Automated and Intelligent Data Migration Strategy in High Energy Physical Storage Systems

Zhenjing Cheng[1,2], Lu Wang[1,2], Yaodong Cheng[1,2,3], Gang CHEN[1]

chengzj@ihep.ac.cn, wanglu@ihep.ac.cn, chyd@ihep.ac.cn, gang.chen@ihep.ac.cn
[1]IHEP computing center, 19B Yuquan Road, Beijing 100049, China
[2]University of Chinese Academy of Sciences, 19A Yuquan Road, Beijing 100049, China
[3]Tianfu Cosmic Ray Research Center, IHEP, 9 Renmin South Road, Chengdu 620500, Sichuan, China

chengzj@ihep.ac.cn

Abstract. As a data-intensive computing application, high-energy physics requires storage and computing for large amounts of data at the PB level. IHEP computing center is beginning to use tiered storage architectures, such as tape, disk or solid state drives to reduce hardware purchase costs and power consumption. At present, automatic data migration strategies are mainly used to resolve data migration between memory and disk. So the rules are relatively simple. This paper attempted to use the deep learning algorithm model to predict the evolution trend of data access heat as the basis for data migration. The implementation of some initial parts of the system were discussed, as well as the file trace collector and the LSTM model. At last some preliminary experiments are conducted with these parts.

1. Introduction

With data-intensive computing [1] requirements, high-energy physics computing needs to store and process massive physical data at the PB level every year. The data access performance of the storage system is the key factor. More and more high-energy physics experiment data puts higher and higher requirements on storage systems, including capacity, performance, scalability, reliability, long-term preservation and cost performance. On one hand, storage systems based on traditional mechanical hard disks or tapes cannot provide higher IOPS(read/write bandwidth cannot be further improved) and the unit price of new solid-state disks is still high, so it has no advantage in terms of storage capacity and cost performance. On the other hand, according to one survey[2], only a small part of the data in the storage system is active during a short period of time, while most of the data will not be accessed. The Chinese Academy of Sciences High Energy Computing Center plans to use hierarchical storage system with a unified namespace to store HEP data, including tape, traditional mechanical hard drives and solid-state drives.

To improve the resource utilization of the storage system, a migration strategy is needed to migrate data between different storage tiers. Heuristic algorithms based on personal experience of system administrators are currently used, which may have empirical biases and lack of load versatility and adaptability. Two major massive storage systems, such as Lustre [3] and EOS [4], provide hierarchical storage techniques. But data migration between all levels of storage is completely specified by the administrator through the file migration list, which has a lot of labor costs.

In high-energy physics storage systems, data access requests are usually not completely random. They are closely related to the user/application computing model [5]. This paper first introduces the research status and related work basis of the data migration strategy, then it introduces the collection of file access training samples from the actual high-energy physical storage system. Finally, it presents how we trained a predictive model with supervised learning to predict future access of files based on the deep learning algorithm, and further developed migration rules based on changes in predicted access heat.

2. Research status

Data migration strategies have always been an important area of research for storage systems. Literature [6] reviews describe algorithm-driven file migration strategies such as LRU, CLOCK, 2Q, GDSF, LFUDA and FIFO. These strategies are mainly used to solve the problem of data migration between volatile main storage and external disks in a stand-alone computer. The essence is to set a threshold on file access feature (such as the last access time). To run in the operating system kernel, a trade-off must be made between prediction accuracy and calculation efficiency, making it impossible to exploit complex file access features or build complex models. Further literature [7] introduced the ULNS model which would consider user information for predictions. The prediction accuracy is better than the normal LS model, but it also relies on a large number of historical file access sequences. Other literature [8] proposed the idea of formulating migration rules based on prediction of file access frequency. The method assumed that each time a user would read the files in a complete and sequential manner. A supervised learning model based on the support vector machine (SVM) algorithm was trained to perform the prediction task. It obtained good prediction results on the specific WEB data set. The SVM algorithm is difficult to implement for large-scale training samples because it supports vectors by means of quadratic programming, which would consume a lot of machine memory and computation time. In recent years, technology based on deep learning broke through the limitations of traditional neural networks on the number of layers and the number of nodes per layer. The training methods are quite different from the traditional machine learning algorithms. This paper uses a deep learning algorithm based on long short-term memory neural networks to solve this problem.

