Using WAL in HBase (Write Ahead Log) in HBase ensures that HBase can also provide effective data recovery capability even if the data node failed, and the logs can also guarantee the persistence and availability of data that have not yet been written to disk [3].

Certainly, the complex security measures greatly affect the actual operating efficiency of the database, especially for the writing of a large number of data, because disk IO operation brought by frequent write file system is bound to reduce the write operation efficiency of database. Therefore, this paper aims to propose a new data processing flow, in the process of writing operation, transfer the log backup to other idle nodes, reduce the frequency of disk writes in current node, improve the performance of data write operation time and the utilization of cluster, at the same time reduce the sensitivity of HBase writes operation time performance to MemStore size.

II. HBASE STORE PROCEDURE

A. *Hbase Write Operation*

HBase provides two kinds of mechanisms to ensure the persistence and availability of data: persist the data from non-volatile memory to disk and log system WAL (Write Ahead Log) [4]. For the write operation of HBase, the application of the two means could be reflected fully during its execution. The write operation process of HBase shows as in Figure I.

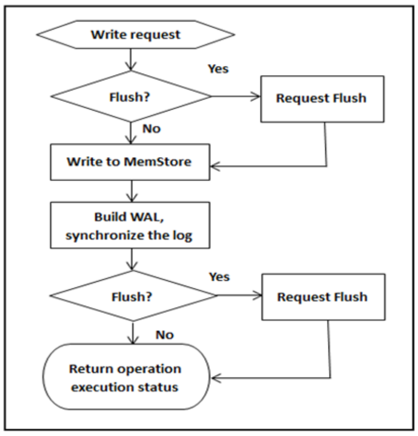

FIGURE I. HBASE WRITE OPERATION EXECUTION PROCESS

a) After HRegion receiving a write operation request, first check the current HRegion, and check whether the data persistence request (Request Flush) needs to be sent, if necessary, first request the current HRegion MemStore in Flush, otherwise process the next action;

b) Write data to MemStore, because MemStore is in memory, this process does have Flush request, th en, continue to carry out the third step;

c) Build WAL, write log, then synchronize the log, and write the log record to file system as soon as possible;

d) check whether the current HRegion needs to initiate a Flush request, if necessary, send Request Flush;

e) after the end of write action, return to the state of execution.

B. Data Persistence

Data persistence is a basic means to ensure data persistence. It stores data and data modification in a non-volatile storage medium. For HBase, namely, write data to relying on file system, such as HDFS [5]. When HRegion Server starts, it will immediately start a thread (MemStore Flusher) which is used to check the status of all MemStore in the current HRegion Server, and the thread is also responsible for the Request Flush of HRegion, and then flush data in MemStore to the file system [6]. For any HRegion, when the data in MemStore needs to be written into file system, this request will add related information to queue, and it will be processed later, there are two situations in which the Flush action is triggered:

(1)When the data in a MemStore of a HRegion is greater than the set threshold, the Flush request is issued.

(2)When the sum of all the MemStore data contained in all HRegions in the HRegion Server is greater than a given threshold value, the persistence process will select several appropriate HRegions, flush their data to the file system, until the sum of the MemStore data does not exceed the set threshold.

C. Write Ahead Log

WAL (Write Ahead Log) is a log that HRegion Server records in the process of handling data manipulation requests (increase, delete, change). In each data operation process, HBase will encapsulate a WAL, then add it to the HLog of current HRegion Server, and write HLog to the file system in time, ensure the persistence and availability of log. HLog shared by all HRegions which are carried by HRegion Server, which could completely record the modification to database that user has taken [7]. Data can be restored in the node or even under the condition of system downtime, but also ensure that the data still in the MemStore will not lose easily.

III. IMPLEMENTATION OF DATA PERSISTENCE AND AVAILABILITY BASED ON OTHER LOGS

A. Other Logs Process

The core essence of OLP (Other Log Process) is to ensure the availability and persistence of data by using data backup between cluster nodes, that is to say, in the process of writing, not to persist the log to file system in a hurry, instead, send the local WAL to pre-specified remote target node (Figure II), this process is called OLP.

