frontiers | Frontiers in Astronomy and Space Sciences

TYPE Original Research
PUBLISHED 14 March 2024
DOI 10.3389/fspas.2024.1371249

EDITED BY
Massimiliano Galeazzi,
University of Miami, United States

REVIEWED BY
Giuseppe Riccio,
National Institute of Astrophysics (INAF), Italy
Vladimir A. Sreckovic,
University of Belgrade, Serbia

*CORRESPONDENCE
Hai-long Zhang,
zhanghailong@xao.ac.cn

RECEIVED 16 January 2024
ACCEPTED 27 February 2024
PUBLISHED 14 March 2024

CITATION
Ye X-C, Zhang H-L, Wang J, Zhang Y-Z, Du X and Wu H (2024), Study on tiered storage algorithm based on heat correlation of astronomical data.
Front. Astron. Space Sci. 11:1371249.
doi: 10.3389/fspas.2024.1371249

Study on tiered storage algorithm based on heat correlation of astronomical data

Xin-Chen Ye[1,2,3], Hai-Long Zhang[1,2,3,4]*, Jie Wang[1,3], Ya-Zhou Zhang[1,2], Xu Du[1,2] and Han Wu[1,2]

[1]Xinjiang Astronomical Observatory, Chinese Academy of Sciences, Urumqi, China, [2]University of Chinese Academy of Sciences, Beijing, China, [3]National Astronomical Data Center, Beijing, China, [4]Key Laboratory of Radio Astronomy, Chinese Academy of Sciences, Nanjing, China

With the surge in astronomical data volume, modern astronomical research faces significant challenges in data storage, processing, and access. The I/O bottleneck issue in astronomical data processing is particularly prominent, limiting the efficiency of data processing. To address this issue, this paper proposes a tiered storage algorithm based on the access characteristics of astronomical data. The C4.5 decision tree algorithm is employed as the foundation to implement an astronomical data access correlation algorithm. Additionally, a data copy migration strategy is designed based on tiered storage technology to achieve efficient data access. Preprocessing tests were conducted on 418GB NSRT (Nanshan Radio Telescope) formaldehyde spectral line data, showcasing that tiered storage can potentially reduce data processing time by up to 38.15%. Similarly, utilizing 802.2 GB data from FAST (Five-hundred-meter Aperture Spherical radio Telescope) observations for pulsar search data processing tests, the tiered storage approach demonstrated a maximum reduction of 29.00% in data processing time. In concurrent testing of data processing workflows, the proposed astronomical data heat correlation algorithm in this paper achieved an average reduction of 17.78% in data processing time compared to centralized storage. Furthermore, in comparison to traditional heat algorithms, it reduced data processing time by 5.15%. The effectiveness of the proposed algorithm is positively correlated with the associativity between the algorithm and the processed data. The tiered storage algorithm based on the characteristics of astronomical data proposed in this paper is poised to provide algorithmic references for large-scale data processing in the field of astronomy in the future.

KEYWORDS

tiered storage, astronomical data processing, load prediction, decision tree, high performance computing

1 Introduction

With the continuous accumulation of data and technological advancements in the field of astronomy, the volume of data involved in modern astronomical research is growing at an unprecedented rate. Research by Cavuoti et al. (2015) indicates that the emergence of new telescopes and sensing devices has transformed astronomy into a data-rich scientific domain. Studies by El Bouchefry and de Souza (2020) suggest that astronomy is facing a data deluge propelled by significant technological advancements in telescopes, detectors, electronic devices, and computer technology. The complexity and dimensions of astronomical data are also rapidly increasing. The quantity and complexity of this data pose unprecedented challenges in terms of storage, processing, and access. La Plante et al. (2021) argue that the real-time analysis of these vast astronomical datasets presents unique computational challenges. Simultaneously, long-term storage and archiving are crucial for generating reliable and reproducible results in scientific research. The continuous growth of astronomical data renders standard laptops and desktop computers inadequate for handling such datasets (Antunes et al., 2022). The use of high-performance computing (HPC) platforms has become an inevitable trend in astronomical data processing. Research by Goz et al. (2020) indicates that astronomy and astrophysics face the challenge of conducting computationally intensive simulations. To address the massive datasets generated by new-generation observational facilities in astronomy, computing facilities, including HPC, are indispensable.

Existing HPC systems typically access storage management systems such as Lustre, BeeGFS, etc., through I/O nodes, and the speed of data access is often constrained by the performance of these I/O nodes (Lüttgau et al., 2018). In astronomical data processing, there are unique data access characteristics. Cheng W et al.'s research (Cheng et al., 2021) indicates that astronomical data primarily includes images, spectra, time series data, and simulation data. Each data item has multiple characteristics, and there are various data types, including structured, semi-structured, unstructured, and hybrid. Ladeyschikov D A et al.'s study (Ladeyschikov et al., 2019) implemented an improved dataset of methanol masers, revealing that the current trend involves using large-scale surveys to address various astrophysical issues, leading to an increase in the correlation between data. In the processing of astronomical data, different files exhibit certain correlations in terms of temporal order, spatial relationships, spectral correlations, and data access patterns due to factors such as observation sequence, observed targets, and processing methods.

Tiered storage provides effective support for high-performance I/O applications in the field of computing by managing data in tiers based on access frequency and performance requirements. However, there is limited targeted research on data access in specific domains, including astronomical studies. The current data placement schemes do not sufficiently reflect the I/O performance requirements of data processing. This paper, based on the characteristics of astronomical data access, explores and proposes a tiered storage approach to address the insufficient I/O node performance and limited high-speed storage in astronomical data processing, aiming to enhance data access speed. The main content of this paper:

- This paper innovatively utilizes the characteristics of astronomical data access, employing the C4.5 decision tree for access correlation analysis and providing load prediction information for data copy migration.
- Building upon tiered storage, an adaptive copy migration strategy is designed and implemented. It dynamically adjusts weights based on specific data processing conditions.
- The effectiveness of the proposed algorithm is tested in a real-world data processing environment with concurrent execution of the astronomical data processing workflow.

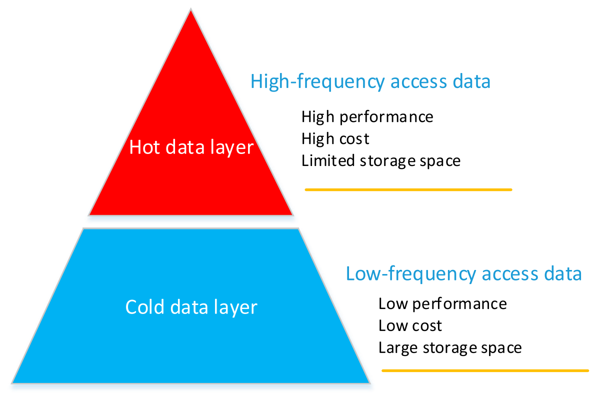


FIGURE 1
Tiered storage structure.

