SmartRE: An Architecture for Coordinated Network-wide Redundancy Elimination

Ashok Anand*, Vyas Sekar† and Aditya Akella*
*University of Wisconsin-Madison, †Carnegie Mellon University
{ashok,akella}@cs.wisc.edu, vyass@cs.cmu.edu

ABSTRACT

Application-independent Redundancy Elimination (RE), or identifying and removing repeated content from network transfers, has been used with great success for improving network performance on enterprise access links. Recently, there is growing interest for supporting RE as a network-wide service. Such a network-wide RE service benefits ISPs by reducing link loads and increasing the effective network capacity to better accommodate the increasing number of bandwidth-intensive applications. Further, a network-wide RE service democratizes the benefits of RE to all end-to-end traffic and improves application performance by increasing throughput and reducing latencies.

While the vision of a network-wide RE service is appealing, realizing it in practice is challenging. In particular, extending single-vantage-point RE solutions designed for enterprise access links to the network-wide case is inefficient and/or requires modifying routing policies. We present SmartRE, a practical and efficient architecture for network-wide RE. We show that SmartRE can enable more effective utilization of the available resources at network devices, and thus can magnify the overall benefits of network-wide RE. We prototype our algorithms using Click and test our framework extensively using several real and synthetic traces.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network management*

General Terms

Algorithms, Design, Management

Keywords

Redundancy Elimination, Caching

1. INTRODUCTION

Redundancy Elimination (RE) for network transfers has gained a lot of traction in recent years. RE is widely used by data centers and enterprise networks to improve their effective network capacity, to reduce their wide-area footprint, and to improve end-to-end application performance. The importance of RE is reflected in the emergence of a huge market for RE solutions (e.g., [4, 3, 2, 8, 5]) and their rapidly growing adoption [6, 9].

The success of such deployments has motivated researchers, equipment vendors, and ISPs to explore the potential of network-wide RE. For example, Anand et al. [12] have recently shown the benefits of supporting RE as a primitive IP-layer service on network routers. In similar vein, network equipment vendors have highlighted network-wide support for content caching and duplicate suppression as a key focus area in their future development efforts [3, 2]. Broadly speaking, these efforts argue for deploying RE at multiple points across a large network and using it as a generic service which is transparent to end-to-end applications.

This vision of network-wide RE is promising for two reasons. First, a network-wide deployment spreads the benefits of RE to all end-to-end applications, as opposed to just benefiting transfers on the individual links of enterprises. Second, it benefits ISPs by improving their effective network capacity and allowing them to better accommodate the increasing number of bandwidth intensive multimedia and file-sharing applications we see today, and by giving them better control over traffic engineering operations [12].

While RE has been well-studied in the context of point deployments (e.g., enterprise WAN access links), there has been little work on how best to design network-wide RE. Thus, the promise of network-wide RE remains unfulfilled. In this paper, we study how to build an effective and practical network-wide RE architecture.

We start by observing that a network-wide RE architecture should meet three key requirements:

(1) **Resource-awareness:** RE involves resource-intensive operations such as indexing content, looking up content fingerprints and compressing data, and reconstructing the original content from locally stored information. An ideal approach must explicitly account for the resource constraints on network elements in performing these RE functions. These constraints arise mainly from (a) throughput bounds which depend on the number of memory operations possible per second and (b) memory capacity which limits the amount of data that can be cached for RE purposes. Naive approaches that do not account for these constraints, such as the strawman framework of Anand et al. [12], offer sub-optimal performance. In contrast, using the limited resources available at each node intelligently can offer close to the best possible benefits.

(2) **Network-wide goals:** The architecture should allow network operators to specify network-wide goals such as increasing overall efficiency (e.g., improving the network throughput) or to achieve specific traffic engineering goals (e.g., alleviating congested hotspots).

(3) **Flexibility:** The architecture must be incrementally adoptable providing benefits even under partial deployment, and must supplement, not replace, current network operations such as existing routing and network management practices.

SIGCOMM'09, August 17–21, 2009, Barcelona, Spain.

