AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Referred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 4(2026)

RIPPLE: A DISTRIBUTED FRAMEWORK FOR EFFICIENT DATA DEDUPLICATION AT THE EDGE

Khatija Mohammed Khazi[1], Dr. C. Berin Jones[2]
[1]PG Scholar, Department of CS, Shadan Women's College of Engineering and Technology, Hyderabad
mohammedkhatija23@gmail.com
[2]Professor & Head, Department of CSE, Shadan Women's College of Engineering and Technology, Hyderabad
jonesberin@gmail.com

ABSTRACT:

Edge computing is emerging as a backbone option for numerous latency-sensitive applications due to its benefits in guaranteeing low data retrieval latency and lowering backhaul network traffic. The capacity of edge storage systems is being stretched by the growing volume of data created at the edge. In recent years, improving edge storage systems' resource utilization has grown to be a major concern. Current solutions use data sharing, data segmentation, data location optimization, etc. to try and accomplish this goal. These methods ignore edge storage systems' data redundancy, which results in significant waste of storage resources. This highlights the necessity of a method for edge data deduplication. Nevertheless, current data deduplication techniques depend on centralized control, which isn't always possible in real-world edge computing settings.

A new framework called Ripple makes it possible for edge servers to carry out data deduplication in a completely decentralized way. Ripple creates a local data index on each edge server, enabling them to independently find and remove redundant data, in contrast to conventional methods that depend on centralized coordination. Because of its decentralized architecture, Ripple can:

1. effectively identify and eliminate redundant data;
2. maintain low-latency data retrieval standards; and (removing Unwanted Data)
3. After deduplication, make sure the data is accessible.

Comprehensive trace-driven tests on an actual testbed confirm Ripple's efficacy. Ripple's practical advantages in edge computing environments are demonstrated by its 60.42% reduction in data retrieval time and up to 16.79% improvement in the deduplication ratio when compared to current state-of-the-art solutions.

Keywords: Data redundancy, data deduplication, data index, edge computing, and data retrieval delay.

1. INTRODUCTION

A growing number of mobile and smart devices are producing significant amounts of data at the network edge because to the rapid development of the web of things (WoT) and artificial intelligence (AI). IDC estimates that by 2025, the world's data volume will be 180ZB, with a notable 70% coming from the network edge. Excessive network resources are used during the transfer of these data to the cloud for processing, which puts a significant strain on network traffic. Concurrently, the traditional cloud computing paradigm is unable to satisfy the increasing needs for low latency of latency-sensitive applications, such as autonomous driving, augmented reality, virtual reality, etc. [2], [3]. By moving data and processing power from the cloud to the network edge, multi-access edge computing (MEC), a novel computing paradigm, provides a solution. Low-latency data processing and online services are thus guaranteed [4]. In order to reduce data retrieval latency and enhance user experience, app suppliers such as YouTube and Tik Tok might distribute popular data on edge servers. An edge storage system (ESS), which consists of a number of connected edge servers in a particular region, can provide low-latency data delivery to local users. In contrast to cloud servers, edge servers are typically installed at 5G/6G base stations for users in the vicinity. Their restricted resources are determined by their small physical stature. In particular, their limited storage capacity affect how well the system delivers data. Some recent research have tried to optimize data placement, lower storage overheads via erasure coding, and send data across edge servers on demand in order to increase storage resource usage. In this work, we examine edge data redundancy, a basic and unexplored problem that significantly contributes to network edge storage resource waste. Edge Data Redundancy: 5G base stations, which are powered by mmWave communication technology, have a restricted coverage radius despite offering high capacity and supporting numerous concurrent connections. 5G base stations (as well as edge servers) are placed densely—up to 50 base stations per km2—to prevent blank spaces. Users in a certain location, such as a university or a central business district, frequently have comparable data needs. App providers cache popular content, such as viral videos and software upgrades, on the edge servers that span the region in response to these data requests. Data redundancy between these edge servers is an inevitable consequence of this. We gathered data on the number of likes for 16,342 well-known Tik Tok videos of various lengths across the globe in September 2023. The results are summarized in Fig. 1. The figure makes it evident that popular videos are of more interest to Tik Tok users. For instance, 98.6% of all likes come from 8.53% of the most popular videos. These popular data sets cause significant data redundancy in the system since they are cached across several edge servers. Challenges: There is a lot of data redundancy in cloud storage systems, and over the past ten years, numerous deduplication strategies have

