IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727
PP 13-16
www.iosrjournals.org

Building a Fingerprint Based Deduplication Detection and Elimination Scheme

Jisha Mariam Jacob [1], Sowmya K S [2]
[1](Department of Computer Science, College of Engineering Perumon, India)
[2](Department of Information Technology, College of Engineering Perumon, India)

Abstract: *As digital data is growing uncontrollably, the need for data reduction has become an important task in storage systems. For large scale data reduction, it is important to maximally detect and eliminate redundancy at low overheads. Data deduplication is a data reduction technique that reduces storage space by eliminating redundant data and only one instance of the data is retained on storage media. Delta compression is an efficient method for removing redundancy among non-duplicates but very similar data files and chunks. DARE is a deduplication aware, low overhead resemblance detection and elimination scheme for data reduction that uses duplicate-adjacency information for resemblance detection in a deduplication system. It also uses an improved super feature approach for further resemblance detection when DupAdj information is lacking or limited. DARE attained a superior performance of both throughput and data reduction efficiency among all resemblance detection approaches.*

Keywords: *Data deduplication, Data reduction, Delta compression, Fingerprint, Resemblance detection, Super-feature.*

I. Introduction

Data reduction is the process of reducing the amount of data stored in a data storage environment which increases storage efficiency and reduces cost. Mainly there are 3 types of lossless data reduction techniques for storage systems- traditional compression (e.g. GZIP compression [1]), delta compression [3] and data deduplication [4] [5]. Data deduplication is an efficient data reduction approach that reduces storage space by eliminating redundant data. The subsequent reference to the same data is stored as pointers to already stored copy. It also minimizes data that must be sent across a WAN for remote backups, replication, and disaster recovery. Advantages are improved storage efficiency, cost saving and minimization of bandwidth. Generally the data deduplication can be operated both at the file-level and block level. File-level deduplication also called as Single Instance Storage (SIS) [6] that examine the index backup or archive files need the attributes stored in the file for comparison. If the file is not same then it will store and update index. Otherwise a pointer will be placed to the existing file. Thus only one instance of the file is saved. It is not a very efficient means of deduplication. Block deduplication looks within a file and saves unique iterations of each block. Generally, a chunk-level data deduplication scheme splits the input data block of a data stream into multiple data chunks, which are uniquely identified and duplicate detected by a secure hash signature(e.g., SHA-1 or MD5) also called a fingerprint[11].

The fingerprint-based deduplication scheme fails to detect similar chunks that are largely identical besides for a very few modified chunks because even if only one bye of a data chunk is changed, then their secure hash signature will be totally different. When applying data deduplication to storage datasets that has frequently modified data it becomes a major issue.

Delta compression [3] [7] [8] [9] [10] is an efficient method for removing redundancy among non-duplicate but very similar data files and chunks. Suppose chunk A_2 is similar to chunk A_1 (base chunk). This approach first calculates and then stores the differences (called delta) and the mapping information between A_2 and A_1. So it is an effective technique that complements by identifying similar data. The problem in applying delta compression in deduplication system is how to identify the most similar candidates for delta compression with low overheads. For delta compression similarity is detected by calculating several Rabin fingerprints as feature and paring them into super-fingerprints, also called as super-features (SF). Indexing a large data set is too big to fit in memory. Also the unsymmetric accesses to on-disk super-features results in low throughput of the system.

The existing solutions for the indexing problem of delta compression is either to record the resemblance information for files, instead of data chunks, so that the similarity index entries can be fitted in the memory or to exploit the locality of backup data streams in deduplication-based systems, which avoid the global indexing on the disk. The first solution faces an implementation difficulty since it is hard to record all the

resemblance information of files in a large-scale data deduplication systems.. When the workload has insufficient locality, later fails to detect redundant data. Also the super-feature method faces high overhead in computing the super features.

It has been found out that the non-duplicate chunks that are adjacent to duplicate ones could be considered as candidate for delta compression in data deduplication systems. Thus the approach of Duplicate-Adjacency based Resemblance Detection, or DupAdj has been used. The use of deduplication information avoids high overhead of super-feature computation and reduces the size of index entries for resemblance detection. Two data chunks are considered to be similar (i.e., candidates for delta compression) if the data chunks that are adjacent are duplicate in a deduplication system, and then further enhance the resemblance detection efficiency by an improved super-feature approach.

II. Existing System

Data deduplication is a method for reducing the amount of storage space by removing the redundant data. Thus it saves only one instance of the data. Venti [4] is a network storage system that detects the data blocks by a hash of their contents. Each data block has a unique hash that is, it is collision-free. The advantage of fingerprint based data deduplication is it will be inconvenient to find two distinct inputs that hashes to same value. It also provides integrity checking for data, improves robustness and performance. The drawback is for a large dataset; the fingerprint is too large to fit in the memory and must be moved to the disk.

