Secure and Efficient Client and Server Side Data Deduplication to Reduce Storage in Remote Cloud Computing Systems

Hemanth Chandra N¹, Sahana D. Gowda²,
Dept. of Computer Science
1 B.N.M Institute of Technology, Bangalore, India
2 B.N.M Institute of Technology, Bangalore, India

Abstract: Duplication of data in storage systems is becoming increasingly common problem. The system introduces I/O Deduplication, a storage optimization that utilizes content similarity for improving I/O performance by eliminating I/O operations and reducing the mechanical delays during I/O operations and shares data with existing users if Deduplication found on the client or server side. I/O Deduplication consists of three main techniques: content-based caching, dynamic replica retrieval and selective duplication. Each of these techniques is motivated by our observations with I/O workload traces obtained from actively-used production storage systems, all of which revealed surprisingly high levels of content similarity for both stored and accessed data.

Keywords: Deduplication, POD, Data Redundancy, Storage Optimization, iDedup, I/O Deduplication, I/O Performance.

1. INTRODUCTION
Duplication of data in primary storage systems is quite common due to the technological trends that have been driving storage capacity consolidation the elimination of duplicate content at both the file and block levels for improving storage space utilization is an active area of research. Indeed, eliminating most duplicate content is inevitable in capacity sensitive applications such as archival storage for cost effectiveness. On the other hand, there exist systems with a moderate degree of content similarity in their primary storage such as email servers, virtualized servers and NAS devices running file and version control servers. In case of email servers, mailing lists, circulated attachments and SPAM can lead to duplication. Virtual machines may run similar software and thus create collocated duplicate content across their virtual disks. Finally, file and version control systems servers of collaborative groups often store copies of the same documents, sources and executables. In such systems, if the degree of content similarity is not overwhelming, eliminating duplicate data may not be a primary concern.

Gray and Shenoy[1] have pointed out that given the technology trends for price-capacity and price-performance of memory/disk sizes and disk accesses respectively, disk data must “cool” at the rate of 10X per decade. They suggest data replication as a means to this end. An instantiation of this suggestion is intrinsic replication of data created due to consolidation as seen now in many storage systems, including the ones illustrated earlier. Here, it is referring to intrinsic (or application/user generated) data replication as opposed to forced (system generated) redundancy such as in a RAID-1 storage system. In such systems, capacity constraints are invariably secondary to I/O performance.

Gray and Shenoy[1] have pointed out that given the technology trends for price-capacity and price-performance of memory/disk sizes and disk accesses respectively, disk data must “cool” at the rate of 10X per decade. They suggest data replication as a means to this end. An instantiation of this suggestion is intrinsic replication of data created due to consolidation as seen now in many storage systems, including the ones illustrated earlier. Here, it is referring to intrinsic (or application/user generated) data replication as opposed to forced (system generated) redundancy such as in a RAID-1 storage system. In such systems, capacity constraints are invariably secondary to I/O performance.
2. RELATED WORK

This paper is related to works on Deduplication concepts in primary memory. Nimrod Megiddo and Dharmendra S. Modha[20] discussed about the problem of cache management in a demand paging scenario with uniform page sizes. A new cache management policy is proposed, namely, Adaptive Replacement Cache (ARC), that has several advantages. In response to evolving and changing access patterns, ARC dynamically, adaptively and continually balances between the recency and frequency components in an online and self tuning fashion. The policy ARC uses a learning rule to adaptively and continually revise its assumptions about the workload. The policy ARC is empirically universal, that is, it empirically performs as well as a certain fixed replacement policy—even when the latter uses the best workload-specific tuning parameter that was selected in an offline fashion. Consequently, ARC works uniformly well across varied workloads and cache sizes without any need for workload specific a priori knowledge or tuning. Various policies such as LRU-2, 2Q, LRFU and LIRS require user-defined parameters, and unfortunately, no single choice works uniformly well across different workloads and cache sizes. The policy ARC is simple-to-implement and like LRU, has constant complexity per request. In comparison, policies LRU-2 and LRFU both require logarithmic time complexity in the cache size. The policy ARC is scan-resistant: it allows one-time sequential requests to pass through without polluting the cache. On real-life traces drawn from numerous domains, ARC leads to substantial performance gains over LRU for a wide range of cache sizes.