been put out. Nevertheless, these methods are inappropriate for edge storage systems due to the following three drawbacks. Restrictions 1. Deduplication Granularity: Finding and reducing common data blocks shared by many storage nodes within the system is the fundamental concept that many of these systems have in common. ESSs could also use this concept. The drawback of this concept is that, in order to rebuild data in response to user requests for data retrieval, data blocks (a few hundred or thousands of data blocks) must be retrieved from various storage nodes. The Web of Things (WoT) and artificial intelligence (AI) are driving the exponential expansion of mobile and smart devices, which is producing previously unheard-of volumes of data at the network edge. By 2025, the world's data volume is expected to surpass 180ZB, with almost 70% coming from the edge. Cloud computing used to manage this kind of data processing and storage, but it is becoming less and less appropriate for latency-sensitive applications like AR, VR, and autonomous driving. Centralized cloud systems are unable to meet the demanding needs of developing applications due to their high latency and heavy backhaul traffic. By moving data storage and computing closer to consumers at the network edge, Multi-access Edge Computing (MEC) has evolved as a new paradigm to address these issues. Usually installed at 5G/6G base stations, edge servers offer low-latency localized services. But their limited resources—especially their storage capacity—create additional difficulties for data distribution. App providers cache these frequently requested data items on several edge servers spread over a certain region as user demand for popular content (such as viral videos and software upgrades) keeps increasing. This lowers access latency, but it also adds a lot of redundant data, which wastes a lot of the limited edge storage resources.

2. STIMULATING

The design, development, and implementation of a decentralized data deduplication framework for Mobile Edge Computing (MEC) environments are all included in the project's scope. By locating and removing redundant data while preserving low-latency information access, the framework aims to increase storage efficiency across dispersed edge servers. As part of the project, local indexing techniques will be developed so that each edge server may keep track of the storage state of nearby servers. Without the need for centralized coordination, these indexes will facilitate effective data retrieval, duplicate detection, and discovery processes. In order to satisfy the needs of delay-sensitive applications, the system will employ latency-aware deduplication algorithms to guarantee that data stays available within predetermined hop-count or latency limitations. In order to promote decentralized decision-making and enhance scalability, the scope also includes neighbor-based communication and coordination among edge servers. To preserve data consistency and integrity, mechanisms for secure data lookup, duplicate verification, and redundancy elimination will be included. By cutting down on needless data replication and eliminating resource waste throughout the network, the project will address storage optimization. By facilitating data sharing and retrieval straight from neighboring edge servers, it will also concentrate on lowering backhaul network traffic. In order to account for variations in storage availability and network conditions, the framework will enable dynamic updating of local indexes. Metrics including deduplication ratio, storage savings, retrieval latency, communication overhead, and resource consumption will be used to evaluate performance. Even when duplicate data is eliminated, the system will be built to ensure high data availability and fault tolerance. The research will also investigate resource management methods appropriate for large-scale MEC installations, decentralized coordinating methodologies, and effective indexing structures. Content distribution, IoT data management, smart cities, autonomous systems, healthcare services, and real-time multimedia applications are just a few of the edge computing applications that the framework can be used for. The project's overall goal is to offer a scalable, dependable, and effective storage management system that strikes a balance between data accessibility, service quality, and storage optimization in contemporary edge computing infrastructures.

3. OBJECTIVE

In order to increase storage efficiency and decrease duplicate data storage, the main goal of this project is to design and deploy Ripple, a decentralized data deduplication framework created especially for Mobile Edge Computing (MEC) environments. The framework seeks to remove redundant data from dispersed edge servers while preserving high data availability and low latency data retrieval. Ripple makes it possible to efficiently index, search, and identify duplicate material among nearby edge nodes by using the shortest-path Bloom filter tree (Bloom tree) data structure. Allowing each edge server to autonomously manage its local index and carry out deduplication operations without depending on a centralized controller is another goal. The system aims to maximize the use of scarce edge resources and minimize storage consumption. By avoiding needless data transfer and storage, it also seeks to reduce backhaul network traffic. Ripple is made to make sure that applications that are sensitive to latency get data within the necessary reaction time limits. By enabling an increasing number of edge servers and rising data volumes, the framework aims to improve scalability. Through decentralized coordination among edge nodes, it seeks to improve fault tolerance and system resilience. Maintaining constant data availability even after duplicate data is eliminated from several sites is