The first chapter of this paper introduces the research background and progress of hierarchical storage. The second chapter presents the designed correlation algorithm and migration strategy. In the third chapter, we conducted performance tests using hierarchical storage on two typical astronomical data processing pipelines. The fourth chapter evaluates the effectiveness of the proposed algorithm in a real astronomical data processing environment.

2 Tiered storage

2.1 Research background of tiered storage

Tiered storage is a data management strategy that categorizes data into multiple levels based on its access frequency and importance, storing it in different media (Zhang et al., 2010). Each tier employs distinct storage technologies and hardware to achieve more efficient data access and management.

A typical tiered storage architecture is illustrated in Figure 1. Generally, tiered storage systems consist of a hot storage layer and a cold storage layer. The hot storage layer is designed for storing frequently accessed hot data and typically utilizes high-speed storage media, such as Solid State Drives (SSD) or memory. These media have faster read and write speeds with low latency, meeting the requirements for quick data access. Hot data often comprises actively used data, such as data undergoing analysis, computation, or visualization. The cold storage layer is designated for storing less frequently accessed cold data and typically employs lower-cost storage media like disks or tapes. These media offer larger storage capacities but slower access speeds, making them suitable for storing historical data, backup data, or infrequently accessed data. For HPC systems, the hot storage layer often corresponds to local SSD storage or memory on computing nodes, while the cold storage layer corresponds to centralized storage accessed via high-speed network interfaces.

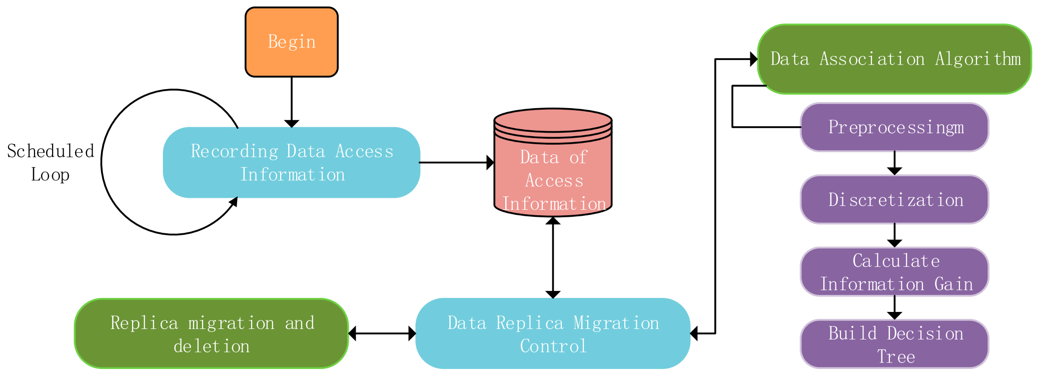

Begin
Scheduled Loop
Recording Data Access Information
Data of Access Information
Data Association Algorithm
Preprocessingm
Discretization
Calculate Information Gain
Build Decision Tree
Replica migration and deletion
Data Replica Migration Control

FIGURE 2
Firstly, the access information of the data is collected and recorded in the database. The Data Replica Migration Control component then transfers the relevant information to the prediction module. After the prediction module completes its forecasting, the Data Replica Migration Control, based on the replica control algorithm, manages the migration and deletion of data replicas.

2.2 Tiered storage research progress

The use of tiered storage or Software-Defined Storage (SDS) methods can effectively support high-performance I/O applications. Torabzadehkashi M et al.'s research (Torabzadehkashi et al., 2019) created a computational storage platform that leverages the high data throughput and ability to handle multiple I/O commands simultaneously provided by SSDs, thus improving data processing speed. Macedo R et al.'s study (Macedo et al., 2020) introduces how SDS enhances the control functions of traditional storage systems. It improves I/O performance through customized I/O paths, copy placement strategies, and data tiering. While tiered storage effectively improves data access speed, the multi-tiered data access control and data copy placement strategies increase the complexity of storage systems (Kougkas et al., 2020). The most direct method of data copy placement is based on empirical heuristic methods, where statistical characteristics of data files (access frequency, storage time, etc.) are used to determine data copy migration in and out based on set thresholds. Although this method is simple to implement, it lacks predictions of data file access and adaptability to different I/O workloads. Migration strategies based on data access prediction are currently essential research directions. Zhang et al. (2010) analyzed factors affecting the efficiency of data migration prediction, designed an algorithm to adaptively determine the optimal prediction window size, optimized the effectiveness of prediction-based data migration, and improved the performance and resource utilization of storage systems. Zhang et al. (2022) designed an Intelligent Tiered Storage System (HSS), proposing an open-source tiered storage framework with a dynamic migration strategy based on Reinforcement Learning (RL). Cheng et al. (2021) designed a persistent tiered cache for the Lustre file system based on NVMM, providing consistent persistent caching services for I/O applications running on storage nodes.

In the field of scientific data processing, Ghoshal and Ramakrishnan (2017) developed the MaDaTS software architecture based on Virtual Data Space (VDS). This architecture is used to manage scientific workflows processing data in multi-tiered storage on HPC systems, providing three data management strategies: Storage-Aware (VDS-STA), Workflow-Aware (VDS-WFA), and User-Driven (VDS-Passive). Experimental results indicate that compared to traditional methods, MaDaTS shows improvements in performance, resource utilization, and storage space. In the research conducted by Khalsa et al. (2020), the OpenAltimetry project was implemented, allowing users to locate, visualize, and download data from the ICESat-2 satellite anytime, anywhere. The data management of the OpenAltimetry project adopts a tiered storage approach, significantly shortening the data retrieval process and enabling rapid processing of newly available data.

Effective migration algorithms can enhance the performance of tiered storage systems. Existing tiering strategies mainly rely on predictive algorithms for data access heat or periodic access to achieve data migration. However, compared to these general strategies, there is limited research focused on migration algorithms with access-specific characteristics, such as those tailored for data in specific domains like astronomical observations or geographic information systems. Utilizing migration algorithms based on data access characteristics allows for better adaptation to data access patterns and requirements, leading to more effective performance improvements.

