Towards improving the efficiency of ICN packet-caches

Yannis Thomas and George Xylomenos
Mobile Multimedia Laboratory, Department of Informatics
School of Information Sciences and Technology
Athens University of Economics and Business, Greece
Email: thomasi,xgeorge@aueb.gr

***Abstract*—In-network packet-level caching is one of the most promising features offered by Information-centric Networking (ICN) architectures. In ICN, routers can use their queueing buffers as temporal storage units, thus allowing on-path caching by exploiting the network's storage resources. Packet-caches can be highly beneficial for content delivery, but they are also known to have three significant weaknesses: (i) packet-granularity produces huge cache indexes, (ii) Zipf-like content popularity penalizes the hit-ratio at core nodes and (iii) any discontinuity in the stored packets disrupts RTT-based congestion control. This paper presents OPC, a novel caching management strategy designed to support wire-speed in-network caching, while dealing with the above problems. OPC works at the object-level, thus reducing indexing requirements, is destined for access routers, thus avoiding the small hit-ratios of caches at core-nodes, and stores contiguous groups of packets, thus easing RTT-based congestion control.**

I. Introduction

The redundancy in Web traffic and the impact of caching has been the topic of many studies [1], [2]. The common conclusion of these works is that the fraction of cacheable content is severely reduced by the object-level structure of application caches. Therefore, it is suggested that caching in the Web should be undertaken by application-independent caching modules that detect and eliminate redundancy at the packet-level [3], [4]. Nevertheless, the IP infrastructure requires specialized caching nodes located at the edges of predefined links, thus limiting the impact of caching.

On the other hand ICN architectures, such as Publish-Subscribe Internet (PSI) and Named Data Networking (NDN) [5], claim in-network packet-level caching as one of their most promising native features. In these cases, network routers utilize their queueing buffers as short-term caches that operate on autonomous packets. ICN transport protocols, such as [6], [7], adopt the pull model, according to which the receiver sends numerous atomic and self-identified packet-requests to the content provider, who responds with the corresponding packets (the actual data). Both requests and responses contain two IDs; the content's unique ID and the packet's sequence ID, thus allowing packet-level redundancy detection at on-path caches.

This research has been co-financed by the European Union (European Social FundESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: Aristeia II/I-CAN.

Nevertheless, numerous studies have questioned the benefits of in-network packet-caching in ICN. In [8], the authors investigate the implementation of the NDN architecture on current hardware and claim that the capacity of a single-chip SRAM memory, the fastest memory type used in routers, is not adequate for holding packet-level indexes. Furthermore, in [9] authors state that in-network caching is ineffective when content popularity follows a Zipf-like distribution, which is the case of web traffic, therefore it is no better than locating caches exclusively at the access nodes of the domain.

II. The OPC approach

A. Design Goals

This paper presents the *Object-oriented Packet Cache* (OPC), a novel packet-level caching management strategy for stand-alone network caches, whose basic goal is to reduce indexing costs, so that the cache index can fit in router SRAM. Thereupon, we introduce a novel replacement policy, that shrinks the size of the cache index by keeping *one entry per content object* - instead of one entry per packet - via a simple and quick structure for data storage.

B. Insertion and Eviction Algorithm

To achieve our design goals, we formulate a replacement policy that stores a compact part of the content, ranging always from the first packet to an unspecified size. In OPC, a cache-enabled router inspects all passing-by traffic and stores a packet-response when one of two conditions is satisfied: (i) it is the first packet of the content, (ii) the previous packet is already stored. Along these lines, when the cache is full and a new packet needs to be stored, OPC evicts the last stored packet of the *least important* content. Defining the least important content is not crucial for our design, hence the cached contents may be organized in an LRU, LFU or FIFO structure.

OPC maintains a hash table called $index\_map$ which maps the unique ID of a content ($content\_id$) to a pair of values: the ID of the last stored packet ($packet\_id$) and a pointer to the first packet in memory (mem_ptr). The $packet\_id$ is a counter that gets incremented at packet insertions and decremented at packet evictions. In other words, the $packet\_id$ denotes the number of stored packets or, equivalently, the sequence number of the last cached packet for that content.

When a packet-request is received, the caching node parses the packet's $content\_id$ and $packet\_id$ and searches the

QSHINE 2014, August 18-20, RHODES, Greece
Copyright © 2014 ICST
DOI 10.4108/icst.qshine.2014.256370

$index\_map$ for that object. If the $index\_map$ returns a $packet\_id$ value that is bigger or equal to the $packet\_id$ of the request, then the packet-response can be found using the pointer mem_ptr; assuming fixed-sized content packets stored in sequence, packet i will be at $mem\_ptr + (i * packet\_size)$.

