SDRE: Selective Data Redundancy Elimination for Resource Constrained Hosts

Yan Zhang and Nirwan Ansari
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
Newark, NJ 07012
{yz45, nirwan.ansari}@njit.edu

Mingquan Wu and Heather Yu
Media Technologies Research
Huawei Technologies USA
Bridgewater, NJ, 08807
{heatheryu,Mingquan.Wu}@huawei.com

***Abstract*—Data redundancy elimination (DRE), also known as data de-duplication, reduces the data amount to be transferred or stored by identifying and eliminating both intra-object and inter-object duplicated data elements. It is one of the key content delivery acceleration techniques over wide area networks (WANs) to reduce delivery latency and bandwidth consumptions by reducing the amount of data to be transferred. Deploying DRE at the end hosts maximizes the bandwidth savings and latency reductions, because the amount of content sent to the destination hosts is minimized. However, standard DRE used to identify redundant content chunks is very expensive in terms of memory and processing capability especially on resource constrained hosts. By analyzing the web application traffic traces, we find out that some types of contents have more redundant contents than others. Thus, it is possible to apply DRE selectively and opportunistically on those contents with more redundant data elements than other content types to save the memory and processing resources at the hosts. In this paper, we propose content-type based selective DRE (SDRE), which deploys DRE selectively on the contents which have the most opportunities for redundant content identification. We explore the benefits of deploying SDRE on smartphone traffic traces. The results show that SDRE can achieve almost the same bandwidth savings as that of standard DRE with less computation and smaller memory.**

***Index Terms*—Data redundancy elimination (DRE), data de-duplication, content delivery acceleration, wide area network (WAN) optimization.**

I. Introduction

Today's IT organizations tend to deploy their infrastructures geographically over a wide area network (WAN) to increase productivity, support global collaboration, and minimize costs, thus constituting today's WAN-centered environments. As compared to a local area network (LAN), a WAN is a telecommunication network that covers a broad area; WAN may connect across metropolitan, regional, and/or national boundaries. Traditional LAN-oriented infrastructures are insufficient to support global collaboration with high application performance at low cost. Deploying applications over WANs inevitably incurs performance degradation owing to the intrinsic nature of WANs such as high latency and high packet loss rate [1]. Many factors, not normally encountered in LANs, can quickly lead to performance degradation of applications which are run across a WAN. As reported in [2], the WAN throughput degrades greatly with the increase of transmission distance and packet loss rate.

The need for speedup over WANs spurs on application performance improvement over WANs. WAN optimization, also commonly referred to as WAN acceleration, describes the idea of enhancing application performance over WANs. A variety of WAN acceleration techniques have been proposed. Some focus on maximizing bandwidth utilization, others address latency, and still others address protocol inefficiency which hinders the effective delivery of packets across the WAN. Data compression is very important to reduce the amount of bandwidth consumed on a link during transfer across the WAN, and it can also reduce the transit time for given data to traverse over the WAN by reducing the amount of transmitted data. Data redundancy elimination (DRE) [3, 4], also known data de-duplication, is a data reduction technique and a derivative of data compression. Data compression [5] reduces the file size by eliminating redundant data contained within an object, while DRE can identify and eliminate the transmission of redundant content from both intra-object and inter-object, such as an entire file and a data block, to reduce the amount of data to be transferred or stored. When multiple instances of the same data element are detected, only one single copy of the data element is transferred or stored. The redundant data element is replaced with a reference or pointer to the unique data copy. A study of large-scale traffic traces [6] indicated that network traffic exhibits a large amount of redundancy. DRE is one of the important content delivery acceleration techniques to enhance application performance over WANs and meet quality of service (QoS) requirements by reducing the amount of data to be transferred, thus reducing the delivery latency and bandwidth consumptions.

