A Transport Protocol for Content-Centric Networking with Explicit Congestion Control

Feixiong Zhang*, Yanyong Zhang*, Alex Reznik†, Hang Liu‡, Chen Qian§, Chenren Xu*
*WINLAB, Rutgers University, North Brunswick, NJ, USA
†InterDigital Communication LLC, King of Prussia, PA, USA
‡Catholic University of America, Washington, DC, USA
§University of Kentucky, Lexington, KY, USA

***Abstract*—Content-centric networking (CCN) adopts a receiver-driven, hop-by-hop transport approach that facilitates in-network caching, which in turn leads to multiple sources and multiple paths for transferring content. In such a case, keeping a single round trip time (RTT) estimator for a multi-path flow is insufficient as each path may experience different round trip times. To solve this problem, it has been proposed to use multiple RTT estimators to predict network condition.**

In this paper, we examine an alternative approach to this problem, *CHoPCoP*, which utilizes explicit congestion control to cope with the multiple-source, multiple-path situation. Protocol design innovations of CHoPCoP include a random early marking (REM) scheme that explicitly signals network congestion, and a per-hop fair share Interest shaping algorithm (FISP) and a receiver Interest control method (RIC) that regulate the Interest rates at routers and the receiver respectively. We have implemented CHoPCoP on the ORBIT testbed and conducted experiments under various network and traffic settings. The evaluation shows that CHoPCoP is a viable approach that can effectively deal with congestion in the multipath environment.

I. Introduction

During the last decade, content retrieval has dominated the Internet usage. To address the challenges posed for content retrieval, content-centric networking (CCN) [12], [15] has been proposed. Being a significant shift in the network design philosophy, CCN is centered on named content instead of host addresses. Routing towards a content is based on the content name instead of the host address, and data retrieval is initiated by issuing *Interest* at the content receiver. Compared to application-layer overlay solutions such as Content Distribution Networks (CDN) and Peer-to-peer systems (P2P), CCN holds the promise of providing a more efficient and cost-effective solution to content dissemination.

CCN's unique characteristics introduce new design challenges for the underlying transport protocol. First, CCN is naturally receiver-driven, since the content receiver needs to issue an Interest first in order to request a Data chunk. Second, hop-by-hop transfer is desired for CCN transport, because content files can be cached along the route to improve throughput. Moreover, since a specific content is often widely disseminated and cached in the network, a CCN flow may have multiple sources – i.e., one content chunk originates from source A, while the next chunk might originate from source B. Such multi-source/multi-path transfer in CCN makes congestion estimation based on a single RTT value fall short. These features of CCN are sufficiently distinct from a traditional end-to-end host-based model that a new transport approach is called for.

Recently transport protocol design for CCN has received attention in the research community and several proposals have been published [3]–[6], [17], [19], [22]. In order to deal with the challenge caused by the multiple-source, multiple-path transfer, a recent study proposes the use of multiple RTT estimators at the receivers to gauge network congestion of each path. Additionally, recent studies also suggest the routers adopt a hop-by-hop Interest shaping scheme to actively prevent network congestion.

In this paper, we propose a new transport scheme, called the Chunk-switched Hop Pull Control Protocol (CHoPCoP), which has the following design elements:

- *Random early marking(REM)*. REM uses explicit congestion signalling instead of RTT-based congestion prediction. Router detects congestion by monitoring the size of outgoing data queue. It then explicitly marks data packets to notify receivers when the network is congested. To our knowledge, this is the first paper that describes, analyzes and evaluates explicit congestion control for CCN.
- *Fair share Interest shaping(FISP)*. CHoPCoP router decides whether to forward an Interest immediately or delay it temporarily based upon the available queue sizes and the flow demands. FISP is triggered to delay Interests when REM can't effectively prevent congestion, for instance in the absence of cooperation from receivers, thus actively protects the router from congestion. FISP also realizes fair bandwidth sharing among flows by fair scheduling of multiple Interest queues at an interface.
- *AIMD-based receiver Interest control(RIC)*. The receiver adjusts its Interest window based on the AIMD (Additive Increase Multiplicative Decrease) mechanism. Here, receiver detects congestion mostly by marked packets from upstream routers.

We have implemented CHoPCoP using the Click Modular Router [13] and evaluated its performance on the ORBIT testbed [16]. We conduct experiments over various network and traffic settings and compare our protocol with several existing ones. Our evaluation shows that *i)* explicit congestion control provides congestion detection timely and correctly in a multi-source/multi-path setting, and significantly alleviates detection errors due to source/path change, thus improves network stability and efficiency; *ii)* our FISP scheme ensures fair sharing of network resources among different flows, it also provides protection against misbehaving receivers while still makes the most of network resources; *iii)* our receiver Interest control scheme guarantees full bandwidth utilization while reacts to REM signal to avoid saturating the network.

The rest of the paper is organized as follows. Section II describes the overall features of CCN and the related work on transport control in CCN. A detailed description of CHoPCoP is given in Section III. Our implementation is presented in Section IV. Section V presents ORBIT-based evaluation results, and concluding remarks are given in Section VI.

II. CCN and transport control

In this section, we give an overview of CCN's basic features [12] and discuss the related work on transport control.

A. CCN Background

There are two types of CCN packets: *Interest* and *Data*. A receiver asks for content by issuing an Interest packet and the corresponding Data chunk is returned in response to that Interest. The receiver can thus control the progression of content retrieval by adapting the Interest sending rate.

