Cache the Queues: Caching and Forwarding in ICN From a Congestion Control Perspective

Dinh Nguyen, Kohei Sugiyama, and Atsushi Tagami
KDDI R&D Laboratories, Japan
{nguyen, ko-sugiyama, tagami}@kddilabs.jp

***Abstract*—Caching and multipath forwarding are essential ingredients of the Information-Centric Networking (ICN) architecture resulting from ubiquitous content storage and content-based node-by-node forwarding nature of ICN. Much is yet to see how they jointly act towards improving user's quality of experience. To this end, we formulate in this paper a unified problem of caching and multipath forwarding as a network optimization problem to maximize user satisfaction which is expressed by their utility function. The formulation allows us to see caching and multipath forwarding in ICN from a congestion control perspective and to reinforce the advantage of a class of *congestion-aware* caching where in-network caches are used to absorb network congestion and to enhance user's satisfaction. In that context, multipath forwarding plays the role of directing content delivery to less congested paths where network capacity is abundant or where requested content has been stored by in-network caches. We evaluate such a congestion control-based, coupled caching and multipath forwarding approach in simulations. The result confirms the advantage of our approach compared with existing ones.**

I. Introduction

Ubiquitous in-network caching adopted in the new information-centric networking (ICN) architectures [1] is reviving research interest in caching and calling for new approaches to fully integrate it into network operations. A large body of research in ICN has been devoted to address the question of cache management, i.e. cache insertion and cache eviction policies, whose main objective is to increase cache hit and reduce server/network load. Recent work begins to move the focus on the latter, not less essential part of the problem: finding the way to fully integrate caching into network functionality.

As having been observed in the literature [2], [3], [4], without proper integration, caching could do more harm than benefit. That is especially true in ICN due to its ubiquity and the interference it causes to other network operations. One evidence is the difficulty of congestion control at the end users when chunk-based in-network caching on the content delivery paths makes conclusion on whether congestion is happening or not ambiguous [5], [6]. In-network caching, moreover, can alter fairness characteristic of the network as a result of working together with AIMD congestion control: bandwidth is allocated in favor of users of popular content and in discrimination against ones who request for less popular content [7]. In case of content streaming, caching can even make it harder for video players to select the right video bitrate [8], [9].

Another important factor in ICN cache management is how to cooperate the network of caches. General speaking, cooperative caching approaches can deliver better performance compared to uncooperative ones with the cost of signaling overhead among caching nodes and/or between caching and forwarding. It is therefore desirable to limit out-of-band signaling to the minimum to reduce implementation complexity, and at the same time, retain an effective level of cooperativeness among individual caches.

In this paper, we devise a caching mechanism which works towards both integration and cooperation issues mentioned above. Our goal is to cooperate caching and multipath forwarding in order to maximize user's satisfaction. We view caching as a facility for absorbing network congestion, and as a result, increasing user's quality of experience. To this end, we formulate a unified problem of caching, multipath forwarding, and congestion control with the objective to maximize user utility. Under the formulation, caching and multipath forwarding work in concert, coordinated by the congestion feedback signals, to minimize the congestion incurred to users and to maximize their satisfaction expressed by their utility functions.

A. Background on ICN Caching and Multipath Forwarding

Major efforts have been made following this line of research dealing with other aspects of the combined caching and multipath forwarding problem, different from the one we consider in this paper. The virtual interest packet (VIP) framework [10] employs back-pressure forwarding and popularity caching to maximize the service rate region of a network. Its objective is, therefore, to stabilize the network at high content request rates other than concerning with individual user experience. In fact, back-pressure algorithms in general are known to result in long delay for low-rate requests as it gives absolute preference to high-pressure, popular requests [11], [12]. Furthermore, one matter here is the high-speed signaling among neighboring routers and the speed of cache management and routing decisions in the time scales of milliseconds and microseconds, which places a heavy processing load on content routers. Our proposed approach avoids this overhead by exploiting the available congestion feedback signal for the purpose of caching and multipath forwarding.

Coupling of caching and forwarding has also been avocated in [13] where in-network caching is interestingly shown to have significant benefit, not by itself but jointly with forwarding. The discussion, however, tights to an ideal nearest

978-0-9883045-1-2/16 $31.00 © 2016 ITC

replica routing (iNRR) which sends requests to the nearest node where content exists and a leave-copy-down (LCD) caching policy. iNRR requires out-of-band content discovery which is essentially broadcast in nature. The model considered therein does not account for link bandwidth constraints, thus ignoring network congestion. One of the first joint multipath forwarding and congestion control optimizations in ICN [6], on the other hand, presumes a static caching scenario where content placement in caches is preset and does not change (modeled by a mapping function) and solves an optimization problem of multipath multi-source congestion control and forwarding.

In contrast to existing work, we look into the combined problem of caching and multipath forwarding from a different, congestion-centric angle. Motivated by a recent work on congestion caching [14] and promising results reported in our previous work on caching using congestion price feedback [15], we fully develop in this paper a unified framework of caching, multipath forwarding, and congestion control which serves to unite caching and multipath forwarding for the sake of user performance. As we will show in the main parts, congestion feedback helps both (1) coordinate individual caches and (2) couple caching and multipath forwarding with the goal to minimize congestion incurred to content users.

B. Contributions

1) We formulate a unified optimization problem of congestion control, multipath forwarding, and caching in ICN (Section III). To the best of our knowledge, these triple problems have never been combined so far. Such coupling allows us to use congestion feedback for the purpose of caching and multipath forwarding, eliminating the need for out-of-band signaling.
2) We illustrate the effectiveness of the coupled caching and multipath forwarding coordinated by the congestion control feedback in the network (Section IV).
3) We implement and evaluate the coupled delay-based caching and multipath forwarding scheme in realistic networks by simulations to show its advantage over existing schemes (Sections V and VI).

Compared to our previous work [15], which reported the effectiveness of congestion price as a caching criterion, in this paper we shift our focus to the interaction of caching and multipath forwarding over congestion feedback with new formulation, analysis, and evaluation results explicitly carried out in multipath settings.

