Intra-AS Cooperative Caching for Content-Centric Networks

Jason Min Wang
jasonwangm@cse.ust.hk

Jun Zhang
zhangjunalex@cse.ust.hk

Brahim Bensaou
brahim@cse.ust.hk

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Hong Kong, China

ABSTRACT

The default caching scheme in CCN results in a high redundancy along the symmetric request-response path, and makes the caching system inefficient. Since it was first proposed, much work has been done to improve the general caching performance of CCN. Most new caching schemes attempt to reduce the on-path redundancy by passing information on content redundancy and popularity between nodes. In this paper, we tackle the problem from a different perspective. Instead of curbing the redundancy through special caching decisions in the beginning, we take an orthogonal approach by pro-actively eliminating redundancy via an independent intra-AS procedure. We propose an *intra-AS cache cooperation* scheme, to effectively control the redundancy level within the AS and allow neighbour nodes in an AS to collaborate in serving each other's requests. We show via trace-driven simulation, that intra-AS cache cooperation improves the system caching performance and reduces considerably the traffic load on the AS gateway links, which is very appealing from an ISP's perspective.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network communications; C.2.3 [**Network Operations**]: Network management

Keywords

Content Centric Network, Caching, Redundancy Elimination

1. INTRODUCTION

Content-Centric Networking (CCN) [9], proposes to redesign the Internet architecture to natively support today's main usage of the Internet – i.e., as the pervasive vehicle of content generation, sharing, and distribution. CCN has attracted a wide interest from the networking community, and along with other similar proposals [10, 18], it elevates named-data to being the first-class citizen in the network. In particular, one of the defining features of CCN is that caching – as an *underlay* capability instead of an *overlay* capability – has been extended to common CCN nodes, by turning the buffer memory in the routers into a cache space.

In CCN, named-data exists in the form of a sequence of chunks that are uniquely identified by hierarchical names, and will be requested by the receiver via separate *interest* packets. Upon a cache hit at a Content Store, the *data* packet is sent back to the receiver following the symmetric reverse path of the interest packet, as illustrated in Figure 1. In case of a cache miss, the interest packet shall be forwarded out in compliance with a name-based routing strategy, if a similar request is not already pending. As such, CCN performs *request aggregation* to avoid sending duplicate requests upstream. When in transit, the chunk data is doomed to be cached by all CCN nodes that it crosses, and the LRU strategy is invoked by default for selecting the victim when cache replacement is necessary. We refer to this approach of caching everywhere with LRU replacement hereafter as the "*ubiquitous LRU*". Although this is simple and fast strategy, it could result in a very poor caching performance, especially due to the highly synchronized cache redundancy along the return path of the chunk data.

Several recent works on CCN have been dedicated to designing new caching schemes to reduce such path redundancy as a means to improving the systematic caching performance. In [17], the so-called PCP scheme proposes to avoid caching unpopular data in access routers and bring popular data to the end users in a progressive way. ProbCache in [13] chooses to cache content probabilistically along the content delivery path. In [12], an age-based implicit cooperative caching scheme is proposed where the age value reflects the content location and popularity. WAVE in [6] maintains a value that indicates the content popularity for each object in order to decide the number of chunks that each object should cache.

In this paper, we explore caching issues in CCN from a different perspective. Instead of directly determining whether a specific item should be cached or which item should be purged to make room for the new item, we devote to building an *intra-AS cache cooperation* scheme that allows nearby nodes to eliminate redundancy of their cached items and to collaboratively serve each other's requests, thereby improving the utilization of limited cache resources. More specifically, each CCN node is freely subject to any caching scheme. Albeit, periodically, neighbour nodes collaboratively purge

ICN'13, August 12, 2013, Hong Kong, China.

