HDTFS: Cost-effective Hadoop Distributed & Tiered File System for High Energy Physics

Xiaoyu Liu[1,2], Libin Xia[1,2], Xiaowei Jiang[1,2] and Gongxing Sun[1]

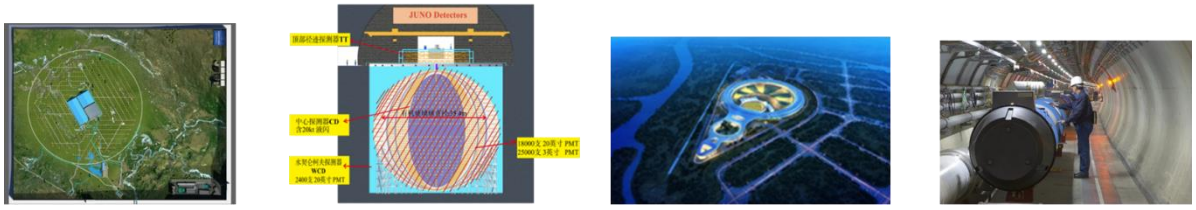
1. Institute of High Energy Physics, Chinese Academy of Sciences, Beijing 100049, China
2. University of Chinese Academy of Sciences, Beijing 100049, China

Abstract. With the scale and complexity of High Energy Physics (HEP) experiments increase, researchers are facing the challenge of large-scale data processing. In terms of storage, Hadoop Distributed File System (HDFS), a distributed file system that supports the data-centric processing model, has been widely used in academia and industry. HDFS can support Spark and other distributed data localization calculation. The study of HDFS in the field of HEP is the basic for securing the application of upper tier computing in this field. However, HDFS expands the cluster capacity by adding more cluster nodes, which cannot meet the high cost-effective system requirements for the persistence and backup process of massive HEP experimental data. In response to the above problems, researching Hadoop Distributed & Tiered File System (HDTFS) supporting both disk and tape storage to achieve hierarchical storage of hot and cold data and solve the defects of high cost of horizontal expansion of HDFS clusters. HDTFS make full use of the characteristics of fast disk access speed and the advantages of large capacity, low price, and long-term storage of tape storage. The system provides users with a single global namespace, and avoids dependence on external metadata servers to access the data stored on tape. In addition, tape tier resources are managed internally so that users do not have to deal with complex tape storage. The experimental results show that this method can effectively solve the massive data storage of HEP Hadoop cluster.

1. Introduction

In order to process massive data, the processing mode has changed from computing-centric to data-centric in the field of Internet. After continuous development, several big-data processing frameworks such as Hadoop and Spark have been formed. Compared with the traditional computing-centric mode, this processing mode does not require network transmission of data, which can greatly relieve network pressure and improve processing efficiency. In order to introduce this processing mode into the field of HEP, researchers have made a series of attempts to building a HEP Hadoop computing cluster[1]. However, with the continuous expansion of the scale of HEP experiments and the continuous improvement of the complexity of experiments, the annual data volume and cumulative data generated by major physics experiments are increasing year by year. For example, the total raw and reconstructed data of the upgraded Beijing Electron Positron Collider (BEPCII) has reached 7.1PB (by the end of 2020). The total raw and reconstructed data of the Daya Bay Neutrino Experiment has reached 3.2PB (by the end of the experiment data collection in December 2020). The raw data collected by the Jiangmen Neutrino Experiment is about 2PB every year (planned to be completed and start taking data in 2022). The High Altitude Cosmic Ray Observatory LHAASO accumulates at least 10PB of data per year. And the upgraded HL-LHC experiment at the LHC will produce 600 PB of raw data per year (upgrade to be completed in 2026)[2]. Figure 1 shows the various HEP experiments. The native file system of the Hadoop system is HDFS. It requires a large storage cost investment if the storage capacity is expanded simply by adding cluster nodes. Through analysis, most of the massive experimental data in HEP are

cold data that are infrequently accessed or require persistent backup, which can be stored and backed up by a lower-priced and higher-capacity tape library. Therefore, this paper reports on building the Hadoop Distributed & Tiered File System (HDTFS) to solve the massive data storage problem faced by HEP Hadoop cluster, improve the cluster storage cost performance, and perfect the HEP Hadoop ecosystem research.



LHAASO JUNO HEPS LHC

Figure 1. Various high energy physics experiments.