II. System Model

We start with a general CCN/NDN system model as a realization of ICN. We assume a typical CCN system [16] of which users are equipped with a congestion control algorithm regulating the flow of requests for chunks of their interested content. Intermediate routers can route requests for chunks of a content object, each to one of the (multiple) faces associated with the content name. To support multipath forwarding, FIB table should allow to associate multiple faces to one content name. Routing in ICN is an active research area (see [17], [18] for examples of current approaches). In this paper, we assume routing information is pre-computed and stored in FIB of all nodes by an independent routing mechanism.

In addition to multipath forwarding, each router has a cache store controlled by a cache management by which content objects (as a whole) are chosen for storing in or purging from cache. All content objects are cacheable. We assume uncoordinated decentralized cache management.

A user requests for a content object at a consumer node which generates and sends requests for chunks of the content. An intermediate router receives requests and forwards them using the set of faces associated with the content name in its FIB table. Request forwarding is made per chunk request.[1] When receiving a packet for delivery from an ingress face, a router immediately forwards the packet to the appropriate egress face. Each egress face has a FIFO *sending queue* for keeping packets in transmission.

III. Problem Formulation

We model the network as a graph $G = \{V, E\}$ where V is the set of nodes and E is the set of directed links. A directed link i, j is from node i to node j. The graph is assumed to be connected. Let N be the set of all content retrievals. A flow, i.e. content retrieval, $n \in N$ originates from a node where the requester resides and requests for content $k \in K$ where K is the set of all content objects. Each flow n has a utility function $U_n(\cdot)$ over the flow rate. For *elastic* traffic such as file download, the utility function is increasing, strict concave, and twice differentiable. *Inelastic* traffic such as video streaming, on the other hand, can have a non-concave sigmoidal utility function.[2] The notations in Table I are used in our development. Notation n is used for a flow, i.e. user, and k for a content object.

A. The Unified Problem

We formulate a combined problem of congestion control, multipath forwarding, and caching as a utility maximization problem (Problem 1). The objective is to maximize the total utility of all users (1), with capacity constraints on links (2), and flow balance constraints at intermediate nodes (3). In multipath setting, flow rate to a user is the total rate over all incoming links. The inequality in flow balance constraint (3) addresses the case when content is cached at intermediate nodes. If a node hosts or caches a content object, the hosted or cached content is replied from that node without data coming from upstream nodes (4). The last constraint (5) corresponds to cache capacity constraint at each node where z_i^k is a binary variable indicating content k is cached at node i ($z_i^k = 1$) or not ($z_i^k = 0$) and B_i is the cache capacity of node i.

[1]For simplicity, we assume no Interest aggregation in the problem formulation and analysis but does account for such aggregation in simulations.

[2]In case of mix elastic and inelastic traffic, multipath forwarding and congestion control should be handle in a more delicate manner to ensure convergence, which requires a new class of algorithms, e.g. [19], for the problems of multipath forwarding and congestion control presented in Sec.III-B. Congestion caching (Sec.III-C), however, works with both types of traffic.

TABLE I
NOTATIONS

Notation	Meaning
N	Set of flows (users)
V, E	Set of nodes, links
K	Set of content objects
$c_{i,j}$	Capacity of link i, j
$x^n_{i,j}$,($\tilde{x}^n_{j,i}$)	Receiving rate (requesting rate) of flow n over link i, j (j, i), $(x^n_{i,j} \geq 0)$
$x^k_{i,j}$,($\tilde{x}^k_{j,i}$)	Receiving rate (requesting rate) of content k over link i, j (j, i)
z^k_i	Caching variable in {1,0} indicating content k is cached at router i or not
$k(n)$	The content requested by user n
$U_n(\cdot)$	Utility function of flow n
$L(n)$/$L_i(n)$	The set of links from the source of content $k(n)$ to user n / to router i
$L^-_k(i)$	The set of links over which content k is delivered to node i
b^k	The size of content k

Problem 1 (System)**.**

$$\max_{\mathrm{x}\geq 0} \sum_{n\in N} U_n\left(\sum_{\ell,n\in L(n)} x^n_{\ell,n}\right) \tag{1}$$

$$\textit{s.t.} \sum_{n:i,j\in L(n)} x^n_{i,j} \leq c_{i,j}\ \forall i,j \in E \tag{2}$$

$$\sum_{\ell,i\in L(n)} x^n_{\ell,i} \leq \sum_{i,j\in L(n)} x^n_{i,j}\ \forall i \in V, \forall n \in N \tag{3}$$

$$\sum_{\ell,i\in L^-_k(i)} \sum_{n:k(n)=k} x^n_{\ell,i} = 0\ \forall i \in V, \forall k \in K : z^k_i = 1 \tag{4}$$

$$\sum_{k\in K} b^k z^k_i \leq B_i\ \forall i \in V. \tag{5}$$

Note that unlike multipath forwarding in conventional network which mostly is carried out at the transport layer by end users in the form of aggregating multiple transport connections, forwarding in CCN is node-by-node: Intermediate nodes make decision on which is the next hop to forward a particular request for a content chunk. And also, what makes CCN different from conventional networks is that data go back on the reverse path over which requests come. Congestion control and multipath forwarding are, thus, performed on the request sending direction while caching is on the data receiving direction.

Since caching is assumed to work in a much slow time scale than multipath forwarding and congestion control, we subsequently decompose the problem into two sub-problems: (1) congestion control and multipath forwarding (fixing $\mathrm{z} = \{z^k_i\}$), and (2) congestion caching management (when $\mathrm{x} = \{x^n_{i,j}\}$ has reached stable values).

B. Congestion Control and Multipath Forwarding

By fixing cache variables $\{z^k_i\}$, Problem 1 becomes a multipath congestion control and forwarding problem to maximize user utility as follows.

Problem 2 (Congestion Control & Multipath Forwarding)**.**

$$\max_{\mathrm{x}\geq 0} \sum_{n\in N} U_n\left(\sum_{\ell,n\in L(n)} x^n_{\ell,n}\right)$$

s.t. (2), (3).