CCN packet forwarding is performed using three main data structures: forwarding information base (FIB), content store (CS) and pending Interest table (PIT) – FIB is used to forward Interest packets toward the data source, CS is the cache memory to store passing Data chunks, and PIT keeps track of forwarded Interest packets so that returned chunks can be sent to its receiver.

When an Interest packet is received from interface *f*, the router performs the following operations: (1) checks CS and returns a copy through *f* if cache hits; (2) otherwise, conducts a PIT lookup to verify if an entry for the same content name already exists. If so, appends *f* to the entry and discards the Interest; (3) if not, creates a new PIT entry and forwards the Interest through the interface indicated by FIB.

When a Data chunk is received, the router forwards the chunk to all interfaces specified by the corresponding PIT entry and then removes that entry. The router may also choose to cache the chunk in CS, if appropriate.

B. Related Work on Transport Control

There has been huge amount of work on transport protocols in the Internet. Here we focus on transport schemes proposed for CCN. We give a discussion of such work in three aspects.

RTT-based congestion control: Some designs, for instance ICP [3] and ICTP [19], rely on a single round trip time (RTT) estimator at the receiver to predict network status which are not suitable for CCN because CCN flows may have multiple sources and multiple paths. To adapt to CCN's multipath nature, authors in [5], [6] propose per-route transport control, in which a separate RTT estimation is maintained for each route at the receiver, similar to MPTCP [21]. Multiple RTT based scheme works well for multipath transfer, however, it adds lots of complexity to the receiver and heavily relies on the accuracy of timing. Our scheme, on the other hand, utilizes explicit congestion signalling from network to effectively notify the receiver about network condition. Compared with multiple RTT based scheme, our approach leads to a much simpler receiver design and is not restricted to limitations of timers which have long been criticized [9], [11], [18], [24].

hop-by-hop Interest shaping: Since in CCN one Interest packet retrieves at most one Data chunk, a router can control the rate of future incoming data chunks by shaping the rate of outgoing Interests. In [4], [17], authors propose quota-based Interest shaping scheme to actively control traffic volume. They assign a quota (in terms of the number of pending Interests) to each flow, and if the number of pending Interests for a flow exceeds the quota, that flow's Interests will be delayed or dropped. In [22], authors use per interface rate limit to avoid congestion at an interface and per prefix-interface rate limit to control Interest rate of each content prefix. The quota-based Interest shaping and the rate limiting Interest control require to assign an appropriate quota value for each flow or rate limit value for each content, which is challenging in practice, if not impossible. And they can be rather inefficient under dynamic traffic setting. Our fair share Interest shaping scheme also considers about fairness, however, resources are shared by all flows and Interest from a flow is delayed temporarily only when shared resources become limited and the corresponding flow unfairly consumes resources.

flow-aware traffic control: Authors in [17] propose a flow-aware network paradigm for CCN. They define content flow as packets bearing the same content name identified on the fly by parsing the packet headers. Moreover, routers impose per-flow fair bandwidth sharing by having multiple data queues at each interface for active flows and using deficit round robin (DRR) [20] for fair scheduling among these queues. Different from such scheme, our FISP realizes per-flow fair sharing by having per-flow Interest queues at an interface and utilizing modified DRR to approximate max-min fairness.

III. CHoPCoP Design

In this section, we describe the design of the CHoPCoP protocol in detail. CHoPCoP consists of the following three functional modules: (1) explicit congestion signalling by random early marking; (2) fair share Interest shaping; and (3) AIMD based receiver Interest control. Figure 1 illustrates the functional modules of CHoPCoP content provider, router, and receiver. In our router model, the memory allocated for buffering packets at interfaces is separated from that used for caching (CS). Each interface has separate inbound buffers/queues and outbound buffers/queues for Interest and Data.

CCN data chunk is large in size [12] because of extra fields in the packet (e.g. signature). Large chunk can cause fragmentation, which may drastically harm throughput since the loss of a single fragment will cause retransmission of the whole chunk, like IPv4 fragmentation. Similar to two-level content segmentation in [19], we thus propose that a chunk

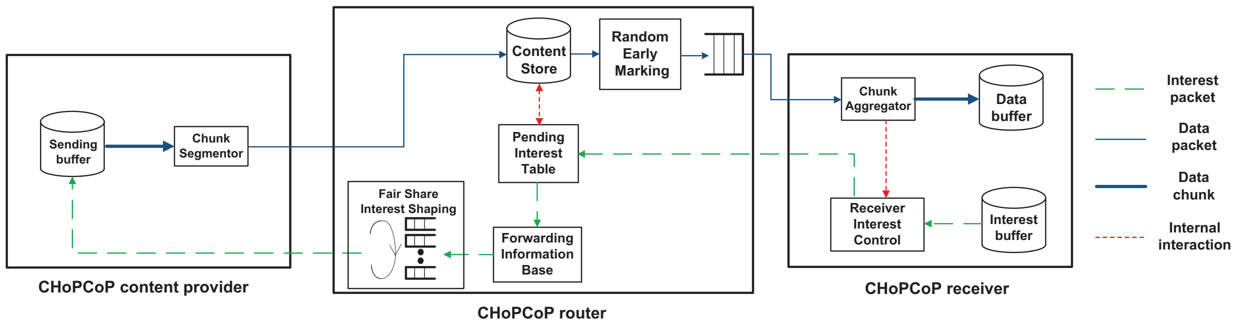


Fig. 1: Functional modules of CHoPCoP content provider, router and receiver (we omit inbound queues in router for simplicity)

is segmented to multiple small packets before the content provider sends it out and the receiver aggregates received packets into a complete chunk. If certain data packets in a chunk are lost, the receiver will issue a specific Interest packet containing the offsets of lost data packets within that data chunk, causing only those lost data packets to be retransmitted.