ACM 978-1-4503-2179-2/13/08 ...$15.00.

duplicate items from their cache space. Consequently, a considerable amount of caching space can be released and further used to cache more fresh and popular items. By following the simple rule that an originally cached item should either be kept in place or be found at one of the neighbour nodes, the same performance level can be preserved while increasing the available number of free caching slots in an AS. We show via trace-driven simulation that the intra-AS caching cooperation improves the average cache hit rate while reducing the traffic load on the institutional link of an AS. It is worth noting that the intra-AS cache cooperation scheme is orthogonal to pure caching schemes [6, 12, 13, 17], and therefore the two can work together.

The rest of this paper is organized as follows. We present the intra-AS cache cooperation scheme in Section 2, and formulate the cooperative redundancy elimination problem in Section 3. We present experimental results in Section 4 and conclude the paper in Section 5.

2. INTRA-AS CACHE COOPERATION

The Internet consists of different levels of autonomous systems (ASs) that belong to independent administrative and economic entities. This work focuses on improving the caching performance with respect to a specific AS through the *intra-AS cache cooperation*. In this section, we first illustrate the framework of cache cooperation scheme. Then we elaborate on the motivations of eliminating cache redundancy within the AS, which is the crucial component of the cache cooperation scheme.

2.1 Framework

The framework of *intra-AS cache cooperation* scheme is described as follows:

- Each CCN node is independently taken care of by a specific cache management scheme, like the default ubiquitous LRU strategy for example. Ideally, the cache space of an AS should abound in popular items requested by its served clients.

- Periodically, each CCN node exchanges with its one-hop neighbours the cache "*summary*" that encodes the information of currently cached items. With these summaries, nearby nodes can implicitly cooperate to serve each other's requests. The size of the summary influences the scalability of the cooperation scheme, due to the overhead of exchanging summaries and storing them in memory. To keep it small, bloom filters can be used to encode the summary. One further caveat is to avoid exposing short-lived items such as ephemeral one-timers in the summary.

- A fresh round of "*cooperative redundancy elimination*" will be launched when the "redundancy monitoring metric" indicates the potential benefits of doing so based on the collected summaries. It undertakes the task of controlling the redundancy level within the AS. The released caching slots from eliminating duplicates serve to cache more diacritical items, which is conducive to enhancing the overall caching performance.

- If a request cannot be served locally, before forwarding the request out following the original CCN forwarding strategy, the node will probe its neighbours to see whether a cache hit might happen there, in light of the exchanged summaries. To maintain the possibly increased delay caused by a false positive probe, the probing action for a quick cache hit is confined within the one-hop neighbourhood and only at the first hop node (the access routers).

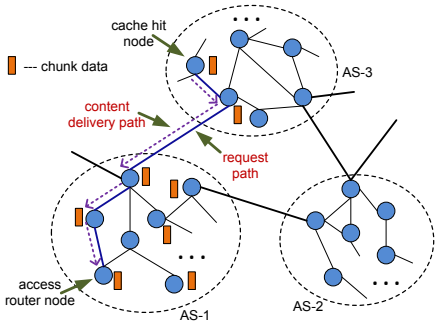


Figure 1: The request and response scenario in CCN with multiple ASs. The default caching strategy causes high redundancy within the AS.

2.2 Motivation of Redundancy Elimination

Network traffic exhibits naturally high redundancy, as has been observed and validated repeatedly in past works [2, 3]. The huge traffic redundancy arises from the fact that a large amount of popular content is often accessed by a great number of users from time to time. Many diverse systems [2, 14] commit to eliminating such redundancy. The central abstraction of named-data in CCN embraces such endeavour as well. The unique identity of named-data already provides a means to remove redundancy at the level of individual nodes. However, redundancy is free to appear among nodes with any degree, especially when each CCN node is exposed to the default ubiquitous LRU caching scheme and also when the door of *multi-path* routing is opened [9]. By default, the requested chunk data will be cached everywhere along the return path, as illustrated in Figure 1. Thus requests for the same data issued by clients at different locations can make the AS brim with copies of the same data. Controlling the redundancy level is critical to systematically improve the caching performance of CCN, as highlighted by the following two aspects:

- First, a CCN node's caching performance is highly related to its cache size, while the available caching resource is rather limited. Recall that CCN's caching capability is directly elevated from the buffer memory of the incumbent IP router. Thus the inherited cache space is around 10 GB size[1]. Even though the average cost of storage is continuously decreasing and extra storage might be extended to the Content Store (cache space) in the future, it is not clear whether it can keep up the pace with the sharply increasing scale of video content distribution over the Internet.