This is essentially a multi-source multipath congestion control and node-based multipath forwarding problem similar to the one constructed in [20], [21] which however does not consider multi-source and caching, and the one in [6] where caching is fixed, i.e. content is stored in multiple predefined locations. By converting to its dual, the problem can be broken into sub-problems which are separately solved by each user and each router in a distributed manner as shown in [20], [21]. In short, users adjust their request rates x_n to maximize their benefit, i.e. the utility minus congestion cost.

Problem 3 (User)**.**

$$\max\left(U_n(\sum_{\ell,n\in L(n)} x^n_{\ell,n}) - \sum_{i,j\in L(n)} \lambda_{i,j} x^n_{i,j}\right) \tag{6}$$

where $\lambda_{i,j}$ can be interpreted as the *congestion price* for each data unit traveling over link i, j. The second term in (6), $\sum_{i,j\in L(n)} \lambda_{i,j} x^n_{i,j}$, is thus the total *congestion cost* incurred to user n. This problem is solved by the congestion control algorithm at end-users in reaction to congestion feedback [22], e.g. using queuing delay-based TCP Vegas or loss-based TCP New Reno.

In multipath-enabled conventional networks, the problem carried out at intermediate routers is to minimize congestion cost to each destination by splitting traffic among multiple paths to that destination. At equilibrium, congestion prices over paths that carry traffic settle at the same minimum. ICN routers, however, do not know the source and destination of a flow, but only know the content being delivered. The multipath forwarding problem in ICN thus becomes minimizing the cost to retrieve each content object from upstream sources or caches. Let us look at the multipath forwarding problem solved by router i.

Problem 4 (Multipath Forwarding)**.**

$$\min_{\mathrm{x}} \quad q^k_i \triangleq \mu^k_i \sum_{\ell,i\in L^-_k(i)} x^k_{\ell,i} \tag{7}$$

$$= \sum_{\ell,i\in L^-_k(i)} \mu^k_\ell x^k_{\ell,i}. \tag{8}$$

Here q^k_i is the congestion cost and μ^k_i is the congestion price to retrieve content k at router i. Also, content delivery rate over a link $x^k_{\ell,i}$ is the aggregate of all flows which request for the content $x^k_{\ell,i} = \sum_{n:k(n)=k} x^n_{\ell,i}$. Denote $x^k_{\ell,j,i}$ as the delivery rate of content k to router i over link ℓ, j, congestion cost of content k at router i is the total cost on all upstream links over which content k is delivered to router i:

$$q^k_i = \sum_{\ell,j\in L^-_k(i)} \lambda_{\ell,j} x^k_{\ell,j,i}. \tag{9}$$

As of (8), for each content k, router i minimizes congestion cost retrieving the content by controlling the amount of traffic $x^k_{\ell,i}$ it receives from each upstream neighbor ℓ. To do that, router i chooses the optimal request rate $\tilde{x}^k_{i,\ell}$ it sends to neighboring router ℓ to minimize the cost (that is, splitting the requests it receives among the forwarding faces in FIB table). This process starts from upstream routers near the source or caches and carries on until it reaches the users of the content. The result is that, by means of multipath forwarding, congestion cost is minimized for all users of the content. One elegant solution to this multipath forwarding problem is to let each router i adjust *forwarding weights* to its next hops in order to minimize the congestion cost q^k_i by a gradient algorithm [20].

C. Congestion Caching Management Problem

Caching is, in essence, to replicate and move the source of a content closer to its users. The impact of moving contents closer to users is twofold. First, requesters of the cached content can enjoy higher delivery throughput since the congestion price of the content at the cache, i.e. μ^k_i, becomes zero. What is more, the load on upstream links $L^-_i(k)$ from the source to router i where a content object is cached decreases, allowing other users sharing the links to increase their own rates and utilities. Assuming a time-scale separation that allows congestion control and multipath forwarding to settle down, the congestion caching management problem is to figure caching variables $\{z^k_i\}$ in order to maximize total user utility:

Problem 5 (Cache Management)**.**

$$\max_{\mathrm{z}} \quad \mathcal{U}(\mathrm{z}) \triangleq \max_{\mathrm{x}\geq 0} \sum_{n\in N} U_n \left(\sum_{\ell,n\in L(n)} x^n_{\ell,n} \right)$$

s.t. (5).

This content placement problem however is combinatory in nature and requires global knowledge of user utility which is impractical in ICN given the huge number of content requests. Our goal here is to find a congestion-guided cache management which maximizes $\mathcal{U}(\mathrm{z})$ as much as possible without additional coordination and knowledge sharing among routers.

The benefit of caching content k at router i is the saving in congestion cost q^k_i associated with the content delivery from upstream nodes as expressed in (9). Given that routers know the congestion cost retrieving content q^k_i by accumulating the cost on the right hand side of (9), and denote $y^k_i = \sum_{\ell,i\in L^-_k(i)} x^k_{\ell,i}$ as the total retrieving rate of content k at router i, we are now able to formulate the following cache management problem to maximize saving in congestion cost.

Problem 6 (Congestion-Cost Caching)**.**

$$\max_{\mathrm{z}} \quad \Phi(\mathrm{z}) \triangleq \sum_{i\in V} \sum_{k\in K} \mu^k_i y^k_i z^k_i \tag{10}$$

s.t. (5).

This saving reflects itself as a reduction in congestion cost to the users, i.e. the second term in (6). Lower congestion cost, e.g. loss probability or queuing delay, in reality means a feedback signal to let the congestion control at the user increase its rate and intermediate routers to divert more traffic to the cheap paths with cached content. The users, as a result, enjoy higher equilibrium rates which maximize their benefit by solving Problem 3. In that sense, caching can be considered as a facility to reduce congestion and to move the system to a new equilibrium where total utility of all users is increased. Eqs. (7) and (10) summarize the two actions of forwarding and caching made by content routers which are coupled by the congestion cost (9) to maximize the total utility of all users.

Problem 6 can be greedily solved by letting each router store the most expensive content in terms of aggregate congestion cost over a predefined period of time T whose implementation is given in Algo.3. Our conjecture is that such *congestion-cost caching* is optimal in reducing system-wide congestion which we leave for future work for its proof. In the next section, we make a few important observations on this class of caching.

