Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Hybrid Random Forest and Differential Equation Control for Fragmentation Reduction in De-duplication Storage Systems

Mudasiru Hammed

mudasiru.hammed@federalpolyilaro.edu.ng

The Federal Polytechnic Ilaro https://orcid.org/0000-0001-6949-4852

Friday Thomas Ibharalu

Federal University of Agriculture, Abeokuta

Adio Taofiki Akinwale

Federal University of Agriculture< Abeokuta

Junoke Soyemi

Federal Polytechic Ilaro

Research Article

Keywords: Control System, De-duplication, Differential Equation, Lyapunov stability, Random Forest, Storage Space

Posted Date: August 26th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10784870/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Hybrid Random Forest and Differential Equation Control for Fragmentation Reduction in De-duplication Storage Systems

Mudasiru Hammed[1,2]*, Friday T. Ibharalu[3], Adio T. Akinwale[3] & Jumoke Soyemi[1,2]

[1]Department of Computer Science, Federal Polytechnic Ilaro, Nigeria
[2]Federal University of Technology, Ilaro,
[3]Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria
*e-mail: mudasiru.hammed@federalpolyilaro.edu.ng

Abstract

Efficiency of storage system can be incremented through de-duplication method, but it introduces many challenges. Fragmentation is one of the most serious problems in de-duplicated storage systems, which causes latency, storage inefficiency, and poor performance. This study employed a random forest along with a differential equation (RF-DE) based controller for managing fragmentation predictively and compare its performance with No Fragmentation Control (NC) and Storage Fragmentation-Informed Garbage Collection (SFI-GC) approaches using a storage Input and Output (I/O) trace dataset. The prediction part is implemented by a random forest model, whereas a differential-equation-based controller dynamically tunes the intensity of garbage collection operations. Experimental results show that the RF-DE technique performs better compared to NC and SFI-GC methods throughout the experiments. The proposed method decreases the fragmentation ratio from 0.38 and 0.33 (for NC and SFI-GC, respectively) to 0.29, reduces the cumulative latency by 22% to 25% and 15% to 18% for NC and SFI-GC methods respectively, and increases the storage health index by 40% and 75%. These results indicate that the utilization of Random Forest and Differential Equations in this study, provides an efficient storage system by means of an effective fragmentation control.

Keywords*:* Control System; De-duplication; Differential Equation; Lyapunov stability; Random Forest; Storage Space.

1 Introduction

Data de-duplication has been extensively employed in the area of backup solutions for storage space (Lin et al. 2015; Ma et al. 2017; Sun et al. 2016; Zhu et al. 2012)**.** This process breaks the incoming data stream into small, variable-sized fragments named chunks. Variable-sized algorithms use different techniques to measure the chunk length (Jie et al. 2019; Kaur et al. 2022; Xia et al. 2016), which are recognized by their short-length signatures called fingerprints. If chunks have the same fingerprint, they are regarded as duplicate and a fingerprint index is developed to map all fingerprints to their place in physical spaces that allows the index to find duplicated chunks (Xia et al. 2016). After the detection of duplicate chunks, all the legitimate chunks are kept in the containers which is the fundamental storage read and write operations. De-duplication process removes identical duplicates of a chunk while each de-duplicated data is replaced by a pointer references the address of an existing chunk. Every backup maintains its own metadata index that contains the identifiers of the containers which hold its chunks. If expired backups are removed by the users, the chunks will not be referenced by any backup; that is, the chunks become invalid and they must be recovered (Xia et al. 2016; Lin et al. 2021). Literature revealed that data de-duplication is a powerful tool for reducing the size of backup workloads. However, with the increasing and changing nature of the data streams, de-duplication system has the demerit of creating fragmentation challenge, leading to deficient restore and garbage collection (GC) performance (Tan et al. 2018; Tan et al. 2024). Without de-duplication systems, files are normally stored with high locality, that is, their blocks are positioned near each other. However, in the presence of de-duplication systems, consecutively stored blocks may be physically distant from each other, particularly when one of the blocks is a duplicate. This occurs because a duplicate block is referenced from an existing location, while a non-duplicate block is written to the next available free space. This phenomenon is referred to as fragmentation (Zhang et al. 2021).

The fragmentation is a major problem, especially when de-duplicated data are read from High Disk Drive systems, as these systems offer a relatively small number of non-sequential read Input and Output operations (I/O) per second. Literature revealed that one of the possible techniques to improve GC and eliminate fragmentation includes reference counting and mark-and-sweep (Lin et al. 2021). Although the current study proposed rewriting of de-duplicated data (Jehlol and George 2022) or storing data in cache memory such as Solid-State Drive (SSD) or High Disk Drive (HDD) (Fu et al. 2016; Park et al. 2017; Kaczmarczyk and Dubnicki, 2015), Data could be stored on a disk in multiple

localities (Lin et al. 2016; Dagnaw et al. 2020) to maintain and exchange space for data locality, defragment data, improve the restore and GC performance (Allu et al. 2018).
Fragmentation, however, keeps deteriorating the restore and GC performance (Tan et al. 2024; Cao et al. 2019). The rewritten techniques utilize the universally acceptable strategy, which involves using the existing copy of the block as the block to be rewritten. This indicates that most rewriting techniques (container process rewriting techniques) have the capability to preserve only a part of the total stream during the backup process (Zou et al. 2022). Jehlol and George (2022) developed the first history-aware rewriting technique to decrease fragmentation rewriting method, the results indicate that the method loses almost half of the de-duplication ratio in favor of restoring performance (Zou et al. 2022). Therefore, there is need for an improved technique that will avoid unnecessary rewriting, eliminate fragmentation and yield good restore and better GC performance. This study proposed a Random forest augmented with differential equation model for controlling fragmentation dynamics, reducing restore latency and maintenance cost. This achieves better performance improvements in de-duplicated distributed storage systems. In inline data de-duplication systems, substantial storage savings are achieved; still, inline de-duplication systems may cause chunk fragmentation, which affects restore performance. Traditional methods depend on periodic defragmentation, heuristic locality constraints, or static clustering strategies. These methods are typically reactive, rather than predictive, and may become unstable for dynamic workloads. They may also have high computational costs. As motivation, this study proposed the direct coupling of a Random Forest predictor with a differential equation control model to overcome the issues encountered by the traditional methods. The basic concept is to leverage machine learning for predicting fragmentation and restore risk; these predictions are directly injected into the system's differential equations and ensure convergence using control-theoretical stability analysis.
The paper helps to extend knowledge in the field with its proposed predictive control mechanism that uses RF predictions along with DE model to make forecasts about, control, restore latency and garbage collection dynamics in distributed storage systems. Unlike the traditional approaches of reactive nature and heuristics-based, the paper offers a proactive approach based on RF-DE that allows for making predictions regarding fragmentations, providing high levels of reliability of the system and minimizing fragmentations to near zero.