another goal. By effectively managing indexing structures, the project also aims to lower memory overhead. Additionally, Ripple seeks to ensure dependable user experiences in edge computing applications by striking a balance between storage optimization and service quality. The framework is intended to promote effective resource allocation, improve overall system performance, and enable intelligent data management in distributed edge situations. In the end, the project offers a scalable, dependable, and decentralized deduplication solution to address important storage and performance issues in MEC systems.

3.1 What The Project Is About

Massive amounts of data are being produced at the network edge due to the quick expansion of smart and mobile devices driven by artificial intelligence (AI) and the Web of Things (WoT). Multi-access Edge Computing (MEC) is a promising alternative to traditional cloud computing because it can meet the strict low-latency requirements of new applications like virtual reality, augmented reality, and autonomous driving. However, handling duplicated data becomes extremely difficult due to edge servers' limited storage capabilities, particularly when they are placed densely at 5G/6G base stations. In order to satisfy user demand, popular content—like viral videos and software updates—is frequently cached across several adjacent edge servers, which causes significant storage waste. Three major drawbacks prevent existing deduplication techniques intended for cloud environments from being directly applied to edge systems: (1) fine-grained deduplication necessitates excessive data reconstruction time; (2) excessively aggressive replica reduction results in undesired latency in MEC environments; and (3) reliance on centralized control introduces delays and backhaul traffic overheads. These difficulties show how urgently a decentralized, latency-aware deduplication strategy designed especially for edge storage systems is needed to increase storage efficiency without sacrificing service quality.

4. EXISTING SYSTEM:

- Because edge computing may enable latency-sensitive applications by processing data closer to the source, it has garnered a lot of attention in recent years.
- Existing systems have concentrated on methods like data location optimization, partitioning, and data sharing to maximize the limited storage resources available on edge servers.
- By distributing data over several edge nodes, reducing access time, and balancing demand, these techniques seek to effectively use storage.
- However, the growing problem of data redundancy in edge contexts, which continues to waste important storage space, is not addressed by these systems, which mainly focus on data delivery.

4.1 Existing System Disadvantages:

- Deduplication support is lacking.
- Backhaul traffic and delay are increased when deduplication metadata is communicated with a central server.

4.2 THE PROPOSED SYSTEM

- The proposed system, Ripple, introduces a ground breaking approach to data deduplication in edge computing environments by eliminating the need for centralized control.
- Each edge server can independently carry out deduplication tasks and maintain a local data index thanks to this decentralized design.
- Ripple greatly increases storage efficiency while maintaining the low-latency requirements necessary for real-time applications by allowing each server to identify and remove redundant data at the edge.
- The system guarantees good data availability even after deduplication and is built to function under the limitations of edge computing, such as scarce resources and high dispersal.
- For contemporary edge infrastructures, Ripple stands out as a scalable and useful solution that solves the drawbacks of conventional centralized approaches.

4.2.1 PROPOSED SYSTEM ADVANTAGE

- Easily expands to many edge nodes without experiencing bottlenecks.
- Decreased Backhaul Traffic

5. METHODOLOGIES

1. Design of User Interfaces

The user must enter their username and password in order to connect to the server. The user can log in to the server directly if they have previously left; otherwise, they must register their information, including their username, password, email address, city, and country. To manage upload and download rates, the database will create an account for every user. The user ID will be used as the name. To access a certain page, you typically need to log in. The query will be searched and displayed.

2. User

Users upload data files for processing through the User module, which serves as the system's entry point. Files that need to be examined for unnecessary content and duplication can be submitted by users. The comparison module receives the uploaded data and uses it for additional analysis. A straightforward interface for controlling data uploads is offered by this module. It starts the deduplication process from scratch.