IV. Features of Congestion-cost Caching and Multipath Forwarding

Cache network analysis is generally hard and is an active research area with works analyzing common caching algorithms such as LRU, FIFO, RANDOM (see e.g. [23], [24] and references therein). Its main difficulty lies in the fact that arrival processes to intermediate caches cannot be accurately characterized. Congestion caching analysis is even more challenging for the reason that caching decisions at a node do not only depend on decisions made at other nodes but are also tied to the congestion level of the network. In this section, we instead provide some important observations on congestion caching and multipath forwarding to illustrate their behaviors under delay-based congestion control. As congestion caching works by accumulating congestion feedback signals over a comparably long period of time which allows congestion control and multipath forwarding to stabilize, it is appropriate to make an fluid analysis of the system. Such a fluid approximation simplifies the analysis and enable us to reach meaningful results.

We assume congestion control ensures fairness of bandwidth allocation. Considering delay-based congestion control such as TCP Vegas and FAST TCP, rates are allocated according to logarithm utility functions $U_n(x_n) = \alpha_n log(x_n)$ and the congestion cost can be interpreted as the total queue occupancy at all network links which a given flow occupies [25]. At equilibrium, queue occupancy allocated to flow n is equal to α_n and utility derivative balances with the congestion price. That is, $\frac{\alpha_n}{x_n} = \mu_n$ where μ_n is the mean queuing delay incurred to user n and x_n is the equilibrium rate.

A. Standalone Caches

We first consider standalone caches which in practice can be deployed in the form of caches at the network edge. We

extend the static LFU caching analysis in [26] with a concept of *queue hit*.

Assume there are K content objects, w.l.o.g. sorting in descending order of congestion cost, i.e. queue occupancy in case of delay-based congestion control, $q_1 \geq q_2 \geq ... \geq q_K$. Request rates are $r_1, r_2, ..., r_K$ and the congestion cost per request are $\tilde{q}_1, \tilde{q}_2, ..., \tilde{q}_K$ respectively. For simplicity of expression, in this section we assume same-size content objects and the cache size m is in number of objects. When a content object is found in cache, the network is *freed* from carrying the cost, i.e. queue occupancy, due to this content, thus we have a *queue hit*. Let X_i be a random variable denoting the the index of content requested by i-th request, and $Y_{k,i}$ be the indicator function of whether content k is in the cache at i-th request. The queue hit H_π of a caching policy π is thus

$$H_\pi = \lim_{t\to\infty} \frac{1}{t} \sum_{i=1}^{t} \sum_{k=1}^{K} E[\mathbf{1}(X_i = k) Y_{k,i}] \tilde{q}_k. \tag{11}$$

With an assumption that $\{Y_{k,i}\}_i$ is stationary and $E_\pi[Y_{k,i}] = p_k(\pi)$ is the probability that content k is in cache following caching policy π, we have

$$\begin{aligned} H_\pi &= \sum_{k=1}^{K} r_k \tilde{q}_k p_k(\pi) \\ &= \sum_{k=1}^{K} q_k p_k(\pi) \\ &\leq \sum_{k=1}^{m} q_k \end{aligned} \tag{12}$$

The last inequality comes from the fact that $\sum_{k=1}^{K} p_k(\pi) = m$ and can be proved using a similar technique as in [26]. By construction, queue hit rate of congestion cost caching which stores content with highest congestion cost is $H_{\text{cost}} = \sum_{k=1}^{m} q_k$. Eq.(12) is thus showing that in case of standalone caches, congestion-cost caching is optimal in relieving queue occupancy inside the network.

B. Coordinating Caches by Congestion Cost

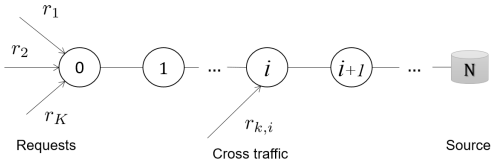

Fig. 1. A delivery path of N nodes with cross traffic.

Consider the topology in Fig. 1 which represents the case when multiple content objects are delivered along a path. Requests arrive at node 0 with intensity $r_1, r_2, ..., r_K$. There is cross traffic arriving at intermediate routers $1, .., N-1$ which is modeled by intensity $r_{k,i}$ for content k at router i ($i \geq 1$). Under delay-based congestion control, congestion cost of a content object observed at a router becomes the total residual queue occupancy which all flows carrying a content object take from the network. The residual queue occupancy of content k at router i can be expressed as

$$q_i^k = \left(r_k + \sum_{\ell=1}^{i} r_{k,\ell} \right) \sum_{j=i}^{N-1} \lambda_{j+1,j} \tag{13}$$

where $\lambda_{j+1,j}$ is the queuing delay on link $j+1, j$. As all content experiences the same queuing delay $\sum_{j=i}^{N-1} \lambda_{j+1,j}$ along the same path, (13) tells us that at each router congestion-cost caching works the same way as LFU which caches content with highest request rate $\hat{r}_{k,i} = r_k + \sum_{\ell=1}^{i} r_{k,\ell}$. The difference however lies in the way individual caches are coordinated by congestion feedback which does not exist in LFU.

Suppose content k is cached at router i due to high request rate $\hat{r}_{k,i}$. As a result of caching decision at router i, the congestion cost feedback to all downstream routers is reduced. Such lower congestion cost is, in a sense, a coordinating signal which makes the content less likely to be cached downstream. The reduction in congestion cost of content k perceived at downstream router j ($j < i$) is

$$\Delta q_j^k = \hat{r}_{k,j} \sum_{\ell=i}^{N-1} \lambda_{\ell+1,\ell} \tag{14}$$

This distinctive feature of downstream coordination by congestion feedback makes congestion-cost caching more efficient in utilizing cache storage since content is not likely cached again downstream unless it experiences significant congestion along the downstream the path.

C. Interaction of Caching and Multipath Forwarding

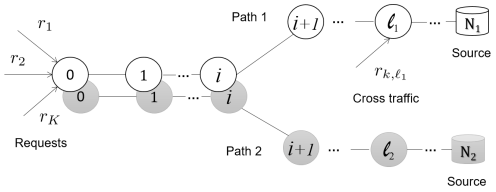

Fig. 2. Topology with 2 paths to the content sources.