A. *Explicit Congestion Signalling*

Since content flow in CCN may have multiple sources and multiple paths, using a single RTT estimation cann't work well. Although using multiple RTT estimations [5] is proposed to address the issue, we take a different approach that each intermediate router estimates its congestion level and then notifies the receiver of the congestion event. Upon reception of such a notification, a properly functioning CHoPCoP receiver slows down the Interest issuance rate using the method presented in Subsection III-C.

The technique we propose to achieve explicit congestion signalling is referred to as *Random Early Marking* (REM), similar to mechanism in RED [11] and ECN [9]. In REM, before a router forwards a data packet through interface *f*, it samples the occupancy of the corresponding outbound data queue, denoted by q. The router then smooths the sampling by calculating the moving average of queue occupancy $\bar{q}$ as

$$\bar{q} = (1-\mu)\bar{q} + \mu q, \tag{1}$$

where $0 < \mu < 1$ is a design parameter which sets the weight of the current sampling. In this way, we can avoid the adverse impact caused by temporary increases in the data queue. The router marks the packet with a probability $P = \frac{\bar{q}-q_{min}}{q_{max}-q_{min}} P_{max}$ if the value of $\bar{q}$ is between q_{min} and q_{max}, and with a probability $P = P_{max} + \frac{\bar{q}-q_{max}}{q_{max}}(1-P_{max})$ if $\bar{q}$ is between q_{max} and $2 * q_{max}$ (shown in Figure 2). If the queue occupancy is above the threshold $2 * q_{max}$, the router always marks the packet. Such gentle variation of packet marking probability is proved to make explicit congestion control much more robust to the setting of parameters [10].

REM predicts network congestion much more accurately than single RTT estimation in a multi-source environment. It actively notifies the receiver prior to congestion taking place, thus keeps the network stable. REM enjoys other additional benefits same as RED/ECN, including the avoidance of global synchronization and bias against bursty traffic, and the guarantee of statistical fairness [11]. Compared to multiple RTT

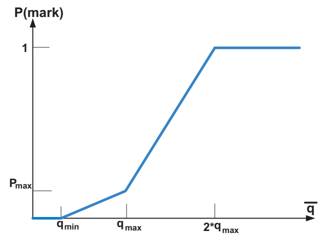


Fig. 2: Mark probability function for REM

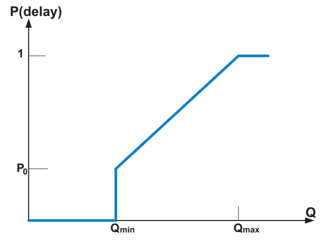


Fig. 3: Delay probability function for FISP

estimation proposal [5], REM vastly simplifies congestion detection at receiver and is free from limitations of timer [9], [11], [18], [24]. Compared with RED/ECN, in REM the congestion signal is delivered to the receiver and directly used for controlling rate instead of being reflected back to the sender, thus causing less delay.

B. *Fair Share Interest Shaping (FISP)*

A lightweight flow-aware network paradigm is well-adapted to CCN and can bring significant advantages [17]. This requires to realize per-flow fair bandwidth sharing at router and overload control even when the receiver is non-cooperative.

In CHoPCoP, we propose a fair share Interest shaping scheme, FISP in short, to address these issues. As seen in Figure 1, multiple Interest queues are allocated in an interface, one for each active content flow, identified by the content name. An active content flow corresponds to a FIB entry that has at least one PIT entry in active, thus flow information can be easily extracted from FIB and PIT. A modified DRR [20] is further used for fair scheduling among different queues: the deficit counter of a queue here is decreased by the size of the corresponding Data after servicing an Interest. To deduct such size value, we can use the segmentation information from the Interest (e.g. in the CCNx prototype [7]), or define a standard field as suggested in [4]. The analysis in [14] shows that the number of active flows that need scheduling remains in hundreds even though there may be tens of thousands of flows in progress, thus demonstrates that fair sharing is scalable.

Moreover, FISP is triggered to delay certain Interests when a router's data queue continues to grow even though REM has marked outgoing data packets, which is particularly helpful when the receiver is non-cooperative. FISP realizes this using an algorithm that consists of the following four phases.

In the first phase, FISP checks whether delaying Interests should take place. To do so, FISP periodically counts the total queue requirement at an interface, Q, as

$$Q = q_d + \gamma q_i, \tag{2}$$

where q_d quantifies the occupancy of outgoing data queue which is directly extracted from REM, q_i quantifies buffer resources needed by data chunks that will arrive at a response to outstanding Interests, and γ is a weight parameter. q_i is implemented along with PIT. It is increased when an Interest corresponding to the interface is sent upstream and decreased when data comes back. If the value of Q exceeds a preset threshold value, Q_{min}, the router starts Interest shaping for this interface.

