Minimizing Bandwidth on Peering Links with Deflection in Named Data Networking

Chadi Barakat, Anshuman Kalla, Damien Saucez, Thierry Turletti
INRIA Sophia Antipolis-Méditerranée, France

***Abstract*—Content dissemination is the primary usage of the Internet today, whereas the existing Internet architecture based on TCP/IP is mainly designed for point-to-point communications. Information-Centric Networking (ICN) has been proposed to conciliate Internet usage and technology. The idea behind ICN is to omit the notion of host and location and establish content as the first class citizen of the network. ICN advocates in-network caching, i.e., to cache content on the path from content providers to requesters. This on-path caching achieves good overall performance but is not optimal as content may be replicated on routers so reducing the total volume of content that can be cached. To overcome this limitation, we introduce the notion of off-path caching where we allocate content to well defined off-path caches within the network and deflect the traffic off the optimal path toward these caches that are spread across the network. Off-path caching improves the global hit ratio by efficiently utilizing the network-wide available caching capacity and permits to reduce egress links' bandwidth usage.**

I. Introduction

We are moving from an Internet used to connect us to well defined hosts (e.g., FTP, telnet, POP3) to an Internet where content delivery and retrieval are prominent. Our main focus nowadays is on finding contents and consuming them without caring about where they are located. Unfortunately, the TCP/IP architecture is designed for point-to-point communications, causing a gap between the technology and its usage and reducing the efficiency of communication.

To conciliate technology and usage, *Information-Centric Networking* has been proposed [1]. ICN removes the notion of end-to-end communication by accessing content directly with its name. Instead of delivering packets to hosts with well defined IP addresses, an ICN network fetches content with some universal name without bothering the end user with tasks as finding the host where the content is located and downloading it himself. Different copies of the content can be available and the ICN architecture automatically should find and download the most appropriate copy with a single request for the content made directly to the network. Several ICN proposals came up ([1]) but even though their conception differs, they all rely on assigning a universal name (hierarchical or flat) to each content rather than identifying it by the IP address of its host. The name of a content is directly used to fetch it, while with "legacy Internet", the name is first resolved into the IP address of the hosting machine. In this paper, we focus on *Content-Centric Networking* (CCN) [2] but our study and findings can be applied to other ICN proposals.

CCN leverages in-network caching to place popular contents close to their consumers, hence reducing the load on the network and improving QoS. This opportunistic caching at routers on the path traversed by content reduces the overall content retrieval time and bandwidth consumption [3], [4] as requests for the same content only traverse a portion of the path until finding a cached copy. However, this on-path caching inside an Autonomous System (AS) (or a domain) is not optimal in terms of provider links' bandwidth usage. Indeed, on-path caching results in an uncontrolled content duplication within the AS, hence wasting cache space and reducing overall hit rate. Better overall hit rate can be obtained if caches within an AS coordinate their effort to avoid replicas and cache more contents. This raises the natural question of *how to perform caching within an AS such that provider links' usage is minimized while keeping the AS links' usage below their nominal capacities*. We address this question in this paper and propose a theoretical formulation together with a practically deployable solution.

Provider links' usage is minimized in a stub AS if it caches as many contents as possible, meaning that content replicas must be avoided. [5], [6] propose to reduce the number of replicas inside an AS while keeping the on-path caching assumption. Our claim is to realize a better gain by allowing traffic deflection (i.e., indirect routing) within the AS in such a way that all requests for the same content traverse the same cache. Therefore, if the content is cached, then it is certain that whichever client (associated with the AS) asks for it, the request will eventually arrive at the cache handling the content. This avoids content replication and allows optimal usage of overall cache space, hence maximizing the overall hit rate and minimizing the usage of provider links. We assess the benefit of our approach with simulations on Rocketfuel topologies [7]. We show that *optimal off-path caching* performs far better than on-path caching. However, the optimal is hardly computable in real environments so we propose a practically deployable placement heuristic called *Caching All Contents by Hashing* (CACH) that utilizes a hash function to deterministically determine in which cache content must be placed.

This paper is organized as follows. Sec. II provides background on CCN and overviews early work on in-network caching. Sec. III discusses optimal off-path caching and presents an optimization framework to minimize the provider links bandwidth usage in stub ASes, then defines CACH, a practical approximation of the optimal. Sec. IV illustrates the pros and cons of each approach with simulations on real topologies and the potential gain obtained when content is not replicated. Sec. V concludes this work.