Multipath topology can be interpreted as overlaid single paths as illustrated in Fig.2. Denote $\mu_{i,a}^k = \sum_{\ell_a=i}^{N_a} \lambda_{\ell_a+1,\ell_a}$, i.e. the total queuing delay from the source to node i on path $a, a \in \{1, 2\}$. As a result of delay-based multipath forwarding, traffic carrying content k is split at router i in such a way that queuing delay is the same over the two paths [3], i.e. $\mu_{i,1}^k = \mu_{i,2}^k$. Consider the case content k is cached on one path, say at node ℓ_1 on path 1, e.g. due to high total request rate $\hat{r}_{k,\ell_1}$. The effect on node i due to such caching decision is that queuing delay retrieving content k over path 1 decreases by $\Delta\mu_{i,1}^k = \sum_{j=\ell_1}^{N_1-1} \lambda_{j+1,j}$. In reaction to this change, node i forwards more requests for content k over path 1 so as to balance queuing delay on the two paths. By equaling queue

[3]Suppose both paths carry traffic at equilibrium.

occupancy of content k before caching happens at node ℓ_1 with the queue occupancy after caching, the ratio of rate increase can be shown to be approximately

$$w_i^k \propto \frac{\mu_{i,1}^k}{\mu_{i,1}^k - \Delta\mu_{i,1}^k}. \quad (15)$$

Eq.(15) shows the interaction between caching and multipath forwarding in this representative scenario: Regarding the content being cached, caching reduces queuing delay by $\Delta\mu_{i,1}^k$ over the path, and multipath forwarding takes advantage of this reduction by increasing content delivery by a factor of $w_i^k > 1$ to the path where the content is cached.

V. Implementation

A. Delay-based Congestion Control and Multipath Forwarding Implementation

A delay-based congestion control similar to TCP Vegas [27] is deployed at ICN users (Algorithm 1) which consists of two parts: slow-start (lines 4–9) and congestion avoidance (lines 10–14). We note that TCP Vegas works on queuing delay feedback and keeps queue occupancy of each flow within a preset range $[Q_{\min}, Q_{\max}]$ (lines 12 and 14). Since data packets in ICN can be replied from the source and multiple in-network caches, it is inaccurate measuring queuing delay by subtracting propagation delay $RTT_{\min}$ from RTT as originally proposed in TCP Vegas. Multipath forwarding, furthermore, makes this measurement impossible due to variation of $RTT_{\min}$ along different paths. We thus use explicit queuing delay feedback for congestion control. Accumulated queuing delay is stored in an additional field inside the data packet and updated by routers on the delivery path from the source or in-network caches to the end-user.

One important feature which needs emphasizing is that unlike loss-based congestion control, the delay-based congestion control can work more *peacefully* with multipath forwarding as the congestion cost, i.e. queue occupancy, can be summed up over individual paths. Loss-based congestion control however incorrectly perceives loss on any one path as congestion signal of the whole aggregate path and as a result reduces congestion window or rate undesirably.

For multipath forwarding, to keep the implementation simple, routers use deficit round-robin to split requests among available faces using the inverse of queuing delay as the forwarding weight of each face (Algorithm 2).[4]

B. Congestion-Cost Caching Implementation

We implement a caching algorithm to solve Problem 6 by which each router stores content objects with highest congestion costs. The cache management algorithm is given in Algorithm 3 which utilizes explicit queuing delay feedback in data packets of each content object denoted as $p(D_\tau^k)$. Intermediate routers need to accumulate the congestion cost q^k of each content object k, i.e. the total congestion cost of all data packets of that content, over a predefined period of time T. T is chosen much longer than RTT for the cost to

[4]Without anticipation, multipath forwarding is shown to have oscillation in [20]. However, the oscillation does not effect caching behavior since caches accumulate congestion cost over long-time periods.

Algorithm 1: Delay-based Congestion Control

Input: Measurements: RTT a and queuing delay q of a content retrieval

```
/* Q_max & Q_min are the max/min queue occupancy */
cwnd = 2; // initiated congestion window in chunks
ssthresh = 4; // initiated slow-start in chunks
/* Compute current queue occupancy */
queue = cwnd/a * q;
/* Adjust congestion window */
if cwnd<ssthresh then
    if queue> Q_max then
        ssthresh = ssthresh*2;
        cwnd = max(cwnd*7/8, 2);
    else
        cwnd = cwnd+1;
else
    if queue> Q_max then
        cwnd = max(cwnd-1, 2);
    else if queue< Q_min then
        cwnd = cwnd+1/cwnd;
/* Re-initiate when timeout */
OnTimeout() { cwnd = 2; ssthresh = 4;}
```

Algorithm 2: Delay-based Multipath Forwarding

```
/* Choose a face f to forward Interest for
   content k at router i */
```

Input: queuing delay q_f^k to retrieve content k over face f, and a constant threshold C_{th}

```
f = start_ptr;
while true do
    credit[f] += 1/q_f^k;
    if credit[f] ≥ C_th then
        credit[f] -= C_th;
        start_ptr = (f != last_face)? f → next : first_face;
        break;
    f = (f != last_face)? f → next : first_face;
return f;
```

Algorithm 3: Implementation of Congestion-cost Caching

Input: chunk D_τ^k of content k received at time τ and its congestion price $p(D_\tau^k)$

```
/* H is cache store of this router */
if τ > t_end then
    /* reset cost measurement */
    q^k = 0;
    t_end = t_end + T;
/* update congestion cost */
q^k = q^k + p(D_τ^k) * sizeof(D_τ^k);
if H_free ≥ sizeof(D_τ^k) then
    /* store if there is free space */
    H ← D_τ^k;
    return;
if q^k ≥ min_cost(H) then
    /* remove all chunks of cheapest content */
    /* LRU to break tie */
    m = min_cost_content(H);
    H.purge(m);
    H ← D_τ^k;
```

TABLE II
CACHE MANAGEMENT, MULTIPATH FORWARDING, AND CONGESTION CONTROL USED IN SIMULATIONS.