2 Related Works

The research gaps and opportunities in the recent literature on data de-duplication and compression systems include various aspects, such as locality-aware indexing, chunking optimization, and storage-aware indexing. Although there have been improvements in certain aspects of performance, there are still challenges in achieving a balance between the de-duplication ratio, data locality, and restore performance, especially in inline and locality-optimized de-duplication systems. There are various studies that aim to improve performance and compression ratios by optimizing data locality using inline de-duplication and data rewriting approaches. Some studies integrate delta compression with inline de-duplication to improve storage efficiency and locality. Studies by Yang et al. (2021) and Zhang et al. (2023) use two-layer locality tracking and cache-aware filtering to minimize additional backup and restore I/O operations, with inverted delta compression enhancing chunk compressibility. According to studies by Zhang et al. (2024),, optimizations entail extra CPU and memory overhead because of the lack of efficient garbage collection mapping in large-scale environments, potentially degrade compression efficiency by skipping highly compressible base chunks, and increase tracking complexity in restore and layout operations.
Management-friendly de-duplication framework (MFDedup) that was proposed by Cao et al. (2019) employs Across-Version-Aware Reorganization (AVAR) for the minimization of random I/O access during the restore process. Although the AVAR method optimizes data locality, it also results in high computational overhead. Moreover, the two-version constraint of the AVAR method limits the redundancy detection capability of the system for long periods of time. Similarly, the MFDedup method balances the restore speed with the ratio of de-duplication through workload-aware rewriting modes. However, the improper selection of the model and the amount of rewriting may result in the reduction of the efficiency of the system and the proliferation (Tan et al. 2024). Kiran et al. (2012) introduced an inline method called iDedup, which leverages spatial and temporal locality to reduce metadata overhead by using an in-memory cache. While locality-aware strategies significantly enhance restore speed and I/O efficiency, they rely on selective de-duplication, offline reorganization, or limited version awareness. These dependencies can severely limit the achievable de-duplication ratio and result in high computational and memory overhead during reorganization or garbage collection processes
The metadata-guided restore mechanism (MGRM) employs the method of adaptive mode switching that changes between the best state and the aggressive state to ensure that de-duplication and restore operations have better efficiency and fragmentation handling in inline de-duplication systems (Zhang et al. 2022). This approach enables aggressive redundancy reduction when needed while being conscious of restoration, thus resolving the conflict between storage space savings and restoration delay. But the approach may cause instability, lack fine-grained control,

performance fluctuations, and extra operational complexity, thus failing to achieve an optimal balance between de-duplication ratio and restoration speed in the system (Zhang et al. 2022).

A novel approach to an "inline lightweight locality-based chunking and hashing" has been proposed by Ahmed and George (2022). This approach relies on the “detection of frequent patterns" to establish chunk boundaries and incorporates a "lightweight hashing scheme” (LT-LH) to minimize CPU and memory usage, thus facilitating inline de-duplication. However, it is largely dependent upon repetitive patterns, lacks generalization ability, it is also exposed to hash collision danger because of the lightweight hash function, and cannot identify proper breakpoints within non-repetitive data Ahmed and George (2022). Study (Zhang et al. 2021b) introduced the Hot Index Distillation (HID) approach to address the performance bottlenecks and fragmentation associated with index lookups in de-duplication systems using disk-based indexes. Hot Index Distillation classifies the obtained fingerprint information into hot, fragmented, and useless categories to minimize the index size while maximizing the locality of the containers and the backup performance. Although HID reduces the index size, the use of rewriting techniques may lead to additional I/O operations, memory usage, and write amplification (Chakraborttii and Litz, 2021). Moreover, the maintenance of the fingerprint information may lead to metadata limitations (Zhang et al. 2021).

Fan and Song, 2019 proposed RapidCDC, an inline locality-aware chunking optimization method whose performance scales linearly with the de-duplication ratio and is efficient even in low duplicate locality scenarios, supporting high throughput for highly redundant files. Nevertheless, it suffers from higher memory consumption for locality detection, has dataset-dependent performance (Cao et al. 2019; Xu and Zhang, 2021), possibility of errors in chunk boundaries, and slight degradation of the de-duplication ratio in low locality workloads.

InDe was proposed by Lin et al. (2021), the technique employs F-greedy rewriting to boost restore capacity, with the Valid Container Referenced Counts (VCRC) technique being able to effectively locate the appropriate container for rewriting. Although the technique has the benefit of maintaining high backup performance with improved data locality, nevertheless, the technique has scalability challenges in very large-scale systems, with the possibility of write amplification, which affects the lifespan of the SSDs that were used. The technique also has metadata management challenges, with real-time VCRC maintenance being difficult (Lin et al. 2021). The Classified-Metadata based Restoring technique (CRM) that was proposed in (Yang et al. 2021) provides high restoring efficiency, but introduces scalability issues, which relate to memory usage, inefficient prefetching mechanism, inability to adjust to varying workloads, and limited experimental setup restrict the practicality of using CMR in current large-scale and distributed de-duplication systems (Hamandawana et al. 2024). Furthermore, the absence of any theoretical proof and insignificant improvements in de-duplication ratio indicate that more refinements are required for CMR.

Li et al. (2024) proposed Partial Repetition and Coding for minimizing data fragmentation. The method exhibits significant theoretical strength based on graph theoretical models, new stretch and jump measures, and connections with bandwidth for accurate fragmentation analyses and optimizations, but suffers from computationally intensive operations, the assumption of ideal network topologies, difficulty in scaling the computations for metrics, the inability to adjust to workloads, overheads in redundancy measures, and insufficient large-scale empirical testing (Palani et al. 2024; Ubaidillah and Ahmad, 2020).

Additionally, locality-optimal inline de-duplication methods improve restore speed by minimizing random I/O and fragmentation (Lin et al. 2021; Cao et al. 2019),, yet such improvements are normally achieved at the expense of metadata management, rewrite overhead, and sensitivity to workload, thus impeding scalability in large-scale or highly repetitive storage environments (Tan et al. 2024; Chakraborttii and Litz, 2021; Kasu et al. 2025; Zhang et al. 2021).

3 Methodology

This study proposed hybrid method that combines a Random Forest (RF) prediction and a nonlinear dynamic controller based on Differential Equations (DE) to enable proactive storage fragmentation management. This hybrid method works in close collaboration to offer predictive, adaptive, and stable control of fragmentation and restore performance for the de-duplicated storage system. The Random Forest model predicts the future storage fragmentation based on the past storage performance, it examines chunk-level attributes such as reference count, age, access frequency, and locality distance to predict the probability of fragmentation. The model also predicts the fragmentation risk for each chunk and the corresponding restore latency using these attributes. The Random Forest (RF) algorithm detects patterns that cannot be detected using traditional heuristic methods due to ability of the model to learn non-linear dependencies present in the system traces. It is a proactive model that can predict fragmentation before it becomes detrimental to the data recovery process. The Differential Equation (DE) controller constitutes the dynamic control component of the proposed system and it models the temporal behavior of the system variables such as fragmentation degree, chunk locality and restore latency. These variables are described using the differential equations reflecting their mutual dynamics depending on workload parameters and optimization criteria. The differential

equation controller uses the fragmentation predictions made by random forest as input data for adjusting the behavior of the storage. The corrective actions such as adaptive chunking, garbage collection, and data migration are performed by using a closed-loop control law.

A Lyapunov function is used to establish the feedback control policy, this ensures the stability of the system. The Lyapunov function always decreases with time, which shows that the fragmentation dynamics of the closed-loop system is bounded and asymptotically stable. The combination of Random Forest prediction augmented with differential equation-based control offers an efficient feedback-based solution for controlling storage fragmentation.