3. Related work

3.1 EOS Massive Distributed Storage System

Distributed storage systems are an integral part of a high-energy physics computing environment. In 2011, in order to meet the data storage requirements of the LHC Run II experiment, the EOS mass distributed storage system was put into use. Currently CERN EOS manages 15 instances, with more than 2.6 billion files and nearly 250 PB of data. The IHEP computing center also uses EOS to manage more than 10PB of data, and serves more than 1000 users in LHAASO [9]. EOS is composed of management server (MGM), message queue (MQ) and file storage server (FST) based on the Xrootd [10] protocol framework. The file storage server supports a variety of storage media such as mechanical hard disks and solid-state drives. At the same time, EOS CTA services support data reading and writing in the tape library. EOS divides the file storage server into multiple storage groups. RAID mode is used for file backup and striping. EOS has a unified namespace. Data and files in different storage groups do not affect each other. Files can be migrated between groups through EOS admin commands. What's more, EOS has designed a comprehensive hierarchical logging mechanism.

3.2 LSTM(long-term and short-term memory neural network)

Long-Short Term Memory Network(LSTM) is an improved Recurrent Neural Network (RNN) [11]. The neural network was originally a machine learning method based on the human brain. However, in general, humans do not start thinking from a blank brain all the time. For example, when reading an article, human beings infer the meaning of the current text based on the knowledge already possessed

and their own understanding. John Hopfield et al. built a cyclic neural network with combined storage capabilities. Internal nodes were connected to each other, so that the neural network has the ability to remember as humans.

Due to the problem of gradient disappearance in traditional cyclic neural networks, it is difficult to deal with long-term dependent information in learning sequences. FA Gerset al[12]. proposed the network structure of LSTM, which redesigned the computational nodes in the cyclic neural network It uses time memory cells to record the state of the current moment. Those cells are commonly referred to as long-short-term memory neural networks. Each cell has three information transfer gates: the input gate, the output gate, and the forgetting gate as shown in fig1.

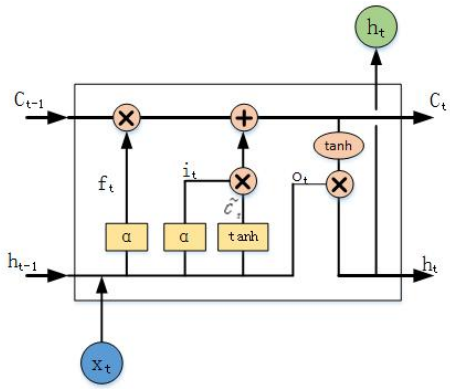

Figure 1. LSTM cell Network.

The information switch door can selectively pass information. The forgetting gate determines in the sequence data at a certain moment, what information is discarded from the cell, and returns the output by a value between 0 and 1 for each cell state C_{t-1} (0 means completely discarded, 1 means completely reserved)). h_{t-1} represents the output of the previous LSTM Cell, x_t represents the input of the current LSTM Cell, and α represents the sigmod function.

$$f_t = \alpha(W_f \cdot [h_{t-1}, x_t] + b_f)$$

The input gate determines what new information should be stored in the LSTM cell state. The input gate contains two processing levels, the sigmod layer determines what values should be updated in the cell state, and the tanh layer creates a new candidate value vector $\tilde{c}_t$ to be added to the cell state.

$$i_t = \alpha(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{c}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

Then C_{t-1} is updated to C_t, and the updated LSTM cell state is the sum of the new candidate value vectors $\tilde{C}_t$ after discarding the old cell information.

$$C_t = f_t * C_{t-1} + i_t * \tilde{c}_t$$

The output gate determines which portion of the cell state to output via a sigmod layer based on the updated LSTM cell state. And a tanh function was used to ensure that only the portion of the LSTM cell that determines the output.

$$O_t = \alpha(W_o[h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * tanh(c_t)$$

The three gates jointly control the inflow and outflow of information, as well as the update of LSTM cell status. Therefore, the LSTM model excels at mining long-term or short-term dependency information in a time series, and is suitable for predicting delayed events with time intervals.