We present the design, implementation, and evaluation of SmartRE, an architecture for network-wide RE that meets the above requirements. In SmartRE, redundancy elimination is performed in a coordinated fashion by multiple devices. SmartRE uses the available resources on RE devices efficiently and naturally accommodates several network-wide objectives.

In describing SmartRE, we focus largely on packet-level RE in ISP networks [12], where RE devices on routers cache packet payloads and strip duplicate strings from individual packets. However, we believe that our design can apply to other deployment scenarios, e.g., in multi-hop wireless networks and datacenters.

In SmartRE, a packet can potentially be reconstructed or decoded several hops downstream from the location where it was compressed or encoded. In this respect, SmartRE represents a significant departure from packet-level RE designs proposed in prior solutions [29, 12], where each compressed packet is reconstructed at the immediate downstream router. Further, SmartRE uses a network-wide coordinated approach for intelligently allocating encoding and decoding responsibilities across network elements.

In general, encoding incurs greater overhead than decoding. Thus, SmartRE allocates encoding to ingress routers to avoid overloading interior routers that operate at higher line-rates and thus have stricter resource constraints. Since the number of edge routers is large, a large number of encoded packets are introduced into the network. Interior routers in SmartRE perform less expensive decoding actions. Decoding is performed in a coordinated fashion with each interior router responsible for storing and reconstructing a fraction of the encoded packets on a path. We use hash-based sampling techniques [31] to facilitate coordination across interior routers while incurring negligible overhead.

When allocating encoding and decoding responsibilities across a network, SmartRE takes into account the memory capacity and packet processing throughput at each RE device along with the prevailing traffic conditions, and configures the actions of different devices so as to best meet an operator-specified network-wide goal. This ensures that no device is overwhelmed and that RE is used optimally to meet the network's objectives.

The duplicate removal and reconstruction logic in SmartRE can be implemented in high-speed two-port switches or middleboxes, which can then be deployed across specific ISP links. These enable incremental adoption in an ISP network. We develop prototypes of the two-port switches in the Click modular router [21]. Using real packet traces, we find that the prototypes can perform duplicate removal at 2.2 Gbps and reconstruction at 8 Gbps.

We conduct an in-depth evaluation of SmartRE as applied to IP-layer RE in ISP networks using controlled simulations based on synthetic and real packet traces over several real and inferred ISP topologies. Across a range of topologies and traffic patterns, the performance of SmartRE is 4-5$\times$ better than naively extending a single-vantage point RE solution to the network-wide case. Further, and more significantly, SmartRE achieves 80-90% of the absolute network footprint reduction of the optimal possible case where RE devices are not limited by any throughput or capacity constraints. We also evaluate partial deployment scenarios and find that enabling SmartRE on a small set of strategically selected routers can offer significant network-wide benefits.

2. RELATED WORK AND BACKGROUND

We start by describing prior work on removing duplicate data from network links, ranging from full object-based approaches to partial packet-based ones. We then present details of packet-level RE and describe prior work on enabling packet-level RE as a router service across ISP networks that forms a key focus in our work.

2.1 Related Work

Object-level caching: Several systems in the past have explored how to remove duplicate data from network links. "Classical" approaches such as Web caches work at the object level, serving popular HTTP objects locally [32]. In similar spirit, CDNs and peer-to-peer caches [7, 1] perform object-level duplicate removal.

Protocol-independent RE mechanisms: In recent years, a class of *application- and protocol-independent* techniques have been developed which can remove redundant strings from any traffic flow. Starting with the pioneering work of Spring et al. [29], several commercial vendors have introduced "WAN optimizers" which remove duplicate content from network transfers. Many of these products [4, 2, 8, 5] work at the level of chunks inside objects and we refer to them as *chunk-level* approaches. In contrast, both Spring et al. [29] and Anand et al. [12] adopt techniques which are similar at the high level but operate at a *packet-level*.

Content-based naming for RE: Content-based naming has emerged as an alternative to enhance web caching (e.g., [19, 26]), content distribution (e.g., [30, 23, 22]), and distributed file systems (e.g., [11]). These approaches use fingerprinting mechanisms [24] similar to packet-level RE to identify addressable chunks. However, these approaches require modifications to end-systems to fully realize the benefits of RE. Network-based, protocol-independent RE approaches are transparent to end-systems and offers the benefits of RE to end-systems that are not content-aware.