In deduplication storage systems, there are several techniques for reducing disk I/O. In [7] an on-disk index of fingerprint in maintained and also a cache is used for index access. Deduplication along with sparseness is an efficient mean for lowering storage consumption.

Data domain file system [9] minimizes index lookups by maintaining the locality of fingerprints for attaining high cache hit ratios. For Wide Area Networks, stream informed delta compression has been proposed for replication of backup datasets. It reduces the index overhead. In this the backup data are stored on backup server and on remote backup repository. The data streams are first divided into content defined chunks [CDC], then for each chunk a secure fingerprint is calculated. Then in the containers, non- duplicate chunks are stored. For each non-duplicate chunk, sketches are calculated which is further used to identify similar chunks.

The content-based chunking [10] approach was proposed for finding more redundant data. Using CBC a file is divided into a sequence of chunks. With this, data is optimized for both high deduplication savings and minimal resource consumption and low overhead. For maintaining the deduplication process and integrity, the system consists of a file system filter driver and a set of background jobs.

SiLo [11] is a deduplication system that exploits similarity and locality for attaining high throughput at low RAM overheads. It either groups strongly-correlated files into a segment or segments a large file for exploiting and exposing their similarity characteristics. As a result of this, smaller similarity index is created so that it can be easily fit into RAM. In storage systems, Resemblance detection with delta compression has been used. REBEL [12] combines compression, duplicate block suppression and delta encoding for removing redundant data. It uses super-fingerprints for reducing the data required to identify similar blocks.

III. Proposed System

DARE is designed to enhance the resemblance detection for additional data reduction in deduplication-based backup/archiving storage systems. The DARE architecture consists of three functional modules, namely, the Deduplication module, the DupAdj Detection module, and the improved super-feature module. In addition, there are five key data structures in DARE, namely, Dedupe Hash Table, SFeature Hash Table, Locality Cache, Container, Segment, and Chunk.

A chunk is the atomic unit for data reduction. The non-duplicate chunks, identified by their SHA-1 fingerprints, will be prepared for resemblance detection in DARE. A container is the fixed-size storage unit that stores sequential and NOT reduced data, such as non-duplicate & non-similar or delta chunks, for better storage performance by using large I/Os. A segment consists of the metadata of a number of sequential chunks (e.g., 1 MB size), such as the chunk fingerprints, size, etc., which serves as the atomic unit in preserving the backup-stream logical locality for data reduction. Here DARE uses a data structure of doubly-linked list to record the chunk adjacency information for the DupAdj detection. The SFeature in the segment may be unnecessary if the DupAdj module has already confirmed this chunk as being similar for delta compression. Dedupe hash table serves to index fingerprints for duplicate detection for the deduplication module. SFeature hash table serves to index the super-features after the DupAdj resemblance detection. It manages the super-features of non-duplicate and non-similar chunks. Locality cache contains the recently accessed data segments and thus preserves the backup-stream locality in memory, to reduce accesses to the on-disk index from either duplicate detection or resemblance detection.