Aayush Gupta et.al[22] have designed CA-SSD which employs content-addressable storage (CAS) to exploit such locality. Our CA-SSD design employs enhancements primarily in the flash translation layer (FTL) with minimal additional hardware, suggesting its feasibility. Using three real-world workloads with content information, statistical characterizations of two aspects of value locality - value popularity and temporal value locality - that form the foundation of
CA-SSD is devised. CA-SSD is able to reduce average response times by about 59-84% compared to traditional SSDs. Even for workloads with little or no value locality, CA-SSD continues to offer comparable performance to a traditional SSD. Our findings advocate adoption of CAS in SSDs, paving the way for a new generation of these devices.

Chi Zhang, Xiang Yu, Y. Wang[23] have answered two key questions. First, since both eager-writing and mirroring rely on extra capacity to deliver performance improvements, how to satisfy competing resource demands given a fixed amount of total disk space? Second, since eager-writing allows data to be dynamically located, how to exploit this high degree of location independence in an intelligent disk scheduler? In this paper, the two key questions were addressed and compared the resulting EW-Array prototype performance against that of conventional approaches. The experimental results demonstrate that the eager writing disk array is an effective approach to providing scalable performance for an important class of transaction processing applications.

Kiran Srinivasan et.al[24] has proposed an inline Deduplication solution, iDedup, for primary workloads, while minimizing extra IOs and seeks. Our algorithm is based on two key insights from real world workloads: i) spatial locality exists in duplicated primary data and ii) temporal locality exists in the access patterns of duplicated data. Using the first insight, only sequences of disk blocks were selectively deduplicated. This reduces fragmentation and amortizes the seeks caused by Deduplication. The second insight allows us to replace the expensive, on-disk, Deduplication metadata with a smaller, in-memory cache. These techniques enable us to tradeoff capacity savings for performance, as demonstrated in our evaluation with real-world workloads. Our evaluation shows that iDedup achieves 60-70% of the maximum Deduplication with less than a 5% CPU overhead and a 2-4% latency impact.

Jiri Schindler et.al[26] introduce proximal I/O, a new technique for improving random disk I/O performance in file systems. The key enabling technology for proximal I/O is the ability of disk drives to retire multiple I/O’s, spread across dozens of tracks, in a single revolution. Compared to traditional update-in-place or write-anywhere file systems, this technique can provide a nearly seven-fold improvement in random I/O performance while maintaining (near) sequential on-disk layout. This paper quantifies proximal I/O performance and proposes a simple data layout engine that uses a flash memory-based write cache to aggregate random updates until they have sufficient density to exploit proximal I/O. The results show that with cache of just 1% of the overall disk-based storage capacity, it is possible to service 5.3 user I/O requests per revolution for random updates workload. On an aged file system, the layout can sustain serial read bandwidth within 3% of the best case. Despite using flash memory, the overall system cost is just one third of that of a system with the requisite number of spindles to achieve the equivalent number of random I/O operations.

### 2.1 EXISTING SYSTEM

To address the existing data Deduplication schemes for primary storage, such as iDedup and Offline-Dedupe, are capacity oriented in that they focus on storage capacity savings and only select the large requests to deduplicate and bypass all the small requests (e.g., 4 KB, 8 KB or less). The rationale is that the small I/O requests only account for a tiny fraction of the storage capacity requirement, making Deduplication on them unprofitable and potentially counterproductive considering the substantial Deduplication overhead involved. However, previous workload studies have revealed that small files dominate in primary storage systems (more than 50 percent) and are at the root of the system performance bottleneck. Furthermore, due to the buffer effect, primary storage workloads exhibit obvious I/O burstiness. The important performance issue of primary storage in the Cloud and the above Deduplication-induced problems, a Performance-Oriented data Deduplication scheme, called POD, rather than a capacity-oriented one (e.g., iDedup), to improve the I/O performance of primary storage systems in the
Cloud is proposed. Figure 1 represents the architecture of POD. By considering the workload characteristics, POD takes a two-pronged approach to improving the performance of primary storage systems and minimizing performance overhead of Deduplication, namely, a request-based selective Deduplication technique, called Select-Dedup, to alleviate the data fragmentation and an adaptive memory management scheme, called iCache, to ease the memory contention between the bursty read traffic and the bursty write traffic.

2.2 PROPOSED SYSTEM (POD vs IDedup)
A possible future direction is to optionally coalesce or even eliminate altogether write I/O operations for content that are already duplicated elsewhere on the disk, or alternatively direct such writes to alternate locations in the scratch space.