In an HBase cluster, it usually has a HMaster node and a number of HRegion Server nodes. HMaster is responsible for managing the HRegion Server node, and distributing backup log to target node for HRegion Server [7]. HRegion Server is the processing unit of OLP. Any HRegion Server node in cluster is both the server receiving OLP and the client initiating OLP process. That is, if there is a HRegion executes write operation process in the current client HRegion Server, the WAL will be written to the pre allocated destination node, at the same time, it will also accept the WAL forwarded by other nodes. The specific operation process shows as follow:

- When the system starts, HMaster will allocate the target node of the backup log for each HRegion Server, and writes the IP information to the Zookeeper;
- HRegion Server starts, initialize the OLP process: start the OLP service, take it as the target node(OLP server) of a node backup log in cluster; at the same time read the target node IP of the current node backup from Zookeeper, and take it as the shared variable of all HRegion in the entire HRegion Server, and send OLP call through it;
- When HRegion calls the write operation, the user data is written to the memory of MemStore, and then construct WAL, write into memory, waiting for the scheduling process to write file system, at the same time start OLP, and send the WAL to remote target node, waiting for written confirmation (OLP Client)which is sent by the object node;
- When the HRegion Server receives a WAL from a node in the cluster, write it to the local log file and returns a confirmation;
- In the setting time of write process, if write process could receive more than half of the write confirmations which are returned by target node, the writing is successful, return write success; otherwise, return write failure and initialize the OLP process.

During the process of system processes write request, the process is no longer frequently to determine whether the current HRegion data is required to write to disk, HBase can significantly reduce the frequency of disk IO.

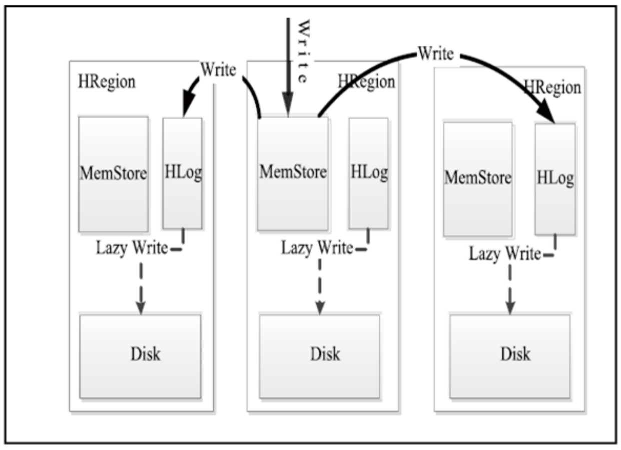

FIGURE II. OTHER LOG BACKUPS.

B. Process Description

After introducing OLP process, the flow that system processes write operation shows as Figure II, the frequency of HRegion write disk is greatly reduced. All of the write file system operations are carried out by other processes in the follow-up time, described as bellows:

a) Check HRegion, but do not check whether the need of the data in the persistent memory MemStore;

b) Write data to the MemStore, MemStore is a memory storage structure, will not request the write file system;

c) Build WAL; write the content of log into memory;

d) Sends local log records to remote backup nodes;

e) Write action is end, return to the state of execution.

C. Persistence and Usability

With the introduction of OLP process, the frequency that HRegion Server initiates data persistence process is greatly reduced. If and only if the sum of all the MemStore data in HRegion Server is more than a threshold value, start the persistence process, and write a number of selected MemStore to disk could achieve the ultimate data persistence. For data that has not yet been written to disk, the system ensures its persistence and availability by local and other node backups.

- If the node is down, it will lose the data which is temporarily stored in the MemStore. At this, when restart the node, then read the local Log file, and restore the data in MemStore.
- If the node completely fails, or the disk is damaged, or the local log has been difficult to fully recover the current node data, system can find and reorganize the other backup log files, and achieve the migration or reconstruction of Region.

After joining in the OLP mechanism, the processing flow of write operation is greatly simplified, and there is no explicit write disk operation in the process of data writing. The flow chart was shown in Figure III.