3 Data association algorithms and migration strategies

In tiered storage, predictive algorithms and migration strategies are the two most crucial components. Firstly, the C4.5 decision tree algorithm was employed as a foundation to implement an access correlation algorithm for astronomical data. Subsequently, the copy migration strategy comprehensively considers both data access heat and the correlation data for a single access, determining the placement of copies in SSD. The architecture of the data copy placement strategy is depicted in Figure 2.

3.1 Association algorithm

Decision tree is an effective supervised learning method designed to partition a dataset into as homogeneous groups as possible. Given that astronomical data often involves attributes with continuous intervals, this paper employs the C4.5 decision tree algorithm (Hssina et al., 2014) as a foundation to implement an access correlation algorithm for astronomical data. The algorithm extracts metadata from FITS(Flexible Image Transport System) files to create a dataset for the decision tree model. Information gain is set as the correlation value, enabling the identification of associated files for data files accessed in a centralized storage during a single access. To ensure algorithm efficiency, we store metadata and statistical information in an in-memory database. The first step is to extract metadata from FITS files as features and preprocess them by extracting keywords, and numericalizing non-numeric features. A crucial aspect of this process is the selection of numerical intervals, which relies on an understanding and informed selection based on the characteristics of astronomical data.

The second step involves discretizing continuous attributes such as observation time and celestial coordinates. In other words, for continuous attributes a, use Formula 1 to generate a candidate partition set containing $n-1$ elements:

$$T_a = \left\{ \frac{a^i + a^{i+1}}{2} | 1 \le i \le n-1 \right\} \tag{1}$$

Namely, using the midpoint $\frac{a^i+a^{i+1}}{2}$ of interval $[a^i, a^{i+1})$ as the candidate partition point.

The third step involves calculating the information gain for different FITS metadata based on historical access information. For discrete values of metadata, the information gain is according to Formula 2.

$$\text{Gain}(D,a) = Ent(D) - \sum_{i-1}^{n} \frac{|D^i|}{|D|} Ent(D^i) \tag{2}$$

$\text{Gain}(D,a)$ represents the partitioning of the sample set D based on feature a.

The formula for uncertainty Ent(D) is Formula 3:

$$\text{Ent}(D) = -\sum_{i=1}^{c} p_i \log_2(p_i) \tag{3}$$

p_i is the probability of the occurrence of the i-th class of data in dataset D.

For continuous attributes such as observation time and celestial coordinates, it is necessary to choose the optimal splitting point for partitioning. Therefore, the calculation method for the information gain is given by Formula 4:

$$\text{Gain}(D,a) = \max_{t\in T_a} Gain(D,a,t) = \max_{t\in T_a} \left(Ent(D) - \sum_{\lambda\in\{-.+\}}^{n} \frac{|D_t^\lambda|}{|D|} Ent(D_t^\lambda) \right) \tag{4}$$

Gain(D, a, t) is the information gain obtained by dividing the sample set D into two parts based on the split point t. That is, choosing the split point t that maximizes Gain(D, a, t).

The fourth step involves constructing the decision tree model and training/testing it with historical data.

When associating data accessed within a specified time period T, the metadata of the accessed data and the metadata of data in centralized storage are input into the constructed decision tree model. The information gain obtained serves as the association value.

3.2 Data copy migration strategy

The migration strategy comprehensively considers the data access heat and the correlation among data to determine the placement of copies in SSD. It primarily involves the following steps:

3.2.1 Data access heat statistics

Record the access frequency of each data block in centralized storage, use real-time monitoring tools to collect data access information.

3.2.2 Correlation analysis

Use the algorithm in Section 3.1 to analyze the correlation of other data accessed in a single instance during the statistical period, using the information gain obtained as the correlation value.

3.2.3 Copy placement strategy

At the beginning, half of the replica data in the local SSD comes from high-access-heat data, and the other half of the replicas comes from data correlated with the associated data. Within a specified time interval, count the number of hits for replicas and the frequency of hits for individual replicas. The next time interval determines the replica ratio according to the Formula 5:

$$\begin{cases} \dfrac{P_{\hbar eat}}{P_{ass}} = \dfrac{\sum_{i=1}^{Num_{heat}} h_{heat-i} \times H_{\hbar eat}}{\sum_{i=1}^{Num_{ass}} h_{ass-i} \times H_{\text{ass}}} \\ \min\limits_{1\le i\le Num_{heat}} \quad h_{heat-i} > 1 \end{cases} \tag{5}$$

Where h_{heat-i} is the number of hits for the ith high-access-heat replica in the previous time interval, H_{heat} is the number of high-access-heat replicas hit in the previous time interval. Similarly, h_{ass-i} is the number of hits for the ith data correlation replica in the previous time interval, and H_{ass} is the number of data correlation replicas hit in the previous time interval. When the access frequency of high-access-heat replicas does not exceed once within the time interval ($h_{heat-i} \nless 1$), the access frequency of data blocks in centralized storage is used for proportional calculation with correlated data.

3.2.4 Data movement and migration

The migration algorithm designed in this paper involves placing data replicas in the local SSD. Each time interval only updates data replicas, and replicas with access frequency below a threshold are removed from the SSD and replaced with new data replicas. The data in centralized storage remains unchanged, and the access address of the data is determined by the unified data access framework.

3.2.5 Iteration and optimization

Regularly review the effectiveness of the strategy and make adjustments and optimizations based on actual performance and requirements. The main adjustment goals are as follows:

1) Periodically assess the performance of the data access heat and correlation analysis algorithm. Adjust the statistical period according to the actual situation to ensure the real-time nature of data access information.
2) For data movement and migration, implement dynamic adjustments to the threshold for replicas, ensuring that