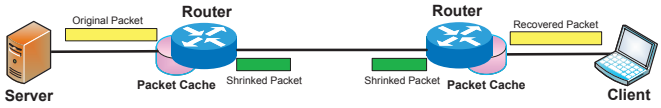
Based on the algorithm granularity, DRE algorithms can be classified into three categories: whole file hashing [7, 8], sub-file hashing [9–17], and delta encoding [18]. Traditional DRE operates at the application layer, such as web caching, to eliminate redundant data transfers. With the rapid growth of network traffic in the Internet, DRE techniques operating on individual packets have been deployed [9–14] based on different chunking and sampling methods. The main idea of packet-level DRE is to identify and eliminate redundant chunks across packets. A large scale trace-driven study on the efficiency of packet-level DRE [6] showed that packet-

978-1-4673-0921-9/12/$31.00 ©2012 IEEE

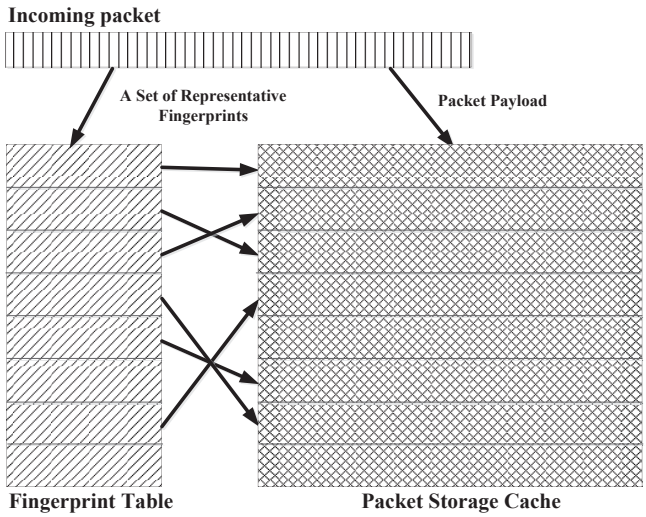
level DRE can obtain average bandwidth savings of 15-60% when deployed at access links of the service providers or between routers. Experimental evaluations on various DRE technologies are presented in [3, 6, 12].

Standard DRE for identifying redundant content chunks is very expensive in terms of memory and processing capability especially on resource constrained hosts. By analyzing the web application traffic traces, we find out that some types of contents have more redundant contents than others, e.g., texts have more redundant data elements than videos since videos are stored in compressed formats in general and almost do not exhibit redundant data blocks within one video. Thus, it is possible to apply DRE selectively and opportunistically on those contents with more redundant data elements than other types to save the memory and processing resources at the hosts. In this paper, we propose content-type based selective DRE (SDRE), which deploys DRE selectively on the contents which have the most opportunities for redundant content identification. The main benefits of SDRE include achieving almost the same bandwidth savings with less computation resources and improving memory utilization. However, the benefits of SDRE come at the cost of less bandwidth savings since the redundant content is not identified across all of the contents transferred from the source to the destination.

The rest of the paper is structured as follows. We present a brief outline of a popular mechanism for packet-level DRE, and analyze its computation complexity in Section II. Then, we describe the proposed content-type based SDRE in Section III. The benefits of SDRE over the standard DRE are explored in Section IV. Finally, Section V concludes the paper.



(a) A typical packet-level DRE implementation.



(b) The packet-level DRE elements.

Fig. 1: The classical packet-level data redundancy elimination.

Algorithm 1 MODP DRE Algorithm

```
1:  for i = w − 1; i < S;i++ do
2:      fingerprint = RabinHash(data[i : i + w − 1]);
3:      if (fingerprint mod p == 0) then
4:          if (fingerprint exists in the fingerprint table) then
5:              Remove the matched region from the arriving
                packet and replace the matched region with a
                fingerprint description;
6:          else
7:              Insert fingerprint in the fingerprint table;
8:          end if
9:      end if
10: end for
11: Insert packet payloads in the packet store cache.
```

II. DRE Algorithms and Computational Overhead

In this section, we briefly describe the typical implementation of packet-level DRE techniques. We outline one popular DRE mechanism, MODP [9], to identify redundant chunks across packets. We also analyze its computational overhead.