C. Two Levels of Operation

OPC defines two levels of operation. The first is the $object$ level operation, which shrinks the cache index, requiring one entry per content. This allows the placement of the cache index in router SRAM, thus supporting wire-speed performance during lookups. The second is the $packet$ level operation, which is utilized for managing the stored packets. The simplicity of the used structure, which forces the sequential storage of a compact group of packets ranging always from the first packet to an arbitrary sequence ID, not only allows quick packet retrievals, but also favors the application of TCP-like congestion control.

The most evident problem of packet caching, with regard to RTT-based congestion detection, is the difference of the communication delays from the original content provider and from the cache. The cache, which constitutes an additional transparent source of content, returns packets much faster than the original source and perturbs the time-out estimation of the receiver. Thereafter, the requesting host (i) decreases unreasonably the estimated time-out on the delivery of a cached response and inevitably (ii) considers the next packet delivered by the original source to be late. This problem is addressed by our scheme, which stores only one, the initial, compact part of a content. OPC makes sure that a false timer expiration can happen only once in every download, right after the last cached packet is delivered. Nevertheless, with TCP-like congestion control, this expiration will lead the flow to slow-start state, which is identical to launching a transfer without the intervention of caches. Consequently, the service will boost its performance on the beginning of the transfer by fetching data from the on-path cache, and will not be negatively affected by caching implications.

Finally, the greatest advantage of OPC is the boosting of cache efficiency, expressed by the hit-ratio, due to always storing the initial part of a content. Ordinary packet-caches achieve a successful hit when $d * r \leq cache\_size$, where d is the time interval between two consequent requests for the same packet and r is the rate with which distinct packet-responses arrive at the cache. A pathological case is when two consequent downloads of the same content take place and $cache\_size < content\_size$: in this situation, the second download will not get any cache hits, since the last packets of the first download have displaced the first ones, and as the second download proceeds, the first packets displace the last ones. In contrast, in this situation OPC will find cached the first part of the content that fits in the cache, since packets are always evicted from the end of the content.

D. Implications and Future Work

Despite its potential advantages, OPC has two weaknesses. Firstly, the insertion algorithm is not expected to work efficiently during multiflow transmissions. Multiflow transfers distribute data amongst multiple routes and normally the requests over each path are not sequential. In that case, the cache will either store only the first packet or no packet at all, decreasing the hit-ratio and inducing pointless computational overhead. However, if the cache is placed at access nodes or routers with high connectivity where paths are likely to converge, this problem can be mitigated.

Secondly, OPC requires a complex algorithm for allocating memory for data-storage. As described previously, the packets of the same content must be stored in sequence, thus when the first packet of a content is stored, a certain amount of memory must be allocated to that content. However, the cache module neither knows the number of packets forming a content, nor the number of packets that will pass-by and get cached. Hence, the allocation of memory per content is not trivial and can result in poor utilization of resources (allocating more memory than needed reduces memory utilization), and/or poor performance (conservative memory allocation either invokes dynamic memory rearrangements too often, or limits caching efficiency with static allocations).

III. Conclusion

We presented OPC, a novel packet-cache management strategy for ICN networks that operates on two levels and manages (i) to shrink the cache index, so that it fits in router SRAM, (ii) to interoperate with TCP-like congestion control avoiding false time-outs and (iii) to significantly increase the hit-ratio by keeping stored the first packets to be requested by a later user for as long as possible. These promising gains will be experimentally evaluated in our future work, where the improvement of packet-level caching brought by OPC will be measured and compared with existing caching solutions.

References

[1] S. Ihm and V. S. Pai, "Towards understanding modern web traffic," in *Proc. of the 2011 ACM Internet Measurement Conference*, 2011, pp. 295–312.

[2] B. Ager, F. Schneider, J. Kim, and A. Feldmann, "Revisiting cacheability in times of user generated content," in *Proc. of the IEEE Global Internet Symposium*, 2010.

[3] 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 Computer Communication Review*, vol. 38, no. 4, 2008, pp. 219–230.

[4] N. T. Spring and D. Wetherall, "A protocol-independent technique for eliminating redundant network traffic," *ACM SIGCOMM Computer Communication Review*, vol. 30, no. 4, pp. 87–95, 2000.

[5] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *Communications Surveys Tutorials, IEEE*, vol. 16, no. 2, pp. 1024–1049, Second 2014.

[6] Y. Thomas, C. Tsilopoulos, G. Xylomenos, and G. C. Polyzos, "Accelerating file downloads in publish subscribe internetworking with multisource and multipath transfers," in *Proc. of the World Telecommunications Congress*, 2014.

[7] G. Carofiglio, M. Gallo, and L. Muscariello, "Icp: Design and evaluation of an interest control protocol for content-centric networking," in *Proc. of the IEEE INFOCOM Workshop on Name Oriented Mobility*, 2012, pp. 304–309.

[8] D. Perino and M. Varvello, "A reality check for content centric networking," in *Proc. of the ACM SIGCOMM Workshop on Information-Centric Networking*, 2011, pp. 44–49.

[9] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-centric networking: seeing the forest for the trees," in *Proc. of the ACM Workshop on Hot Topics in Networks*, 2011.