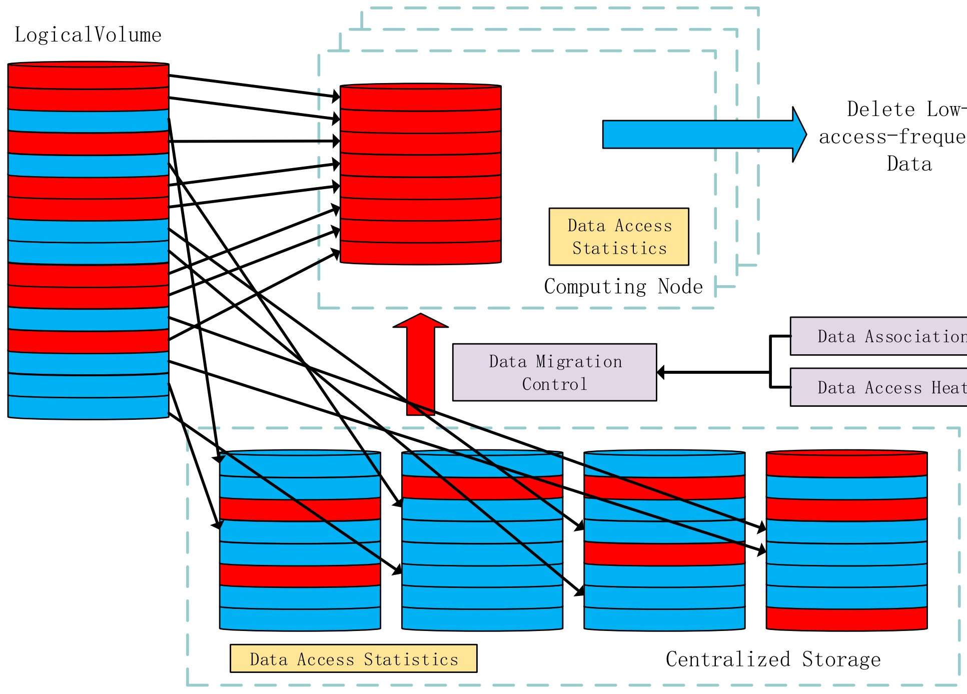


FIGURE 3
Tiered storage architecture.

only replicas with access frequencies below a certain threshold are deleted and replaced to reduce unnecessary data migration cost.

3.3 Unified access architecture design

By constructing a tiered storage architecture and deploying it on computing nodes using the FUSE(Filesystem in Userspace)[1] framework, a custom file system has been implemented. This system can place data file replicas based on the migration strategy proposed in this paper. The architecture design is shown in Figure 3. The application of the FUSE framework ensures that file access continues to occur through a unified file path, making the entire architecture highly versatile. This design eliminates the need for modifications to data processing software, as file replica placement and migration are handled internally within the FUSE framework. Simultaneously, data access statistics are collected through the Linux file system interface. The data association algorithm predicts the access probability of associated data, and the information about the data to be migrated is written into the database according to the data replica migration strategy.

The data replica control module utilizes this information to determine which data needs replica placement and executes the corresponding data replica migration. For applications, the tiered storage of data is a black-box operation, and there is no need to know the level at which the data resides. This integrated tiered storage architecture not only ensures the efficiency and flexibility of the system but also provides a user-friendly access method.

1 https://www.kernel.org/doc/html/next/filesystems/fuse.html

TABLE 1 Data processing node information.

Name	Informantion
CPU	Intel i9-10900K
RAM	64 GB RAM
Software	TEMPO2 2020.04.1
OS	Debian 10.10.0
Network Interface	10GbE
Hard Disk	1TB NVMe SSD

4 The impact of tiered storage on the performance of astronomical data processing

To investigate and analyze the impact of tiered storage on the performance of astronomical data processing, two specific experiments were conducted, involving the preprocessing of formaldehyde molecular spectral line data and pulsar search data. The first experiment tested the preprocessing pipeline for formaldehyde molecular spectral line data, comparing the

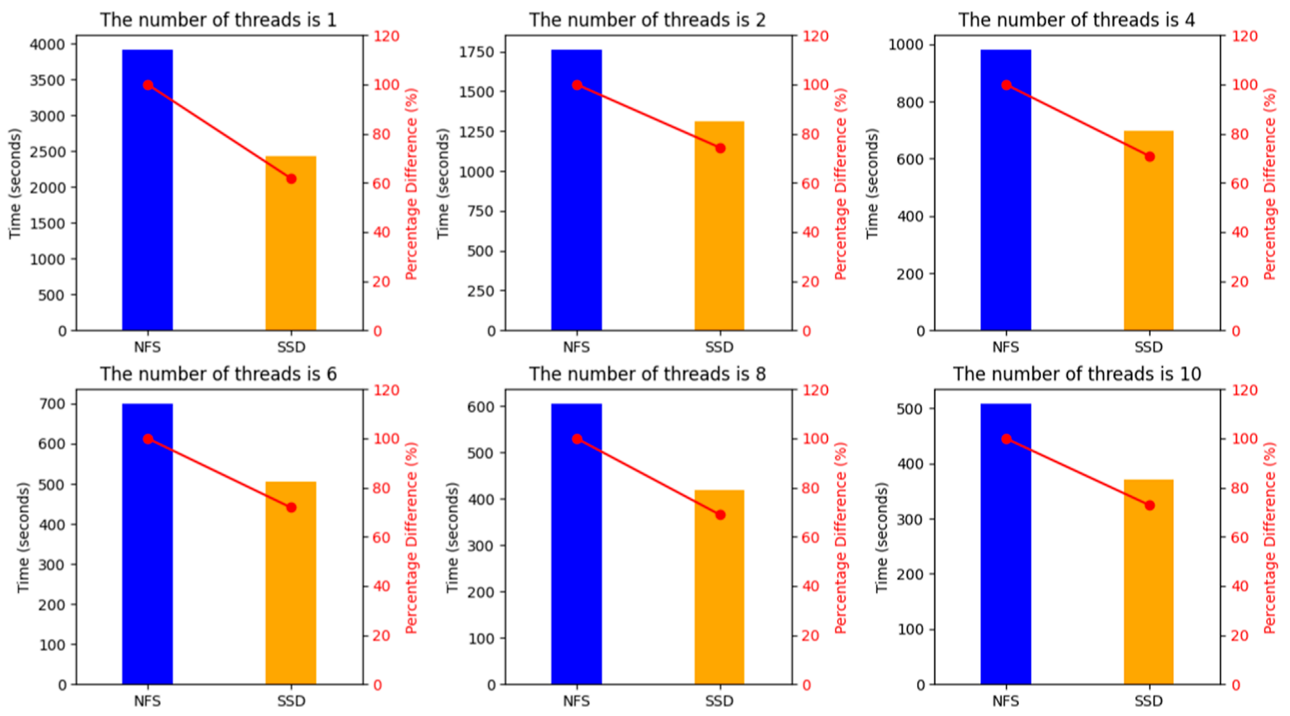


FIGURE 4
The comparison of Xfb data processing time at different storage locations is illustrated by bar charts showing the actual processing time and line charts depicting the percentage of time consumption. For various data processing thread counts, storing data in SSDs consistently proves effective in reducing data processing time.

processing speed differences when the data is stored in centralized storage and local SSD. The second experiment focused on the pulsar search pipeline of the FAST 19-beam receiver, comparing the impact of various data placement schemes on data processing speed.