- Second, from the ISP's perspective, one of the biggest and most urgent concerns today is how to sustain the ever-increasing network traffic in the AS. The dominant video traffic surging on both wired and wireless networks poses a great burden on ISP link bandwidth, let alone the cost paid for the huge amount of cross-traffic. CCN, equipped with the in-network caching capability, is expected to greatly offload the cross-AS traffic.

[1]The memory size of commodity routers is limited by both the price of DRAM or SRAM and the requirement of fast packet processing.

These two facts dictate the frugal usage of limited caching resources within the AS by only storing valuable content and avoiding storage waste in storing duplicates. Therefore, controlling the redundancy level occupies a significant role in improving the AS-level caching performance.

2.3 Dimensions of Redundancy Elimination

Broadly speaking, redundancy elimination in the caching network of CCN can occur in two spacial dimensions: i) *vertically*, by controlling the number of copies of the chunk data along the return path; and ii) *horizontally*, by controlling the duplication over neighbour nodes within the AS. Moreover, it can be undertaken in two manners: i) *actively* by assessing in real time whether the passing chunk data should be cached or not; or ii) *passively* by controlling redundancy through an independent procedure after data has been cached. The caching schemes proposed in [6, 12, 13, 17] all fall into the category of vertical-active redundancy elimination. In contrast, here our efforts go to achieving performance improvements via a horizontal-passive redundancy control.

3. COOPERATIVE REDUNDANCY ELIMINATION

In this section, we start by formalizing the *cooperative redundancy elimination* problem, then after discussing several issues on the problem formulation, we present a greedy heuristic algorithm to eliminate redundancy.

3.1 Problem Formulation

Consider a network managed by a single administrative entity denoted by the undirected graph $G = (N, E)$, where N is the set of network nodes and E is the set of connecting links between nodes. Let S_i be the cache size of node i (in units of chunks). Denote by K_i the set of cached items at node i, and let N_i be the set of neighbouring nodes of node i that are within its cooperation scope. The implications of N_i are two-fold: (1) node i can own the view of cached items of nodes in N_i and then can utilize them to serve its locally-unsatisfied requests; (2) from node i's perspective, for each cached item $k \in K_i$, if there is already one copy of k in any node of N_i, it can safely purge item k from its cache to release one slot of cache space.

Let x_{ik} ($i \in N, k \in K_i$) be the 0-1 decision variable that indicates if node i should keep item k, when $x_{ik} = 1$), or evict it, otherwise. Then $(S_i - \sum_{k \in K_i} x_{ik})$ is the amount of free space released by the cooperative redundancy elimination in node i. Each node quantifies its benefits achieved from the eviction via utility function $U_i(\cdot)$ assumed to be twice-differentiable, increasing, and strictly concave. For example, we can use $U_i(v) = \ln(1+v)$, $v \geq 0$ to achieve the proportional fairness. The intra-AS cooperative redundancy elimination problem, referred as CRE-P, can be formulated as follows:

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_{i \in N} w_i U_i\big(S_i - \sum_{k \in K_i} x_{ik}\big) \\ \text{s.t.} \quad & S_i - \sum_{k \in K_i} x_{ik} \geq 0, \ \forall i \in N \\ & x_{ik} + \sum_{j \in N_i, k \in K_j} x_{jk} \geq 1, \ \forall i \in N, k \in K_i \\ & x_{ik} \in \{0,1\}, \ \forall i \in N, k \in K_i \ . \end{aligned} \tag{1}$$

The first constraint is the storage capacity constraint. The second constrain represents the "*lossless redundancy elimination*" principle, i.e., the guarantee that the cache hit rate of a node will not deteriorate after elimination, as the original locally cached item will either still reside in the current node or can be found in one or many neighbours in N_i.