2. Related Work

Research on HDFS tiered storage is currently focused on developing storage policies for tiered data hotness, using heterogeneous resources to improve data access efficiency, and using resources such as tapes to store cold data to save storage, etc. HDFS-2832[3] proposed Heterogeneous Storage in HDFS, and with continuous development, the study now supports four storage types: memory, SSD, HDD and ARCHIVE[4]. Users can configure different storage policies for files to achieve heterogeneous storage of files with different hotness, but the actual storage resources of ARCHIVE storage in this study are disks without really using tape resources. Subramanyam propose a storage resource management technique[5] to automatically and dynamically move data across tiered storage based on data hotness, thus effectively using scarce and expensive storage space. Yuxin et al. proposed to use SSM (Smart Storage Management) system[6] for data optimization to improve the efficiency of storage systems. Ciritoglu et al. proposed a heterogeneous aware replica deletion scheme (HaRD)[7] to maintain data distribution balance on heterogeneous clusters. Krish et al. proposed hatS system[8] for hierarchical storage of files in an HDFS cluster by using different DataNode processes to manage SSD and HDD resources separately on a single machine node. Elena et al. proposed a multi-tier distributed file system OctopusFS[9], which can utilize memory, SSD, HDD, and remote cloud storage. Nusrat et al. proposed Triple-H[10], a new design idea of HPC cluster HDFS based on heterogeneous storage architecture. In summary, there is no experimental case of using tape library to build HDFS tiered storage system. In this study, tape library resources are used to build HDFS tiered storage system to complete the storage of cold data and improve the cost effectiveness of archival storage of massive experimental data.

3. Design and Implementation

In response to the massive data storage challenges faced by HEP Hadoop cluster, we propose to build a cost-effective HDTFS to achieve archival storage of cold data. The system provides a unified access interface for users, so that the original users can use the tiered storage system without code modification. HDTFS is a distributed, multi-tier file storage system that utilizes disk and tape. It is designed to provide a scalable, high-performing, and cost-effective storage system for the HEP Hadoop platform.

3.1. Overall architecture design of system

The system consists of master and worker nodes. The master node runs NameNode to manage the metadata of the system, the DataNode on the worker node to manage the disk resources, and the Tape server manages the tape resources. A message server is introduced to manage the request messages to the tape tier. The file operation request initiated by the client is sent to the message queue after the NameNode queries the file metadata if it found to involve the tape tier files, and the Tape server responds to complete the read/write operations. For the management of file metadata, NameNode is used to manage the metadata of the tiered storage system. Through the unified metadata management

method proposed in this paper, the unified management of files on tiered resources is realized. The native HDFS includes Namenode and Datanode components. HDTFS builds on this by modifying the in-memory directory tree, which the original namenode is responsible for managing, so that it can also manage file tape copies, and by adding tapeservers responsible for managing the tape files, and a message queue responsible for managing requests for files in the tape layer. The overall architecture design of the system is shown in Figure 2.

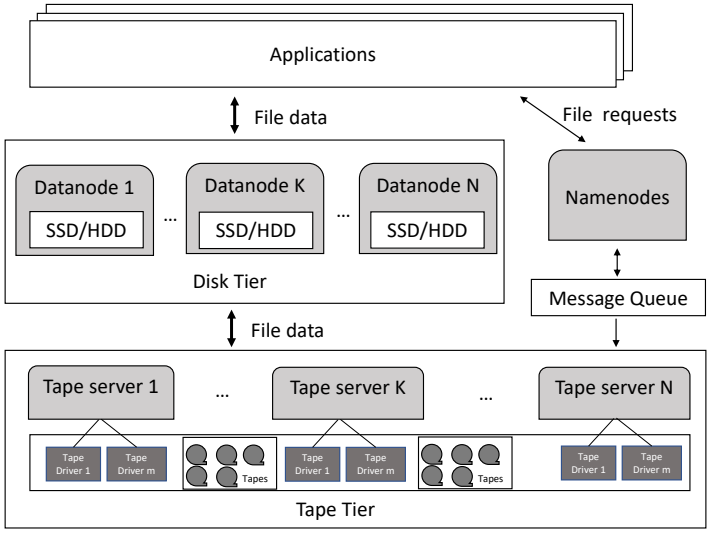


Figure 2. Architecture of HDTFS.

