Storage System Trace Characterization, Compression, and Synthesis using Machine Learning – An Extended Abstract

Pratik Poudel
ppoud001@fiu.edu
Florida International University
Miami, USA

ABSTRACT

This study addresses the knowledge gap in request-level storage trace analysis by incorporating workload characterization, compression, and synthesis. The aim is to better understand workload behavior and provide unique workloads for storage system testing under different scenarios. Machine learning techniques like K-means clustering and PCA analysis are employed to understand trace properties and reduce manual workload selection. By generating synthetic workloads, the proposed method facilitates simulation and modeling-based studies of storage systems, especially for emerging technologies like Storage Class Memory (SCM) with limited workload availability.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning approaches*; **Modeling methodologies**; **Discrete-event simulation**.

ACM Reference Format:
Pratik Poudel. 2023. Storage System Trace Characterization, Compression, and Synthesis using Machine Learning – An Extended Abstract. In *SIGSIM Conference on Principles of Advanced Discrete Simulation (SIGSIM-PADS '23), June 21–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/XXX.XXX

1 INTRODUCTION

The use of storage traces to analyze workload characteristics, compression, and synthesis has been researched extensively. However, previous studies have focused on only one of these aspects. In this study, the researchers aim to address all three aspects using machine learning techniques and investigate trace properties across different time granularities. The goal is to provide representative workloads for simulation and modeling-based studies of data placement for emerging technologies like Storage Class Memory (SCM). No existing studies have addressed all three aspects together, making this research unique and valuable ([1], [2],[3], [4],[5]).

Workload characterization is the process of analyzing and understanding the behavior of a workload in a system, such as IO patterns and resource utilization. *Workload compression* is the process of reducing the number of traces needed to evaluate the storage system while maintaining the original workload's key characteristics, which helps in selecting a unique and non-redundant subset of the original traces as simulation points. *Workload synthesis* is the process of generating a synthetic workload that simulates real-world user and application behavior, enabling storage simulation and modeling works in scenarios where the workload of the target system is unavailable or limited. Figure 1 shows the block diagram of the proposed method. Request-level storage traces shown are *FIU* trace [6] and *Rocksdb* [7], taken in different time granularity windows (e.g., 1-hour time intervals). Workload characterization helps in understanding the behavior and characteristics of the available workload, which is used to find unique representative subsets of the workload and generate more workloads based on the desired IO pattern. The process makes use of machine learning methods like K-means clustering and PCA analysis to reduce the need for manual selection and learn hidden dependencies in the workload.

The proposed work enables understanding of storage trace behavior, enhancing the potential for workload characterization, compression, and synthesis together. The aim is to aid in the simulation of various scenarios and system performance using storage-level workloads. By using synthetically generated workloads solely or along with the available workloads, it facilitates simulation and modeling-based studies for emerging storage systems like Storage Class Memory (SCM) where workloads are not available or existing ones have limited availability. One such simulator, NVMain, supports various emerging non-volatile memory technologies, including DRAM and SCM. NVMain offers advantages in its ability to run trace-based simulation or be patched to a CPU simulator like gem5 for full system understanding. Configured with the appropriate parameters for the chosen SCM technology and fed with synthesized or compressed workloads, NVMain can simulate and model the system's performance. This method helps make informed choices for storage system optimization and resource allocation, resulting in enhanced performance and efficiency.

2 EXISTING STORAGE TRACES

Public repositories offer various storage traces for studying storage systems, focusing on request-level storage block traces in this study. The SNIA repository offers MSR Cambridge Traces, Virtual Desktop Infrastructure Traces, and Mobile Storage Subsystem Traces provide extensive collections for different systems. We have selected the FIU and RocksDB traces for our study [7][6]. The FIU traces consist of 21 days block traces from HDD-based production systems at Florida International University's Department of Computer Science. We consider three FIU trace types: *home*, *mail*, and *web*, consisting four separate workloads of home directories, the department's email inboxes, and a mix of *online* and *webresearch* workloads for web.

SIGSIM-PADS '23, June 21–23, 2023, Orlando, FL, USA
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-ABCD…$15.00
https://doi.org/10.1145/XXX.XXX

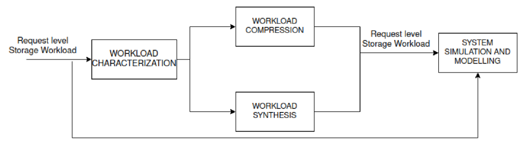

Figure 1: Workload Characterization, Compression and Synthesis process block diagram

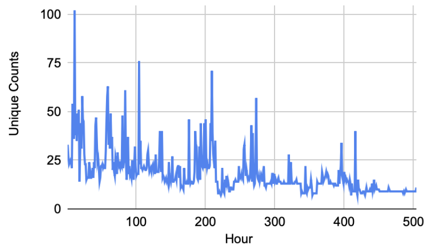

Figure 2: FIU home trace number of unique counts of lba over the course of 21 days taken at a granularity of one hour time interval

3 PROPOSED METHOD

The proposed method consists of three stages: (1) Workload Characterization, (2) Workload Compression, and (3) Workload Synthesis. Figure 1 shows the block diagram. Request-level storage workload traces are used for characterization, enabling compression while maintaining representativeness. Compressed workloads, together with artificial workloads, facilitate simulation-based studies of storage systems using simulators like NVMain.