Cache management& Congestion control	**Explanation**
Cost: Congestion-cost caching	Cache content with highest congestion cost (Algorithm 3).
LCE: Leave-Copy-Everywhere	Cache all, LRU eviction
POP/Popular: Popularity caching	Cache most popular content [10]
Vol: Highest data volume	Cache content with highest data volume (ignoring congestion cost)
Vegas	TCP Vegas-like delay-based congestion control (Algorithm 1)

Multipath forwarding	**Explanation**
Queuedelay	Multipath forwarding using queuing delay feedback
VIP: Back-pressure forwarding	Back-pressure multipath forwarding used in [10] (chunk-based)
PI: Pending interest forwarding	Inverse of number of pending requests as forwarding weight [6]
RTT: RTT forwarding	Inverse of RTT as forwarding weight [28]
Singlepath: Shortest path forwarding	Multipath forwarding is disabled

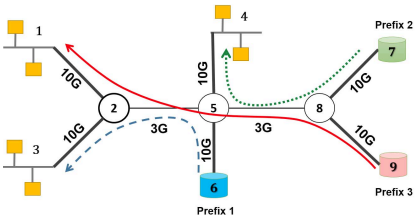

Fig. 3. Topology with two bottlenecks (link capacity in Gbps).

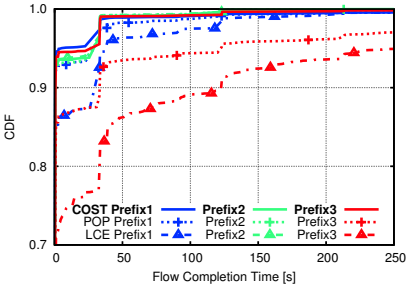

Fig. 4. Completion Time by Prefix

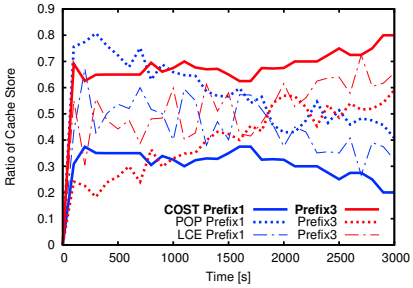

Fig. 5. Cache composition resettles after an increase of request rate for content of prefix 3 at t=1500s.

stablize (default value of 250s in our implementation). We use the technique proposed in [15] to measure and update the congestion costs of content objects (for both cached and not yet cached content). In brief, congestion cost measurement for caching works as follows.

- The congestion cost of a given content object is computed by accumulating the congestion price fed back by data packets of the content object for a preset time duration T.
- To keep track of change in congestion cost after a content object has been cached, congestion cost of a content object is increased proportionally to the (byte) hit of the content.

Overhead collecting congestion cost for the purpose of caching can be reduced by letting routers measure the cost periodically in multiple short intervals T_0 within T ($T_0 << T$).

VI. SIMULATION EVALUATION

We create an event-driven C++ simulator based on CCN architecture [16] for evaluating the performance of our congestion-based coupled caching and multipath forwarding in comparison with several combinations of well-known caching algorithms and mulitpath forwarding strategies (Table II).

A. Caching Behaviors

We first verify the behavior of congestion-cost caching in a bottlenecked topology (Fig.3) with two cascaded bottleneck links of 3Gbps in the middle of the topology. The topology mimics typical content delivery over Internet where there are multiple bottlenecks on the delivery paths. There are three repositories at node 6, 7, and 9 hosting content prefix 1, 2, and 3 respectively, each consists of 2000 content objects (50MB in size). A 2GB cache store is placed at node 2. Poisson request arrivals (Zipf 0.8, intensity 3 request/s) from access network 3, 4, and 1 request for content of prefix 1, 2, and 3 respectively. Simulations last for 3000 seconds.

Completion time distribution by prefix is reported in Fig. 4. Compared with popularity caching (POP) and LCE, congestion-cost caching (COST) has superior performance with shortest completion time across all prefixes. Notably, the proposed cache management (solid lines) maintains similar completion time distribution among all content prefixes. This is a result of selectively caching high congestion cost content, i.e., prefix 3, which significantly reduces network congestion and shortens delivery time of all prefixes.

To see this selective caching behavior and how it reacts to change in congestion cost, we additionally run a second set of simulations in which request rates for content of prefix 3 doubles at time t=1500s (Fig.5). When request rate increases at t=1500s, more cache storage is used to store content of prefix 3 as the congestion cost associated with this prefix increases with the request rate. Ratio of cache used to store content of prefix 3 raises from about 0.65 to 0.8 which benefits users who request for this content prefix since it can be directly retrieved from the cache and relieves congestion on the two bottlenecks. A similar trend is observed with POP and LCE but the ratio is not as much and as consistent as with congestion-cost caching.

B. Interaction of Caching and Multipath Forwarding

We use a two-level multipath network (Fig.6) which models a simplification of CDN networks for simulations. There are two repositories of prefix 1 and 2 at node 8 and 9 respectively, each hosts 2000 contents (Zipf 0.8). Routes to prefixes 1 and 2 and request intensity from access networks are given in the

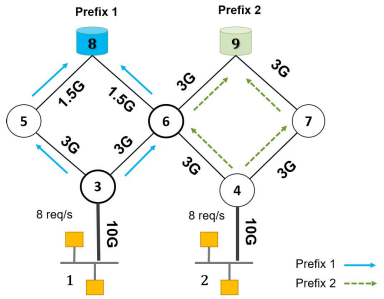

Fig. 6. Two-level topology (link capacity in Gbps).

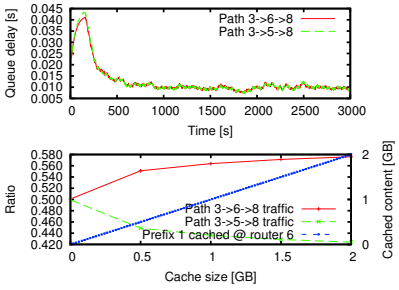

Fig. 7. Performance of coupled caching and multi-path forwarding in two-level topology.