3.2. Communication management

Communication management for building tiered storage mainly includes two aspects: 1) Requests for tape tier files are managed by message queues, which facilitates the decoupling of functions between systems and improves system stability. Through technical selection, RabbitMQ was selected for development, which has a more flexible routing policy and supports priority queues compared with other message queues. The requests involving tape files will be sent by Namenode to the message queue for storage management. The tape library manager consumes the messages in the message queue, takes corresponding actions according to the request type, and complete file archiving or retrieving according to the request parameters. 2) Communication management for metadata is implemented using Hadoop RPC. Namenode is the server side of RPC requests for metadata management. It realizes the management of tiered file metadata and efficiently handles RPC requests from multiple clients. The implementation of this part first defines the relevant RPC protocols, then implements the above defined RPC protocols, starts the RPC Server, constructs the RPC client and completes the sending of RPC requests in the business logic implementation.

3.3. Write/Read process

The archiving/retrieving process occurs during the writing/reading of files stored on the tape tier in the system. The process of read (only disk tier), write (only disk tier), archive, retrieve of file stored in tiered storage system are shown as Figure 3.

3.3.1. Write process. The process of application writing to the tiered storage system is the same as that of writing to the original HDFS disk tier. First, the application initiates a file creation request to the NameNode, and then writes data to the DataNode after creating a new file. After adding the tape tier, the application can call the archiving interface and initiate a file archiving request to the NameNode (the system also supports automatic archiving of cold data). After

NameNode confirms that the file has not been archived by judging the file metadata, it will send the archiving request to the message queue for queuing, and the underlying Tape Management System (TMS), which consists of a Tape server, will respond to the request by requesting file data from DataNode, and return the file metadata information to NameNode after archiving is completed.

3.3.2. Read process. The application reads a file by first requesting the file's metadata information from the NameNode. If the file is on the disk tier, the NameNode will return the metadata information directly to the application, which then requests the file data from the corresponding DataNode. If the file is on the tape tier, the NameNode sends the retrieve request to the message queue, and the Tape server responds to the retrieve request and extracts the file to the DataNode before the system executes the client's file read request.

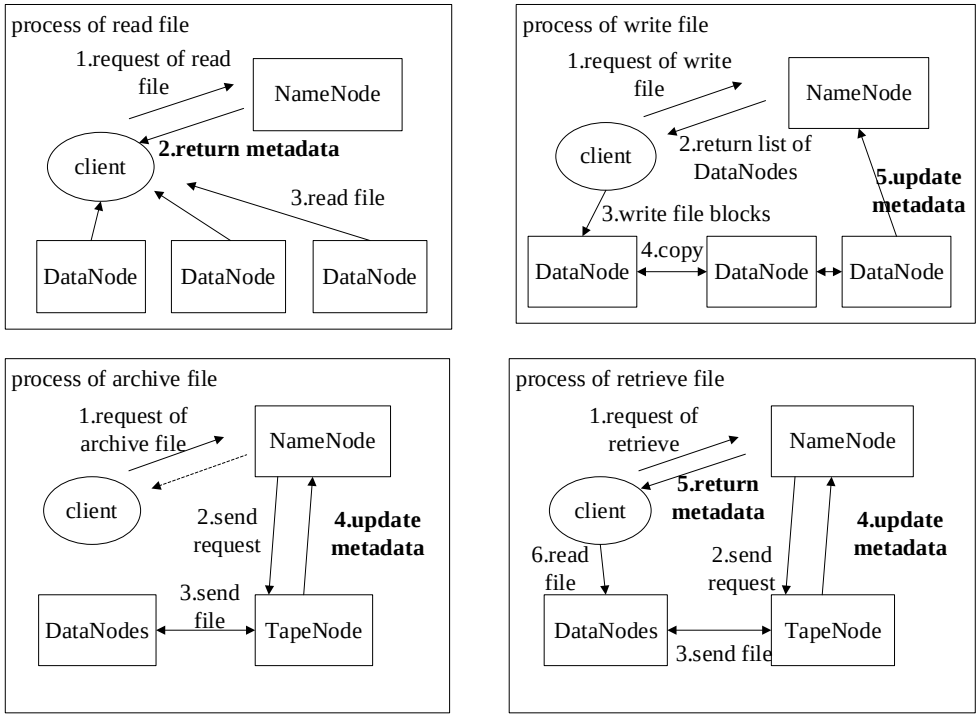


Figure 3. Process of different operation for file stored in tiered storage system.

4. Tests and Results

4.1. Test environment

The test environment is a cluster of 7 nodes, including 1 master node and 6 worker nodes. The NameNode is deployed on the master node, and the DataNode and Tape Server are deployed on the worker nodes. The software and hardware of each node are shown in Table 1.

Table 1. Software and hardware environments of the testing cluster.