4. Design and Implementation

As shown in the figure 2, the system interacts with high-energy physics storage systems such as Lustre and EOS through an intelligent migration management system, which consists of feature collection

nodes, central databases, model training nodes, and policy configuration nodes. The system deploys an I/O log collector on each file storage server of EOS After filtering out irrelevant information, I/O logs are stored in the central key-value database in the format of <timestamp, parameter field, value>. Then the system integrates, normalizes, and batches the feature data according to the needs of different models, and writes to the online data queue for model training. The model training node is a multi-GPU computing cluster based on Tensorflow, caffe and other deep learning frameworks. The trained models and test results are stored in a distributed file system for persistent store and visible to all nodes. The policy configuration node scans the file list in the tape, mechanical hard disk, and solid state hard disk in the background, and performs the migration actions according to the model prediction results and the pre-set migration condition.

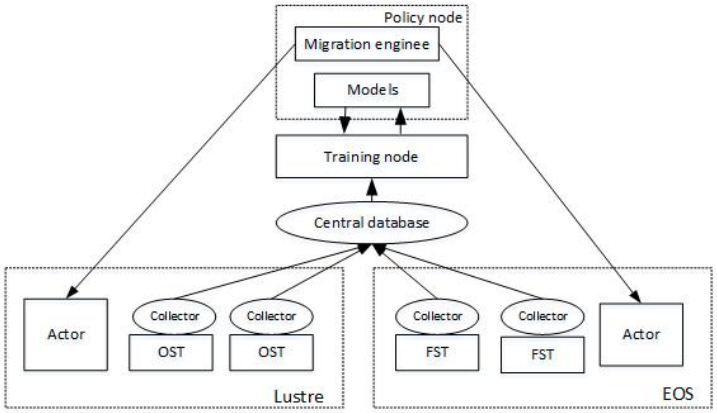

Figure 2 System design

Training a predictive model requires input of a large number of training samples. In general, in high-energy physics storage systems, file access is related to the computing mode of different users and different applications. In the Linux operating system, file operations are based on file descriptors and file pointers, including file creation, opening and closing, file read/write point movement, file reading and writing, etc. [13]. EOS's logging system provides historical access records in file name units: <timestamp, access category, file read and write bytes, file read/write point movements, file size>. These records can be organized into a multi-dimensional vector according to the type of operation, which is defined as the access feature vector of the file.

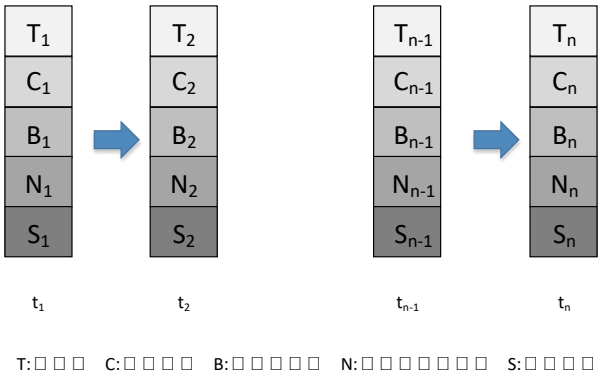

Figure 3 file access feature vector

5. Experiments

5.1 Experiment data source

We collected access I/O logs for files in high-energy physics EOS storage systems from 2018-04-01 to 2018-05-01, and selected the following files to access I/O features.

1. numHourlyAccesses: The number of accesses to the file per hour. A complete file access refers to the process of including file open, read and write data, and file close.

2. avgHourlyReads: The average number of read times during a full access during the hour.
3. stdDevHourlyReads: The mean square error of read number d during a complete access during the hour.
4. avgHourlyBytesRead: The average value of read bytes during a full access during the hour.
5. stdDevHourlyBytesRead: The average variance of read bytes during a complete access during the hour.
6. avgHourlyWrites: The average number of written during a full access during the hour.
7. stdDevHourlyWrites: The mean square error of written number during a complete access during the hour.
8. avgHourlyBytesWritten: The average of written bytes during the full access of the file per hour.
9. stdDevHourlyWritten: The mean square error of written number during a complete access during the hour.
10. avgHourlySeeks: The average value of seeks number for the read and write positions during the complete access of the file during the hour.
11. stdDevHourlySeeks: The mean square error seeks number in the read and write position of the file during a complete access during the hour.