II. State of the art

A. CCN in a nutshell

Content-Centric Networking proposed by Jacobson et al. in [2] is a future networking paradigm with the key idea of establishing the content, identified solely by its name, as self-sustaining entity while excluding the notions of host and location. Entire communications in CCN are carried out by two types of packets: (i) *Interest*, being generated and sent by a CCN client when requesting a content, and (ii) *Data*, containing the requested content, being sent in response to an Interest. Every CCN router maintains three data structures: (i) *Pending Interest Table* (PIT) that maintains a list of interested interfaces (called faces) for each unsatisfied content request, (ii) *Forwarding Information Base* (FIB) that holds a list of faces that can potentially serve a requested content, and (iii) *Content Storage* (CS) that retains a copy of content (in the limit of available storage) even after forwarding it to requesting faces. CS enables in-network caching transforming every CCN router into a legitimate source of data. An Interest packet is forwarded by intermediate CCN routers based on their FIBs while updating their PITs so that to be able to later match content to requesting faces. In return, a content is sent by the origin server or any CCN router encountered over the path followed by the Interest packet and holding a copy of the requested content. The Data packet back-tracks the path laid down by the Interest packet at each intermediate router and the carried content can be eventually cached at those routers.

B. In-network caching

Chai et al. in [5] show that it is possible to obtain a better hit rate, and hence less traffic on provider links, by constraining the location where content can be cached in CCN. In [5], content is cached on well chosen routers on the path between producers and consumers, improving so the hit rate. In addition, [8] studies the impact on performances of homogeneous and heterogeneous cache sizes in CCN and shows that allocating content store space heterogeneously across the CCN network brings very limited gain and that the simple *degree centrality* metric to decide where to store content provides good performance. Finally, in [6], caching entities collaborate to ensure that globally popular contents are cached. However, in all these solutions contents are always cached on the path between consumers and providers, yielding content replication because of lack of collaboration between paths under utilization of the total caching capacity, and suboptimal traffic on provider links.

Even though caching problem is brought to the light today with the emergence of Information-Centric Networking, the problem of caching and more particularly the routing between caches, has been broadly studied for hierarchy of web caches [9], [10], [11]. Among those proposals, the work from Ross in [10] is the closest to ours. Indeed, Ross discusses hash-based routing protocols which use hash functions to select web caches from where to retrieve objects in a hierarchy of caches.

Finally, Iyer et al. use a P2P overlay to build a decentralized web cache and the evaluation shows performance similar to centralized web caches [12]. We somehow confirm this result by showing that deflection to select caches in a stub network can outperform solutions without deflection.

III. On optimal caching in a network domain

Ubiquitous in-network caching offered by CCN at the router level [2] raises the question of the optimal placement of the different contents over caches available between content consumers and content producers. Generally speaking, for the set $\mathbf{C}$ of contents and the set $\mathbf{R}$ of caching routers, optimal content placement over caches amounts to finding a function $f : \mathbf{C} \rightarrow \{0,1\}^{|\mathbf{R}|}$ that provides, for each content, a binary vector whose value at position i determines whether the content must be cached at router i or not. Function f must fulfill some optimization requirements and is thus determined by an objective function and a set of constraints. Our objective is to minimize the total bandwidth usage on the provider links of a stub AS (e.g., enterprise network, campus network) by means of in-network caching. This objective stems from the fact that the cost for a stub network of using its provider links is much higher than the one of operating local links, hence the benefit of satisfying the maximum of clients' interests locally. To realize this objective, we must ensure that (i) routers cache the maximum number of contents by caching each content at most at one place inside the domain, and (ii) the cache hit rate is maximized by caching most popular contents. These two conditions can be formulated as in Proposition 1, see [13] for proof and more details.

Proposition 1: Assuming contents of equal size, the amount of inter-domain content-retrieval traffic of a stub AS operating CCN that can cache N contents is minimized if the N top most popular contents are cached.

It follows that *on-path caching*, the default behavior of CCN, is not optimal from a provider-link bandwidth-usage point of view. With on-path caching, contents are cached over paths between content consumers and providers. Since these paths are not the same for all consumers, contents can find themselves cached at different places inside the network, which violates Proposition 1.