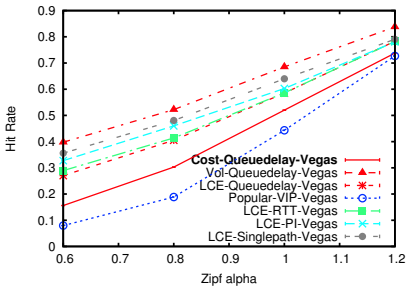

Fig. 8. Real ISP Network: Hit Rate

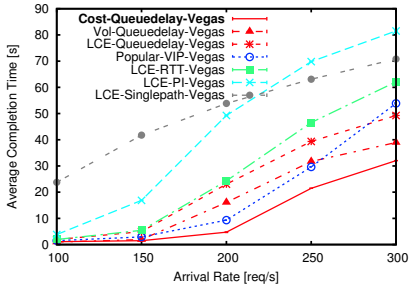

Fig. 9. Real ISP Network: Completion Time

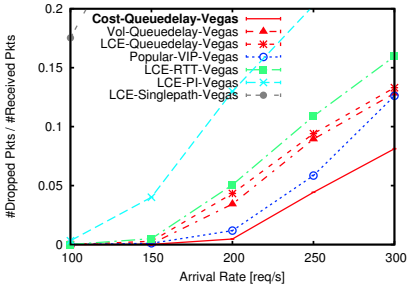

Fig. 10. Real ISP Network: Dropped Packets

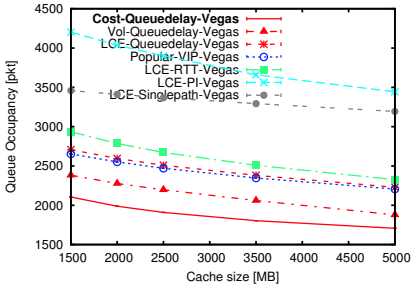

Fig. 11. Real ISP Network: Queue Occupancy

figure. We are interested in how multipath forwarding in router 3 and the cache in router 6 cooperate by congestion feedback.

Starting from equal split without caching (cache size=0), as cache size at router 6 increases, router 3 forwards more traffic over the path to router 6 and less traffic to the path to router 5 since the former path is cheaper in terms of congestion price (Fig. 7 bottom).[5] Also here we can see almost all cache space is used to store content of prefix 1 which is much more expensive to retrieve from the source due to limited link bandwidth of 1.5Gbps on link 8→6 (compared with 3Gbps of prefix 2 over link 9→6). Delay-based multipath forwarding at router 3 does a very good job of balancing queuing delay between the two paths to prefix 1 (Fig. 7 top), which helps direct more traffic via router 6's cache when the cache size increases and congestion price, i.e. queuing delay, decreases on that path. This verifies the coordinating feature among caching and multipath forwarding mentioned in Sec.IV-C.

C. Performance in Real ISP Topology

Next, we run simulations in a real, large-scale network using an ISP topology from Rocketfuel [30] (ASN1239). Link bandwidths are set inversely proportionally to the link weights. The widest links have bandwidth of 5Gbps and weight of 1. Multipath routing is pre-configured in the network following the max-flow from the sources to a given router. Other simulation parameters are given in Table III.

The proposed caching and multipath forwarding using congestion cost are compared against combinations of existing

[5]Mean ratio in the last 1500 seconds is reported; the first 1500-second period is considered as cache warm-up and discarded.

TABLE III
CONFIGURATION PARAMETERS

Parameter	Value
Content catalog	10000 objects
Content size / chunk size	50MB / 0.2MB
Content popularity	Zipf, $\alpha = 0.8$
Request arrivals	Poisson
Cache size	2.5GB (i.e. 0.5% content catalog size)
Queue length	8MB
Multipath scheduler	Back-pressure [10], and DRR [29] (default)
Simulation length	1000 seconds (x 10 runs)

cache management and multipath forwarding strategies. Notation *cache-forwarding-congestion* or *cache-forwarding* is used to denote the corresponding combination. Table II lists the algorithms used in our simulations. The combination *LCE-Singlepath-Vegas* serves as baseline scenario of an ordinary CCN network.

1) Completion Time and Hit Rate: As a metric for user performance, we first report the completion time, i.e. the average time it takes users to complete downloading their interested content in Fig. 9. At low request rate (e.g. 100 req/s), there is not much difference in the performance of all the schemes under evaluation. The improvement due to caching and multipath forwarding however is significant at high request rates when the heavy load builds up queues inside the network. The proposed *Cost-queuedelay* achieves best performance compared with all other simulated caching and multipath forwarding combinations. The reason is because caching and forwarding on queuing delay feedback can

substantially reduce network load, i.e. queue occupancy at bottlenecked links, and as a result, speeds up content delivery. If caching is done based on the volume of data without considering congestion price, i.e. *Vol-queuedelay*, completion time is consistently longer compared with *Cost-queuedelay* which considers both data volume and the congestion price as in (9). Completely replacing congestion-cost caching with LCE, i.e. *LCE-queuedelay*, has even worse performance. All other *uncoupled* combinations of caching and multipath forwarding (*Popular-VIP*, *LCE-RTT*, and *LCE-PI*) achieve not as good performance as the proposed *Cost-queuedelay*. Although *Popular-VIP* performs comparably well compared to *Cost-queuedelay*, its performance deteriorates with high arrival rates as it suffers higher loss rate due to queue overflow (Fig. 10).

Hit rate, i.e. the ratio of hits to total number of requests, is reported in Fig.8. As expected, *Cost-queuedelay* request hit rate is not quite high compared to existing methods since it does not totally prefer high request rate content.

2) Queue Occupancy and Loss Rate: Fig.11 reports queue occupancy as an indicator of the congestion level of the network. Caching and multipath forwarding play a key role in reducing congestion inside the network. The coupled congestion-cost caching with queuing delay forwarding *Cost-queuedelay* guarantees the lowest queue occupancy among all the schemes at high request rates. As before, replacing congestion-cost caching with LCE (*LCE-queuedelay*) or volume-based caching (*Vol-queuedelay*) results in higher queue occupancy compared with *Cost-queuedelay*. This result highlights the advantage of congestion-cost caching and multipath forwarding coupled by delay-based congestion control to deal with high network load and absorb congestion.