6. TECHNIQUE ALGORITHM USED

6.1 Proposed Algorithm

- **Decentralized Redundant Data Detection and Elimination Algorithm (DRDDEA)**

In a distributed setting, the Decentralized Redundant Data Detection and Elimination Algorithm (DRDDEA) is intended to find and eliminate redundant or duplicate data while preserving data integrity and storage effectiveness. The program starts by gathering data records from many network-participating decentralized nodes. A distinct fingerprint or hash value that represents the content of each data item is produced through processing. By comparing records using these hash values, security and privacy are maintained without disclosing the real data. For previously saved records, the method keeps track of metadata and matching hash references in a distributed index. A new data item's hash value is compared to already-existing entries in the distributed index upon submission. The method marks the data as possibly redundant and carries out further verification to confirm duplication if a matching hash is discovered. When redundancy is verified, the system generates a reference link to the current data rather than keeping a duplicate copy. The new data is saved and its metadata is added to the index if no match is found. To guarantee that all participating nodes concur on redundancy detection findings, consensus procedures are employed. To ensure uniformity throughout the network, the algorithm updates metadata records on a regular basis. By removing needless duplication while maintaining data availability, storage resources are optimized. By distributing data references among several nodes, DRDDEA additionally facilitates fault tolerance. Decentralized validation procedures and cryptographic hashing improve security. The approach eliminates network overhead brought on by recurrent data storage, lowers storage costs, and increases retrieval efficiency. It is especially helpful in distributed databases, blockchain systems, cloud storage platforms, and large-scale data-sharing settings where effective data management and integrity preservation are essential.

- **Content-Aware Data Placement**

An sophisticated data management system called Content-Aware Data Placement chooses the best place to store data based on its relevance, access patterns, and content characteristics. In order to extract pertinent parameters such file type, size, frequency of access, sensitivity level, and usage needs, the algorithm first analyzes incoming data. The data is divided into several categories based on these attributes. By taking into account variables including storage capacity, computing power, network latency, dependability, and present workload, the algorithm assesses available storage nodes or servers. Next, a placement decision is made to store the data on the best node that can offer effective resource usage and access. To minimize retrieval delays, frequently requested or high-priority data may be stored on high-performance storage systems or closer to users. Costs can be optimized by allocating less often used data to less expensive storage resources. To detect shifts in usage behavior, the algorithm continuously tracks data access patterns and system performance. Data can be dynamically moved to more suitable places if needed without compromising availability. Replication techniques can also be used to increase accessibility and guarantee fault tolerance. To guarantee that sensitive data is kept in secure settings, security and privacy concerns are taken into account during placement. To enable effective search and retrieval processes, metadata related to the data is kept up to date. To avoid resource bottlenecks, the technique distributes storage usage among several nodes. The algorithm increases overall system efficiency by taking into account the context and content of data instead of employing random placement techniques. In distributed systems, cloud computing platforms, big data environments, and blockchain-based storage networks, content-aware data placement lowers access latency, improves scalability, optimizes storage resource allocation, and enables dependable data management.

7. SYSTEM ARCHITECTURE

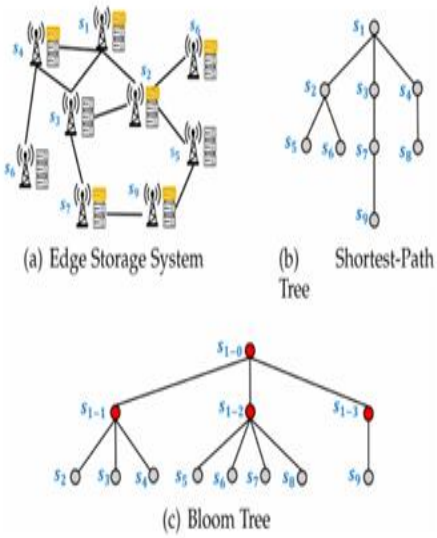


FIG:7 System Architecture

The system is made up of several networked edge servers set up in a Mobile Edge Computing (MEC) setting. Every edge server shares storage information with nearby servers and saves data locally. To arrange edge servers according to communication latency and network closeness, a shortest-path tree is built. In order to find the closest servers for data retrieval, the root node keeps track of connectivity statistics. On top of the shortest-path tree, a Bloom Tree structure is constructed to facilitate effective duplicate detection. Every node in the Bloom Tree has Bloom filters that provide an overview of the information kept in the relevant edge server and its offspring. The local edge