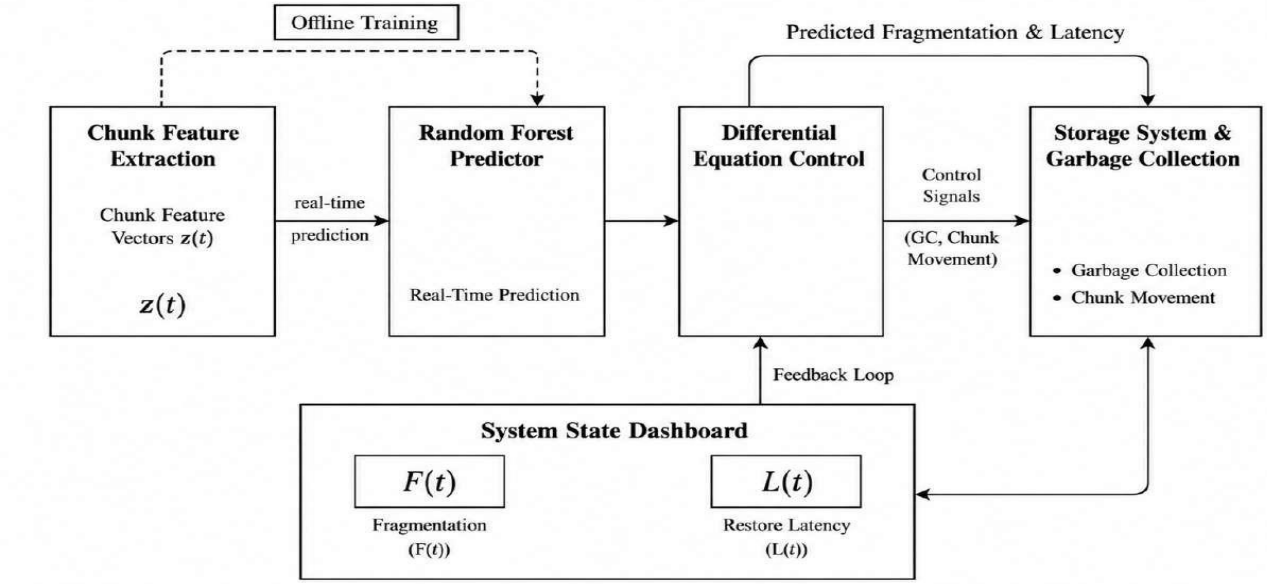


Fig. 1 Data Fragmentation Control System Architecture

3.1 Random Forest-Based Fragmentation Predictor

The Random Forest-based prediction model is used for predicting future storage fragmentation and restore latency based on historical storage performance observations. The capable of learn nonlinear relations between chunk properties and system dynamics and therefore enable effective proactive management of fragmentation prior to any degradation that may affect the restore performance. The normalized training dataset is modelled in equation (1)

$$D = \{\bar{x}_i, y_i\}\sum_{i=1}^{N}, \tag{1}$$

The input feature vectors in equation (2) describes input and output variables for Random Forest algorithm

$$\bar{x}_i = [\bar{r}, \bar{a}_i, \bar{l}_i, \bar{f}_i, \bar{s}_i]^T \tag{2}$$

where $\bar{r}_i$ is defined as **n**ormalized reference count of i^{th} chunk, $\bar{a}_i$ defines normalized chunk age, $\bar{l}_i$ is defined as the normalized locality distance, $\bar{f}_i$ is defined as the normalized access frequency, and $\bar{s}_i$ is expressed as normalized storage-tier indicator.

The output vector in equation (3) contains the target variables related to the i^{th} training sample which determine the storage system's performance at that particular point in time and are taken as the actual output to train the Random Forest predictor.

$$y_i =]F_i, L_i]^T \tag{3}$$

where F_i represents the storage fragmentation ratio, and L_i refers to the actual restoration latency.

The equation (4) expressed the Random Forest predictor that is an aggregation of (K) regression trees, trained separately and whose prediction is made using the average output from all (K) regression trees.

$$\hat{y} = \frac{1}{K}\sum_{K=1}^{K} T_k(\bar{x}, \theta_k), \tag{4}$$

where $T_k(\cdot)$ refers to the k^{th} regression tree constructed from bootstrap and θ_k denotes the random variables controlling the construction of the tree.
Therefore, the aggregation of all predictions of the regression trees builds a strong prediction with low variance.

Every regression tree discovers the non-linear mapping between the past properties of the chunks and storage performance. Aggregation of the predictions of multiple separately trained regression trees allows the Random Forest to produce accurate predictions about future values of the fragmentation level and restore latency . The predicted state of the storage becomes the input to the differential equation-based controller. These two predictions in equation (5) expressed the expected future behavior of the storage system after the de-duplication process. In particular, the prediction of the future fragmentation level defines how much the chunks will be dispersed, while the prediction of the future restore latency describes the effect of dispersed chunk on the data restore performance. The predictions give the estimates of the expected future storage configuration needed for the controller to operate.

$$\hat{y}(t+\Delta t) = \begin{bmatrix} \hat{F}(t+\Delta t) \\ \hat{L}(t+\Delta t) \end{bmatrix}, \tag{5}$$

where $\hat{y}(t+\Delta t)$ refers to the predicted future storage vector condition at the prediction horizon, $\hat{F}(t+\Delta t)$ and $\hat{L}(t+\Delta t)$ define the predicted storage fragmentation rate and restore latency respectively, while t denotes current time of the system, and Δt defines prediction horizon. Mean squared error between the measured and predicted storage state vectors is minimized across all training samples as it is written in equation (6). This leads the random forest to learn the nonlinear mapping between chunk attributes and the storage system behavior. The optimal predictor then predicts the future state of the storage to the differential equation controller to enable fragmentation control and stable restore times.

$$L = \frac{1}{N}\sum_{i=1}^{N} \| y_i - \hat{y}_i \|_2^2, \tag{6}$$

3.2 Convergence and Predictive reliability of the Random Forest

The predictive reliability of the proposed Random Forest based fragmentation is proved by the convergence attribute of the Random Forests. Let assumed that $h(\bar{x}, \theta_k)$ is the prediction of the k^{th} regression tree with random variables of θ_k which representing bootstrap sampling and feature selection. The confidence in the prediction of the ensemble is measured using the margin function in equation (7).

$$mar(\%, y) = P_\theta(h(\%, \theta) = y) - \max_{j=1} P_\theta(\%, \theta) = j), \tag{7}$$

where a large margin corresponds to high confidence in predicting the right storage condition. The generalization error is usually converge to a finite value as the number of trees grows (Breiman, 2001), the Breiman's convergence theorem is mathematically represented in equation (8).

$$PE^* = P(mr(\%, y) < 0), \tag{8}$$

Therefore, this indicates that an increase in the number of ensembles enhances prediction stability without inducing over-fitting. In addition to prediction stability, the prediction reliability depends on the quality of the forest and the average correlation (ρ) of the regression trees which is mathematically represented in equation (9).

$$s = E[mr(\%, y)], \tag{9}$$

By applying Chebyshev's inequality, the upper bound of the generalization error is written as equation (10).

$$PE^* \le \frac{\rho(1-s^2)}{s^2}, \tag{10}$$

So therefore, the Random Forest convergence and reliability indicates that high fragmentation prediction accuracy can only be achieved through maximizing prediction strength and minimizing the correlation between individual trees. Bootstrap aggregating and feature randomization in the (RF) model decrease the correlation between individual trees and increase prediction strength. Therefore, the Random Forest can provide accurate predictions on fragmentation and restore latency, which are used by the differential-equation controller in controlling fragmentation.