3.2 Modelling Issues

- *What is the motivation for the objective function?*

The objective function of (1) is the weighted sum of utilities that characterize the gain from the procedure of redundancy elimination in terms of released caching slots. Similar to [17], we distinguish two types of nodes in the AS: i) *access nodes*, which serve as the-first-hop nodes directly receiving requests from end-users; and ii) *intermediate nodes*, which connect access nodes to the outside world. Due to the "*cache filtering effect*" in CCN, the caching performance of access nodes plays the major role in the systematic caching performance [17]. Therefore, we value more the gain achieved at access nodes than that of intermediate nodes by setting a higher weight w_i for access nodes. Furthermore, the concavity of the utility function helps impose a certain fairness on node gains.

- *Why narrow the scope of cooperation to be one-hop?*

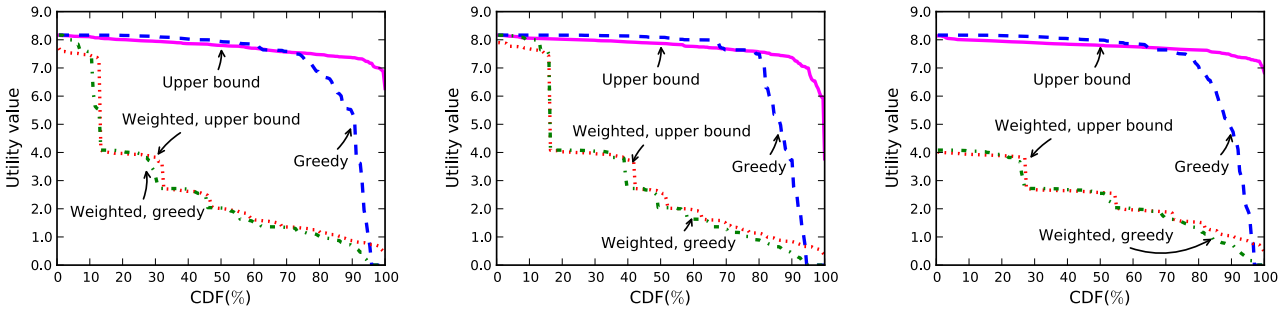
What is the optimal number of copies for each cached item? Where should each copy be placed? The accepted redundancy level reflects the trade-off between the system goal of reducing cross-traffic by maximizing the diversity of cached items and the performance goal of minimizing the access latency. Acquiring the optimal degree of duplications itself is a challenging problem [8]. Moreover, previous studies on cooperative caching [4, 7] reveal that this problem is tied with the network topology and needs to be handled jointly with request routing, rendering the proposed solutions often non-scalable. To avoid being trapped in such a dilemma, we circumscribe the scope of redundancy elimination to be the one-hop neighbourhood. The benefits of one-hop cooperation are four-fold. First, it helps control the AS-level redundancy by eliminating duplicates of small groups. Second, it frees us from routing concerns – to discover where purged items have been preserved. Third, this simplification reduces the amount of signalling traffic for cooperation and enforces no change to the native routing strategy in CCN. Fourth, the prolonged access latency of those requests for evicted items would be subtle, as the additional transport distance is only at most one-hop longer.

3.3 Greedy Heuristic

CRE-P is a Nonlinear Integer Programming problem and can easily be shown to be NP-hard. In particular, with no fairness consideration ($U(\cdot)$ being linear) and in the absence of storage capacity constraints, it degenerates to solving a sequence of the Minimum Dominating Set (MDS) problems – MDS is already NP-hard. More specifically, each separate cached item k leads to an induced graph $G_k = (N, E_k)$ of $G = (N, E)$, where $(u, v) \in E_k$ if $(u, v) \in E$ and k is stored at both u and v. Then we need to calculate the MDS for each induced graph accordingly.