In the second phase, FISP determines which flow's Interests should be delayed by looking at each flow's queue requirement. We use Q_j to denote flow j's queue requirement calculated similar to Equation 2. If Q_j exceeds its *fair share*, i.e., $Q_j \geq \frac{Q_{min}}{n}$ (n is the number of flows the interface currently has), then flow j's packets are delayed with a probability $P = \frac{Q - Q_{min}}{Q_{max} - Q_{min}} + P_0$ as pictorially shown in Figure 3. The Interests that are delayed are sent to a delay queue instead of being sent upstream and will re-enter outgoing Interest queues after a certain interval. The delayed Interests will not be counted towards the queue requirement for the interface.

In the third phase, we consider the overly-congested scenario. Once the queue requirement Q exceeds another threshold, Q_{max}, then any incoming Interest will be delayed.

In the fourth phase, if the router finds that the queue requirement, Q, falls below $Q_{release}$, then it will release all the delayed Interests to the outgoing Interest queues. We note that the relationship between the three threshold values is $Q_{release} < Q_{min} < Q_{max}$.

Note that Q_{min} in FISP is larger than $2 * q_{max}$ in REM, and the delay probability starts from P_0 instead of 0, ensuring FISP is triggered to delay Interests after REM, when router queue continues to accumulate even after packet marking.

FISP actively protects the network from congestion even with non-cooperative receivers. Compared with quota-based Interest shaping [4], [17], FISP is more efficient in resource utilization, while it also ensures fairness among different flows. Moreover, the whole protocol is kept simple since the delaying algorithm in FISP is not triggered in normal condition.

C. AIMD Based Receiver Interest Control (RIC)

In CHoPCoP, each receiver maintains an Interest window that keeps track of pending Interests. The size of this window determines the Interest rate from the receiver, which in turn impacts the traffic volume in the network. We need to keep the receiver Interest window large enough to enjoy the available bandwidth, while not saturating network capacity. Here, we use the AIMD (Additive Increase Multiplicative Decrease) mechanism to manage receiver Interest control (RIC). Specifically, the receiver adjusts its Interest window proportional to congestion level implied in explicit congestion signalling.

RIC consists of two phases, namely, the slow start phase and the congestion avoidance phase. The slow start phase begins when the receiver sends out the first Interest for a given content. In this phase, the Interest window rapidly increases to utilize the network capacity by incrementing the window size by one every time it receives a complete data chunk. After the window size, W, reaches a threshold, or when the network is congested, the receiver starts the congestion avoidance phase. Here, the receiver window is either increased at a much slower rate, or decreases, depending upon whether the congestion is detected. Before congestion is detected, every time after the receiver receives W data chunks, we increase the window size by α where $\alpha < W$. After congestion is detected, however, the window size decreases to βW where $\beta < 1$. The values of α and β determine how aggressive the user is in tracking available bandwidth, as has been well studied in TCP [8].

In CHoPCoP, the receiver detects congestion either when it has a timeout or receives marked packets. We use different β values in these situations. In the former case, we simply use a fixed value, β_0, and in the latter, we calculate β as

$$\beta = \beta_2 - \frac{(\beta_2 - \beta_1) N_{marked}}{N}, \tag{3}$$

where N_{marked} and N denote the number of marked packets and the total number of packets in a chunk respectively.

Equation 3 shows that the receiver reacts to REM in proportion to the extent of congestion, not only its presence. Note that in normal condition the receiver detects congestion through receiving REM notification; timeout only takes place when REM fails.

Whenever a timeout occurs, the receiver needs to retransmit the Interest. RIC retrieves the offsets of lost data packets within the data chunk from the chunk aggregator and sends out the Interest with the offset information. The timeout value is calculated as:

$$R\bar{T}T = \sigma R\bar{T}T + (1 - \sigma) RTT, \tag{4}$$

$$TimeOut = \delta R\bar{T}T, \tag{5}$$

Where RTT denotes the current RTT sample value, while $R\bar{T}T$ denotes the moving average of RTT value. Note that we would set the timeout parameter relatively large because REM is adopted and timer is not critical any more.

IV. Implementation

We have implemented a complete network stack for our protocol as a user-level daemon, using the Click Modular Router. In our implementation, packets are directly fetched from physical interfaces to the user space or pushed to physical interfaces from the user space through PCAP sniffing. A link-state routing scheme is implemented as the name-based routing substrate where each node broadcasts its link state advertisement (LSA) with the content information it has.

Packet and header format: The following packet types are defined in our implementation: Interest, Data (chunk/packet), "hello" message and LSA. Each packet header contains the

pkt_type field that identifies the packet's type. "hello" message and LSA are used for building the routing table, similar to OSPF discussed in [12]. Both Interest and data contain $content\_name$ and $chunk\_ID$ identifying the requested chunk within the specified content. A data chunk is segmented into multiple data packets, each containing seq_num that is the offset of the packet in the chunk. An Interest can specify seq_num asking for a specific data packet. A $service\_ID$ byte is included in the data packet header with the last bit reserved for congestion signalling.

Data chunk: In our current implementation, a data packet is 1KB in size and a chunk consists of 32 data packets, which complies with the multi-segment setting specified in [19].

Memory management: The queues for each interface can be dimensioned using the traditional bandwidth-delay product rule [1]. We thus set the outbound data queue size equal to the average RTT value multiplied by the link bandwidth. Here we assume 300ms as the average Internet RTT value. Since the inbound data queue is some transient buffer where data packets are processed rather fast by the router engine, its size is set to a small fixed value as 1MB. For an Interest queue, its size is set to 100KB accordingly.