3.3 Estimation of Out-of-Bag Prediction Error

Prediction accuracy of the Random Forest fragment predictor is estimated using Out-of-Bag (OOB) samples, which makes it unnecessary to have a dedicated validation sample set. In the process of bootstrap sampling, about one-third of the training samples are omitted from the regression tree and are used to evaluate its prediction accuracy. The OOB prediction of the i^{th} training sample is computed as the average of the predictions of all trees that omit this sample from their bootstrap sets. This is mathematically computed in equation (11).

$$\hat{y}_{OOB,i} = \frac{1}{S_i} \sum_{k \in S_i} T_k(x, \theta_k), \tag{11}$$

where S_i defines the collection of regression trees that omit the i^{th} sample while being trained using bootstrapping. More so, the out-of-bag prediction error within this context is then estimated in equation (12).

$$E_{OOB} = \frac{1}{N} \sum_{i=1}^{N} \| y_i - \hat{y}_{OOB,i} \|_2^2, \tag{12}$$

where N denotes total number of training data points, y_i defines actual storage condition vector, and $\hat{y}_{OOB,i}$ is known as the out-of-bag (OOB) estimate. Out-of-bag (OOB) error serves as an unbiased estimate of the generalization capability of the Random Forest and helps in validating the fragmentation predictor before its use. After performing the model verification step, the trained Random Forest predicts the future condition of the storage as it is written in equation (13).

$$x_{RF}(t + \Delta t) = \begin{bmatrix} \hat{F}(t + \Delta t) \\ \hat{L}(t + \Delta t) \end{bmatrix}, \tag{13}$$

The forecasted storage condition vector is then input to the differential equation controller for adaptive organization of the chunks. The Random Forest augmented with Differential Equation model, which predicts future fragmentation before it affects restore performance, and also helps to proactively minimizing fragmentation while keeping the restore latency consistent.

3.4. Fragmentation Controller Using Differential Equations

After fragmentations have been predicted using the random forest method, the future state of the storage is used in a first-order differential equation to control storage fragmentations. The storage state which is presented in equation (5) indicates that both fragmentation rate and restore latency have been predicted. Therefore, let the equation (14) represents the true storage state.

$$Y(t) = \begin{bmatrix} F(t) \\ L(t) \end{bmatrix}. \tag{14}$$

Applying the first-order non-homogeneous differential equation, the model for storage behavior is written in equation (15).

$$\frac{dY(t)}{dt} + BY(t) = K\hat{y}(t + \Delta t). \tag{15}$$

In this case, (B) is the storage fragmentation behavior matrix that describes the natural behavior of fragmentation and restoration delay, while (K) is the feedback gain matrix. The term at the right-hand side is the forcing function that comes from the Random Forest prediction model. The control input is defined in equation (16).

$$u(t) = K\hat{y}(t + \Delta t). \tag{16}$$

Using the control input from equation (16) in the storage dynamics from equation (15), the closed-loop fragmentation dynamics obtained in equation (17).

$$\frac{dY(t)}{dt} = -BY(t) + u(t), \tag{17}$$

where $Y(t)$ defines storage state vector. Equation (18) describes the dynamics fragmentation and restores latency over time, whereas the Random Forest prediction serves as the reference signal used to adaptively reorganize chunks in the storage based on the predicted future state of the system. **.** Hence, the closed-loop control action drives the storage system to its desired state, which implies low fragmentation and good restore latency with stable storage performance.

3.5 Prediction and Control of Fragmentation Stability Analysis

The Lyapunov's direct method is used to prove the stability of the proposed Random Forest (RF) augmented with Differential Equation (DE) fragmentation controller. The tracking error between the real storage states and the random forest predicted storage states is written in equation (18).

$$e(t) = Y(t) - \hat{y}(t + \Delta t), \tag{18}$$

The quadratic Lyapunov function in equation (19) is employed to analyze the stability of the (RF) and (DE) fragmentation controller.

$$V(e) = \frac{1}{2} e^T e, \tag{19}$$

where the superscript $(e)^T$ refers to the transpose operator, and ($e^T e$) denotes the squared Euclidean norm of the tracking error. It is shown in equation (20) that the Lyapunov function represents the energy $V(e)$ related to the tracking error, which is utilized to measure the difference between the true and predicted storage state. The scaling factor $(\frac{1}{2})$ is used for mathematical simplicity because its presence does not affect the stability results but makes the derivative calculation easier.

$$V(e) > 0, \quad \forall e \neq 0 \text{, and } V(0) = 0. \tag{20}$$

Therefore, the Lyapunov function is positive definite, it meets the first requirement of Lyapunov stability. Positive definiteness means that the Lyapunov function is always greater than zero if the current values of fragmentation and restore latency differ from those predicted by the Random Forest model. It reaches its minimum value only when the tracking error is zero. . However, the Lyapunov function is used to assess the deviation of the proposed system from its required operation state. But, the system proved that the injection of random forest into differential equations to deal with data fragmentation is globally asymptotically stable.

Furthermore, usage of random forest augmented with the differential equations system allows both accurate predictive control and stable behavior of the control system. In contrast to traditional methods that use static heuristics or periodic defragmentation, the proposed system uses a closed-loop approach to continuously predict fragmentation risk and take corrective measures, leading to better restore performance without compromising system throughput.

4 Experimental Setup

4.1 Experimental Data

Experiments were carried out with the help of a realistic storage I/O trace (websearch.csv), which includes timestamps, logical block offsets, operation type (either read, write or delete), size of request and latency. In the pre-processing stage, incomplete data points were discarded, numeric variables were converted into proper format and requests for storage were split into logical chunks of 8KB size each to create unique IDs for chunks. In the analysis of storage, twelve (12) features were extracted, including: fragmentation ratio, jump count, stretch, rewrite density, locality, sequential, access frequency, chunk age, chunk lifetime, access entropy, read-to-write ratio, and health index of storage. All numeric features were normalized using Min-Max Scaling.

4.1.1 Experimental Approaches

Two storage optimization techniques were studied and compared: Storage Fragmentation Index with Garbage Collection (SFI-GC), which employs partial repetition and coding technique introduced by Li et al. (2024), and the RF-DE method proposed in this study. The baseline for evaluation was conducted using two approaches of the SFI-GC method: No Control (NC) when no fragmentation control or garbage collection is applied, and SFI-GC where garbage collection is controlled using Storage Fragmentation Index. The same experimental setup was applied.

No Control (NC) (Red color): This baseline technique does not apply any fragmentation control or garbage collection optimization.
Storage Fragmentation Index with Garbage Collection (SFI-GC) (Blue color): This is a benchmarking technique that computes the weighted Storage Fragmentation Index (SFI) to determine the degree of garbage collection, whether low, medium, or high. The garbage collection level is the determinant used to tune fragmentation, latency, jump, and stretch.
Random Forest Dynamic Engine (RF-DE) (Purple colour): In this method, three different random forest models, which independently predict the storage fragmentation, latency, and health based on an optimized set of storage attributes, were trained. The hyper-parameters were optimized using the Randomized Search Cross-Validation technique with fivefold cross-validation based on the number of trees, maximum tree depth, minimum samples for splitting the nodes and leaf nodes, and feature selection methodology. The predicted storage state is then exploited by the adaptive controller to dynamically adjust garbage collection strength.

4.1.2 Training Procedure

The training of the Random Forest models was done using randomly selected 50,000 instances of storage request datasets. Splitting was done in the ratio of 80% to 20%, and the hyper-parameter tuning was done using Randomized Search CV with 5-folds. Best-performing models for each prediction problem were kept for further evaluation.