5.2 Experiment results

In order to prevent the model from over-fitting, the training data is randomly divided into a training set (80%), a verification set (10%) and a test set (10%). A LSTM model is used to predict file access heat. Precision, recall and F-Score rate were used to evaluate the trained model. In large-scale data sets, precision and recall are generally mutually constrained. Precision is a description of random errors. It's the ratio of the correctly labeled files to the whole pool of files. Recall is the ratio of the correctly labeled hot files by our model to all who are hot in reality.

F-Score [14] comprehensively considers both precision and recall rate, and reconciles it at a certain ratio. F-Score is best if there is some sort of balance between precision and recall in the prediction system. The calculation formula is as follows:

$$\mathrm{F-Score} = (1+\beta^2) * \frac{Precision * Recall}{(\beta^2 * Precision) + Recall}$$

Beta balances the importance of recall and precision. Beta was 1 in the calculation of F-Score. It means precision and recall were equally important to the test results. Since the ratio of the number of hot files and cold files in the training set is quite different, F-Score is used as the main indicator for model evaluation.

	LRU	**SVM**	**MLP**	**LSTM**
PRECISON	0.561	0.734	0.758	0.780
RECALL	0.514	0.694	0.665	0.741
F-SCORE	0.536	0.713	0.708	0.760

6. Conclusion and Outlook

In HEP storage, performance demands and data access imbalances in mass storage systems are increasing. As the amount of stored data grows, tiered storage requires data management software to migrate less active data to lower cost storage devices. Thus an automated data migration strategy is needed. This paper proposed a method of file access heat prediction. Data heat trend is used as the basis for migration to a relatively low-cost storage device. Due to the limitations of traditional models, it is difficult to achieve good results in predicting at such nonlinear scenes. This paper attempted to use the LSTM model to predict the evolution trend of data access heat. But until now it didn't consider impact brought by data migration to the storage performance. Next, the concept of migration cost will be introduced, which would consider impact on storage performance.

Acknowledgments

This work was supported by the Youth Innovation Promotion Association project of CAS No. 2015012.

References

[1] Reed D A, Dongarra J. Exascale computing and big data[J]. Communications of the ACM, 2015, 58(7): 56-68.
[2] Ranganathan K, Foster I. Identifying dynamic replication strategies for a high-performance data grid[C]//International Workshop on Grid Computing. Springer, Berlin, Heidelberg, 2001: 75-86.
[3] Schwan P. Lustre: Building a file system for 1000-node clusters[C]//Proceedings of the 2003 Linux symposium. 2003, 2003: 380-386.
[4] Peters A J, Sindrilaru E A, Adde G. EOS as the present and future solution for data storage at CERN[C]//Journal of Physics: Conference Series. IOP Publishing, 2015, 664(4): 042042.
[5] Allcock B, Bester J, Bresnahan J, et al. Data management and transfer in high-performance computational grid environments[J]. Parallel Computing, 2002, 28(5): 749-771.
[6] T. J. Gibson. Long-term UNIX File System Activity and the Efficacy of Automatic File Migration. PhDthesis, University of Maryland, Baltimore County, May 1998.
[7] Yeh T, Long D D E, Brandt S A. Performing file prediction with a program-based successor model[C]//MASCOTS 2001, Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems. IEEE, 2001: 193-202.
[8] Eads, Damian and Karen A. Glocer. “Viewing Adaptive Migration Policies for Tiered Storage Systems as a Supervised Learning Problem.” (2006)
[9] Cao Z. A future project at tibet: the large high altitude air shower observatory (LHAASO)[J]. Chinese Physics C, 2010, 34: 249-252.
[10] Dorigo A, Elmer P, Furano F, et al. XROOTD-A Highly scalable architecture for data access[J]. WSEAS Transactions on Computers, 2005, 1(4.3): 348-353.
[11] Gers F A, Schmidhuber J, Cummins F. Learning to forget: Continual prediction with LSTM[J]. 1999.
[12] Gers F A, Schraudolph N N, Schmidhuber J. Learning precise timing with LSTM recurrent networks[J]. Journal of machine learning research, 2002, 3(Aug): 115-143.
[13] LaRosa C, Xiong L, Mandelberg K. Frequent pattern mining for kernel trace data[C]//Proceedings of the 2008 ACM symposium on Applied computing. ACM, 2008: 880-885.
[14] Sokolova M, Japkowicz N, Szpakowicz S. Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation[C]//Australasian joint conference on artificial intelligence. Springer, Berlin, Heidelberg, 2006: 1015-1021.