REM (random early marking): Parameters of the packet marking probability function is set as follows: $q_{min} = 0.1C$, $q_{max} = 0.2C$, $P_{max} = 0.002$, where C denotes the queue capacity. For REM smoothing, we set $\mu = 0.05$ in Equation 1. The parameter setting is based on discussions in RED [11]. Note that the value of P_{max} is rather small because a data chunk is regarded as marked at the receiver as long as any of its segmented packets gets marked.

FISP (Fair-Share Interest shaping): For delay probability function, $Q_{release} = 0.3C$, $Q_{min} = 0.4C$, $Q_{max} = 0.9C$, $P_0 = 0.3$, where C denotes buffer capacity for the interface.

RIC (receiver Interest control): The threshold of Interest window for slow start phase is set to 20. AIMD has the following parameters: $\alpha = 1, \beta_0 = \beta_1 = 0.5, \beta_2 = 0.8$. We set $\sigma = 0.7$ in Equation 4 and $\delta = 6$ in Equation 5.

V. Evaluation

We describe our evaluation effort on ORBIT testbed [16], [23] and present detailed experimental results here. Our evaluation focuses on REM's capability in stabilizing network condition and adapting to CCN's multipath. We also evaluate the capability of FISP in ensuring flow fairness and protecting against non-cooperative receivers. For such purpose, we conduct detailed experiments on several simplified but representative network topologies and compare CHoPCoP with existing CCN transport protocols that utilize a single RTT estimation (i.e. ICP [3], HR-ICP [4] and ICTP [19]) and protocols that use quota-based hop-by-hop Interest shaping (i.e. HbH in HR-ICP). We don't compare CHoPCoP with multiple RTT estimation proposals here since our scheme can be an alternative to those proposals in controlling multipath transfer.

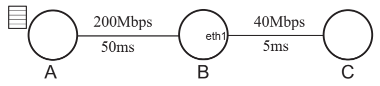


Fig. 4: A three node baseline topology, consisting of a source (A), a router (B) and a receiver (C).

A. Experimental Setup

The ORBIT testbed provides a realistic network environment to support protocol evaluation in both wired and wireless settings. Each ORBIT node in our experiments has a 2.93GHz Intel i7 quadcore processor, 4GB memory, and runs Linux 2.6. Also, each node is equipped with two wired interfaces, eth0 and eth1. Both interfaces on all the ORBIT nodes are connected, each forming a gigabit LAN. We use the link layer packet filtering techniques discussed in [2] to create logical topologies. In addition, we use Click Modular Router's link emulation elements to configure wired link parameters such as link bandwidth and delay.

B. A Single-Source, Single-Destination Benchmark Scenario

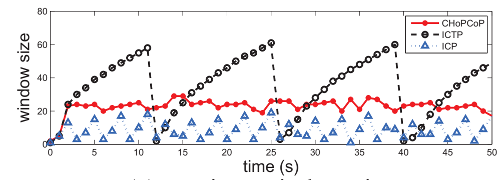
We first conduct several benchmark experiments using a three node baseline topology, shown in Figure 4. Here, the source node (A) is connected to the router (B) via a long Internet connection (with a 200Mbps bandwidth and a 50ms latency), and the router is connected to the receiver (C) via a local Internet access (with a 40Mbps bandwidth and a 5ms latency). We set the outbound data queue size of the router's eth1 interface to be $40Mbps \times 300ms = 1500KB$ according to the bandwidth-delay product rule [1]. We consider a 320MB content file that consists of 10,000 chunks in total, each chunk 32KB in size. There is only one flow in this setting.

1) The Effectiveness of REM and RIC: First, we compare CHoPCoP with existing protocols: ICTP, ICP, and HR-ICP. Here, the receiver is cooperative and slows down Interest issuing when marked packets are observed, so the main components that are effective in CHoPCoP are REM and RIC. We will show that CHoPCoP stabilizes the network condition and achieves higher throughput.

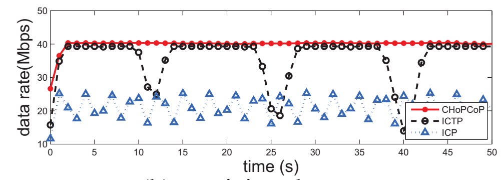
Figures 5(a)-(d) summarizes the results, showing how the receiver window size, the transient receiving data rate, the number of timeout instances, and the router queue size, change over time. The results show that CHoPCoP significantly outperforms the others. The receiver side Interest window is much smoother, with an average size of 23.27 and a standard variance of 2.57 (Figure 5(a)); the receiving data rate is much higher (also smoother), with 39.91Mbps at the steady state (Figure 5(b)); and no timeout is observed at the receiver (Figure 5(c)). Specifically, the throughput[1] of CHoPCoP is 85.75% and 13.69% higher than that of ICP/HR-ICP and ICTP respectively. This is because it effectively smooths the outgoing data queue at the router (with an average of 235.8KB and standard variance of 116.5), as shown in Figure 5(d).

The inferior performance of ICTP/ICP/HR-ICP can be explained below. In ICTP, after the receiver window reaches a

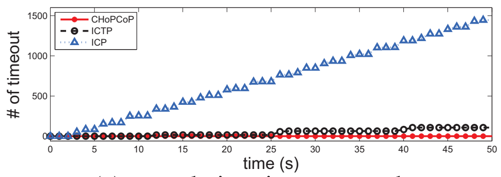
[1]We use the term throughput for the average data delivery rate, while the term receiving rate for instant data delivery rate.