4.1.3 Performance Evaluation

The accuracy of prediction was measured using MAE, RMSE, R^2 score, and EVS. The performance of storage optimization was measured by calculating the average fragmentation, latency, storage health, jump count, stretch, and throughput for NC, SFI-GC, and RF-DE methods. Percentage improvement of RF-DE over SFI-GC was computed for each measure, and then the results were shown using comparative graphics and tables.

4.2 Results and Discussion

4.2.1 Controller Intensity

The controller intensity is compared between the new proposed Random Forest augmented with Differential Equation (RF-DE) controller and the Storage Fragmentation-Informed Garbage Collection (SFI-GC) approach using about 520,000 storage requests in Figure 2. The controller intensity denotes the degree of the action required to ensure that the performance of storage remains stable. It is also necessary to suppress fragmentation in the system. In an ideal situation, the controller should be able to sustain the same level of intensity even when the workloads change. The controller intensity result from figure 2 shows that the suggested RF-DE-based controller performs a smooth and adaptive control behavior in the course of the entire experiment. After an initialization phase, the intensity of the controller increases steadily from about 0.45 to 0.67 as the storage requests accumulate. This performance shows that the controller constantly adjusts to the anticipated fragmentation state with the proper amount of effort without sudden corrections. The steady increase in controller intensity proves the Random Forest algorithm's successful prediction of future fragmentation growth, combined with differential equations. On the other hand, the SFI-GC controller shows clearly reactive control. Even though it has a lower average controller intensity (around 0.20-0.39), there are many spikes in the range of 0.80 that occur through the entire process, especially during storage request in the latter part of the process. This means that there are multiple instances of emergency garbage collection after fragmentation occurs. This means that the SFI-GC controller needs to make many corrections in order for the storage process to stabilize again.

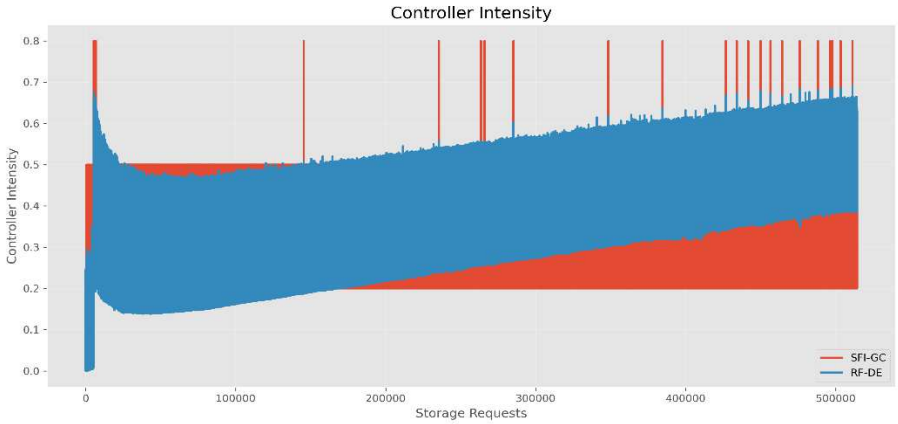


Fig. 2 Controller Intensity for RF-DE and SFI-GC

This comparison shows clear essential distinction that lies between these two methods. The SFI-GC technique functions mainly in a reactive mode, performing garbage collection only when the level of fragmentation surpasses certain thresholds. Hence, fragmentation builds up before taking any measures to correct the situation, leading to sporadic but intense control activations. On the other hand, the RF-DE technique relies on a predictive control strategy, whereby the Random Forest method predicts the development of fragmentation and the differential equation control adjusts the process of storage management.

4.2.2 Fragmentation Ratio

In figure 3, the fragmented level from the fragmentation results showed that there was a rapid decrease in the fragmented level as the number of storage requests was increased. The uncontrolled system (NC) had the largest fragmented level, and it stabilized at around 0.38. The fragmented level was decreased to around 0.33 by the SFI-GC method, but the method with the smallest fragmented level was RF-DE method with a fragmented level of about 0.29. This shows that the RF-DE method was better in reducing the fragmentation in storage allocation.

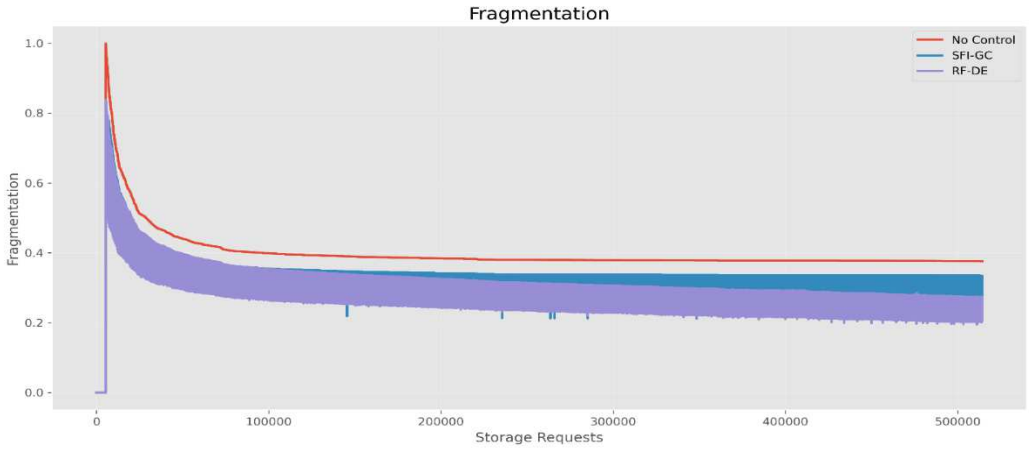


Fig. 3 Fragmentation Ratio

4.2.3 Storage Latency Analysis

The latency analysis in figure 4 shows a comparison of cumulative latency in NC, SFI-GC, and the RF-DE controller using almost 520,000 storage requests. The latency of all techniques increased linearly with the increase in the number of storage requests, but there were variations in their growth rate. No Control (NC) technique produced the maximum

cumulative latency of about 2300 latency units. This shows that without fragmentation control, there will be an increase in disk seeks and restore latency. The SFI-GC showed better cumulative latency values (2100 latency units; around 8% to 10% less than No Control technique) by selectively employing a garbage collection mechanism. However, being a reactive strategy, the latency increased when workload request continued. On the other hand, RF-DE achieved the lowest cumulative latency (1750 to 1800 latency units), which is around 22% to 25% and 15% to 18% lower than No Control and SFI-GC, respectively. The increasing distance between the curves RF-DE and SFI-GC highlights the high level of scalability of the proposed algorithm under growing workloads. However, the usage of machine learning augmented with dynamic control is more efficient for managing fragmentation than classical garbage collection algorithms.