To implement Proposition 1, each of the N most popular contents must be allocated to one and only one router and the remaining contents must be retrieved from outside the domain without being cached. Therefore, Interests for the N most popular contents must be redirected to the caching router associated to the content. As the caching router associated to the content might not be on the shortest path between the content consumer and producer, *deflection* has to be used. With deflection, Interest packets for a content follow an indirect path that passes by the associated router. Over this indirect path, they are only handled by the CCN router associated to the content and cross the other routers in a transparent way. We call caching that stems from deflection *off-path caching*.

Whereas [5] keeps caching contents over shortest paths and optimizes the placement of contents over them by correlating the demand on other paths, we target an optimal, and more generally, network-wide content placement. Note that several

allocation matrices A can implement Proposition 1 leading to a minimal inter-domain traffic. The choice of A must account for two operational concerns. First, as deflection aims at concentrating traffic for a given content toward one *deflection point*, it might cause over-utilization of the links around the deflection point, especially that we are deflecting popular contents. Second, deflection induces path-stretch inside the AS and consequently, internal delays and traffic may increase. One can be interested in finding the subset of matrices A that limits the internal delays and makes the intra-domain traffic increase under some constraints. Note that internal delays are different from delays perceived by consumers, as the latter ones depend on the hit rate as well, which we are minimizing. Simulation results in [13] indicate that consumers' delays can in overall decrease if one accounts for the gain in the hit rate and the external delays to reach the servers.

Next, we present one possible mathematical formulation of this optimization problem, which is NP-*complete* in general, but that can be reduced to a simple linear problem if internal links capacity is omitted. In Sec. III-B, we propose a practically doable solution that is a tradeoff between practical deployment issues and the theoretical optimal.

A. Optimal content placement

Denote by μ_r the caching capacity in router r, μ_r contents among the N most popular ones must be deflected to (and hence cached at) r. Moreover, the deflection must be such that link capacity is not exceeded and delay is minimized. Minimizing the delay can be seen as a way to minimize internal traffic as well. Our optimization problem thus aims to find an allocation matrix A such that: (i) only the N most popular contents are cached, (ii) the deflection does not cause internal link over-utilization and, (iii) the overall delay toward deflection points is minimized. Formally, this gives the following optimization problem, where the allocation matrix $A = (A_{r,c}), r \in \mathbf{R}, c \in \mathbf{C}$ is expected as output:

$$\text{minimize} \sum_{c \in \mathbf{C}^*} \sum_{e \in \mathbf{E}} \lambda_{c,e} \sum_{r \in \mathbf{R}} A_{r,c} d_{e,r} \tag{1}$$

$$A_{r,c} \in \{0,1\}, c \in \mathbf{C}^*, r \in \mathbf{R}, \tag{2}$$

$$A_{r,c} = 0, \forall c \in \mathbf{C} \setminus \mathbf{C}^*, \forall r \in \mathbf{R}, \tag{3}$$

$$\sum_{r \in \mathbf{R}} A_{r,c} = 1, \forall c \in \mathbf{C}^*, \tag{4}$$

$$\sum_{c \in \mathbf{C}^*} A_{r,c} = \mu_r, \qquad \forall r \in \mathbf{R}, \tag{5}$$

$$\sum_{c \in \mathbf{C}} \sum_{e \in \mathbf{E}} \lambda_{c,e} \cdot v_c \cdot \delta_{l,e,c,A} \leq \kappa_l, \qquad \forall l \in \mathbf{L}. \tag{6}$$

Here, $\mathbf{C}^* \subseteq \mathbf{C}$ is the set composed of the N most popular contents, $\mathbf{C} \backslash \mathbf{C}^*$ being the complementary set. N is set equal to the overall network caching capacity, $\sum_{r \in \mathbf{R}} \mu_r = |\mathbf{C}^*| = N$. $\lambda_{c,e}$ is the demand for content c in Interest packets/s seen at edge router e. $\mathbf{E}$ is the set of edge routers directly connected to the CCN clients and from where the requests enter the network. $d_{e,r}$ is the shortest path length between e and r. $\mathbf{L}$ is the set of network links and for each link, κ_l is its capacity and $\delta_{l,e,c,A}$ is a binary value that indicates whether the link l is on the path between the edge router e and the deflection point. Finally, v_c models the size of content c.