This inspires us to ponder over the relationship between cooperative redundancy elimination problem and the minimum dominating set problem. On one hand, as we know, the simple greedy heuristic leads to a $\ln\Delta$-approx[2] solu-

[2] $\ln\Delta$ can be safely claimed to be the smallest approximation ratio for any polynomial time algorithm.



(a) AS-1755 (b) AS-3967 (c) AS-brite1

Figure 2: The upper part shows the CDF of node utility values achieved by the optimal solution to the integer-relaxed cooperative redundancy elimination problem (*Upper bound*) and the greedy heuristic across three topologies; the lower part shows the CDF of the weighted utilities.

tion to the MDS, where Δ is the largest node degree of the graph. The greedy heuristic is inclined to pick those nodes with high degrees. On the other hand, concerning the cooperative redundancy elimination, we have the following two observations:

- Nodes with high degrees tend to be intermediate nodes that are of less importance to the systematic caching performance, compared to access nodes with small degrees.
- A node with high degree locates in a better position for eliminating redundancy, since preserving one copy of the duplicates there can serve more neighbouring nodes.

We are encouraged to apply the same greedy heuristic to CRE-P, due to the above two observations and in viewing that an efficient and practical implementation – working in a locally distributed manner – of the greedy algorithm for MDS has been proposed by F. Kuhn et at. in [11]. Note that the locally distributed implementation is critical to the practicality of intra-AS cache cooperation scheme.

4. PERFORMANCE EVALUATION

The performance evaluation consists of two parts. We first examine the greedy heuristic in yielding an approximate solution to CRE-P. Then we explore the core benefits brought by intra-AS cache cooperation scheme from different aspects.

4.1 Experimental Setup

Network topologies. Considering that network topologies keep evolving, we evaluate the intra-AS cache cooperation scheme under both realistic and synthetic topologies so as to better understand the potential impacts of topology on the performance and thus to verify the applicability of the cooperation mechanism. We extract two router-level topologies from the Rocketfuel project [15]: AS 1755 (Ebone, in Europe) and AS 3967 (Exodus, in US). In addition, we use BRITE [1] to generate two hierarchical top-down topologies: Brite1 and Brite2. Table 1 gives further details on the four topologies.

Workloads. The essence of caching is utilizing the redundancy of workloads and its performance heavily depends on the concrete traffic characteristics. This highlights the importance of using realistic workloads in evaluating the caching performance of CCN. ProWGen [5] characterizes well the observed realistic Web workloads: zipf-like popularity distribution and the temporal locality. We generate two groups of traces for access nodes of the four topologies: (1) for Trace-1, the zipf-slope is randomly chosen in $[0.90, 0.96]$, while for Trace-2, it is chosen from $[0.70, 0.76]$; (2) the number of requests at each access node is $6,000$; (3) the object size is normalized to be $[5, 20]$ chunks; (4) the percentage of one-timers is $[60, 70]$%; and (5) the percentage of unique objects is $[35, 40]$%. This setting reflects in our opinion different traffic characteristics of different types of content, particularly in terms of content popularity distribution.

Table 1: Network Topologies

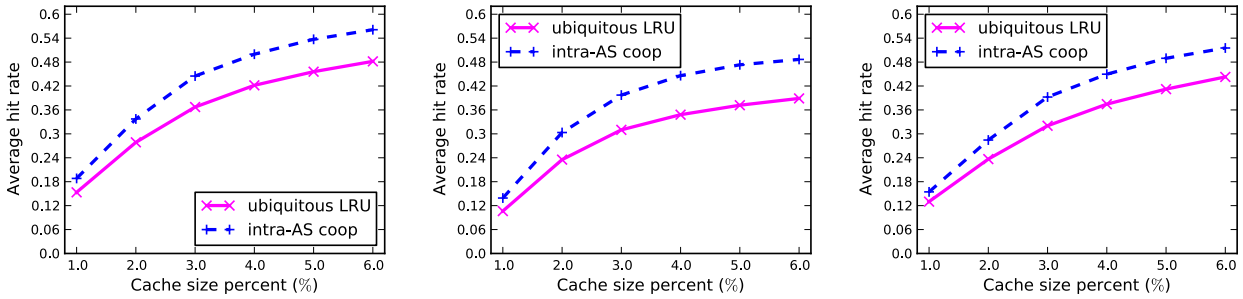
Topology	Nodes	Edges	Max degree
AS 1755	172	381	15
AS 3967	201	434	15
Brite1	200	404	16
Brite2	200	404	11