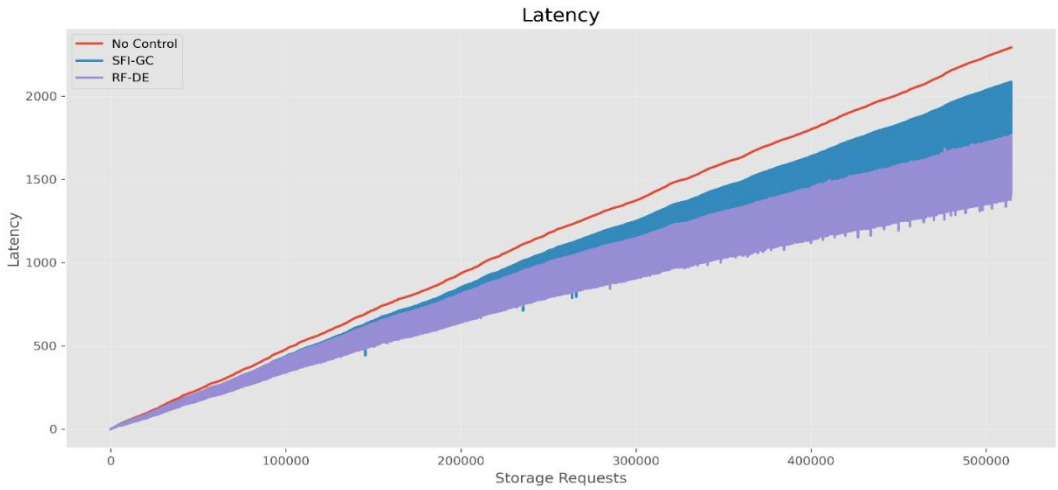


Fig. 4 Storage Latency

4.2.4 Storage Health

Moreover, the storage health in Figure 5 shows the efficiency of RF-DE in terms of storage control. In terms of the health value of the system, RF-DE achieved the highest value which was about 35, while the uncontrolled and SFI-GC systems recorded values of 25 and 20, respectively. From this high health value of the RF-DE, it can be inferred that RF-DE maintained a better state of storage during the experimentation process even as the number of requests increased.

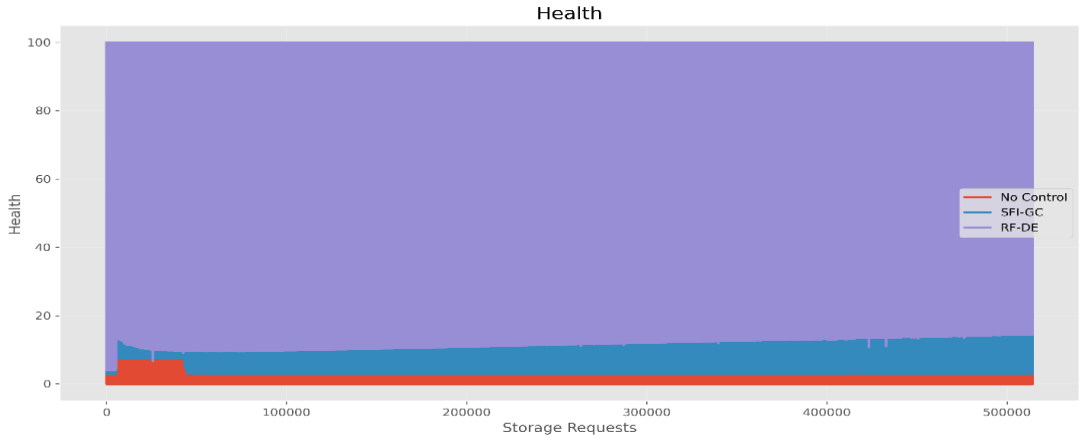


Fig. 5 Storage Health

4.2.5 System Jump Count

The findings from the jump-count metric in figure 6 were almost identical in all the approaches analyzed, being almost constant at one throughout the entire experiment. This implies that the enhancements in performance achieved through the use of RF-DE algorithm which were not accomplished by moving too much around or conducting many jumps. Also, the value for the stretch was fairly low with just some few variations.



Fig. 6 System Jump Count

4.2.6 Analysis of Stretch

The stretch analysis in figure 7 presents a comparison of the stretch behavior of NC, SFI-GC, and the suggested RF-DE controller during about 520,000 storage transactions. In general, stretch was low, with most values lying in the range of 0.30 to 0.45, this shows that the scheduling is performed steadily within normal working conditions. Spikes of 0.65–0.80 and a maximum value of about 1.0 were observed during the periods of higher storage conflict and fragmentation. In addition, all three techniques faced the same workload pattern, the RF-DE showed a smaller number of stretch spikes compared to other techniques (NC and SFI-GC). The predictive random forest algorithm augmented with a differential equations-based controller (RF-DE) prevents garbage collection and storage allocation before fragmentation becomes extreme and the slowdown of the requests became steady throughout the workload. The SFI-GC method reduced stretch through garbage collection and selective processing, but, due to the nature of the approach (threshold-based and reactive), it experienced more spikes than the RF-DE approach. On the other hand, the No-Control strategy showed high stretch variance since there was no fragmentation management, leading to increased seek time and scheduling delays. All the approaches ensured that stretch remained within bounds, the RF-DE performed better in terms of minimizing spikes.

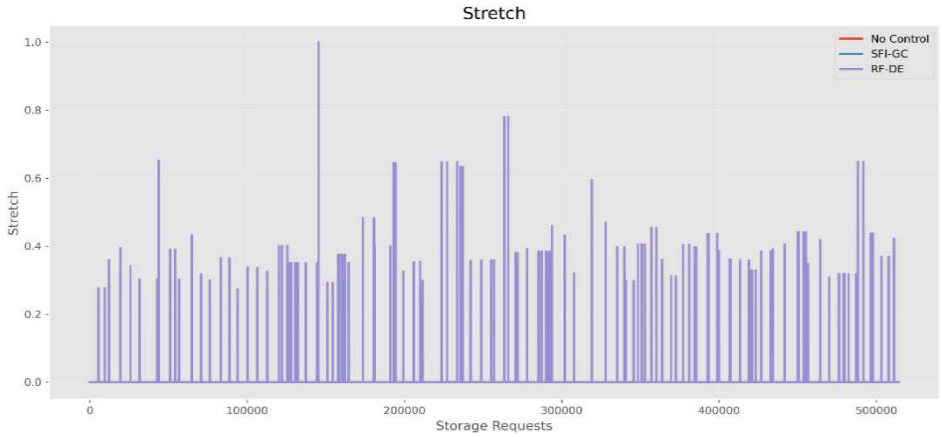


Fig. 7 Analysis of Stretch

4.2.7 Summary of Quantitative Comparison of NC, SFI-GC and RF-DE Results

Table 1 provides the comparative performance analysis of the No Control (NC), Storage Fragmentation Informed Garbage Collection (SFI-GC), and Random Forest-Differential Equation (RF-DE) methods with regard to controller intensity, fragmentation ratio, latency, storage fitness, jump count, and stretch values. The result clearly shows that the RF-DE technique performs better than the other two methods of measurement.

