A Hierarchical Storage Mechanism for Hot and Cold Data Based on Temperature Model

Shicong Ma, Tianhai Zhao, Jianhua Gu, and Yunlan Wang(✉)

Northwestern Polytechnical University, Xi'an, Shaanxi, China
masc@mail.nwpu.edu.cn, {zhaoth,gujh,wangyl}@nwpu.edu.cn

Abstract. The storage of hot and cold data play a crucial role in improving data access efficiency and reducing storage expenses. This paper proposed a temperature model to quantify the real-time hotness of data. Based on the temperature model, we proposed a hierarchical storage mechanism for hot and cold data, managing dynamic data migration among local cold database, local hot database, and remote cold database. Experimental results show the advantages of the proposed method in terms of hot data hit rate, hot data hit rate for key data, migration count, and average response time. It can improve data access performance and the satisfaction of important users, and significantly reduce expenses.

Keywords: hot and cold data · hierarchical storage · data temperature model · dynamic migration · remote storage

1 Introduction

In recent years, the data volume expanded rapidly and the demand for data management and processing capabilities has steadily increased, posing unprecedented challenges in the storage and management of data. The Pareto Principle shows the majority of accesses concentrate on a small portion of data over a specific period [1], enabling us to allocate data with varying hotness to distinct storage media for hierarchical storage. Commonly used storage devices include Hard Disk Drive (HDD) and Solid State Drive (SSD). HDD provide economical and substantial storage capacity solutions. Conversely, SSD offer faster read and write speed, but come with higher price. As urban land development approaches saturation in many developed cities, essential resources like water and electricity face strain. The concepts of channel more computing and storage resources from developed regions to less developed but resource rich regions have gained considerable attention and development in country like China [2]. An effective hierarchical storage strategy can optimize data access performance while conserving storage expenses, but it also imposes higher requirements for accurately identifying the degree of data hotness.

C. Strauss et al. (Eds.): DEXA 2024, LNCS 14910, pp. 153–159, 2024.
https://doi.org/10.1007/978-3-031-68309-1_13

This article proposes a **h**ierarchical **s**torage mechanism for hot and cold data based on **t**emperature **m**odel (HSTM). Firstly, we construct the data temperature model. Then, based on the temperature model, we propose hierarchical storage mechanism. The main contributions of HSTM are:

- **Data temperature model.** Propose a model for computing temperature based on access time, access frequency, user priority, and data priority. This model provides a comprehensive assessment and quantification of the real-time hotness of data.
- **LSTM network based prediction.** Employ Long Short Term Memory (LSTM) networks to forecast the volume of new data, striking a balance between the utilization of hot database space and data migration count.
- **Periodic migration pattern analysis.** Mine the periodic migration pattern of data. Pre-migration of data with identifiable periodic migration patterns can mitigate time latency of migration, further enhancing the hot data hit rate.
- **Cost-based decision for local and remote storage.** When the local cold database storage space is insufficient, migrate the data with higher benefits of storing remotely to the remote cold database, substantially reducing storage expenses.

2 Related Work

The predominant methods for identifying hot and cold data encompass two classical cache replacement algorithms: the Least Recently Used algorithm (LRU) [3] and the Least Frequently Used algorithm (LFU) [4]. Various optimizations and variations, such as LRU-K [5] and LFU-Aging [6], have been proposed to enhance the adaptability to different scenarios.

In subsequent investigations, scholars have adopted intricate data structures to discern between hot and cold data. HashKV proposes a distinction strategy between hot keys and cold keys [7]. HashKV stores the hot keys in the segment of vLog, and separates the cold key, stores in the disk then. HotRing proposed an innovative Key-Value Store (KVS) with hotspot awareness, specifically designed for massively concurrent access to a small portion of items [8]. Facebook identified hot and cold data based on the status of a bucket of objects instead of the status of a single object [9].

Some algorithms have achieved commendable performance, extending beyond the realm of data structures. Xie compared the data's access frequency with a pre-defined threshold [10], then determine whether the data is hot or cold.

Recent years have introduced approaches that assess data hotness through the evaluation of data temperature values. Song have devised a temperature calculation model that exhibits sensitivity to temporal changes, drawing upon Newton's Law of Cooling [11].

Most current methods designed for the identification and management of hot and cold data rely on basic access characteristics, such as access frequency and

time. However, these methods often neglect the opportunity to enhance important users' satisfaction and do not optimize methods from the perspective of mining data access and migration patterns. Meanwhile, there is little consideration given to the performance and storage cost of local and remote storage.

3 System Architecture

HSTM comprises two components: the data temperature model and the hierarchical storage mechanism for hot and cold data.

The initial section quantifies the real-time hotness of data. In this paper, data stored in the hot database is categorized as hot data, while data stored in the cold database is categorized as cold data. The subsequent section, built upon the temperature model, manages the data dynamic migration.