4.1 Testing of the formaldehyde molecular spectral line data preprocessing pipeline

Testing of the Data Preprocessing Pipeline for formaldehyde Molecular Spectral Line Data involves calibrating data obtained from the 26-m radio telescope at the Nanshan Observatory of Xinjiang (Tang et al., 2013). The data preprocessing pipeline reads raw data recorded by the Xfb digital terminal, calibrates the intensity and velocity of spectral line data. Intensity calibration considers factors such as Tsys, gain curve, and atmospheric transparency. Velocity calibration uses the astropy[2] software package to calibrate the observed velocity to the local standard of rest velocity. The calibrated spectral line data is encapsulated into Fits format files. The test data consists of 772 files, totaling 418 GB. The data is placed in both centralized storage mounted via a 10Gbe network and the local SSD of computing nodes. The processing is performed using 1, 2, 4, 6, 8, and 10 threads, and data processing time is compared. Each test group is run 5 times, and the final data processing time is the average of these five tests. The configuration information for the data processing nodes is shown in Table 1. The test results are presented in Figure 4.

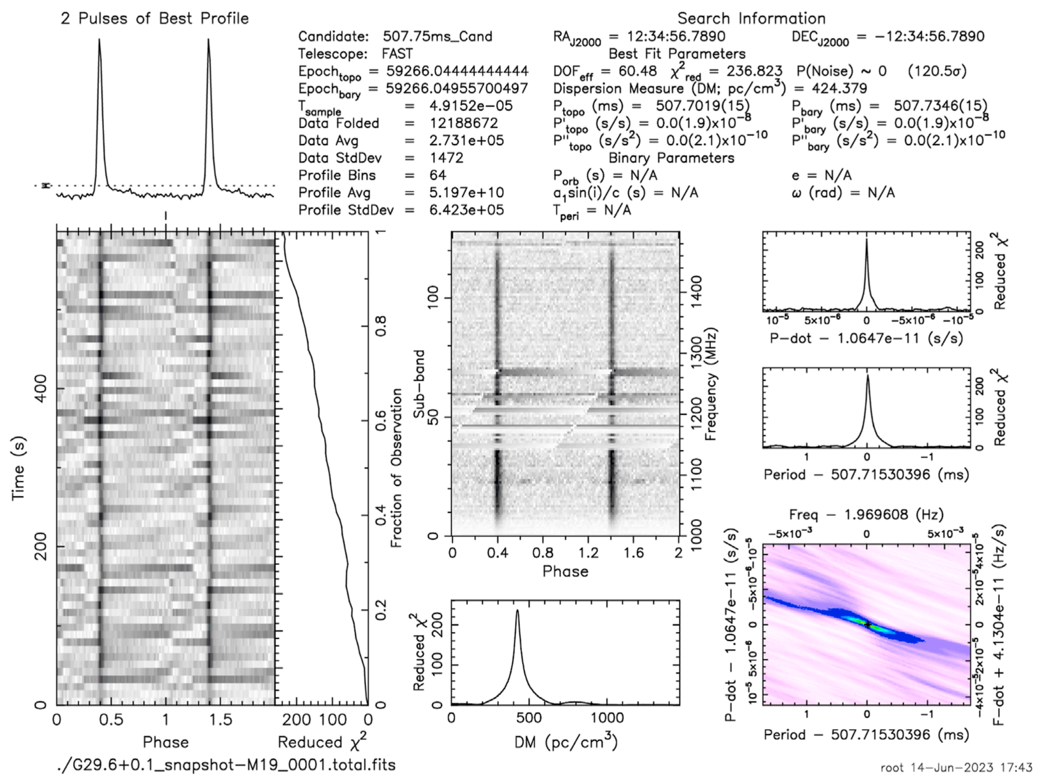
The experiments indicate that, under different thread configurations, directly reading data from the local SSD of the computing nodes for preprocessing of formaldehyde molecular spectral line data results in faster processing speeds compared to accessing centralized storage. In single-threaded data processing, the local SSD exhibits the optimal acceleration effect, reducing processing time by 38.15%. This is because in multi-threaded processing, threads queuing for data access can occupy idle bandwidth, thereby achieving more efficient data access. In the subsequent tests, our focus will primarily be on multi-threaded data processing.

4.2 Pulsar search data testing

In the data processing of the FAST 19-beam receiver for pulsar search (Jiang et al., 2020), the data processing pipeline[3] used in this paper first reads the observation data of a specific coordinate from the first beam for concatenation. It then reads data from three offset positions of that beam. Intermediate data will be frequently read to traverse different DM(Dispersion Measure) values for pulsar search. The DM value of a pulsar refers to the dispersion measure, which is a measure of the dispersion effect caused by the interstellar

2 https://www.astropy.org/

3 https://github.com/gajjarv/PulsarSearch


```
Candidate:  507.75ms_Cand
Telescope:  FAST
Epoch_topo  = 59266.04444444444
Epoch_bary  = 59266.04955700497
T_sample        =  4.9152e-05
Data Folded     =  12188672
Data Avg        =  2.731e+05
Data StdDev     =  1472
Profile Bins    =  64
Profile Avg     =  5.197e+10
Profile StdDev  =  6.423e+05

                         Search Information
RA_J2000 = 12:34:56.7890              DEC_J2000 = -12:34:56.7890
            Best Fit Parameters
DOF_eff = 60.48  χ²_red = 236.823  P(Noise) ~ 0   (120.5σ)
Dispersion Measure (DM; pc/cm³) = 424.379
P_topo (ms) =  507.7019(15)           P_bary (ms) =  507.7346(15)
P'_topo (s/s) = 0.0(1.9)x10^-8        P'_bary (s/s) = 0.0(1.9)x10^-8
P''_topo (s/s²) = 0.0(2.1)x10^-10     P''_bary (s/s²) = 0.0(2.1)x10^-10
            Binary Parameters
P_orb (s) = N/A                       e = N/A
a_1sin(i)/c (s) = N/A                 ω (rad) = N/A
T_peri = N/A
```

FIGURE 5
This is a result plot of pulsar search data processing, corresponding to the celestial position of a known pulsar. The result plots display information such as time domain, frequency domain, DM value, etc. Astronomers can use these result plots to determine whether the observed target is a pulsar through manual inspection or machine learning methods.

medium (primarily the ionized gas in the interstellar space) during the propagation of the pulsar's radio pulse signal.