Table 1 Comparative Analysis of the Results

Performance Metrics	**No Control Technique**	**SFI-GC Technique**	**RF-DE Technique**	**RF-DE performance enhancement over NC**	**RF-DE performance enhancement over SFI-GC**
Controller Intensity	Nil	0.20–0.39 $(spikes \approx 0.80)$	0.45-0.67 (smooth adaptive)	Stable adaptive control	RF-DE alleviates large control spikes
Fragmentation Ratio	0.38	0.33	0.29	23.7% reduction	12.1% reduction
Storage Latency (units)	≈ 2300	≈ 2100	$\approx 1750-1800$	22-25% reduction	15-18% reduction
Storage Health Index	25	20	35	40% enhancement	75% enhancement
Jump Count	≈ 1.0	≈ 1.0	≈ 1.0	No significant variability	No significant variability
Stretch	0.35-0.45 with frequent spikes $(\max \approx 1.0)$	0.30-0.45 with moderate spikes	0.30-0.45 with lower spikes	Lower changes	RF-DE reduced spike frequency

4.2.8 Discussion

It can be deduced from table 1 that the Random Forest model enhanced by Differential Equations (RF-DE) outperforms No Control (NC) and Storage Fragmentation-Informed Garbage Collection (SFI-GC). The analysis of the controller intensity reveals that RF-DE was working with smooth increasing controller intensity ranging from 0.45 to 0.67, while SFI-GC demonstrated low controller intensity (0.20-0.39) but with periodic spikes up to 0.80. Those spikes suggest that there are frequent emergency garbage collections due to reactive fragmentation management that were taking place. As opposed to that, the intensity of RF-DE increased gradually, which proves the ability of predictive control to prevent fragmentation. As regards to fragmentation ratio, RF-DE demonstrated the best result

(0.29) among all the options in comparison to SFI-GC (0.33) and NC (0.38). This implies that 23.7% improvement compared with NC and 12.1% improvement compared with SFI-GC. The latency results further support the efficacy of RF-DE method. Although NC registered about 2,300 latency units, while the SFI-GC managed to bring it down to 2,100 units with its reactive approach to garbage collection. However, RF-DE had the least latency of (1750-1800 units), with value range between 22-25% improvement on NC and 15-18% improvement on SFI-GC. The increasing interval between the RF-DE and SFI-GC lines shows that the predictive approach has better scalability as the number of storage requests grows. The storage health index also significantly improved in the case of RF-DE and stood at 35, while NC managed 25 and SFI-GC only 20 points. This amounts to a 40% improvement on NC and a 75% improvement on SFI-GC. However, the jump count stayed roughly constant across all three approaches, which demonstrating that the performance improvement of RF-DE was not because of the higher amount of storage relocation or excessive data movements. But, this was achieved by better prediction and adaptively control. Finally, even though the stretch was in the acceptable operation range (0.30-0.45) for all the three methods, the RF-DE method had the lowest stretch value and also the minimum changes in stretch during contention. But, contrary to SFI-GC that suffered from more frequent moderate spikes because of its reactive threshold approach, while NC suffered the most variance because of uncontrolled fragmentation. The analysis shows that the RF-DE technique performed better than the NC technique and the SFI-GC technique in all of the important performance metrics. Using the Random Forest model along with the adaptive control method using differential equations, RF-DE managed to minimize fragmentation and latency, keep the system healthy, maintain low variability in the stretch, and do so without any additional overhead.

5 Conclusion and Future Works

5.1 Conclusion

In this study, the Random Forest model enhanced with differential equations (RF-DE) controller was employed for proactive fragmentation management in de-duplicated storage. With machine learning predictions combined with an adaptive dynamic controller, the approach successfully maintained the process of garbage collection based on predicted storage conditions. The experiments have proven that RF-DE controller has achieved the lowest ratio of fragmentation, reduced the total latency by up to 25%, increased the level of storage health by up to 75%, decreased the variability of stretch and kept the stability of controller intensity at the same time. Contrary to the reactive approach based on the SFI-GC algorithm, the RF-DE did not let fragmentation appear even when performance of storage system was still good. Thus, the proposed method showed higher scalability and responsiveness of the storage system in case of growing load. The experiments show that the method of predictive control outperforms the classic threshold-based garbage collection in terms of minimizing fragmentation and disk bottlenecking.

5.2 Future Work

The RF augmented with the DE model suffers from the dependence on high-quality training data, requires fixed control parameters, and computational burden. The RF and DE methods can be further enhanced using optimization techniques such as workload-aware chunk redistribution, adaptive control schemes (model predictive control and reinforcement learning, among others), and distributed system support. Other areas include the integration of multi-objective optimization (such as energy conservation, Solid State Drive (SSD) wear-leveling, and I/O overhead) and the use of hybrid ensemble and nonlinear adaptive controllers to improve convergence properties during workload changes. The application of the system to distribute or cloud-scale storage environments promotes intelligent, self-stabilizing de-duplication systems with low rewrite overheads while maintaining Lyapunov-based stability properties.

5.3 Statements and Declarations

Funding
Authors did not receive support from any organization for the submitted manuscript.

Financial or Non-financial Interests
Authors have no competing interests to declare that are relevant to the content of this manuscript.

Data Availability
The workload trace dataset files used in this study are publicly available in the trace repository of the University of Massachusetts. The trace files are as follows:
WebSearch Trace. https://skulddata.cs.umass.edu/traces/storage/WebSearch3.spc.bz2

Financial Trace. https://skulddata.cs.umass.edu/traces/storage/Financial2.spc.bz2

Consent
No consent to be declared, all figures and table in this manuscript are designed by the authors.

Disclosure and Statement of the Use of Artificial Intelligence Generated Content (AIGC)
In preparing this manuscript of "Hybrid Random Forest and Differential Equation Control for Fragmentation Reduction in De-duplication Storage Systems", Grammarly and ChatGPT were utilized by the authors only for language and grammatical improvements while preparing the Results and Discussion, and Related Works sections. The authors have critically examined the content after the utilization of these software and are wholly responsible for the publication of the content.

Authors' contribution statements
All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Mudasiru Hammed, Friday T. Ibharalu, Adio T. Akinwale, and Jumoke Soyemi. The first draft of the manuscript was written by Mudasiru Hammed, and all authors commented on previous versions of the manuscript. All authors reviewed and approved the final manuscript.

References

Ahmed ST, George LE (2022) Lightweight hash-based de-duplication system using the self-detection of most repeated patterns as chunks divisors. *Journal of King Saud University – Computer and Information Sciences* 34(7):4669–4678. https://doi.org/ 10.1016/j.jksuci.2021.04.005

Allu Y, Douglis F, Kamat M, Prabhakar R, Shilane P, Ugale R (2018) Can't we all get along? Redesigning protection storage for modern workloads. In: *Proceedings of the 2018 USENIX Annual Technical Conference (USENIX ATC '18)*, pp 705–718.

Breiman L (2001) Random forests. *Machine Learning* 45(1):5–32. https://doi.org/10.1023/A:1010933404324

Cao Z, Liu S, Wu F, Wang G, Li B, Du DH (2019) Sliding look-back window assisted data chunk rewriting for improving deduplication restore performance. In: *Proceedings of the 17th USENIX Conference on File and Storage Technologies (FAST '19)*, pp 129–142.

Chakraborttii C, Litz H (2021) Reducing write amplification in flash by death-time prediction of logical block addresses. In: Proceedings of the 14th ACM International Systems and Storage Conference (SYSTOR '21), pp 11:1–11:12. https://doi.org/10.1145/3456727.3463784

Dagnaw G, Hua W, Zhou K (2020) SSD assisted caching for restore optimization in distributed deduplication environment. In: *Proceedings of the IEEE International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS)*, pp 1–8. https://doi.org/10.1109/HPBDIS49115.2020.9130572

Fan N, Song J (2019) RapidCDC: Leveraging duplicate locality to accelerate chunking in CDC-based deduplication systems. In: *Proceedings of the ACM Symposium on Cloud Computing (SoCC '19)*, pp 220–232. https://doi.org/10.1145/3357223.3362731

Fu M, Feng D, Hua Y, He X, Chen Z, Liu J, Xia W, Huang F, Liu Q (2016) Reducing fragmentation for in-line deduplication backup storage via exploiting backup history and cache knowledge. *IEEE Transactions on Parallel and Distributed Systems* 27(3):855–868. https://doi.org/10.1109/TPDS.2015.2410781

Hamandawana P, Cho DJ, Chung TS (2024) Speed-Dedup: A new deduplication framework for enhanced performance and reduced overhead in scale-out storage. *Electronics* 13(22):4393. https://doi.org/10.3390/electronics13224393

Jehlol HB, George LE (2022) Big data backup de-duplication: A survey. *International Journal of Scientific Research in Science, Engineering and Technology*:174–191.