While the first option might seem similar to data Deduplication at a high-level, a primary focus on the performance implications of such optimizations rather than capacity improvements has been suggested. Any optimization for writes affects the read-side optimizations of I/O Deduplication and a careful analysis and evaluation of the trade-off points in this design space is important and shares data with existing users if Deduplication found in the client or server side.

Advantages:
1. Requires less space in the storage server
2. Data Deduplication will be done completely in all blocks

3. IMPLEMENTATION
POD
In this paper, the system proposes POD, a performance-oriented Deduplication scheme, to improve the performance of primary storage systems in the Cloud by leveraging data Deduplication on the I/O path to remove redundant write requests while also saving storage space. In Figure 2 shows the sequence diagram where one can understand how the sequence flow happens between each module. POD takes a request-based selective Deduplication approach (Select-Dedup) to Deduplicating the I/O redundancy on the critical I/O path in such a way that it minimizes the data fragmentation problem. In the meanwhile, an intelligent cache management (iCache) is employed in POD to further improve read performance and increase space saving, by adapting to I/O burstiness. Our extensive trace-driven evaluations show that POD significantly improves the performance and saves the capacity of primary storage systems in the Cloud.
iDedup
In iDedup system, the system describes iDedup, an inline Deduplication system specifically targeting latency-sensitive, primary storage workloads and comparison with latest techniques like POD also known as I/O Deduplication. With latency sensitive workloads, inline Deduplication has many challenges: fragmentation leading to extra disk seeks for reads, Deduplication processing overheads in the critical path and extra latency caused by IOs for Dedup-metadata management.

3.1 COMPARATIVE ANALYSIS
In this section, the performance of POD vs iDedup Deduplication models is evaluated through extensive trace-driven experiments.

Figure 3. The time delay performance of the different Deduplication schemes.

A prototype of POD as a module in the Linux operating system and use the trace-driven experiments to evaluate its effectiveness and efficiency has been implemented. In this paper, POD with the capacity oriented scheme iDedup is compared. Two models (POD vs iDedup) with two parameters are compared, they are Time delay and Cpu utilization.
Figure 4. The Cpu utilization performance of the different Deduplication schemes.

In the first parameter comparison i.e., Time delay. Four production files were compared for our trace driven evaluation. In Figure 3(a) the time delay for the iDedup model can be seen and in Figure 3(b) the time delay for same files in Pod model is shown. For both the deduplicator models, same files were uploaded with name cmain, owner, clog, search. Results shows that Pod is efficient than iDedup.

In the second parameter comparison i.e., Cpu utilization. a single file named abc.txt was uploaded to both Pod and iDedup models. In Figure 4(a) Cpu utilization in iDedup model can be seen and in Figure 4(b) the Cpu utilization in Pod model can be seen. From the results it states that Pod is more efficient than the iDedup model.

4. CONCLUSION

System and storage consolidation trends are driving increased duplication of data within storage systems. Past efforts have been primarily directed towards the elimination of such duplication for improving storage capacity utilization. With I/O Deduplication, a contrary view is taken that intrinsic duplication in a class of systems which are not capacity-bound can be effectively utilized to improve I/O performance – the traditional Achilles’ heel for storage systems. Three techniques contained within I/O Deduplication work together to either optimize I/O operations or eliminate them altogether. An in-depth evaluation of these mechanisms revealed that together they reduced average disk I/O times by 28-47%, a large improvement all of which can directly impact the overall application-level performance of disk I/O bound systems. The content-based caching mechanism increased memory caching effectiveness by increasing cache hit rates by 10% to 4x for read operations when compared to traditional sector-based caching. Head-position aware dynamic replica retrieval directed I/O operations to alternate locations on-the-fly and additionally reduced I/O times by 10-20%. Selective duplication created additional replicas of popular content during periods of low foreground I/O activity and further improved the effectiveness of dynamic replica retrieval by 23 35%.

FUTURE WORK

I/O Deduplication opens up several directions for future work. One avenue for future work is to explore content-based optimizations for write I/O operations. A possible future direction is to optionally coalesce or even eliminate altogether write I/O operations for content that are already duplicated elsewhere on the disk or alternatively direct such writes to alternate locations in the scratch space. While the first option might seem similar to data Deduplication at a high-level, a primary focus on the performance implications
of such optimizations rather than capacity improvements is suggested. Any optimization for writes affects the read-side optimizations of I/O Deduplication and a careful analysis and evaluation of the trade-off points in this design space is important and shares data with existing users if Deduplication found in the client or server side.

REFERENCES