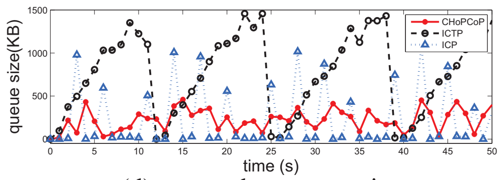
(a) receiver window size



(b) receiving data rate

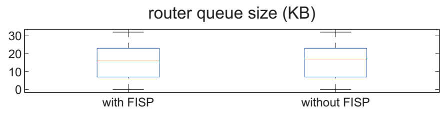


(c) cumulative timeout number

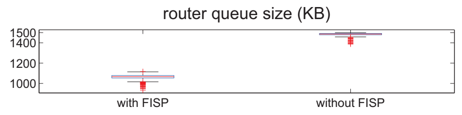


(d) router data queue size

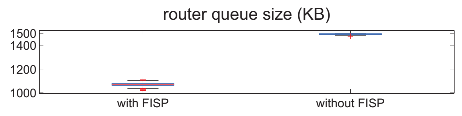
Fig. 5: The performance results for a 3-node baseline topology. ICP and HR-ICP behave very similarly, we thus use the ICP curve to represent both schemes.



(a) Interest rate: 140 per second



(b) Interest rate: 160 per second



(c) Interest rate: 200 per second

Fig. 6: Boxplot of router queue size with and without FISP under different CIR. At a low CIR, both schemes perform similarly. At a high CIR, FISP nicely controls the router queue size, and thus avoids queue overflow. Without FISP, the router queue overflows when the CIR is above 160 Interests per second.

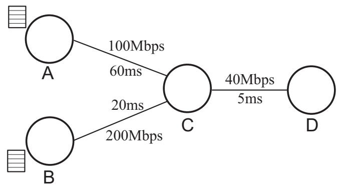


Fig. 7: A topology that consists of two sources (A and B) that serve the same request from D.

certain value (∼22 in our experiment), the receiving Interest rate remains the same even if the window keeps increasing. This results in a large queue at the router, which may easily cause congestion. On the other hand, in ICP/HR-ICP, a relatively small timeout setting causes a large number of false timeouts and thus poor throughput.

2) The Effectiveness of Fair Share Interest Shaping (FISP): Next, we consider a non-cooperative receiver who issues Interests at a constant Interest rate (CIR), even when the router has signalled congestion through packet marking. In this case, FISP is triggered to actively delay Interest in order to mitigate congestion since the receiver doesn't respond to REM signalling.

The router's outgoing data queue is 1500KB in size, and we have $Q_{max} = 0.9C = 1350KB$. We consider CIR of 140, 160, and 200 Interests per second in each run, requesting a 320MB file. We show router queue size with and without FISP in Figure 6, and delivery ratio and throughput in Table I.

The results show that with FISP, the router's outgoing data queue can be kept at $\sim 1050KB$ when $CIR > \frac{40Mbps}{32KB} \approx 150$ Interests per second. Without FISP, router queue overflows and the router keeps congested.

C. A Multi-Source, Single-Flow Scenario

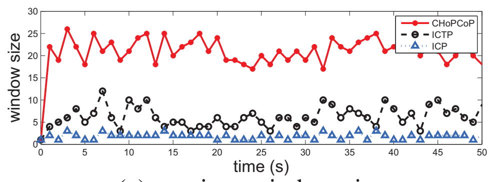
In CCN, a content request might be served by multiple sources, resulting in multiple paths. Single RTT based protocols, however, implicitly assume there is only a single path for a content flow, thus can't accurately estimate the network condition. In this section, we take a close look at the issues caused by having multiple sources for the same flow, as well as the effectiveness of CHoPCoP in handling these issues.

1) The Impact of Timeout Parameters at the Receiver: In this set of experiments, we show that a large timeout parameter is essential to avoid false timeouts at the receiver because of the inherent large RTT variation in a multi-source environment. To demonstrate the point, we consider a topology that consists of two sources. The detailed topology and the link parameters are shown in Figure 7. Here, we use a relatively small content file with only 100 chunks to eliminate the router queuing delay from the RTT measurement. These 100 chunks are randomly placed at the two sources, and thus both sources serve the same flow.

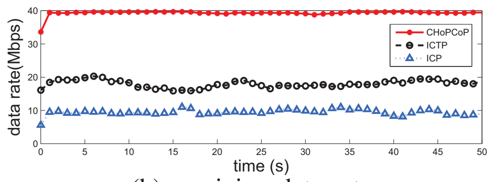
We run CHoPCoP with two timeout values using different δ values in Equation 5: $\delta = 2$ for a small timeout value while $\delta = 6$ for a large value. Table II shows the number of timeouts and throughput observed at the receiver. The results clearly

CIR	Data Delivery Ratio		Throughput	
	With FISP	Without FISP	With FISP	Without FISP
140	100%	100%	36.37 Mbps	36.36 Mbps
160	100%	5.86%	37.44 Mbps	2.4 Mbps
200	100%	1.91%	37.62 Mbps	977.9 Kbps

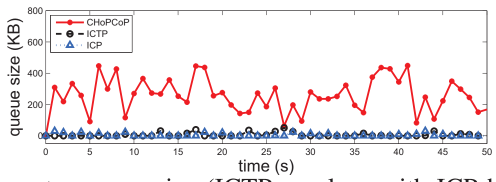
TABLE I: FISP leads to better delivery ratio and throughput. Without FISP, throughput decreases significantly when queue overflows



(a) receiver window size



(b) receiving data rate



(c) router queue size (ICTP overlaps with ICP here)

Fig. 8: Comparison of CHoPCoP, ICTP, ICP, and HR-ICP when two sources serve the same content flow. We use the ICP curve to represent both ICP and HR-ICP because of their similar performance.

show that small timeout value leads to many more timeouts. In fact, timeout takes place almost every time when the source switches from B to A. On the other hand, we observe no timeout if we choose a large timeout value, resulting in 55% higher throughput.