Eq. (1) models the mean internal delay of the top-N popular contents to be minimized in our search for the optimal placement among all those allocation matrices A that minimize the egress traffic. The other equations are the constraints for our optimization problem. Eq. (2) indicates that a popular content is either cached (i.e., 1) or not (i.e., 0). Eq. (3) indicates that non popular contents must not be cached by any router. Eq.(4), combined with Eq. (2), imposes that a content in the list of top-N popular contents is cached once and only once in the network. Eq. (5) imposes that the caching capacity μ_r of a router is never exceeded, and that the overall caching capacity of the network is exploited. Finally, Eq. (6) ensures that no link is used more than its capacity by both the deflected and non deflected contents (hence the sum of the set $\mathbf{C}$).

Our optimization problem can be casted as a general problem of finding paths subject to multiple constraints, and hence is NP-*complete* [14]. If we remove the bandwidth constraint in Eq. (6) and assume similar popularity distribution over edges, the problem can be made of linear complexity and solved exactly as follows. First, we calculate for each router r its mean delay to all edges waited by the traffic demand per edge, i.e., $\sum_{e \in \mathbf{E}} \lambda_e \sum_{r \in \mathbf{R}} d_{e,r}$. Using this delay metric, routers are ranked from most central to least central, and contents are allocated to them by decreasing order of popularity (most popular on most central, least popular on least central, etc). We detail this heuristic the evaluation section.

B. CACH: Caching All Contents by Hashing

The cost of deploying optimal off-path caching requires tracking popular content, solving an NP-*complete* optimization problem, and running very large routing tables to account for the placement of each content. Instead of claiming to implement the optimal solution, we consider it as a reference solution that points to the maximum gain an operator can achieve by performing in-network caching. We start from it and construct a mechanism that requires no popularity knowledge and where routing table size is independent of N. To do so, content is still cached in one and only one place but non popular content can be cached as well. In other words, every content is deflected. As a result, caching routers will have to cache more content than their capacity, meaning that popular content might not be cached for some period of time and will have to be downloaded from the Internet. By doing so, we do not strictly respect Proposition 1 but we follow the same principle of *one content, one replica* and rely on cache eviction policies (e.g., Least Recently Used LRU) to keep most popular content cached at maximum one place inside the network.

To achieve the above goal, edge routers use a hash function to decide on the placement of every content on the routers, independently of its popularity. The hash function is applied to the content name and the returned value determines the position of the caching router for each content. If every router

knows the complete list of caching routers in the network and uses the same hash function with the same seed, then deflection respecting the principle of *one content, one replica* can be performed on-line. Practically, when an Interest packet is received by a router, the router extracts the content name, hashes it, determines the deflection point associated to the content and forwards the Interest packet toward the chosen deflection point. Similarly, when a caching router receives a Data packet, it determines whether or not it must cache the content by looking at the hash value of the content name. We call this mechanism CACH, which stands for *Caching All Contents by Hashing*. CACH is a means to randomly select one single deflection point per content independently of its popularity and in a distributed and stateless way.

Note that routing tables with CACH are bound to the number of caching routers ($\mathcal{O}(|\mathbf{R}|)$) which is no more than what is observed in intra-domain routing today in stub networks. Moreover for each ICN packet, routers have to apply a hash function on the content name, which is already commonly done in traffic monitoring and engineering, e.g., ECMP [15].

IV. Evaluation

In this section, we compare the different approaches in term of provider link bandwidth usage and discuss the impact of deflection on link utilization inside the network. For ease of presentation, we consider the variant of the problem with no limit on the capacity of internal links (the variant of linear complexity discussed in Sec. III-A). The comparison is done with an event-driven simulator we have built using Rocketfuel network topologies [7]. Each node of the topology (i.e., router) is assigned a role, a caching capacity, and a forwarding table and a weight is set to every link. Nodes are either edges, routers, or servers. Edge nodes equally generate demands for content according to a Zipf law. Routers implement caching with the Least Recently Used (LRU) policy and forwarding follows shortest paths according to link weights.