Environment	Master Node	Worker Node
CPU	Intel Xeon E5-2680 v3 @ 2.50GHz	Intel Xeon E5-2630L v2 @ 2.40GHz
Memory	128GB	64GB
Disk	2*2TB SAS	5*1TB SAS
Network	1Gbps Ethernet	
OS	CentOS Linux release 7.9.2009(Core)	

Mhvtl Version	Mhvtl1.5.0
Java Version	openjdk version "1.8.0_282"
Hadoop Version	Hadoop3.2.2

4.2. Test result

In order to test the tiered storage function of HDTFS, create a test directory /user/xy/test and write files. According to the above writing process, the files are all stored on the disk tier at this time. You can use the HDFS command to view the disk space occupied by the directory as shown in the Figure 5(a). After archiving the test directory, the disk space occupied is shown in Figure 5(b). At this time, the files are stored in the tape library, and the disk space is released.

```
[hadoop@helion01 ~]$ hdfs dfs -du -h /user/xy/test
5.0 G  14.9 G  /user/xy/test/data031322
3.0 G  8.9 G   /user/xy/test/data031422
980 M  2.9 G   /user/xy/test/data031522
4.0 G  12.0 G  /user/xy/test/data031622
2.0 G  6.0 G   /user/xy/test/data031722
[hadoop@helion01 ~]$ hdfs dfs -du -h -s /user/xy/test
14.9 G  44.6 G  /user/xy/test
```

(a). Disk space occupied before archiving.

```
[hadoop@helion01 ~]$ hdfs dfs -du -h /user/xy/test
0  0  /user/xy/test/data031322
0  0  /user/xy/test/data031422
0  0  /user/xy/test/data031522
0  0  /user/xy/test/data031622
0  0  /user/xy/test/data031722
[hadoop@helion01 ~]$ hdfs dfs -du -h -s /user/xy/test
0  0  /user/xy/test
```

(b). Disk space occupied after archiving.

Figure 5. Disk space occupied before/after archiving.

5. Conclusions

The HDTFS realizes cost-effective storage of massive data. Using tape libraries to store cold data can save storage costs without affecting the access efficiency. It implements the cold data archiving of the HEP Hadoop cluster and perfects the HEP Hadoop ecosystem.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Contracts No. 12275295 and No. 11775249.

References

[1] Qiulan Huang, Zhanchen Wei, Gongxing Sun, et al. Using Hadoop for High Energy Physics Data Analysis[C]// International Conference on Big Scientific Data Management(BigSDM 2018): Beijing, China. Springer Cham, 2018.

[2] Gang Chen. Data and computing for high energy physics experiments[J]. Scientia Sinica:Physica,Mechanica & Astronomica, 2021, 51(9): 14-23.

[3] Apache Software Foundation. Enable support for hetero-geneous storages in HDFS [EB/OL]. [2021-12-26]. https://issues.apache.org/jira/browse/HDFS-2832.

[4] Apache Software Foundation. Archival Storage, SSD & Memory[EB/OL]. [2021-12-26]. https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/ArchivalStorage.html
[5] Subramanyam R. HDFS heterogeneous storage resource management based on data temperature[C]//2015 International Conference on Cloud and Autonomic Computing. IEEE, 2015: 232-235.
[6] Yuxin Guan, Zhiqiang Ma, Leixiao Li. HDFS Optimaization Strategy Based On Tiered Storage of Hot and Cold Data[J]. Procedia CIRP, 2019, Volume 83: 415-418.
[7] Ciritoglu H E, Murphy J, Thorpe C. Hard: a heterogeneity-aware replica deletion for hdfs[J]. Journal of big data, 2019, 6: 1-21.
[8] K. R. Krish, A. Anwar and A. R. Butt. hatS: A Heteroge-neity-Aware Tiered Storage for Hadoop[C]// 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing: Chicago, IL, USA. IEEE, 2014.
[9] Elena Kakoulli and Herodotos Herodotou. OctopusFS: A Distributed File System with Tiered Storage Manage-ment[C]// SIGMOD '17: Proceedings of the 2017 ACM In-ternational Conference on Management of Data: Illinois, Chicago, USA. ACM, 2017.
[10] Nusrat Sharmin Islam, Xiaoyi Lu, Md. Wasi-ur-Rahman, et al. Triple-H: A Hybrid Approach to Accelerate HDFS on HPC Clusters with Heterogeneous Storage Architecture [C]// 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing(CCGRID), Shenzhen, China. IEEE, 2015.