2.2 Packet-level RE Explained

The central idea of packet-level RE is to remove strings in packets that have appeared in earlier packets. To perform RE across a single link, the upstream device stores (in memory) packets it has transferred on the link over a certain period of time. Packet contents are indexed using *fingerprints* which essentially form content-based hooks pointing to content in random locations within the packet. For each incoming packet, the upstream RE device checks if the packet's fingerprints have appeared in earlier in-memory packets. Each matching fingerprint indicates a certain region of partial overlap between the incoming packet and some earlier packet. The matching packets are compared to identify the maximal region of overlap. Such overlapping regions are removed from the incoming packet and a shim is inserted to tell the downstream device how to decode the packet using its local memory. A packet can carry multiple shims, each potentially matching a different in-memory packet. Decoding is simple: the downstream device uses the shim in the encoded packet to retrieve the matching packet(s), and fills in the corresponding missing byte range(s). Chunk-level approaches work similarly.

2.3 Network-wide RE

Why packet-level RE: Both packet- and chunk-level RE are agnostic to application protocols and can be implemented as generic network services that need not understand the semantics of specific applications. Prior studies have shown that both approaches are significantly better than caching entire objects [29]. However, chunk-level approaches require terminating TCP connections and partially reconstructing objects before applying compression. This interferes with the end-to-end semantics of connections and also imposes high overhead on the RE devices since they must maintain per-flow state. Packet-level approaches do not interfere with end-to-end semantics of connections, and where technology permits, can be transparently supported in routers or middleboxes.

Extending packet-level RE to a network: Since packet-level RE brings significant compression benefits while operating in a transparent and application-agnostic fashion, Anand et al advocate its

use as a router primitive for network-wide RE [12]. In their proposal, each router in an ISP network maintains a cache of recently forwarded packets. Upstream routers on a link use the cache to identify common content with new incoming packets and strip these redundant bytes on the fly. Downstream routers reconstruct packets from their local cache. This process repeats in a *hop-by-hop* fashion along a network path inside an ISP. Anand et al. evaluate an ideal, unconstrained setting where they assume memory operations take negligible time and that the caches on each router are infinite. Under this model, they show that network-wide RE could offer significant benefits in terms of reducing overall network load and absorbing sudden traffic overload in situations such as flash crowds. The central goal of our paper is to design a practical architecture that can achieve these benefits when RE elements operate within realistic throughput and memory capacity constraints.

The hop-by-hop approach proposed by Anand et al. is a naive approach because it takes a very link-local view of RE and does not account for constraints of the RE devices. In the next section, we discuss why this naive approach offers poor performance in practice and show how smarter caching and coordination can offer vastly improved benefits.

3. BENEFITS OF COORDINATION

We start by describing the practical limits on the throughput of the two packet-level RE primitives, namely, encoding and decoding. Then, we present qualitative examples highlighting the benefits arising from assigning encoding and decoding responsibilities across a collection of routers in an intelligent, coordinated fashion. In particular, we show how this: (1) leads to efficient memory usage, (2) ensures RE-related tasks can be performed at full capacity, and (3) enables incremental deployment. We contrast this against a naive approach that does not account for resource constraints.

In this section, we assume a hypothetical intelligent, coordinated approach. This has two implications. First, we have the flexibility to specify where a packet should be cached along a routing path. In particular, this allows us to split caching responsibilities along a path. This is in contrast to the hop-by-hop approach, where each packet is explicitly cached at every hop along the path. For example, if packets $p_1, \ldots, p_4$ traverse a path $I, R_1, \ldots, R_4$, we can specify that each p_i is cached at (and only at) R_i. Second, we assume that RE devices that are separated by multiple hops in the network can either implicitly or explicitly maintain a consistent view of each other's caches. This means that an encoded packet can potentially be decoded several hops downstream from the point where it was encoded. In the above example, this means that I can encode packet p_4 with respect to p_3 and R_3 is responsible for decoding it. Again, in the hop-by-hop approach, this would not be possible; each packet would have to be encoded and decoded per-link.