A. Packet-Level DRE

The classical packet-level DRE technique is implemented at routers, which work cooperatively. A typical packet-level DRE implementation is shown in Figure 1(a). The router near the source removes the identified redundant data chunks, while the router located at the other end recovers the original packet by inserting the identified redundant data chunks. To deploy packet-level DRE algorithms, a fingerprint table, which stores representative fingerprints calculated from each packet, and a packet cache, which stores packet payloads, are required as shown in Figure 1(b). Only those representative fingerprints and packet payloads observed over some past interval of time are stored in the fingerprint table and the packet cache, respectively. The end-to-end DRE pushes the DRE implementation to the sources and the destinations. The benefits of end-to-end DRE is multi-folder; for examples, it can be applied to operate on encrypted data traffic and it may minimize the redundant data elements transmitted over the networks.

B. MODP DRE Algorithm

MODP was first applied to the DRE mechanism by Spring *et al.* [9] to remove redundant content chunks across packets. For every packet arriving in a particular direction, MODP first computes a set of Rabin fingerprints by applying the Rabin-Karp hash [19] function over sliding windows of w contiguous bytes of the packet's payload (line 2). According to the findings of redundant match distribution [6], $w = 32$ bytes is suggested to maximize the effectiveness of DRE. Thus, for an S-byte packet ($S \geq w$), a total of $S - w + 1$ fingerprints are generated. It is impossible to store all fingerprints in the fingerprint table, and only those fingerprints whose value is 0

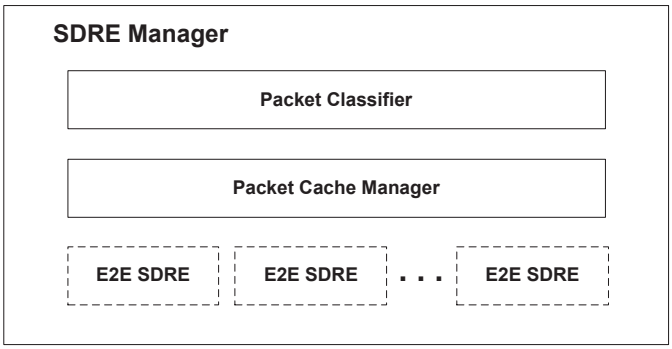


Fig. 2: The components of a SDRE manager.

$mod\ p$ are chosen as the representative fingerprints (lines 3-10), which will be stored in the fingerprint table. Hence, $1/p$ of the calculated fingerprints are sampled as the representative fingerprints for every packet. The packets are stored in the packet store cache as shown in Figure 1(b). Pointers from each fingerprint to the corresponding packet are also stored.

Each representative fingerprint is then checked against the fingerprints stored in the fingerprint table. If one matched fingerprint already exists in the table (line 4), the matching region is then found in the packet store cache. In general, two matching mechanisms can be used for DRE techniques. One is called chunk-match, and the other one is maximum-match. If one matched representative fingerprint is found in the fingerprint table, the w byte representative region used to compute this representative fingerprint is identified as the redundant chunk by the chunk-match mechanism. While with the maximum-match mechanism, the matched region is maximized by expanding to the left and right of the representative region. After the matched region is identified, it is removed from the arriving packet and replaced with a fingerprint description (line 5), which consists of the offset of the fingerprint in the packet store cache as well as the amount of the redundant bytes before and after the matched representative region. If there is no matched fingerprint in the fingerprint table, a new representative fingerprint will be inserted into the fingerprint table (line 7). Finally, the packet payloads will be stored in the packet cache (line 11). A formal description of the MODP DRE algorithm is shown in Algorithm 1.

C. Computational Overhead

As described above, the representative fingerprints for each packet are selected from the calculated Rabin fingerprints, which are calculated for every w-byte length substring of the packet. Given a sequence of bytes $[t_1, t_2, \cdots, t_w]$ of length w bytes, a Rabin fingerprint can be expressed as:

$$RF(t_1, t_2, \cdots, t_w) \ = \ (t_1 p^{w-1} + t_2 p^{w-2} + \cdots + t_w)\ mod\ M \quad (1)$$

where p and M are constant integers.