We consider three scenarios: (i) *on-path* caching (in red in all figures), (ii) optimal *off-path* caching (in green), where the *optimal placement* of the N most popular content on the routers inside the AS is pre-determined, and (iii) *CACH* where hashing is invoked to place or retrieve the popular content on the fly (in blue). Simulations are run on several Rocketfuel topologies but only the results for the ASN 3,967 topology are presented as the others show similar trend. We randomly select *edge* routers such that every city in the topology has two edge routers. We then have a total of 44 edges, the 79 remaining nodes are taken as *routers*. Among the routers, 6 have been randomly selected to be attached to provider links. We consider that content can be retrieved via the closest provider link, which is a reasonable assumption for networks using default edge routes. We consider a total of 7,900 contents with a popularity distribution following a Zipf (0.8) law [4]. We limit the caching capacity to 10 entries per router, meaning that total caching capacity of the network is 790 contents (i.e., 10% of content demand). Small cache size exacerbates competition between contents. Each scenario is simulated 11 times and each simulation generates 200,000 Interest packets.

A. Provider link bandwidth usage

The number of Interest packets that leave the AS determines the provider link bandwidth utilization. Fig. 1 shows the cumulative number of Interest packets that leave the AS as function of contents ordered by decreasing popularity. With on-path caching, 83% of Interest packets are forwarded outside the AS to fetch the requested contents, while for optimal off-path caching, only 35% of them are forwarded. CACH lies in between with 47% Interest packets forwarded. Fig. 1 also depicts that with on-path caching, 50% of the provider links traffic is generated by the first 5.5% of most popular contents while the same contents account for less than 0.7% and 22% of provider links traffic for optimal placement and CACH.

Fig. 2 complements Fig. 1 and gives per-content average hit ratio, ordered by content popularity (with 95^{th} confidence interval). With the optimal placement, the hit ratio for the most popular contents is close to one whereas it is close to zero for the remaining ones. Comparatively, the hit ratio is low with on-path caching. With CACH, the hit ratio is high for popular contents thanks to LRU but gracefully degrades when popularity diminishes. The large hit ratio variance observed for CACH is because the hash function seed is changed at every simulation, hence changing the contents allocation to routers. Depending on the allocation, contents with relatively equal popularity (e.g., many contents from the tail) might be deflected to the same router, increasing cache churn. Actually, the slight hit ratio increase for least popular contents is an artifact of the finite number of iterations. Interestingly, even though deflection potentially causes longest path within the domain, the higher hit rate reduces the overall delay. For instance, with an external delay of 100 ms, the average delay is 104 ± 0.05 ms with on-path, while it is only 34 ± 0.09 ms with optimal placement and $69 \pm .11$ ms with CACH.

B. Internal links bandwidth usage

We normalize the bandwidth usage of every internal link by the average bandwidth usage of the edge links. A value higher than one indicates that the link aggregates large amount of traffic from several edges making it central in the topology, and a value below one means that the link transports a small fraction of edge traffic. Internal links are over-provisioned to 10 times the capacity of edge links, which leaves enough margin to assess the increase in traffic they might experience.

Fig. 3 shows, for every link in the topology, the normalized bandwidth for the different schemes. Links are ordered according to the normalized bandwidth in the optimal case. Many links are not utilized and present a normalized bandwidth of 0. These are redundant links that are not part of shortest paths. In the middle, we can see about fifty links having a normalized bandwidth below 1, indicating a reasonable bandwidth usage. Surprisingly, there is less usage of these links with deflection. This result, possibly specific to our topology, comes from the higher hit rate offered by deflection. Indeed, deflection

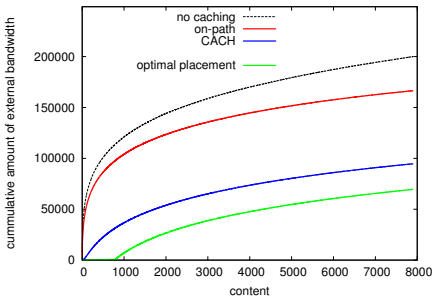

Fig. 1. Median of the aggregate traffic on provider links vs. content ordered by decreasing popularity

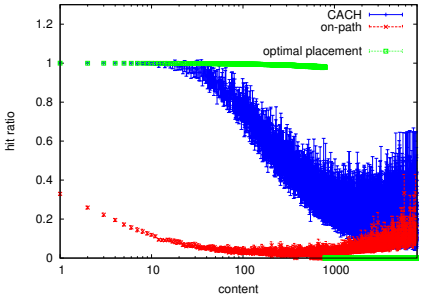

Fig. 2. Average hit ratio with 95^{th} confidence interval vs. content ordered by decreasing popularity