2) The Effectiveness of REM and RIC: Next, we show that REM is effective in detecting networking congestion timely and correctly when multiple sources exist, and thus keeping the network stable and efficient. We use the same topology as shown in Figure 7, with $\delta = 6$. Here, we consider a content file that consists of 10000 chunks, which are randomly placed at the two sources.

We compare CHoPCoP with existing protocols: ICTP, ICP, and HR-ICP, and show results in Figure 8. The results show that CHoPCoP is very effective in fully utilizing network capacity while keeping the network stable by explicitly signalling the receiver about potential congestion in advance. As a result, CHoPCoP improves throughput by 107.6% compared to ICTP, and by a factor of 2 compared to ICP/HR-ICP.

Poor performance of ICP and HR-ICP shows single RTT estimator cann't predict network congestion in multi-source environment. We also find that for ICTP, out-of-sequence packet arrival takes place frequently which triggers fast retransmission and fast recovery and in turn lowers the throughput. This shows that using packet sequence to detect packet loss similar to "triple duplicate ACK" is not reliable either.

δ	Timeout #	Throughput (Mbps)
2	37	25.1
6	0	38.79

TABLE II: In CHoPCoP, a large timeout value (with large δ values) can eliminate false timeouts.

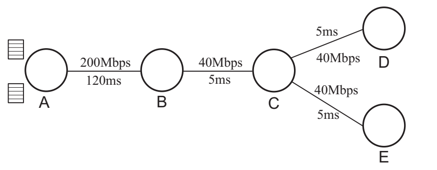
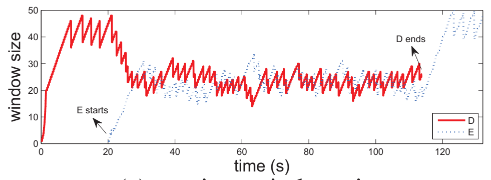
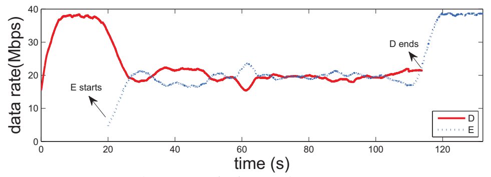


Fig. 9: Multiple flow topology where D and E request two different files from the same source A.



(a) receiver window size



(b) receiving data rate

Fig. 10: CHoPCoP can efficiently utilize the network resource when only one flow exists. When we have multiple flows, it ensures fairness among them.

D. A Multi-flow Scenario

We further investigate how CHoPCoP behaves when multiple flows exist. We conduct the experiments on the topology in Figure 9, which consists of two receivers (D and E). These two receives request different content files hosted by A, resulting in two flows in the topology.

1) Fairness between the Flows: First, we show CHoPCoP adapts to the number of flows in the network – when we have one flow, it allows the flow to efficiently utilize the network resources, while when we have multiple flows, it provides fair sharing among them. For this purpose, in the topology in Figure 9, we have receiver D and E request two different content files (320MB each) from the source A. D starts its request at time 0 while E starts at time 20s.

The results in Figure 10 confirm our point. Before E starts, there is only one flow, initiated by D, and as a result, this flow owns the resources exclusively, with average window size of 42.24 and throughput of 37.89Mbps. As soon as E starts, the two flows share the resources in a fair manner, with average window size of 23.83 and 24.04, and throughput of 19.54Mbps and 19.44Mbps for D and E, respectively. Finally after D ends, the flow initiated by E owns the resources exclusively.

2) Comparison of FISP with Quota-based Interest Shaping: Next, we compare the performance of FISP against quota-based Interest shaping scheme, HbH, presented in HR-ICP [4], using the topology in Figure 9. In this set of experiments, receiver D sends out Interests at a constant rate of 20 Interests per second, while receiver E's Interest rate varies from 40 to 180, with a 20 Interests per second increase in each run.

The results in Figure 11 show that our scheme, FISP, provides better resource utilization, better handles traffic fluc-

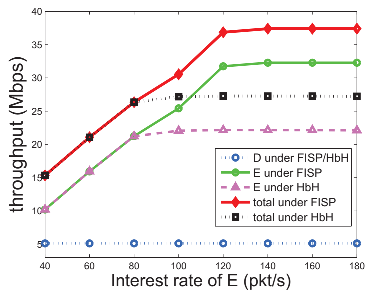


Fig. 11: Throughput comparison of FISP with HbH. Receiver D has the same throughput for both schemes.