For a S-byte packet ($S \geq w$), a sequence of $S - w + 1$ Rabin fingerprints can be calculated using the above equation with the substrings $\{\{t_1, t_2, \cdots, t_w\}, \{t_2, t_3, \cdots, t_{w+1}\}, \cdots, \{t_{S-w+1}, t_{S-w+2}, \cdots, t_S\}\}$. By observing the above equation, the calculation for the next Rabin fingerprint can be

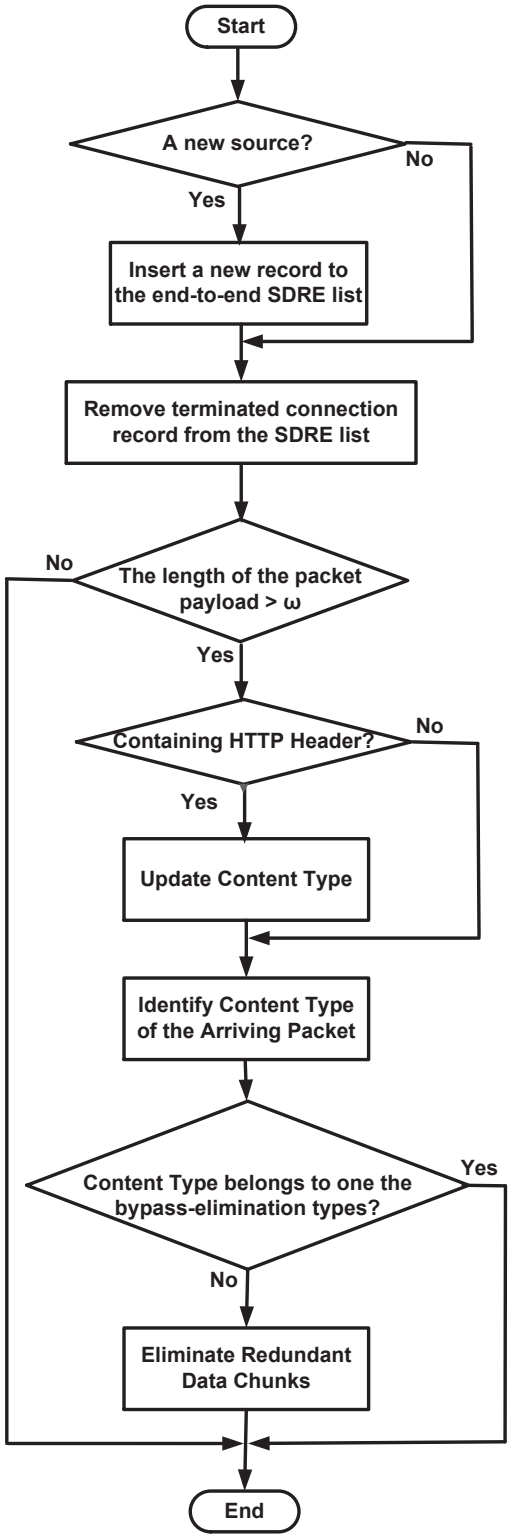


Fig. 3: Selective Data Redundancy Elimination Mechanism.

TABLE I: **Smartphone Traffic Trace Breakdown**

Trace	Volume	Text	Image	Video	Audio	Appl.
1	39.7 MB	19%	10%	39%	0	32%
2	23.7 MB	38%	19%	0	0	43%
3	61.9 MB	23%	9%	0	42%	26%
4	24.4 MB	29%	35%	10%	0	26%
5	24.4 MB	33%	16%	7%	0	44%
6	27.9 MB	34%	27%	0%	0	39%
7	52.7 MB	21%	25%	3%	0	51%

simplified by using the previous one as follows:

$$RF(t_{i+1},t_{i+2},\cdots,t_{i+w}) = (RF(t_i,\cdots,t_{i+w-1}) - t_i \times p^w) \times p + t_{i+w} \; mod \; M \quad (2)$$

Since p and w are constant, $t_i \times p^w$ can be precalculated and stored in a table. With this fast Rabin fingerprint execution, a substraction, a multiplication, an addition and a modular calculation are required to compute one fingerprint.

A subset of these calculated Rabin fingerprints are then selected as the representative fingerprints for the packet. If the maximum-match criterion is used, for each matched fingerprint, the corresponding matched data block needs to be retrieved from the packet cache, and the matched region is expanded byte-by-byte in both directions to obtain the maximal region of redundant data block.