CCN simulator. We have implemented a simplified CCN model in OmNet++ [16]. The model incorporates the three basic components of CCN [9] – CS, PIT, and FIB – and supports chunk-based caching, per-chunk request, and request aggregation. Though CCN supports multi-path routing[3], at present we only evaluate our strategy under single-path routing and assume the shortest-path-routing by convention. On behalf of its served clients, each access router issues requests for objects. In the simulation, we assume the arrival process of requests for objects at each access router follows a Poisson process. In CCN, a request for the object will be translated into a sequence of requests for chunks and the sending window w is set to be 1, meaning only after receiving one chunk will the request for the next chunk be issued. In the following experiments, the object size is normalized to be the number of chunks (10 KB).

4.2 Efficiency of the Greedy Heuristic

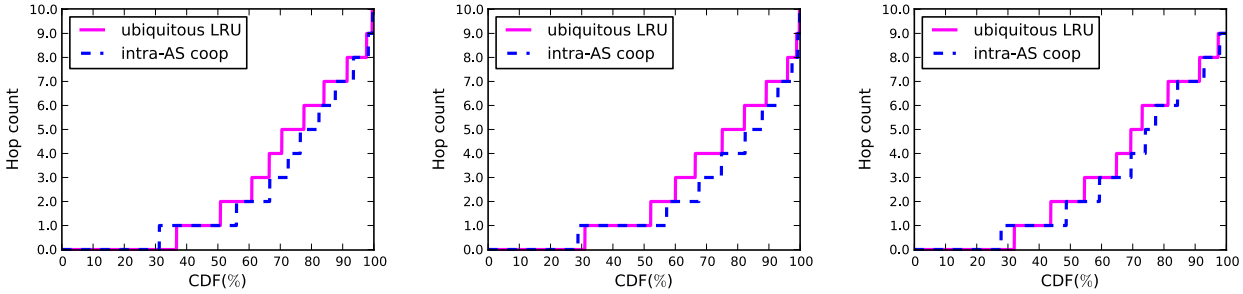
Without being able to derive the approximation ratio of the *greedy* heuristic theoretically, we evaluated its performance by comparing it with an upper bound of CRE-P. By relaxing the integral constraints of x_{ik}, CRE-P degenerates into a convex optimization problem (classed CRE-PR). The optimal solution of CRE-PR provides an *upper-bound* on

[3]It will incur more serious redundancy within the AS.



(a) AS-1755 (b) AS-3967 (c) AS-brite1

Figure 3: The average cache hit rate of access routers with varying cache size under Trace-1 (with high value of the zipf slope).



(a) AS-1755 (b) AS-3967 (c) AS-brite1

Figure 4: The CDF of served hop counts of chunk requests when the relative cache size is 3%, under Trace-1 (with high value of zipf slope).

that of CRE-P. To solve CRE-PR optimally, we derived a decentralized algorithm[4] by using non-linear Gauss-Seidel procedure and Dual Decomposition.

The set of cached items at each node comprises one input to the cooperative redundancy elimination problem (CRE-P), that is, a snapshot of the dynamic caching network of CCN. The following experiments use as input the results obtained from the most frequently used (MFU) strategy. In addition, based on our previous observations, we set $w_i = 1/d_i$ where d_i is the node degree.