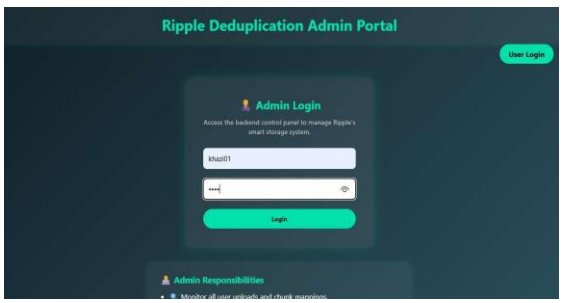
server first determines whether the same data already exists in neighboring servers by consulting the Bloom Tree when new data is received. Redundant storage is removed and only references to the current data are kept if duplicate data is found within the allowed latency range. The data is saved locally and the Bloom Tree is updated if no duplicates are discovered. By enabling decentralized decision-making, the framework does away with the requirement for a centralized controller. To keep data available and update indexes, nearby servers work together. In order to reduce response times, the architecture also makes sure that requests for data retrieval are fulfilled by the nearest server. The system achieves effective storage utilization, decreased network traffic, low retrieval latency, and high scalability in MEC contexts by combining edge servers, shortest-path routing, and Bloom Tree indexing.

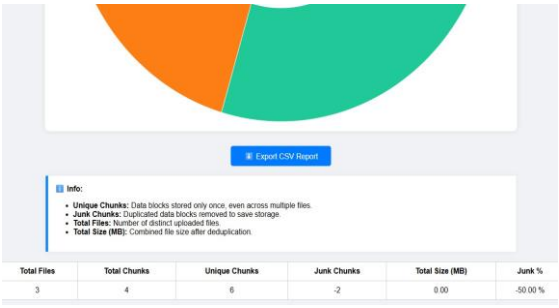
8. RESULTS











9. CONCLUSION

By successfully addressing the issues of distributed data management, low latency, and storage efficiency, the Ripple framework marks a major breakthrough in decentralized data deduplication for edge storage systems. The framework makes it possible to efficiently index and identify duplicate data across several adjacent edge servers without depending on centralized management by introducing the novel Ripple tree data structure. This decentralized method preserves quick access to stored material while enhancing scalability, fault tolerance, and system stability. The system can adjust to shifting storage requirements and maximize memory efficiency thanks to the dynamic tree growth and contraction techniques. Ripple effectively strikes a balance between data availability and deduplication efficiency, guaranteeing that users may obtain data fast even in highly dispersed settings. When compared to current approaches, experimental evaluations show significant gains in retrieval performance and storage optimization. The user experience is improved and real-time applications implemented in Multi-access Edge Computing (MEC) settings are supported by the notable decrease in data retrieval latency. The enhanced deduplication ratio also lowers infrastructure expenses and the amount of redundant storage used. By reducing pointless data transfers, the framework also helps to improve network resource use. It is appropriate for large-scale edge computing deployments where data volumes continue to increase quickly due to its decentralized design. All things considered, Ripple offers a practical, scalable, and resource-efficient solution for contemporary edge storage systems, making it a good starting point for further advancements in edge computing, distributed storage, and intelligent data management technologies.

10. FUTURE ENHANCEMENT

Future improvements to the Ripple framework may concentrate on enhancing the system's overall performance, scalability, and adaptability in more dynamic edge computing contexts. The creation of sophisticated communication protocols that reduce coordination overhead between edge servers while preserving effective duplicate detection and low-latency data access is one crucial path. To minimize

pointless metadata exchanges and maximize network bandwidth consumption, intelligent synchronization techniques can be implemented. System reliability can be further improved by adaptive algorithms that can react to shifting server density, shifting workloads, and shifting network circumstances. Data accessibility and service continuity can be enhanced by support for mobile edge environments, where users and devices move around a lot. For large-scale deployments, hybrid data indexing architectures that integrate hierarchical and decentralized indexing techniques may offer improved scalability. Predictive deduplication, which enables the system to detect and stop redundant data storage before duplication starts, can be achieved by integrating machine learning and artificial intelligence algorithms. To more efficiently distribute computational and storage burdens among edge nodes, dynamic resource allocation techniques can be used. Trust and data integrity can be strengthened by enhanced security features like cryptographic verification and blockchain-based metadata management. In order to lower power consumption in edge devices with limited resources, energy-aware optimization strategies can also be investigated. Data management across several computing layers can be enhanced by integration with cloud-edge collaborative architectures. Additionally, interoperability with new MEC standards and support for diverse edge infrastructures can boost practical adoption. With these improvements, Ripple will be able to provide future edge computing and distributed storage applications with increased efficiency, scalability, responsiveness, and dependability.