Based on the above DRE computation overhead analysis, we can see that standard DRE is very expensive in terms of memory and processing capability to identify redundant content chunks.

III. Selective Data Redundancy Elimination

SDRE is a packet-level content-type based end-to-end data redundancy elimination technique, which deploys DRE selectively on the contents which have the most opportunities for redundant content identification. As shown in Figure 2, SDRE consists of a packet classifier, a DRE packet cache manager, and multiple end-to-end SDRE modules, which are created and terminated dynamically according to the end-to-end traffics. The packet classifier maintains a content-type table according to the TCP flow tuples, consisting of the source IP address, the source port, the destination IP address, and the destination port. For any content transferred from the source to the destination over a TCP connection for web applications, a HTTP header should be transmitted ahead of the content delivery. Thus, the packet classifier can categorize the content-type of the following content packets by identifying the "CONTENT-TYPE" HTTP field in the HTTP header. A DRE packet cache manager is another important component in an end-to-end DRE technique especially for resource-constrained hosts because a content source connects to many end users and an end-user connects to multiple content sources simultaneously, while the size of the packet cache used for redundancy elimination is limited. A properly designed packet cache management algorithm can improve the effectiveness of DRE techniques. The packet cache can be shared evenly among all host-to-host connections, but this might reduce the utilization of the packet cache and effectiveness of DRE because some host-to-host connections may transfer more contents than others. Hence, traffic volume based packet cache assignment could be an effective method for end-to-end DRE techniques than connection based cache assignment. SDRE modules are the components which eliminate the redundant elements from the packets. The starts and terminations of SDRE modules are controlled by the SDRE manager, which maintains an end-to-end SDRE list. When a new source IP address is detected by the SDRE manager, a new record will be inserted to the end-to-end SDRE list. When a source has completed all the content transmission, the SDRE manager will remove its record from the end-to-end SDRE list.

A brief description of the SDRE mechanism is shown in Fig. 3. When a new packet arrives, the SDRE manager checks its source IP address first against the end-to-end SDRE list. If there is no matched record, a new record will be inserted into the end-to-end SDRE list. At the same time, if the content type belongs to one of the bypass redundancy elimination types, a new end-to-end SDRE module will be created, and some space of the packet cache will be assigned to it. Otherwise, the end-to-end SDRE module will not be created and the DRE packet cache will not be assigned for this source-to-destination connection until one packet, which contains the payload does not belong to all of the bypass redundancy elimination types, has been received by the end user. Then, the terminated source-to-destination connections are checked and removed from the list. If the length of the packet payload is smaller than the size of the Rabin sliding window, it will bypass the redundancy elimination procedure; otherwise, it will be passed to one of the SDRE module according to its source IP address. For every arriving packet which is passed to the SDRE module, it will be classified as a HTTP header packet if it contains a completed or partial HTTP header; otherwise, it will be classified as a HTTP content packet. For a HTTP header packet, the HTTP field "CONTENT-TYPE" is filtered out and its field value is used for packet content classification. If the content type of the arriving packet belongs to one of the SDRE elimination types, it will bypass the redundancy elimination; otherwise, redundant data elements in the arriving packet will be identified and eliminated against the DRE packet cache.

IV. Performance Evaluation of SDRE

In this section, we explore the benefits of deploying SDRE on smartphone traffic traces. First, we explain how these traffic traces are collected. Seven smartphone 3G traffic traces were collected from seven persons and used for SDRE evaluation. Each person uses his/her smartphone to access the Internet normally, and the web application traffic will be recorded to a file automatically. Web application traffic of at least seven days was recorded for each person.

The major part of web traffic could be text, image, video, audio, and application. There are also some other types of

content used in the web application, such as message, model and multipart, but they only compose a very small part of the whole web traffics. The traffic breakdown based on the content-type for these seven traffic traces is shown in Table I. The total volumes recorded for these seven persons range from 20 to 60 MB. In these seven traffic traces, the text, image and application types of contents compose of 19%-38%, 9%-35% and 26%-51% of the total volume, respectively. Among these seven persons, only one of them (trace 1) used his/her smartphone to watch some video contents substantially. The video content made up of 39% of the total volume for this trace record. Other six persons rarely used their smartphone to surf videos on the Internet. Only another person listened to some audio content (trace 3), which occupied 42% of his/her total traffic volume, and no other persons downloaded any audio contents in these seven traces.