Since CRE-P seeks to maximize the sum of weighted utilities, *utility value* serves as the natural performance metric. Figure 2 shows that the *greedy* heuristic favours a large portion of nodes at the expense of sacrificing a small portion of nodes, in terms of the node *utility*. In contrast, the *upper-bound* brings forth a fair outcome, as expected. We found that those sacrificed nodes were intermediate node with high degrees and their portion depends on the input (caching status) and the AS topology: 25% in AS-1755, 20% in AS-3967, and 28% in Brite1 and Brite2. Only 5% in AS-1755, 6% in AS-3967, 4% in Brite1, and 1% in Brite2 were extremely sacrificed. In respect of *weighted utilities*, the two curves are close to each other. We believe this demonstrates that the *greedy* heuristic does work well in achieving a competitive result for CRE-P.

4.3 Core Benefits of Cooperation Scheme

The intra-AS cache cooperation scheme dedicates to controlling the redundancy level within the AS through eliminating duplicates among one-hop neighbouring nodes. It also empowers a node to make further use of the cache space of its neighbours. Thus, it helps to enhance the utilization of limited caching resources in the AS. To evaluate its performance, we compare it to the default *ubiquitous LRU* with varying cache sizes. We conduct experiments under two groups of traces and the cooperative redundancy elimination period is set to 120s and 180s. Since similar trends are observed, we only present here the results on Trace-1 with the cooperation period of 120s.

4.3.1 Cache hit rate

The *hit rate* often serves as the performance metric in caching. In our case, we count a request as a cache hit when the requested chunk is either locally stored or is successfully fetched from one of the neighbours. Figure 3 shows that the intra-AS cache cooperative scheme can improve the hit rate of access routers across the four topologies (result on Brite2 is not presented in the figure). Such improvement becomes more prominent when the cache size increases. This is because a larger cache indicates a higher chance of redundancy within the AS, making the cooperation scheme benefit more from the redundancy elimination.

4.3.2 Bandwidth saving

CCN promises to offload the traffic with in-network caching. In our experiments, each AS has one outgoing link to the outside (content server). Those unsatisfied requests by caches within the AS have to fetch the data from the content server (or a nearby AS), yielding the cross-traffic, which the AS-operator wants to minimize. We measure the amount of cross-traffic in GB. Table 2 summarizes the results. With

[4]The CRE-P problem in large scale favours a decentralized algorithm that can be executed in parallel.

Table 2: The amount of cross-traffic (in GB) with varying cache size (in percentage) across the four ASs under Trace-1.

AS	Method	1%	2%	3%	4%	5%	6%
1755	ubi-LRU	4.06	2.70	1.57	0.69	0.34	0.19
	AS-coop	3.91	2.47	1.25	0.47	0.21	0.12
3967	ubi-LRU	5.33	3.62	1.97	0.83	0.37	0.20
	AS-coop	5.13	3.15	1.40	0.42	0.18	0.10
Brite1	ubi-LRU	4.73	3.38	2.06	1.16	0.70	0.45
	AS-coop	4.61	3.19	1.77	0.93	0.53	0.31
Brite2	ubi-LRU	4.86	3.08	1.44	0.54	0.24	0.12
	AS-coop	4.67	2.70	1.00	0.33	0.15	0.11

a larger cache size, the default *ubiquitous LRU* already offloads a significant amount of cross-traffic. Since the chunk size is set to 10 KB and each access router issues $6,000$ requests for objects (with size $5 \sim 20$ chunks), the small value difference, shown in Table 2, corresponds to a great number of data chunks. Therefore, the *intra-AS cache cooperation* scheme performs better in terms of bandwidth saving.

4.3.3 Average hop count of requests

Compared with *ubiquitous LRU*, cache cooperation tends to generate more internal traffic apparently. Thus it takes a potential risk of overloading internal links. To investigate the degree of increased internal traffic arising from the cache cooperation, we measure the distribution of the hop count needed to satisfy each request, since the hop count characterizes the amount of traffic that each chunk request yields within the AS. Figure 4 shows the CDF of hop counts of chunk-requests when the cache size is 3%. The *ubiquitous LRU* causes a larger portion of requests served with 0-hop, while the *cooperation scheme* achieves a larger portion of requests served with 0-hop or 1-hop. The area under the curve reflects the gross amount of internal traffic. To our surprise, *intra-AS cache cooperation* scheme turns out to reduce the internal traffic.