REFERENCES

[1] W. Shi, G. Pallis, and Z. Xu, "Edge computing," Proc. IEEE, vol. 107, no. 8, pp. 1474–1481, Aug. 2019.

[2] AWS, "AWS Wavelength for media & entertainment," 2021. [Online]. Available: https://d1.awsstatic.com/Wavelength2020/AWS-Wavelength for-Media-Entertainment-SolutionBrief-Feb2021-Final.pdf

[3] G. Zhu et al., "Pushing AI to wireless network edge: An overview on integrated sensing, communication, and computation towards 6G," Sci. China Inf. Sci., vol. 66, no. 3, 2023, Art. no. 130301.

[4] Q. He, Z. Dong, F. Chen, S. Deng, W. Liang, and Y. Yang, "Pyramid: Enabling hierarchical neural networks with edge computing," in Proc. ACMWebConf., 2022, pp. 1860–1870.

[5] H. Jin, R. Luo, Q. He, S. Wu, Z. Zeng, and X. Xia, "Cost-effective data placement in edge storage systems with erasure code," IEEE Trans. Serv. Comput., vol. 16, no. 2, pp. 1039–1050, Mar./Apr. 2023.

[6] W. Xiao, Y. Hao, J. Liang, L. Hu, S. A. Alqahtani, and M. Chen, "Adaptive compression offloading and resource allocation for edge vision computing," IEEE Trans. Cogn. Commun. Netw., to be published, doi: 10.1109/TCCN.2024.3400820.

[7] J. Peng et al., "MagNet: Cooperative edge caching by automatic content congregating," in Proc. ACM Web Conf., 2022, pp. 3280–3288.

[8] L. Yuan et al., "CSEdge: Enabling collaborative edge storage for multi accessedgecomputingbasedonblockchain,"IEEETrans .ParallelDistrib. Syst., vol. 33, no. 8, pp. 1873–1887, Aug. 2022.

[9] R. Li, Z. Zhou, X. Zhang, and X. Chen, "Joint application placement and request routing optimization for dynamic edge computing service management," IEEE Trans. Parallel Distrib. Syst., vol. 33, no. 12, pp. 4581–4596, Dec. 2022.

[10] X. Xia, F. Chen, Q. He, J. Grundy, M. Abdelrazek, and H. Jin, "Online collaborative data caching in edge computing," IEEE Trans. Parallel Distrib. Syst., vol. 32, no. 2, pp. 281–294, Feb. 2021.

[11] J. Zhou, F. Chen, G. Cui, Y. Xiang, and Q. He, "FEUAGame: Fairness aware edge user allocation for app vendors," IEEE Trans. Parallel Distrib. Syst., vol. 35, no. 8, pp. 1429–1443, Aug. 2024.

[12] X. Ge, S. Tu, G. Mao, C.-X. Wang, and T. Han, "5G ultra-dense cellular networks," IEEE Wireless Commun., vol. 23, no. 1, pp. 72–79, Feb. 2016.

[13] M. Sihaget al., "A data-driven approach for finding requirements relevant feedback from TikTok and YouTube," in Proc. IEEE Int. Requirements Eng. Conf., 2023, pp. 111–122.

[14] J. Yang, A. Sabnis, D. S. Berger, K. Rashmi, and R. K. Sitaraman, "C2DN: How to harness erasure codes at the edge for efficient content delivery," in Proc. USENIX Symp. Netw. Syst. Des. Implementation, 2022, pp. 1159–1177.

[15] Z. Liu et al., "DistCache: Provable load balancing for large-scale storage systems with distributed caching," in Proc. USENIX Conf. File Storage Technol., 2019, pp. 143–157.

[16] X. Zou, W. Xia, P. Shilane, H. Zhang, and X. Wang, "Building a high performance fine-grained deduplication framework for backup storage with high deduplication ratio," in Proc. USENIX Annu. Tech. Conf., 2022, pp. 19–36.

[17] Y. Shin, D. Koo, and J. Hur, "A survey of secure data deduplication schemes for cloud storage systems," ACM Comput. Surv., vol. 49, no. 4, pp. 1–38, 2017.