Figure 4 shows the bandwidth saving ratios of SDRE over that of DRE versus different sizes of packet store caches, ranging from 100 MB to 1 MB. The MODP fingerprint calculation algorithm and the maximum matching mechanism are deployed in this evaluation. The sliding window size is set to 32 bytes as suggested to maximize the effectiveness of DRE in [6]. Each point represents the bandwidth saving ratio of the SDRE over that of DRE for one traffic trace with some size of packet cache. Connection-based packet cache management is used in this evaluation. Two bypass DRE content-type scenarios are evaluated. One includes videos and audios, and the other bypass DRE content-type set includes images, videos, and audios. The images, videos and audios are stored in compressed formats in general due to the relatively large original sizes of images, videos and audios compared to text files. Thus, redundancy within these compressed content, such as images, videos and audio, is quite limited; this provides opportunities to reduce the computation overhead of DRE algorithms while

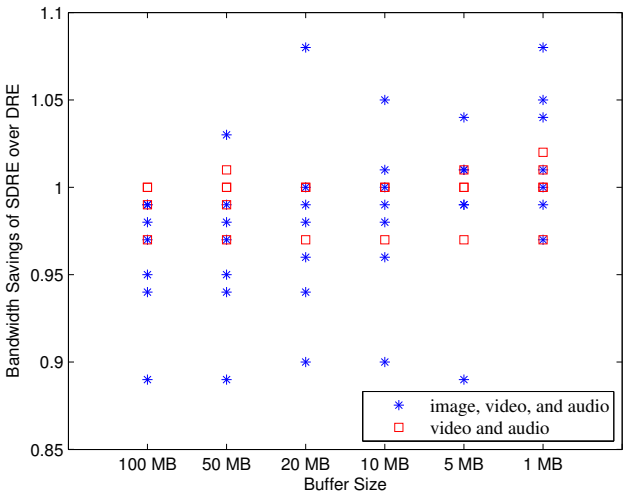


Fig. 4: Bandwidth savings of SDRE over DRE for different smartphone traffic traces.

achieving almost the same DRE effectiveness. With enough packet caches, the bandwidth saving ratio of SDRE over that of DRE represents how much redundancy is contributed by the bypassed DRE content types. In this case, 100 MB packet cache is large enough to remove all the redundant data chunks which can be identified by the DRE techniques. From the results with 100 MB packet cache, we see that all the ratios are smaller than 1 because only parts of packets are checked for redundant data chunks, but more than 90% of the redundant data chunks can be identified by SDRE. This result verifies that redundancy within images, videos and audio is quite limited. With the decrease of the DRE packet cache, SDRE achieves more bandwidth savings than that of standard DRE. Since the content with more redundant data chunks will stay in the DRE packet cache longer and will not be refreshed by the content with less redundancy, the effectiveness of the packet cache for redundant data identification can be improved. Among all of these seven smartphone traces, about 19%-51% and 3%-42% of traffic processed for redundant data identifications can be reduced by setting up the DRE bypass content types with {images, video, audio} and {videos, audio}, respectively. From the results shown in Figure 4, we can see that SDRE can achieve almost the same bandwidth savings as that of standard DRE with less computations and smaller memory.

V. Conclusion

DRE has been developed to eliminate the transmission of redundant content to improve the application performance over WANs by reducing the amount of data to be transferred, hence reducing the delivery latency and bandwidth consumptions. However, standard DRE for identifying redundant content chunks is very expensive in terms of memory and processing capability especially on resource constrained hosts. In this paper, we have proposed a content-type based selective DRE (SDRE) technique to reduce the resource requirement of DRE algorithms by applying DRE selectively on the contents which have the most opportunities for redundant content identification. The benefits of deploying SDRE on smartphone traffic traces were evaluated, and the results showed that SDRE can achieve almost the same bandwidth savings as that of standard DRE with less computation and smaller memory size.