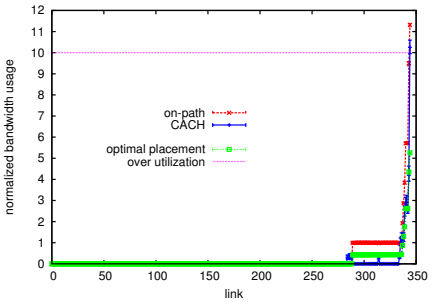

Fig. 3. Normalized internal link bandwidth usage

increases the hit rate and reduces the traffic that leaves the AS, hence contributing to the decrease in the traffic on links close to the provider.

Finally, we can see on the right-hand side of the figure, a few links that are heavily used because of their centrality in the topology and the fact they lay on several shortest paths. These can be also links close to caches storing highly popular content. Optimal placement seems improving the situation as it does its best in placing the most popular content on the most central routers having the maximum degree of connectivity, hence dispatching the traffic on the maximum number of links. CACH does as bad as on-path caching from this regard. To avoid the overload of these links, the best protection would be to take their capacity into account while placing content. This is what our constrained optimization is supposed to solve. We keep its evaluation under capacity constraints for a future research.

V. Conclusion

The existing CCN architecture is a future networking paradigm to cope with new Internet usages. We illustrate in this paper how the opportunistic caching technique advocated by CCN is not optimal in terms of inter-domain bandwidth usage as it results in wastage of limited caching space by duplicating content throughout the network. Following this observation, we propose a way to deploy an effective caching mechanism within a stub AS such that provider link usage is minimized. Our optimal off-path caching solution needs to deflect selectively the Interest packets corresponding to the cached popular contents toward optimally computed off-path routers (caches). With simulations on real topologies, we show that the cost of off-path caching in terms of intra-domain route inflation is compensated by the gain in terms of caching. When caching is optimized, a larger number of popular contents can be cached within the AS and less traffic has to exit the AS via inter-domain links. As the optimal off-path caching solution is NP-*complete*, we propose a hash-based mechanism named CACH as a practically implementable caching solution with low operating overhead to limit the number of replicas of popular content across the AS. We demonstrate by simulations that CACH is a good approximation of the optimal solution.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.

[2] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," in *Proc. of ACM CoNEXT '09*, 2009.

[3] L. Muscariello, G. Carofiglio, and M. Gallo, "Bandwidth and storage sharing performance in information centric networking," in *Proc. of ACM SIGCOMM ICN '11*, 2011.

[4] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance." arXiv, 2012.

[5] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache "less for more" in information-centric networks," in *Proc. of Networking 2012*, 2012.

[6] S. Guo, H. Xie, and G. Shi, "Collaborative forwarding and caching in content centric networks," in *Proc. of Networking 2012*, 2012.

[7] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," in *Proc. of ACM SIGCOMM IMW '02*, 2002.

[8] D. Rossi and G. Rossini, "On sizing ccn content stores by exploiting topological information," in *Proc. of INFOCOM NOMEN '12*, 2012.

[9] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web," in *Proc. of ACM STOC '97*, 1997.

[10] K. W. Ross, "Hash-routing for collections of shared web caches," *IEEE Network*, vol. 11, pp. 37–44, 1997.

[11] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281–293, 2000.

[12] S. Iyer, A. Rowstron, and P. Druschel, "Squirrel: a decentralized peer-to-peer web cache," in *Proc. of ACN PODC '02*, 2002.

[13] "Minimizing Bandwidth on Peering Links with Deflection in Named Data Networking," 2012, under submission ACM ICN 2012.

[14] Z. Wang and J. Crowcroft, "Quality of service routing for supporting multimedia applications," *IEEE Journal on Selected Areas in Communications*, vol. 14, pp. 1228–1234, 1996.

[15] B. Augustin, T. Friedman, and R. Teixeira, "Measuring Multipath Routing in the Internet," *IEEE/ACM Transactions on Networking*, vol. 19, no. 3, pp. 830–840, 2011.