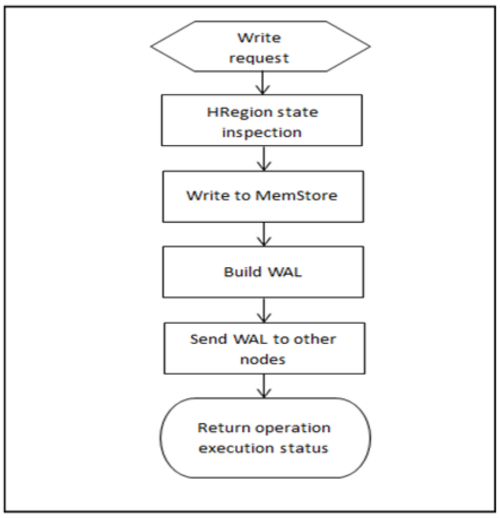

FIGURE III. OLP WRITE OPERATION EXECUTION PROCESS.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In order to demonstrate the performance of the OLP scheme, this paper carries out the write operation test according to the HBase database in fully distributed mode. Select a test case as shown in Table I to carry out test to original HBase system and switched OLP process system respectively. In fully distributed mode, the adapted test environment includes single HMaster, single Zookeeper and five cluster of HRegion Server. All of the nodes are in the local area network, and the bottom layer uses HDFS as the storage support [8]. Table I indicates the time of HBase original ecology and writing data used after joining in OLP in different MemStore sizes and different insertion data. MemStore sizes are 1M, 8M, 32M and 128M respectively. The amount of data insertion are 100 thousand, 200 thousand, 500 thousand and 1 million respectively, in order to convert it into histogram form more intuitively.

TABLE I. TIME CONSUMING COMPARISON

Number / MemStore Size / Time (MS)	**Number of data written(10 thousand lines)**			
	10	***20***	***50***	***100***
1M(HBase)	20166	30123	125000	274530
1M(OLP)	4000	5000	49312	81563
8M(HBase)	25136	34256	110230	227680
8M(OLP)	3104	12304	38000	75432
32M(HBase)	24135	35123	98145	214560
32M(OLP)	2896	12000	47125	73012
128M(HBase)	25001	30971	70125	195610
128M(OLP)	546	10021	46587	69231

A. Write Operation Time Performance Stability

When the original HBase is setting different MemStore size, the time performance of the write operation will also has a significant difference. And the different is more prominent when larger data volume is written, shown in Figure IV, when 1 million lines of data are written, the time of the write operation is significantly decreased as the MemStore becomes larger, and the same trend can be seen in other data volumes.

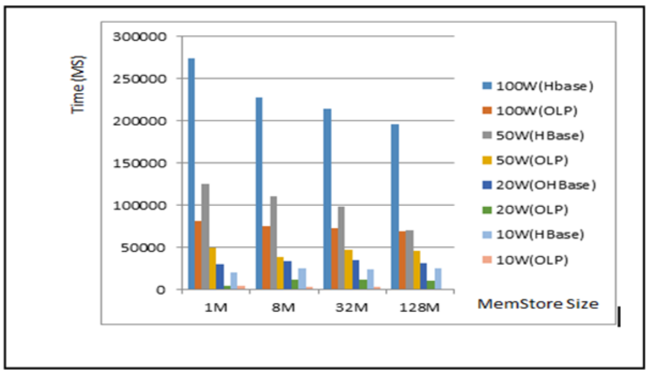

FIGURE IV. PERFORMANCE COMPARISON OF DIFFERENT MEMSTORE TIME IN DISTRIBUTED MODE.

After the introduction of OLP, the sensitivity of the system to MemStore decreases, for different MemStore values, the time of writing same amount of data is nearly same. At the same time, no matter how much the amount of data is written, the time performance of the write operation is only related to the amount of data written, and there is no large fluctuation with the change of the MemStore size. After HBase is added to OLP, the time performance of system write operation already has little relation with the size of MemStore.