5. CONCLUSIONS

In this paper, we studied the caching problem in CCN following a different approach, by advocating cooperative redundancy elimination after data have been cached.

Two benefits are reaped from the *intra-AS cache cooperation* scheme: the caching slots released from redundancy elimination can be used to cache other popular items; and a broader view of cached items at neighbouring nodes helps to serve locally unsatisfied requests. Initial results of trace-based simulation show that: (i) the simple greedy heuristic is very efficient in eliminating redundancy; and (ii) the intra-AS cache cooperation scheme improves the caching performance of access routers and reduces the AS cross-traffic without overloading the internal links.

In the future, we aim to extend the results by combining the *vertical* redundancy elimination approach with the *horizontal* cache cooperation scheme.

6. ACKNOWLEDGMENTS

This work is supported in part by grant: HKUST RPC 11EG17.

7. REFERENCES

[1] Brite. `http://www.cs.bu.edu/brite/`.

[2] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker. Packet caches on routers: the implications of universal redundant traffic elimination. In *ACM SIGCOMM'08*, 2008.

[3] A. Anand, C. Muthukrishnan, A. Akella, and R. Ramjee. Redundancy in network traffic: findings and implications. In *ACM SIGMETRICS'09*, 2009.

[4] I. Baev, R. Rajaraman, and C. Swamy. Approximation algorithms for data placement problems. *SIAM J. Comput.*, 38:1411–1429, Aug. 2008.

[5] M. Busari and C. Williamson. Prowgen: a synthetic workload generation tool for simulation evaluation of web proxy caches. *Comput. Netw.*, 38(6):779–794, Apr. 2002.

[6] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack. Wave: Popularity-based and collaborative in-network caching for content-oriented networks. In *INFOCOM Workshops*, 2012.

[7] J. Dai, Z. Hu, B. Li, J. Liu, and B. Li. Collaborative hierarchical caching with dynamic request routing for massive content distribution. In *IEEE INFOCOM'12*, March 2012.

[8] K. Hosanagar and Y. Tan. Cooperative cashing? an economic analysis of document duplication in cooperative web caching. *Info. Sys. Research*, 23(2):356–375, June 2012.

[9] V. Jacobson, D. K. Smetters, J. D. Thornton, M. Plass, N. Briggs, and R. L. Braynard. Networking named content. In *ACM CoNEXT'09*, 2009.

[10] T. Koponen, A. Ermolinskiy, M. Chawla, K. H. Kim, I. Stoica, B. gon Chun, and S. Shenker. A data-oriented (and beyond) network architecture. In *ACM SIGCOMM'07*, 2007.

[11] F. Kuhn and R. Wattenhofer. Constant-time distributed dominating set approximation. In *ACM PODC'03*, pages 25–32, 2003.

[12] Z. Ming, M. Xu, and D. Wang. Age-based cooperative caching in information-centric networks. In *INFOCOM Workshops*, pages 268 –273, March 2012.

[13] I. Psaras, W. K. Chai, and G. Pavlou. Probabilistic in-network caching for information-centric networks. In *ACM SIGCOMM Workshop ICN'12*, 2012.

[14] S. Sanadhya, R. Sivakumar, K.-H. Kim, P. Congdon, S. Lakshmanan, and J. P. Singh. Asymmetric caching: improved network deduplication for mobile devices. In *ACM Mobicom'12*, pages 161–172, 2012.

[15] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring isp topologies with rocketfuel. *IEEE/ACM Trans. Netw.*, 12(1):2–16, Feb. 2004.

[16] A. Varga. Omnet++. `http://www.omnetpp.org`, 2012.

[17] J. M. Wang and B. Bensaou. Progressive caching in ccn. In *IEEE GLOBECOM'12*, Dec. 2012.

[18] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos, and et al. Named data networking (ndn) project. *October*, pages 1–26, 2010.