Wu J, Hua Y, Zuo P, Sun Y (2019) Improving restore performance in deduplication systems via a cost-efficient rewriting scheme. IEEE Transactions on Parallel and Distributed Systems 30(1):119–132. https://doi.org/10.1109/TPDS.2018.2852642

Kaczmarczyk M, Dubnicki C (2015) Reducing fragmentation impact with forward knowledge in backup systems with deduplication. In: Proceedings of the 8th ACM International Systems and Storage Conference (SYSTOR '15), pp 201–208. https://doi.org/10.1145/2757667.2757678

Kasu P, Hamandawana P, Chung TS (2025) Latency-aware deduplication for efficient object-based big data transfers. In: *Proceedings of the IEEE International Conference on High Performance Computing, Data, and Analytics (HiPC '25)*, pp 55–65. https://doi.org/10.1109/HiPC66333.2025.00015

Kaur R, Bhattacharya J, Chana I (2022) Deep CNN based online image deduplication technique for cloud storage system. *Multimedia Tools and Applications* 81(28):40793–40826. https://doi.org/10.1007/s11042-022-13182-7

Kiran S, Benson T, Ganger GR, Vaidyanathan K (2012) iDedup: Latency-aware inline data de-duplication for primary storage. In: *Proceedings of the 10th USENIX Conference on File and Storage Technologies (FAST '12)*.

Li YH, Sima J, Shomorony I, Milenkovic O (2024) Reducing data fragmentation in data deduplication systems via partial repetition and coding. *arXiv preprint* arXiv:2411.01407. https://doi.org/10.48550/arXiv.2411.01407. https://doi.org/10.48550/arXiv.2411.01407

Lin L, Deng Y, Zhou Y (2021) Improving restore performance of deduplication systems via a greedy rewriting scheme. In: *Proceedings of the IEEE 27th International Conference on Parallel and Distributed Systems (ICPADS '21)*, pp 291–298. https://doi.org/10.1109/ICPADS53394.2021.00042

Lin L, Xiao K, Liu W (2016) Utilizing SSD to alleviate chunk fragmentation in de-duplicated backup systems. In: *Proceedings of the IEEE International Conference on Parallel and Distributed Systems (ICPADS '16)*, pp 616–624.

Lin X, Douglis F, Li J, Li X, Ricci R, Smaldone S, Wallace G (2015) Metadata considered harmful to deduplication. In: *Proceedings of the 7th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage '15)*. https://doi.org/10.5555/2813749.2813760

Ma J, Stones RJ, Ma Y, Wang J, Ren J, Wang G, Liu X (2017) Lazy exact deduplication. *ACM Transactions on Storage.*13(2):1–26. https://doi.org/10.1145/3078837

Palani S, Ajay B, Bhavadharani B, Varshini A (2024) Enhanced security through distributed fragmentation and encryption for cloud-based data storage. In: *Proceedings of the International Conference on Emerging Research in Computational Science (ICERCS)*, pp 1–8. https://doi.org/10.1109/ICERCS63125.2024.10895601

Park D, Fan Z, Nam YJ, Du DH (2017) A look-ahead read cache: Improving read performance for de-duplication backup storage. *Journal of Computer Science and Technology* 32(1):26–40.

Sun Z, Kuenning G, Mandal S, Shilane P, Tarasov V, Xiao N (2016) A long-term user-centric analysis of deduplication patterns. In: *Proceedings of the 32nd IEEE Symposium on Mass Storage Systems and Technologies (MSST)*, pp 1–7. https://doi.org/10.1109/MSST.2016.7897080

Tan Y, Wang B, Wen J, Yan Z, Jiang H, Srisa-an W (2018) Improving restore performance in deduplication-based backup systems via a fine-grained defragmentation approach. *IEEE Transactions on Parallel and Distributed Systems* 29(10):2254–2267. https://doi.org/10.1109/TPDS.2018.2828842

Tan Z, Long L, Shen J, Liu R, Gao C, Zhong K, Jiang Y (2024) Optimizing garbage collection for ZNS SSDs via in-storage data migration and address remapping. *ACM Transactions on Architecture and Code Optimization* 21(4):1–25.
https://doi.org/10.1145/3677286

Ubaidillah SHSA, Ahmad N (2020) Fragmentation techniques for ideal performance in distributed database: A survey. *International Journal of Software Engineering and Computer Systems* 6(1):18–24. https://doi.org/10.15282/ijsecs.6.1.2020.3.0066

Xia W, Jiang H, Feng D, Douglis F, Shilane P, Hua Y, Fu M, Zhang Y, Zhou Y (2016) A comprehensive study of the past, present, and future of data deduplication. *Proceedings of the IEEE* 104(9):1681–1710. https://doi.org/10.1109/JPROC.2016.2571298

Xu Z, Zhang W (2021) QuickCDC: A quick content-defined chunking algorithm. In: *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing with Applications/Big Data and Cloud Computing (ISPA/BDCloud)*.

Yang R, Zhou Y, Huang P (2021) Boosting the restoring performance of de-duplication data by classifying backup metadata. *ACM Transactions on Data Science* 2(2):Article 13. https://doi.org/10.1145/3437261

Zhang D, Deng Y, Zhou Y, Li J, Zhu W, Min G (2022) MGRM: A multi-segment greedy rewriting method to alleviate data fragmentation. *IEEE Transactions on Cloud Computing* 11(3):2503–2516.

Zhang D, Deng Y, Zhou Y, Zhu W, Qin X (2021a) Improving deduplication-based backup systems via hot fingerprint entry distilling. *ACM Transactions on Storage* 17(4):1–23. https://dl.acm.org/doi/10.1145/3459626

Zhang X, Deng C, Xia W, Shilane P, Tan H, Zhang H, Wang X (2021b) Odess: Speeding up resemblance detection for redundancy elimination by fast content-defined sampling. In: *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pp 480–491. https://ieeexplore.ieee.org/document/9458911

Zhang Y, Jiang H, Feng D, Jiang N, Qiu T, Huang W (2023) LoopDelta: Embedding locality-aware opportunistic delta compression in inline deduplication for highly efficient data reduction. In: *Proceedings of the USENIX Annual Technical Conference (USENIX ATC '23)*, pp 133–148.

Zhang Y, Jiang H, Wang C, Huang W, Chen M, Zhang Y, Zhang L (2024) Applying delta compression to packed datasets for efficient data reduction. *IEEE Transactions on Computers* 73(1):73–85. https://doi.org/10.1109/TC.2023.3318404.

Zhu G, Zhang X, Wang L, Zhu Y, Dong X (2012) An intelligent data de-duplication based backup system. In: *Proceedings of the 15th International Conference on Network-Based Information Systems (NBiS 2012)*, pp 771–776. https://dl.acm.org/doi/10.1109/NBiS.2012.150

Zou X, Yuan J, Shilane P, Xia W, Zhang H, Wang X (2022) From hyper-dimensional structures to linear structures: Maintaining deduplicated data's locality. *ACM Transactions on Storage* 18(3):1–28. https://doi.org/10.1145/3507921