B. *Write Operation Rate*

Figure V proves that after the introduction of OLP, the consumption time of HBase writing operation has little change in different MemStore size. The four time curves which use OLP are almost coincide, it has again obviously shown that HBase is no longer sensitive to the MemStore size after the introduction of OLP.

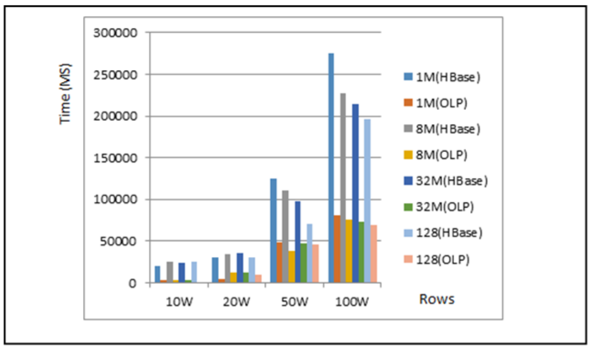

FIGURE V. THE FOUR TIME CURVES USING OLP.

When fully distributed deployment, the introduction of OLP still improves write operation time performance significantly.

According to the experimental results of fully distributed mode, it can be seen that after introducing OLP for HBase, the sensitivity of the system write operation to the MemStore size is reduced obviously, that is, the performance of the write operation time is no longer dependent on the settings of MemStore size. At the same time, when the MemStore does not exceed the default values, OLP can significantly improve the performance of the write operation time of system. On the other hand, the introduction of OLP will increase the network load during the operation of the system, and also increase the log number of the whole cluster.

V. CONCLUSION

HBase relies on the write data to a file system to ensure the persistence and availability of data, so there will be a large number of write file system operations. This paper proposes to use the data backup between nodes in cluster to ensure the persistence and availability of data, reduce the write file system operation of local node, on this basis, and improve the time performance of HBase write operation. The experiments prove that the other node log methods can effectively improve the time performance of the write operation when the MemStore settings is not more than the default settings, by using other node log methods can greatly reduce the sensitivity of the HBase write operation time to the MemStore size, it makes the execution time of the system write operation is no longer closely related to the size of MemStore.

Acknowledgment

This project was supported by the Funding Project for Natural Science Foundation of China (Grant No. 61671070), the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (Grant No.ICDD201608) and the Project of Computer Teaching Reform of Beijing Information Science and Technology University (Grant No. 5111623409).

REFERENCES

[1] Yu Zhang, Zhongyou Ma, and Xiaofeng Meng, "*Efficient Processing of Spatial Keyword Queries on HBase*," Journal of Chinese Computer Systems, vol. 33, pp. 2141–2146, October 2012.

[2] Lin Gao, Xiangqian Song, "*Research on Cloud Computing and Key Technologies*," Microcomputer & Its Applications, vol. 30, pp. 5–11, October 2011.

[3] Changcheng Tang, Feng Yang, Dong Dai, and Mingming Sun "*Research on Cloud Computing and Key Technologies*," computer application systems, vol. 22, pp. 176–180, October 2013.

[4] Jianyong Fan, Ming Long, Wei Xiong, and Mingming Sun "*Research on distributed storage of vector spatial data based on HBase*," Geography and Geo-Information Science, vol. 28, pp. 39–42, Sptember 2012.

[5] Dong Guo, Yong Du, Liang hu "*Research on Cloud Computing and Key Technologies*," Journal of Jilin University (Science Edition), vol. 50, pp. 101–106, January 2012.

[6] Weikang Chen, Song Du, "*On Date Placement Strategies In Distribute Storage System*," Computer Applications and Software, vol. 26, pp. 6–9, January 2009.

[7] Zhenghong He, Ya Zhou, "*Research on Large Scale Data Loading Based on HBase*," Computer Systems& Applications, vol. 25, pp. 231–237, January 2016.

[8] Yan Zhang, Song Guo, "*Research on cloud computing platform experiment based on Hadoop*," Journal of Shenyang Normal University, vol. 31, pp. 85–89, January 2013.