3.1 Encoding and Decoding Throughput

Standalone throughput: The main bottleneck affecting the processing throughput of packet-level RE operations is *memory access*. Encoding a packet requires multiple memory accesses and is much slower than decoding. To see why, suppose that the memory hardware can support R random memory accesses per second. For modern DRAMs, the random access latency is 50ns, hence $R = 2 \times 10^7$. Suppose that each packet has at most k matches, and that we compute F fingerprints for each packet. (Note that since the number of matches can never be more than the number of fingerprints that were computed, $k \leq F$.) Typical values are $F = 10$ and $k = 3$ [12].

The encoding throughput for a standalone RE device is *at most* R/F packets per second. This is because each packet, whether it

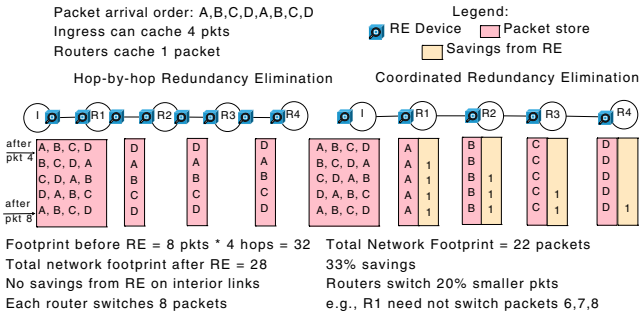

Figure 1: Benefits of a coordinated approach when RE devices have constraints on memory size.

can be encoded or not, requires F random accesses to determine if there is a match or not. Once matches are found, further processing is required to actually create the encodings. On the other hand, decoding throughput is *at least* R/k. This is because each packet has between 0 and k encodings. Thus, in this standalone case, decoding is $\geq F/k$ times faster than encoding. Since $k \leq F$, the decoding throughput is clearly higher.

Throughput on a single link: Given this understanding of the standalone encoding and decoding throughput, we can now consider the throughput across a single link. For simplicity, let us assume all packets are of the same size MSS. Suppose that the link capacity is such that it can carry P MSS-sized packets per second. For instance, if the link speed is 2.4Gbps (OC48), and $MSS = 500B$, then $P = 6 \times 10^5$ and for an OC192 link $P = 2.4 \times 10^6$. Two cases arise:

1. **Slow link** ($R/F \geq P$): This means that *line rate* encoding and decoding are possible; e.g., for an OC48 link where $R/F = 2 \times 10^6 \geq P = 6 \times 10^5$. In this case, the encoder can encode up to P packets per second, each carrying up to k matches. The decoder can decode each encoded packet.
2. **Fast link** ($R/F < P$): This means that *line rate encoding* is not possible. This is the case for OC192 and higher speed links. ($R/F = 2 \times 10^6 < P = 2.4 \times 10^6$). In this case, the encoder can encode no more than R/F packets per second; a fraction of packets are left un-encoded to ensure line-rate operation. Even though the decoder as a standalone operates F/k times faster, its decoding throughput is now limited by the encoding throughput immediately upstream. Thus, it is limited to decoding R/F packets per second.

3.2 Motivating Examples

We present the examples in the context of a "bump-in-the-wire" deployment where an RE middlebox is attached to router linecards. Each RE device has pre-specified *resource constraints*. These capture hardware limitations (e.g., how many decoding actions can the device perform per unit time?) or economic constraints (e.g., DRAM cost which could limit total memory per device).

These examples also apply when there are resource budgets *per router*. For example, processing constraints induced by power/cooling requirements are better modeled on a per-router/per-PoP basis rather than per-middlebox. Also, software or virtualized RE deployments (e.g., [14, 21]) would be characterized by per-router constraints.

As the following examples show, the naive hop-by-hop approach described in the previous section severely constrains the effectiveness of redundancy elimination.

Memory efficiency and router benefits: Consider the scenario in Figure 1. Suppose each RE device on the path has memory to store only 1 packet for this path (since the devices are shared among the paths that traverse the link), but the RE devices on the first link can