References

[1] T. Grevers Jr. and J. Christner, *Application Acceleration and WAN Optimization Fundamentals*. Indianapolis, IN: Cisco Press, 2007.

[2] P. Sevcik and R. Wetzel, "Improving Effective WAN Throughput for Large Data Flows," http://www.silver-peak.com/assets/download/pdf/Netforecast_wp_EffectiveThroughput.pdf, November 2008.

[3] N. Mandagere, P. Zhou, M. A. Smith, and S. Uttamchandani, "Demystifying Data Deduplication," in *Proc. of the Middleware Conference Companion*, Leuven, Belgium, Dec. 1-5, 2008, pp. 12–17.

[4] Q. He, Z. Li, and X. Zhang, "Data Deduplication Techniques," in *Proc. of FITME*, Oct. 2010, pp. 430 –433.

[5] M. Al-laham and I. M. M. E. Emary, "Comparative study between various algorithms of data compression techniques," *International Journal of Computer Science and Network Security*, vol. 7, no. 4, April 2007.

[6] A. Anand, C. Muthukrishnan, A. Akella, and R. Ramjee, "Redundancy in network traffic: findings and implications," in *Proc. of the 11th International Joint Conference on Measurement and Modeling of Computer Systems*, Seattle, WA, USA, 2009, pp. 37–48.

[7] W. J. Bolosky, S. Corbin, D. Goebel, and J. R. Douceur, "Single Instance Storage in Windows 2000," in *Proc. of the 4th conference on USENIX Windows Systems Symposium*, Washington, Aug. 2000, pp. 13–24.
[8] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer, "Reclaiming Space from Duplicate Files in a Serverless Distributed File System," in *Proc. of ICDCS*, Vienna, Austria, July 2002, pp. 617–624.
[9] N. T. Spring and D. Wetherall, "A protocol-independent technique for eliminating redundant network traffic," in *Proc. of SIGCOMM*, Stockholm, Sweden, 2000, pp. 87–95.
[10] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker, "Packet caches on routers: the implications of universal redundant traffic elimination," in *Proc. of SIGCOMM*, Seattle, WA, USA, 2008, pp. 219–230.
[11] A. Anand, V. Sekar, and A. Akella, "SmartRE: an architecture for coordinated network-wide redundancy elimination," *SIGCOMM Computer Communication Review*, vol. 39, pp. 87–98, August 2009.
[12] B. Aggarwal, A. Akella, A. Anand, A. Balachandran, P. Chitnis, C. Muthukrishnan, R. Ramjee, and G. Varghese, "EndRE: an end-system redundancy elimination service for enterprises," in *Proc. of NSDI*, San Jose, California, 2010, pp. 28–28.
[13] G. Lu, Y. Jin, and D. Du, "Frequency based chunking for data deduplication," in *Proc. of Modeling, Analysis Simulation of Computer and Telecommunication Systems*, Aug. 2010, pp. 287 –296.
[14] S. Saha, A. Lukyanenko, and A. Yla-Jaaski, "Combiheader: Minimizing the number of shim headers in redundancy elimination systems," in *Proc. of INFOCOM workshops*, April 2011, pp. 798 –803.
[15] S. Quinlan and S. Dorward, "Venti: A New Approach to Archival Data Storage," in *Proc. of FAST*, 2002, pp. 89 – 101.
[16] A. T. Clements, I. Ahmad, M. Vilayannur, and J. Li, "Decentralized Deduplication in SAN Cluster File Systems," in *Proc. of USENIX Annual technical conference*, San Diego, California, June 2009.
[17] C. Constantinescu, J. Pieper, and T. Li, "Block Size Optimization in Deduplication Systems," in *Proc. of Data Compression Conference*, Snowbird, Utah, March 16-18, 2009.
[18] J. J. Hunt, K.-P. Vo, and W. F. Tichy, "An Empirical Study of Delta Algorithms," in *Proc. of the SCM-6 Workshop on System Configuration Management*, 1996, pp. 49–66.
[19] M. O. Rabin, "Fingerprinting by Random Polynomials," Center for Research in Computing Technology, Harvard University, Tech. Rep. TR-15-81, 1981.