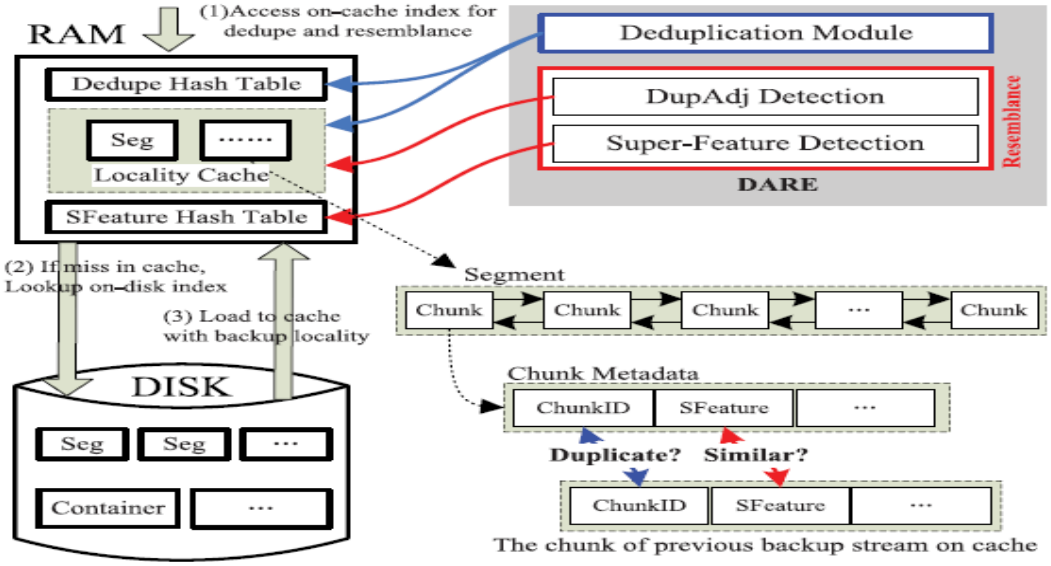

Fig 1: DARE architecture

By using the Deduplication module, DARE will first detect duplicate chunks for the input data stream. The DupAdj approach detects resemblance by exploiting duplicate-adjacency information of a deduplication system. In this module, for each non-duplicate chunk, DARE will first use its DupAdj Detection module to quickly determine whether it is a delta compression candidate; if it is not a candidate, DARE will then compute its features and super-features, using its improved Super-Feature Detection module, to further detect resemblance for data reduction. Because DARE adopts a caching scheme that exploits the backup-stream logical locality in a way similar to the Sparse Indexing, SiLo, and BLC approaches, the indexing hit ratio in the locality cache for both the deduplication and resemblance detection modules will be very high. Upon a miss in the locality cache, DARE will load the missing segment from the latest backup to the RAM with the LRU replacement policy. It is noteworthy that, after deduplication, the cached segments that have preserved the logical-locality of chunks, including the adjacency information of the duplicate-detected chunks, will be further exploited by DARE to detect possible resemblance among the non-duplicate data chunks.

DARE keeps track the backup stream logical locality of chunk sequence using a doubly-linked list, that helps to search the duplicate adjacency chunks for detecting resemblance by traversing the chunks on the list.

IV. Conclusion

DARE is a deduplication-aware, low-overhead resemblance detection and elimination scheme for delta compression built on the top of deduplication on backup datasets. DARE uses a novel resemblance detection approach, DupAdj, which utilizes the duplicate-adjacency information for efficient resemblance detection in the existing deduplication systems, and incorporates an improved super-feature method for detecting resemblance when the duplicate-adjacency information is inadequate or restricted. DARE is an efficient tool for maximizing the data reduction at low overhead. Supplementing delta compression to deduplication can effectively enlarge the logical space of the restoration cache.

Acknowledgements

We thank members of Computer Science and Information Technology Department of College Of Engineering, Perumon for their valuable support and feedback.

References

[1]. The data deluge [Online]. Available: http://econ.st/fzkuDq
[2]. J. Gantz and D. Reinsel, *Extracting value from chaos, IDC Rev., vol. 1142, pp. 1–12, 2011.*
[3]. L. DuBois, M. Amaldas, and E. Sheppard, *Key considerations as deduplication evolves into primary storage, White Paper 223310, Framingham, MA, USA: IDC, Mar. 2011.*
[4]. S. Quinlan and S. Dorward, *Venti: A new approach to archival storage,* in Proc. USENIX Conf. File Storage Technol., Jan. 2002, pp. 89–101.
[5]. D. T. Meyer and W. J. Bolosky, *A study of practical deduplication,* ACM Trans. Storage, vol. 7, no. 4, p. 14, 2012.
[6]. W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur, *Single instance storage in windows 2000,* in Proc. 4th USENIX Windows Syst. Symp., Aug. 2000, pp. 13–24.

[7]. A. Muthitacharoen, B. Chen, and D. Mazieres, *A low-bandwidth network file system*, in Proc. ACM Symp. Oper. Syst. Principles., Oct. 2001, pp. 1–14

[8]. P. Shilane, M. Huang, G. Wallace, and W. Hsu, *WAN optimized replication of backup datasets using stream-informed delta compression*, in Proc. 10th USENIX Conf. File Storage Technol., Feb. 2012, pp. 49–64.

[9]. B. Zhu, K. Li, and R. H. Patterson, *Avoiding the disk bottleneck in the data domain deduplication file system*, in Proc. 6th USENIX Conf. File Storage Technol., Feb. 2008, vol. 8, pp. 1–14.

[10]. A. El-Shimi, R. Kalach, A. Kumar, A. Ottean, J. Li, and S. Sengupta, Primary data deduplication-large scale study and system design, in Proc. Conf. USENIX Annu. Tech. Conf., Jun. 2012, pp. 285–296.

[11]. W. Xia, H. Jiang, D. Feng, and Y. Hua, *Silo: A similarity-locality based near-exact deduplication scheme with low RAM overhead and high throughput*, in Proc. USENIX Conf. USENIX Annu. Tech. Conf., Jun. 2011, pp. 285–298.

[12]. P. Shilane, G. Wallace, M. Huang, and W. Hsu, *Delta compressed and deduplicated storage using stream-informed locality*, in Proc. 4th USENIX Conf. Hot Topics Storage File Syst., Jun. 2012, pp. 201–214.