For the pulsar search pipeline, we designed three sets of comparative experiments: 1) Original data placed in centralized storage, directly reading data from centralized storage for data processing in each iteration; 2) Original data placed in centralized storage, placing intermediate data in local SSD before each iteration; 3) Original data placed in local SSD, directly reading data from SSD for data processing in each iteration.

As the data processing time is long, we selected single-beam data from the FAST telescope's 19-beam receiver for testing. The number of data is 382, with a single file size of 2.1 GB and a total size of 802.2 GB. The nodes and mounted devices used in the test are the same as those in Section 4.1, and the number of parallel threads used by the data processing pipeline is 8. Each test was run three times to obtain an average value. Assuming DM < 1,000, given that the algorithm has been determined, the DM step size has the greatest impact on the pulsar search speed. We tested the processing time for pulsar data with different step sizes. The data processing results for one of the cases are shown in Figure 5, and the test results are shown in Figure 6.

Since pulsar data processing involves multiple accesses to intermediate data, the placement of this intermediate data has a significant impact on data processing speed. In contrast, the generation of concatenated files, created once and subsequently accessed multiple times, constitutes a relatively small proportion of the total processing time. Therefore, the original file's placement has a minor impact on the overall processing time of data. Overall, Scheme 3 only reduces processing time by 2.28% compared to Scheme 2. This indicates that, in data processing where intermediate data is frequently accessed, the placement of intermediate data has a more significant impact on the overall processing time.

Both experiments underscore the significant influence of data placement on data processing speed in astronomical data processing. In cases where local SSD space is limited, using algorithms to pre-copy data to the SSD can enhance processing speed to a certain extent. The comparison of the two experimental results suggests that data copy placement strategies should possess adaptability to effectively handle different types of data access scenarios, enabling flexible switching between strategies based on specific requirements.

5 Astronomical data processing pipeline performance test

To assess the performance of the proposed heat correlation algorithm and replica migration algorithm under different data placement strategies, we placed formaldehyde molecular spectral line data and pulsar search data in centralized storage, simulating the coexistence of multiple characteristic data processing workflows in a real environment. The goal is to compare the data processing durations for data placed in centralized storage, data migration

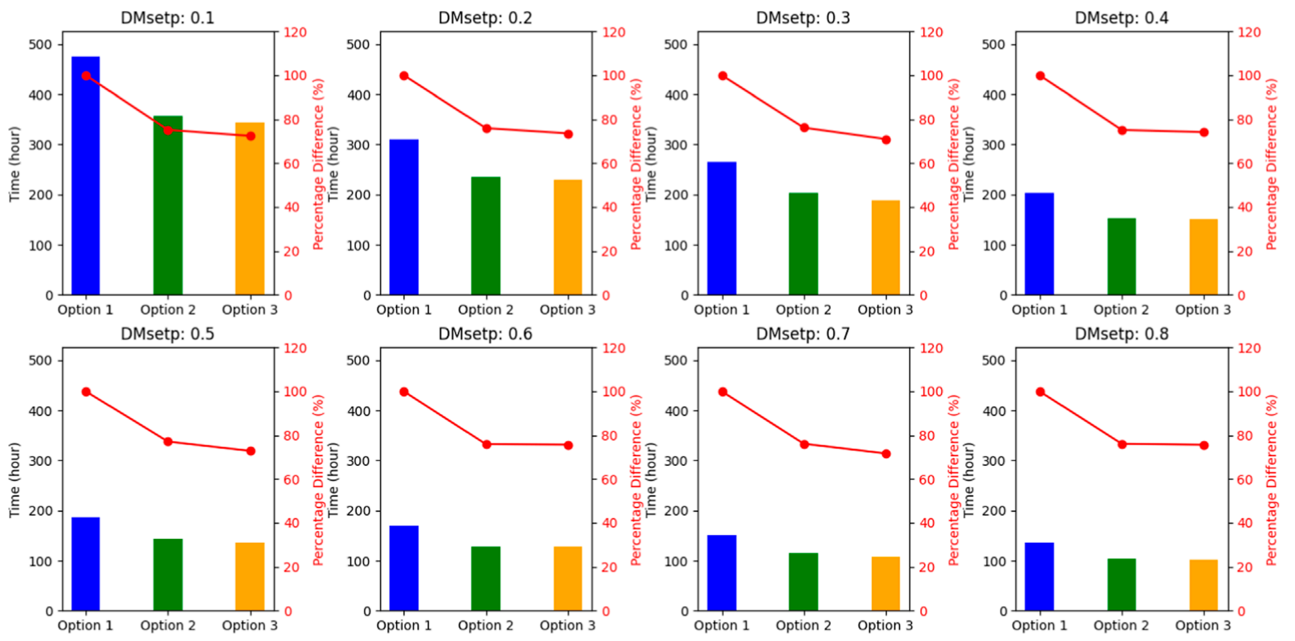


FIGURE 6
A comparison of the duration of pulse star search data processing for different data placement strategies is illustrated by bar charts showing the processing duration and line charts depicting the percentage of time consumption. Both Scheme 2 and Scheme 3 prove effective in reducing the data processing duration, with only a minor difference in processing time between the two.

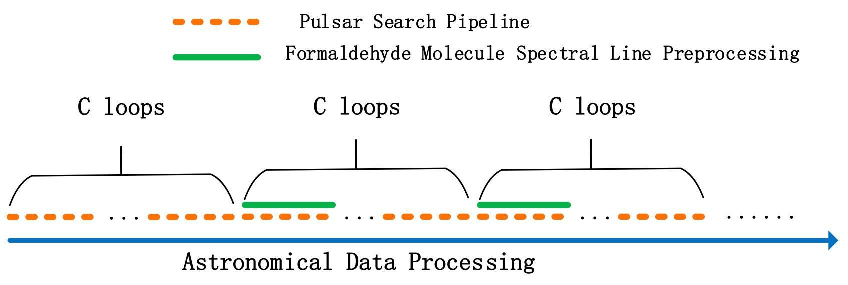

Pulsar Search Pipeline
Formaldehyde Molecule Spectral Line Preprocessing
C loops
C loops
C loops
Astronomical Data Processing

FIGURE 7
Test environment data processing workflow.

based on access heat, and the migration strategy designed in this paper, thereby validating the effectiveness of the proposed algorithms in improving data processing speed. The experimental plan is as follows:

5.1 Data placement strategy

Place formaldehyde molecular spectral line data and pulsar search data in centralized storage, consistent with the data conditions in Section 4.1 and Section 4.2.