3.1 Data Temperature Model

The new data are likely to access in the immediate future and thus warrants storage in the hot database. The initial temperature value $T_{newData}$ is set as the average temperature of hot data.

Using I and Q to represent the user and data priority, where $I = 1$ indicates an important user, $I = 0$ indicates a regular user, $Q = 1$ indicates key data, and $Q = 0$ indicates regular data. Let T_{pre} and T_{new} denote the temperature before and after data access, T_{new} can be formulated as follows:

$$T_{new} = T_{pre} + \alpha + \beta \times I + \gamma \times Q \tag{1}$$

where α, β, and γ denote weights of access, user priority, and data priority.

In order to mitigate the persistent impact of short-term frequent access on data temperature, it is necessary to periodically reduce the temperature of data. Let T_{pre} and T_{new} be the data temperature before and after reduction, and δ be the temperature weakening coefficient. T_{new} is the product of T_{pre} and δ.

3.2 Data Hierarchical Storage Mechanism

Local Hot and Cold Data Migration Management includes time-driven and event-driven migration. Time-driven migration occurs at the end of each period p, event-driven migration occurs in the process of each period p.

To mine the periodic migration pattern of data, when data d is migrated from cold to hot database for the y-th time (where y $\geq$ 4), if the three intervals resulting from the last four migrations are equal (the difference in period numbers between two consecutive migrations form interval), it is inferred that data d exhibits a periodic migration pattern.

In time-driven migration, we utilize LSTM to predict new data volume of upcoming period e based on new data volume of previous period, then compute the upper threshold of the current period ζ as the difference between the hot

database storage capacity c and 0.5 times e, indicating the maximum storage space utilized in the hot database post-migration. By means of migration, the cold data predicted to migrate to the hot database in the next period, along with a portion of the currently high-temperature data, is stored in the hot database.

Event-driven migration involves storing newly generated data and cold data with increased temperature in the hot database. If the storage space in the hot database is inadequate, hot data is evicted to the local cold database in ascending order of temperature until sufficient storage space is adequate.

Local and Remote Data Migration Management includes two types of event-driven migration. When the temperature of remote cold data increases, it initiates the migration of remote cold data back to the local database. When the utilized storage space of the local cold database exceeds 90% of the total storage capacity, it initiates the migration of local cold data to the remote database.

We denote the benefit of storing a unit of data remotely as udy, can be expressed as the difference between the local storage cost $cost_L$ and the remote storage cost $cost_R$, divided by the data size s, and $cost$ is the weighted sum of the monetary cost $expCost$ and the latency cost $dlyCost$. The $expCost$ is calculated as the product of the data size s, the power consumption per unit of stored data e, and the unit cost of electricity ep. The $dlyCost$ is the sum of the sending latency $s/bindwidth(bw)$ and the propagation latency pd, multiplied by the normalized data temperature T'.

$$udy = \frac{cost_L - cost_R}{s} = \frac{s \times e \times epdiff + \theta \times (\frac{s}{bw_L} - \frac{s}{bw_R} - pd_R) \times T'}{s} \quad (2)$$

where θ is the weight of the delay cost relative to the monetary cost, and $epdiff$ represents the difference in electricity cost per unit between local and remote storage, and pd_L in $dlyCost_L$ is negligible.

4 Experimentation and Analysis

This section conducts simulation experiments to compare the HSTM with the LRU-K and LFU-Aging algorithms.

Evaluation metrics encompassed the hot data hit rate for all data, hot data hit rate for key data, data migration count, and average access response time.

4.1 Experimental Environment and Configuration

The experiment utilized two computers to emulate local and remote storage, each equipped with an Intel(R) Xeon(R) Gold 6132 CPU @ 2.60 GHz, 376 GB of memory, and CentOS Linux release 7.4.1708 (Core). One computer was configured with a 1.5 TB SSD and a 20 TB HDD, while the other had a 35 TB HDD.

An initial set of one million data files, ranging in size from 1 MB to 20 MB, were stored in the local database. Each period comprised around 1 TB hotspot

data, with 80% being the previous period's hotspot data, 10% from the previous period's non-hotspot data, and 10% newly generated data in the current period. Each data access had an 80% probability of targeting hotspot data. The period p was set to one day, conducting a total of 60 million accesses over 30 days. Periodic temperature reductions occurred after the completion of each time-driven migration.

The settings for parameters are as follows: the SSD read speed is 500 MB/s, HDD read speed is 100 MB/s, bandwidth of long-distance link is 1G, pd_R is 20 ms, ε is 4, α is 0.5, β is 0.25, γ is 0.25, δ is 0.5 and m is 8.

4.2 Experiment on Local Hot and Cold Data Migration Management

The value of c was set as 0.8 TB, 1 TB, and 1.2 TB. The results are presented below (Table 1).