As a direct result of low queue occupancy, loss rate, the ratio of the number of dropped packets due to queue overflow to the number of successfully received packets, is always lowest using the proposed *Cost-queuedelay* (Fig. 10). Baseline configuration (*LCE-singlepath*) cannot sustain even the lowest request rate when loss rate has already exploded.

VII. Concluding Remarks

We have demonstrated the advantages of a joint scheme of caching, multipath forwarding, and congestion control in accelerating content delivery and reducing network congestion. Tightly coupled with congestion control, multipath forwarding and caching can deliver superior performance to users. Multipath forwarding directs traffic to cheaper, less congested paths while caching itself moves expensive, costly-to-retrieve content nearer to users who are requesting it. Here we do not claim either multipath forwarding or caching is more important for the objective of maximizing user performance. It is in fact the joint effect of both on top of congestion control which maximizes user satisfaction. Congestion control feedback provides the implicit coordination between caching and multipath forwarding to achieve such a desired effect. For future work, we plan to implement the proposed scheme using random early marking, reduce the overhead of congestion price feedback, and deploy the scheme in real-world applications.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *Communications Magazine, IEEE*, vol. 50, pp. 26–36, July 2012.

[2] W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang, "Cooperative content distribution and traffic engineering an isp network," *SIGMETRICS Perform. Eval. Rev.*, vol. 37, pp. 239–250, June 2009.

[3] D. DiPalantino and R. Johari, "Traffic engineering vs. content distribution: A game theoretic perspective," *INFOCOM 2009*, pp. 540–548.

[4] H. Xie, G. Shi, and P. Wang, "Tecc: Towards collaborative in-network caching guided by traffic engineering," *INFOCOM*, March 2012.

[5] L. Saino, C. Cocora, and G. Pavlou, "Cctcp: A scalable receiver-driven congestion control protocol for content centric networking," *ICC*, 2013.

[6] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, and S. Wang, "Optimal multipath congestion control and request forwarding information-centric networks," *IEEE ICNP*, Oct. 2013.

[7] D. Saucez, I. Cianci, L. Grieco, and C. Barakat, "When aimd meets icn: A bandwidth sharing perspective," *Networking*, pp. 1–9, June 2014.

[8] D. H. Lee, C. Dovrolis, and A. C. Begen, "Caching http adaptive streaming: Friend or foe?," NOSSDAV, pp. 31:31–31:36, 2014.

[9] Y. Liu, J. Geurts, J.-C. Point, S. Lederer, B. Rainer, C. Muller, C. Timmerer, and H. Hellwagner, "Dynamic adaptive streaming over ccn: A caching and overhead analysis," *ICC*, pp. 3629–3633, June 2013.

[10] E. Yeh, T. Ho, Y. Cui, M. Burd, R. Liu, and D. Leong, "Vip: A framework for joint dynamic forwarding and caching named data networks," *ACM ICN*, pp. 117–126, 2014.

[11] L. X. Bui, R. Srikant, and A. Stolyar, "A novel architecture for reduction of delay and queueing structure complexity the back-pressure algorithm," *IEEE/ACM TON*, vol. 19, no. 6, pp. 1597–1609, 2011.

[12] E. Athanasopoulou, L. Bui, T. Ji, R. Srikant, and A. Stolyar, "Back-pressure-based packet-by-packet adaptive routing communication networks," *IEEE/ACM TON*, vol. 21, pp. 244–257, Feb 2013.

[13] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," *ACM ICN*, pp. 127–136, 2014.

[14] M. Badov, A. Seetharam, J. Kurose, V. Firoiu, and S. Nanda, "Congestion-aware caching and search information-centric networks," *ACM ICN*, pp. 37–46, 2014.

[15] D. Nguyen, K. Sugiyama, A. Tagami, "Congestion price for cache management in information-centric networking," *INFOCOM'15 WRKSHP*.

[16] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," *CoNEXT* 2009.

[17] J. Garcia-Luna-Aceves, "Name-based content routing information centric networks using distance information," *ACM ICN*, pp. 7–16, 2014.

[18] E. Hemmati and J. J. Garcia-Luna-Aceves, "A new approach to name-based link-state routing for information-centric networks," *ACM ICN*, pp. 29–38, 2015.

[19] J. Jin, W.-H. Wang, and M. Palaniswami, "Utility max-min fair resource allocation for communication networks with multipath routing," *Comput. Commun.*, vol. 32, pp. 1802–1809, Nov. 2009.

[20] F. Paganini and E. Mallada, "A unified approach to congestion control and node-based multipath routing," *IEEE/ACM Trans. Netw.*, vol. 17, pp. 1413–1426, Oct. 2009.

[21] F. Paganini, "Congestion control with adaptive multipath routing based on optimization," *CISS*, pp. 333–338, March 2006.

[22] S. Low, "A duality model of tcp and queue management algorithms," *IEEE/ACM TON*, vol. 11, pp. 525–536, Aug 2003.

[23] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," *INFOCOM*, April 2014.

[24] E. J. Rosensweig, J. Kurose, and D. Towsley, "Approximate models for general cache networks," *INFOCOM*, pp. 1100–1108, 2010.

[25] S. H. Low, L. L. Peterson, and L. Wang, "Understanding tcp vegas: A duality model," *J. ACM*, vol. 49, pp. 207–235, Mar. 2002.

[26] Z. Liu, P. Nain, N. Niclausse, and D. Towsley, "Static caching of web servers," vol. 3310, pp. 179–190, 1997.

[27] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "Tcp vegas: New techniques for congestion detection and avoidance," *SIGCOMM*, 1994.

[28] H. Qian, R. Ravindran, G.-Q. Wang, and D. Medhi, "Probability-based adaptive forwarding strategy named data networking," *IFIP IM*, pp. 1094–1101, May 2013.

[29] M. Shreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," *SIGCOMM CCR*, vol. 25, pp. 231–242, Oct. 1995.

[30] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," *IMW*, pp. 231–236, 2002.