5.2 Data processing workflow

In the pulsar search pipeline loop (with DM step settings as in Section 4.2), run the formaldehyde molecular spectral line preprocessing pipeline once every C loops. The data processing workflow is illustrated in Figure 7.

5.3 Comparison of data placement strategies

Compare the data processing durations for data placed in centralized storage, data scheduled based on access heat, and the scheduling strategy based on heat correlation.

Through this experimental plan, we aim to comprehensively evaluate the performance of different data placement strategies in a combined astronomical data processing workflow, validating the practical effectiveness of the proposed algorithms.

The hardware environment used in the experiment is detailed in Table 1, and all data processing pipelines run in parallel with eight threads. By employing a mixed data processing approach, we

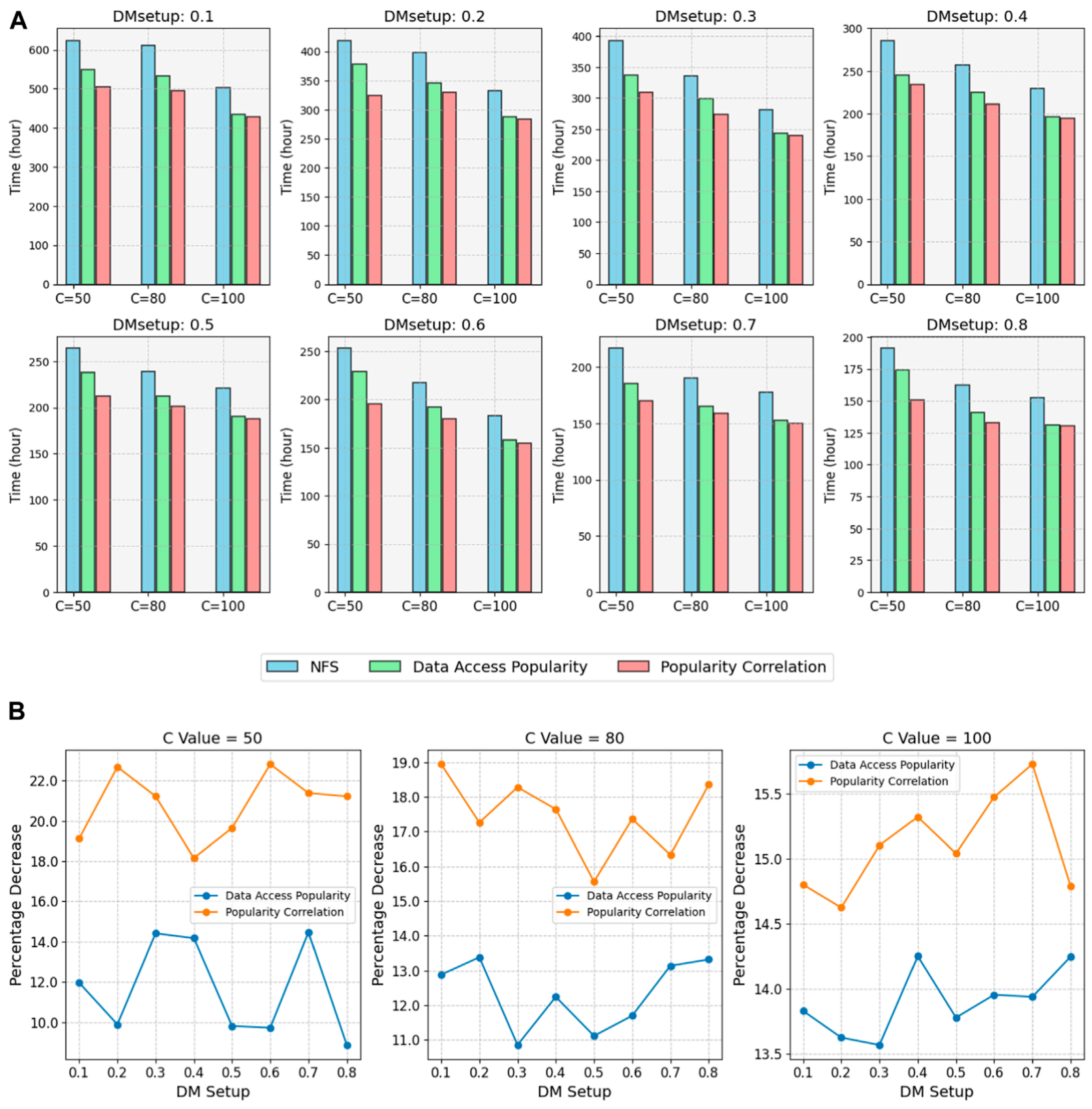


FIGURE 8
(A) Displays the comparison of data processing durations for different data placement strategies. **(B)** Illustrates the percentage of time consumption for the data access heat algorithm compared to the algorithm presented in this paper. The comparison reveals that the algorithm proposed in this paper can effectively reduce data processing time, and its effectiveness is correlated with the inter-data correlation.

simulate the coexistence of multiple characteristic data processing workflows in a real environment. The test results are shown in Figure 8.

The test results indicate that, in the simultaneous execution of multiple astronomical data processing tasks, both the traditional heat algorithm and the algorithm proposed in this paper can improve data processing speed to varying degrees compared to centralized storage access. When the loop value is set to 50, the traditional heat algorithm reduces processing speed by 11.66%, and the algorithm proposed in this paper reduces it by 20.77%. With a loop value of 100, the traditional heat algorithm decreases processing speed by 13.90%, and the algorithm proposed in this paper reduces it by 15.11%. It is evident that the algorithm proposed in this paper demonstrates a more pronounced advantage in data processing with high inter-data correlation.

6 Conclusion

This paper proposes a data access prediction algorithm based on the temporal, spatial, and spectral characteristics of astronomical data, as well as data access patterns. The algorithm utilizes the C4.5 decision tree to perform access correlation analysis, providing crucial information for data replica migration. A replica migration strategy is designed and implemented on top of Tiered storage, considering both data access heat and the correlation of single-accessed data, exhibiting good adaptability.