Table 1. Experimental results of HSTM, LRU-K, and LFU-Aging.

c	Metric	HSTM (=0.25/0.75/1.25)	LRU-10	LRU-15	LRU-20	LFU-Aging
0.8	hit rate (%)	59/59.3/59.3	55.5	56.7	57.2	53.6
	hit rate for key data (%)	72.7/75.8/76.5	65.8	66.6	66.7	68.8
	migration count (millions)	3.13/2.83/2.74	4.96	2.93	2.05	56.1
	average access response time (ms)	52.8/52.56/52.56	55.6	54.64	54.24	57.12
1	hit rate (%)	72.4/72.5/72.4	67.8	67.9	67.5	64.1
	hit rate for key data (%)	87.6/89.6/90	80.6	80.4	79.8	81
	migration count (millions)	1.87/1.64/1.56	3.54	2.09	1.48	43.56
	average access response time (ms)	42.08/42/42.08	45.76	45.68	46	48.72
1.2	hit rate (%)	77.6/77.5/77.2	75.3	74.2	72.2	71.7
	hit rate for key data (%)	93.1/93.3/93.4	89.9	88.4	85.7	88.6
	migration count (millions)	1.25/1.24/1.23	2.68	1.62	1.29	34.46
	average access response time (ms)	37.92/38/38.24	39.76	40.64	42.24	42.64

The experimental results show that HSTM exhibits an advantage in the hot data hit rate, the hot data hit rate for key data, and the average response time. When c surpasses the volume of hotspot data s, HSTM exhibits an advantage in migration count. Moreover, increasing the value of the data attribute weight parameter γ leads to a further improvement in the hot data hit rate for key data.

4.3 Experiment on Local and Remote Data Migration Management

The total storage capacities of the local cold database were set to 1 TB, 2 TB, and 3 TB, respectively. The parameter θ in (2) adjusted dynamically in each migration, ensuring that, subsequent to the data migration to the remote location, the utilized storage space in the local cold database accounted for approximately 60% of its total storage capacity.

The experimental results show that compared to storing all data locally, adopting local and remote hierarchical storage increases the average response time from 42.08 ms to 45.66 ms, 45.5 ms and 45.24 ms when the local cold data capacity is 1 TB, 2 TB and 3 TB, respectively. Assuming the volume of data stored remotely is vPB, the annual cost for the long-distance link is 200,000 yuan, and the power consumption rate for storing data e is 2.5 KW/PB. The price difference in electricity costs between local and remote locations $epdiff$ is 0.4 yuan per KWH. The formula for calculating the annual savings in storage costs $expSavings$ for local and remote hierarchical storage is expressed as follows:

$$expSavings = 2.5 \times 24 \times 365 \times 0.4 \times v - 200000 \tag{3}$$

5 Conclusion

This paper proposes a hierarchical storage mechanism for hot and cold data based on temperature model, implementing a three-tiered data storage and management encompassing local SSD, local HDD, and remote HDD, offering advantages in hot data hit rate and migration count, and reducing storage expenses. Future research should delve deeper into data access patterns in data centers to develop targeted data hierarchical storage strategies.

Acknowledgments. This work is supported by the National Key R&D Program of China (NO. 2022YFB4501701).

References

1. Sanders, R.: The Pareto principle: its use and abuse. J. Serv. Mark. **1**, 37–40 (1987)
2. Eastern Data, Western Computing: China's National Big Data Infrastructure Project. rootaccess.substack.com/p/eastern-data-western-computing-chinas. Accessed 16 Mar 2024
3. Dan, A., Towsley, D.: An approximate analysis of the LRU and FIFO buffer replacement schemes. In: Proceedings of the 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, pp. 143–152. ACM (1990)
4. Robinson, J.T., Devarakonda, M.V.: Data cache management using frequency-based replacement. In: Proceedings of the 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, pp. 134–142. ACM (1990)
5. O'Neil, E.J., O'Neil, P.E., Weikum, G.: The LRU-K page replacement algorithm for database disk buffering. ACM SIGMOD Rec. **22**(2), 297–306 (1993)
6. Arlitt, M., Friedrich, R., Jin, T.: Performance evaluation of web proxy cache replacement policies. In: Puigjaner, R., Savino, N.N., Serra, B. (eds.) TOOLS 1998. LNCS, vol. 1469, pp. 193–206. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-68061-6_16
7. Chan, H.H., et al.: HashKV: enabling efficient updates in KV storage via hashing. In: USENIX ATC 2018, pp. 1007–1019 (2018)
8. Chen, J., et al.: HotRing: a hotspot-aware in-memory key-value store. In: 18th FAST, Santa Clara, pp. 239–252 (2020)

9. Muralidhar, S., et al.: f4: Facebook's warm BLOB storage system. In: 11th OSDI, Broomfield, pp. 383–398 (2014)
10. Xie, Y., et al.: Efficient storage management for social network events based on clustering and hot/cold data classification. IEEE Trans. Comput. Soc. Syst. **10**, 120–130 (2023)
11. Song, Y., et al.: A novel hot-cold data identification mechanism based on multidimensional data. In: 2022 5th DSIT, Shanghai, pp. 1–5 (2022)