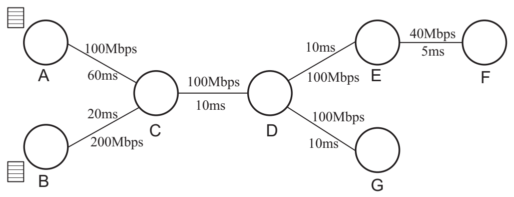


Fig. 12: A larger-scale topology with two sources (A and B) and two receivers (F and G). Link EF is the bottleneck link between A/B and F, while link CD is the bottleneck between A/B and G.

tuation, and thus gains higher network throughput. Specifically, after the network reaches saturation, FISP achieves the throughput as high as 37.40Mbps for FISP while HbH only has 27.26Mbps, with a 37.2% of improvement.

E. A Larger-Scale Network Topology

Finally, we put everything together and look at a larger-scale network topology that has 7 nodes and 2 flows. The topology and detailed link parameters are shown in Figure 12.

We compare the throughput of CHoPCoP with existing protocols: ICTP, ICP and HR-ICP. Here, receiver F and G simultaneously request two different content files C1 and C2 (320M each) respectively. Content C1 is located at source A, while content C2 is randomly distributed at the two sources.

Table III summarizes the throughput of two receivers under different schemes during the period when both flows are active. CHoPCoP achieves a performance gain of 103% over ICTP, and a factor of 5 over ICP/HR-ICP.

VI. Conclusion

In this paper, we present the design, implementation and evaluation of CHoPCoP, a CCN transport protocol. In addition to being receiver driven and hop-by-hop transport, CHoPCoP utilizes explicit congestion signalling to tackle with CCN's multipath nature. We also propose fair share Interest shaping scheme to provide bandwidth sharing among different flows. Moreover, our Interest shaping scheme will actively delay Interests when explicit congestion control can't effectively control network congestion.

We have implemented the complete protocol stack using the Click Modular Router, and evaluated its performance on the ORBIT testbed. Our experimental results show that explicit congestion signalling, when coupled with our AIMD-based receiver Interest control, can successfully stabilize the router queues and improve the throughput in a multi-source/multi-path environment. When there are multiple flows in the network, our fair share Interest shaping scheme ensures fairness and provides more efficient resource utilization than earlier quota-based Interest shaping algorithms. Finally, using a topology with multiple sources and multiple flows, we show that CHoPCoP can improve the total network throughput by at least 103% over existing solutions that use single RTT estimation.

Transport Protocol	Throughput (F) (Mbps)	Throughput (G) (Mbps)	Total (Mbps)
CHoPCoP	37.82	59.24	97.06
ICTP	34.13	13.72	47.85
ICP	7.85	5.90	13.75
HR-ICP	7.68	6.05	13.73

TABLE III: Throughput comparison among protocols.

References

[1] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In *ACM SIGCOMM*, 2004.
[2] G. D. Bhanage, Y. Zhang, and I. Seskar. On topology creation for an indoor wireless grid. In *ACM WiNTECH*, 2008.
[3] G. Carofiglio, M. Gallo, and L. Muscariello. Icp: Design and evaluation of an interest control protocol for content-centric networking. In *IEEE INFOCOM WKSHPS*, 2012.
[4] G. Carofiglio, M. Gallo, and L. Muscariello. Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks. In *ACM ICN*, 2012.
[5] G. Carofiglio, M. Gallo, L. Muscariello, and M. Papalini. Multipath congestion control in content-centric networks. In *IEEE INFOCOM WKSHPS*, 2013.
[6] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, and S. Wang. Optimal multipath congestion control and request forwarding in information-centric networks. In *IEEE ICNP*, 2013.
[7] CCNx open source project. https://www.ccnx.org/.
[8] D.-M. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN systems*, 1989.
[9] S. Floyd. Tcp and explicit congestion notification. *ACM SIGCOMM Computer Communication Review*, 1994.
[10] S. Floyd. Recommendation on using the "gentle_" variant of red. 2000.
[11] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1993.
[12] V. Jacobson and et al. Networking named content. In *ACM CoNEXT*, 2009.
[13] E. Kohler and et al. The click modular router. *ACM Transactions on Computer Systems*, August 2000.
[14] A. Kortebi, L. Muscariello, S. Oueslati, and J. Roberts. Evaluating the number of active flows in a scheduler realizing fair statistical bandwidth sharing. In *ACM SIGMETRICS*, 2005.
[15] Named data networking project. http://www.named-data.net/.
[16] ORBIT testbed. http://www.orbit-lab.org/.
[17] S. Oueslati, J. Roberts, and N. Sbihi. Flow-aware traffic control for a content-centric network. In *IEEE INFOCOM*, 2012.
[18] I. Psaras and V. Tsaoussidis. Why tcp timers (still) don't work well. *Computer Networks*, 2007.
[19] S. Salsano and et al. Transport-layer issues in information centric networks. In *ACM ICN*, 2012.
[20] M. Shreedhar and G. Varghese. Efficient fair queueing using deficit round robin. In *ACM SIGCOMM*, 1995.
[21] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley. Design, implementation and evaluation of congestion control for multipath tcp. In *USENIX NSDI*, 2011.
[22] C. Yi and et al. A case for stateful forwarding plane. *Elsevier Computer Communications*, 2013.
[23] F. Zhang, A. Reznik, H. Liu, C. Xu, Y. Zhang, and I. Seskar. Using orbit for evaluating wireless content-centric network transport. In *ACM WiNTECH*, 2013.
[24] L. Zhang. Why tcp timers don't work well. In *ACM SIGCOMM*, 1986.