In the graded storage test for formaldehyde molecular spectral line preprocessing of NSRT observations, a total of 772 data sets with a size of 418 GB were processed. Graded storage improved data processing speed by 38.15% in single-threaded processing and consistently increased processing speed by over 20% in multi-threaded processing. In the pulsar search test with FAST's 19-beam receiver data, 382 data sets totaling 802.2 GB were processed. Comparing different storage schemes and fixed search accuracies, graded storage achieved a maximum improvement of 29.00% in data processing speed. Due to the significant impact of repeated reads of concatenated files in pulsar searching, the storage location of intermediate files is more sensitive to data processing speed. In multi-concurrent data processing tests, the proposed algorithm improved data processing speed by an average of 17.78% compared to centralized storage. When compared to a replica migration algorithm based on access heat, the proposed algorithm increased data processing speed by an average of 5.15%. The effectiveness of the proposed algorithm is positively correlated with the data associativity. The tests demonstrate that the algorithm effectively leverages the characteristics of astronomical data processing, enhancing processing efficiency in situations where local high-speed storage is limited and alleviating I/O bottlenecks in data processing.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

X-CY: Writing–original draft, Writing–review and editing. H-LZ: Writing–review and editing. JW: Writing–review and editing. Y-ZZ: Writing–review and editing. XD: Writing–review and editing. HW: Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work is supported by National Key R&D Program of China Nos. 2021YFC2203502 and 2022YFF0711502; the National Natural Science Foundation of China (NSFC) (12173077 and 12003062); the Tianshan Innovation Team Plan of Xinjiang Uygur Autonomous Region (2022D14020); the Tianshan Talent Project of Xinjiang Uygur Autonomous Region (2022TSYCCX0095); the Scientific Instrument Developing Project of the Chinese Academy of Sciences, Grant No. PTYQ2022YZZD01; China National Astronomical Data Center (NADC); the Operation, Maintenance and Upgrading Fund for Astronomical Telescopes and Facility Instruments, budgeted from the Ministry of Finance of China (MOF) and administrated by the Chinese Academy of Sciences (CAS); Natural Science Foundation of Xinjiang Uygur Autonomous Region (2022D01A360).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Antunes, A. K., Winter, E., Vandegriff, J. D., Thomas, B. A., and Bradford, J. W. (2022). Profiling heliophysics data in the pythonic cloud. *Front. Astronomy Space Sci.* 9, 1006839. doi:10.3389/fspas.2022.1006839

Cavuoti, S., Brescia, M., De Stefano, V., and Longo, G. (2015). Photometric redshift estimation based on data mining with PhotoRApToR. *Exp. Astron.* 39, 45–71. doi:10.1007/s10686-015-9443-4

Cheng, W., Li, C., Zeng, L., Qian, Y., Li, X., and Brinkmann, A. (2021). Nvmm-oriented hierarchical persistent client caching for lustre. *ACM Trans. Storage (TOS)* 17 (1), 1–22. doi:10.1145/3404190

El Bouchefry, K., and de Souza, R. S. (2020). "Learning in big data: introduction to machine learning," in *Knowledge discovery in big data from astronomy and earth observation* (Netherlands: Elsevier), 225–249.

Ghoshal, D., and Ramakrishnan, L. (2017). "Madats: managing data on tiered storage for scientific workflows," in Proceedings of the 26th International Symposium on High-Performance Parallel and Distributed Computing, DC, Washington, USA, June 26 - 30, 2017 (ACM), 41–52.

Goz, D., Ieronymakis, G., Papaefstathiou, V., Dimou, N., Bertocco, S., Simula, F., et al. (2020). Performance and energy footprint assessment of FPGAs and GPUs on HPC systems using astrophysics application. *Computation* 8 (2), 34. doi:10.3390/computation8020034

Hssina, B., Merbouha, A., Ezzikouri, H., and Erritali, M. (2014). A comparative study of decision tree ID3 and C4. 5. *Int. J. Adv. Comput. Sci. Appl.* 4 (2), 13–19. doi:10.14569/specialissue.2014.040203

Jiang, P., Tang, N. Y., Hou, L. G., Liu, M. T., Krčo, M., Qian, L., et al. (2020). The fundamental performance of FAST with 19-beam receiver at L band. *Res. Astronomy Astrophysics* 20 (5), 064. doi:10.1088/1674-4527/20/5/64

Khalsa, S. J. S., Borsa, A., Nandigam, V., Phan, M., Lin, K., Crosby, C., et al. (2020). OpenAltimetry-rapid analysis and visualization of Spaceborne altimeter data. *Earth Sci. Inf.* 15, 1471–1480. doi:10.1007/s12145-020-00520-2

Kougkas, A., Devarajan, H., and Sun, X. H. (2020). I/O acceleration via multi-tiered data buffering and prefetching. *J. Comput. Sci. Technol.* 35, 92–120. doi:10.1007/s11390-020-9781-1

Ladeyschikov, D. A., Bayandina, O. S., and Sobolev, A. M. (2019). Online database of class I methanol masers. *Astronomical J.* 158 (6), 233. doi:10.3847/1538-3881/ab4b4c

La Plante, P., Williams, P. K. G., Kolopanis, M., Dillon, J., Beardsley, A., Kern, N., et al. (2021). A Real Time Processing system for big data in astronomy: applications to HERA. *Astronomy Comput.* 36, 100489. doi:10.1016/j.ascom.2021.100489

Lüttgau, J., Kuhn, M., Duwe, K., Alforov, Y., Betke, E., Kunkel, J., et al. (2018). Survey of storage systems for high-performance computing. *Supercomput. Front. Innovations* 5 (1). doi:10.14529/jsfi180103

Macedo, R., Paulo, J., Pereira, J., and Bessani, A. (2020). A survey and classification of software-defined storage systems. *ACM Comput. Surv. (CSUR)* 53 (3), 1–38. doi:10.1145/3385896

Tang, X. D., Esimbek, J., Zhou, J. J., Wu, G., Ji, W. G., and Okoh, D. (2013). The relation of H2CO, 12CO, and 13CO in molecular clouds. *Astronomy Astrophysics* 551, A28. doi:10.1051/0004-6361/201219809

Torabzadehkashi, M., Rezaei, S., HeydariGorji, A., Bobarshad, H., Alves, V., and Bagherzadeh, N. (2019). Computational storage: an efficient and scalable platform for big data and hpc applications. *J. Big Data* 6, 100–129. doi:10.1186/s40537-019-0265-5

Zhang, G., Chiu, L., Dickey, C., Liu, L., Muench, P., Seshadri, S., et al. (2010). "Automated lookahead data migration in SSD-enabled multi-tiered storage systems," in 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Nevada, USA, 3-7 May 2010 (IEEE), 1–6.

Zhang, T., Hellander, A., and Toor, S. (2022). Efficient hierarchical storage management empowered by reinforcement learning. *IEEE Trans. Knowl. Data Eng.* 35, 1–5793. doi:10.1109/TKDE.2022.317 6753