In the *first* step, we analyze the FIU traces to observe IO patterns, subjecting them to trace characterization using diverse features determining data points over different intervals. After the initial analysis, we examine the trace in smaller windows based on time or requests. We describe each interval group using 13 features, used to analyze similar data points over the 21-day period to find similar phase-based information. Data points are grouped based on their cluster, preceded by normalizing the features and reducing the dimensionality of the feature while retaining initial information.

The *second* step, workload compression, uses the clustering results from characterization, selecting cluster centers as representative clusters. This reduces simulation points in further studies by removing redundant behavior. The *third* step, workload generation, leverages characterized traces to synthesize data based on the distribution and nature of data points in each cluster, using machine learning techniques like GANs, Transformers.

Future work involves using synthetic traces in NVMain to simulate and evaluate data placement strategies in SCM technologies. By comparing algorithms like LRU, LFU, and ARC, the impact on storage system performance and efficiency can be assessed, aiding in optimization for future applications.

4 PRELIMINARY RESULTS

Initial findings suggest that the I/O pattern is periodic in the first three weeks, with bursts of requests occurring every few days for FIU home and mail traces, and approximately every eight hours for the web trace. The RocksDB trace showed much higher I/O activity, with around six times as many unique streams and a peak intensity approximately 16 times higher than the other traces, a sample from the FIU home trace is shown in Figure 2 . However, the mail server had more I/O activity in terms of unique streams at peak times (approximately twice as high as RocksDB), although the average run had similar unique streams. In terms of total I/O intensity per hour, RocksDB had a higher value compared to the other traces, with a value around 10 times higher than the mail trace, which had the highest among the FIU traces. The main focus is on observing the periodic nature of I/O throughout the trace duration, rather than comparing the two traces (FIU and RocksDB). The study's second step focuses on workload characterization, using only the FIU trace, with RocksDB-based analysis left for future work. Results were obtained through various steps, such as verifying PCA analysis by checking the percentage of retained information, which in our case was 90 percent. The optimal number of clusters for K-means clustering varied based on the request granularity, with four clusters for the 21-day period of the FIU home trace with 150k requests, and 10 clusters for the same duration with a one-hour time interval. The analysis was conducted for both time and intensity granularity windows. Clustering was observed among trace points, with most centered around a major cluster despite the presence of multiple clusters. Similarity was computed using Euclidean distance and Pearson correlation, revealing a predominant region with dark squares corresponding to the supercluster. These findings show clustering tendencies in time and request window granularity, but further work is required for workload compression and generation using cluster points.

5 CONCLUSION

In conclusion, this study presents a comprehensive approach using machine learning techniques for storage-level trace characterization, compression, and synthesis. Analyzing FIU block and RocksDB traces over a 21-day period reveals insights into server performance across various time granularities. Utilizing normalization, PCA, and K-means clustering, the research uncovers patterns in IO activity and trace properties, significantly contributing to storage trace analysis. The findings enable a holistic understanding of storage trace behavior, empowering informed decisions on storage system optimizations and resource allocations. Future work explores machine learning techniques for trace compression, cluster data point selection, and trace synthesis to enable tasks like simulation, modelling of the storage systems to assess and improve storage system performance and efficiency.

REFERENCES

[1] J. Basak, K. Wadhwani, and K. Voruganti. 2016. Storage workload identification. *ACM Transactions on Storage (TOS)* 12, 3 (2016), 1–30.
[2] O. Desai, S. Shin, E. Lee, and B. S. Kim. 2022. A principled approach for selecting block I/O traces. In *Proceedings of the 14th ACM Workshop on Hot Topics in Storage and File Systems.* 52–58.
[3] Chris Ruemmler and John Wilkes. 1992. *UNIX disk access patterns.* Technical Report. Hewlett-Packard Laboratories Palo Alto.
[4] K. Sindhu, K. Seshadri, and C. Kollengode. 2022. Workload characterization and synthesis for cloud using generative stochastic processes. *The Journal of Supercomputing* 78, 17 (2022), 18825–18855.
[5] Tarasov Vasily. 2012. Extracting flexible, replayable models from large block traces. FAST.
[6] Akshat Verma, Ricardo Koller, Luis Useche, and Raju Rangaswami. 2010. Energy proportional storage using dynamic consolidation. *Proceedings of the File and Storage Systems* (2010).
[7] Gala Yadgar, Moshe Gabel, Shehbaz Jaffer, and Bianca Schroeder. 2021. SSD-based workload characteristics and their performance implications. *ACM TOS* (2021).