[18] X. Zou, J. Yuan, P. Shilane, W. Xia, H. Zhang, and X. Wang, "The dilemma between deduplication and locality: Can both be achieved?," in Proc. USENIX Conf. File Storage Technol., 2021, pp. 171–185.

[19] Z. Cao, S. Liu, F. Wu, G. Wang, B. Li, and D. H. Du, "Sliding look back window assisted data chunk rewriting for improving deduplication restore performance," in Proc. USENIX Conf. File Storage Technol., 2019, pp. 129–142.

[20] W. Xia et al., "A comprehensive study of the past, present, and future of data deduplication," Proc. IEEE, vol. 104, no. 9, pp. 1681–1710, Sep. 2016.

[21] W. Y. B. Limetal., "Decentralized edge intelligence: A dynamic resource allocation framework for hierarchical federated learning," IEEE Trans. Parallel Distrib. Syst., vol. 33, no. 3, pp. 536–550, Mar. 2022. [22] R. Kübler, "How to build popularity-based recommenders with polars," 2023.
[23] F.NiandS.Jiang,"RapidCDC:Leveragingduplicatelocalitytoaccelerate chunking in CDC-based deduplication systems," in Proc. ACM Symp. Cloud Comput., 2019, pp. 220–232.
[24] R. Kisous, A. Kolikant, A. Duggal, S. Sheinvald, and G. Yadgar, "The what, the from, and the to: The migration games in deduplicated systems," ACMTrans. Storage, vol. 18, no. 4, pp. 1–29, 2022.
[25] R. Luo, H. Jin, Q. He, S. Wu, and X. Xia, "Enabling balanced data deduplication in mobile edge computing," IEEE Trans. Parallel Distrib. Syst., vol. 34, no. 5, pp. 1420–1431, May 2023.
[26] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. S. Shen, "Content popularity prediction towards location-aware mobile edge caching," IEEE Trans. Multimedia, vol. 21, no. 4, pp. 915–929, Apr. 2019.
[27] J. Gao, S. Zhang, L. Zhao, and X. S. Shen, "The design of dynamic probabilistic caching with time-varying content popularity," IEEE Trans. Mobile Comput., vol. 20, no. 4, pp. 1672–1684, Apr. 2021.
[28] Y. Li, C. Tian, F. Guo, C. Li, and Y. Xu, "ElasticBF: Elastic Bloom filter with hotness awareness for boosting read performance in large key-value stores," in Proc. USENIX Annu. Tech. Conf., 2019, pp. 739–752.
[29] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: A scalable wide-area web cache sharing protocol," IEEE/ACM Trans. Netw.,vol.8, no. 3, pp. 281–293, Jun. 2000.
[30] A. Pagh, R. Pagh, and S. S. Rao, "An optimal Bloom filter replacement," in Proc. Annu. ACM-SIAM Symp. Discrete Algorithms, 2005, pp. 823–829.
[31] N. Dayan, I. Bercea, P. Reviriego, and R. Pagh, "InfiniFilter: Expanding filters to infinity and beyond," in Proc. ACM Int. Conf. Manage. Data, 2023, pp. 1–27.
[32] P. Pandey, A. Conway, J. Durie, M. A. Bender, M. Farach-Colton, and R. Johnson, "Vector quotient filters: Overcoming the time/space trade-off in f ilterdesign,"inProc.ACMInt.Conf.Manage.Data,2021 ,pp.1386–1399.
[33] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher, "Cuckoo filter: Practically better than Bloom," in Proc. ACMInt. Conf. Emerg. Netw. Experiments Technol., 2014, pp. 75–88.
[34] F. Liu, G. Tang, Y. Li, Z. Cai, X. Zhang, and T. Zhou, "A survey on edge computing systems and tools,"Proc.IEEE,vol.107,no.8,pp.1537–1562, Aug. 2019.
[35] S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, "Delay aware microservice coordination in mobile edge computing: A reinforce ment learning approach," IEEE Trans. Mobile Comput., vol. 20, no. 3, pp. 939–951, Mar. 2021.
[36] G. Cheng, D. Guo, L. Luo, J. Xia, and S. Gu, "LOFS: A light weight online file storage strategy for effective data deduplication at network edge," IEEE