Preprint | Preprint version of the paper Published in IEEE/ACM Transactions on Networking (Volume: 29, Issue: 5, Oct. 2021) DOI: 10.1109/TNET.2021.3090174

Delay-based Network Utility Maximization Modelling for Congestion Control in Named Data Networking

Yuhang Ye, Brian Lee, *Member, ACM*, Ronan Flynn *Member, IEEE*, Jin Xu, Guiming Fang, Yuansong Qiao, *Member, IEEE*

***Abstract*— Content replication and name-based routing lead to a natural multi-source and multipath transmission paradigm in NDN. Due to the unique connectionless characteristic of NDN, current end-to-end multipath congestion control schemes (e.g. MPTCP) cannot be used directly on NDN. This paper proposes a Network Utility Maximization (NUM) model to formulate multi-source and multipath transmission in NDN with in-network caches. From this model, a family of receiver-driven transmission solutions can be derived, named as path-specified congestion control. The path-specified congestion control enables content consumers to separate the traffic control on each path, which consequently facilitates fair and efficient bandwidth sharing amongst all consumers. As a specific instance, a Delay-based Path-specified Congestion Control Protocol (DPCCP) is presented, which utilizes queuing delays as signals to measure and control congestion levels of different bottlenecks. In addition, a set of high-performance congestion control laws are designed to accelerate bandwidth and fairness convergence towards the optimum defined by the NUM model. Finally, DPCCP is compared with state-of-the-art solutions. The experimental evaluations show that DPCCP outperforms existing solutions in terms of bandwidth utilization, convergence time and packet loss.**

***Index Terms*—Named Data Networking, Congestion Control, Multipath, Multi-source.**

I. Introduction

Content-centric applications have dominated Internet usage. To improve the performance of content distribution, Named Data Networking [1] (NDN) has been proposed by routing request (Interest) and content (Data) packets using universal resource identifiers (names) without host addresses. Name-based routing and possible content replication (e.g., caching in routers or peers) in NDN naturally lead to a multipath and multi-source transmission paradigm.

This work was supported in part by Irish Research Council under ID GOIPD/2019/874, Science Foundation Ireland (SFI) under Grant Number 16/RC/3918 and 13/SIRG/2178, co-funded by the European Regional Development Fund, National Natural Science Foundation of China under Grant Number 61671126, Science and Technology Program of Sichuan Province of China under Grant Number 2019JDR0022.

Yuhang Ye, Yuansong Qiao, Brian Lee and Ronan Flynn are with the Software Research Institute, Athlone Institute of Technology, Athlone, Westmeath, Ireland (e-mail: yye@research.ait.ie, ysqiao@research.ait.ie, blee@ait.ie, rflynn@ait.ie)

Jin Xu is with School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan (e-mail: jinxu@uestc.edu.cn)

Guiming Fang is with Institute of Software, Chinese Academy of Sciences, Beijing, China (e-mail: fgm@iscas.ac.cn)

The addressless packet forwarding process in the original NDN bring significant challenges to multi-source and multipath transmission control. First, transmission paths and content sources are usually unknown to consumers. Second, content cached in routers is unpredictable to consumers. For example, the content segments cached in a router may be ephemeral and discontinuous (depending on caching policy). As a result, existing end-to-end multipath transmission control schemes (e.g. MPTCP [2] and CMT-SCTP [3]) for TCP/IP architecture no longer fit NDN. They do not support the cases when content fractions may be randomly cached in the middle of paths. Moreover, host addresses are unavailable to NDN consumers to distinguish transmission paths.

The current transmission performance of NDN is affected by two functions [4]: 1) receiver-driven congestion control and 2) hop-by-hop adaptive forwarding. Specifically, receiver-driven congestion control determines how much content traffic a consumer can request from the network, and hop-by-hop adaptive forwarding decides how the requested content traffic is distributed across different paths. In the conventional NDN, 100% of adaptive forwarding is implemented at router-side, i.e., consumers do not control the forwarding of Interest packets. In consequence, each consumer can only use a "coupled" congestion control to control the traffic on all paths as an entirety. In consequence, if one path becomes congested earlier than others, to avoid congestion, the consumer has to reduce its content requesting rate to all paths even when some paths are congestion-free [5].

A traditional way to tackle this bandwidth utilization issue is to develop an optimal adaptive forwarding strategy i.e., routers know how to distribute traffic on paths so that the loads of all bottlenecks are just balanced. The optimal forwarding can be modelled via a multi-commodity flow problem with time-variant capacity constraints that are affected by cache hits [6]. Solving this problem has proven to be pseudo-poly NP-hard [7]. Even though we consider an intuitive approximation that each router balances the congestion perceived from each interface, it is still challenging to implement an accuracy forwarding strategy due to the difficulty of congestion measurement. For example, an inaccurate measurement (e.g. the numbers of pending Interests OMCC-RF [6]) will lead to low bandwidth utilization [5], [8], [9]. By contrast, if routers need accurate measurement, they must discover and maintain the congestion status per bottleneck then distribute traffic only to the congestion-free paths, which will bring routers non-trivial overheads.

Path switching [10] in NDN provides a different way to solve bandwidth utilization issues. It allows a consumer to control the traffic on each path independently. Our previous work Path-specified Transport Protocol (PTP) [5] integrates path switching and MPTCP-like congestion control to improve networking efficiency. However, PTP lacks a mathematical model to present the network utility and system equilibrium.

In this paper, we will formulate the optimal multipath traffic allocation into a global model and present a delay-based congestion control algorithm to solve it. The congestion control framework follows PTP i.e., separating traffic control for each path. In addition, this paper takes a further step to analyze the impact of cache on congestion and optimize the link utilization when caches are presented. This key contribution consists of three parts:

1) Employed Network Utility Maximization (NUM) to model the multi-source/multipath transmission in NDN.
2) Presented a high-performance Delay-based Path-specified Congestion Control Protocol (DPCCP) derived from NUM.
3) Evaluated DPCCP with SOTA solutions. The results show that DPCCP achieves faster bandwidth and fairness convergence.

For the sake of simplicity, the following content will often abbreviate “multi-source and multi-path transmission” into “multipath transmission” unless explicit notations are given.

The remainder of this paper is organized as follows. We present related works in Section II. Section III illustrates basic definitions. Section IV formulates NDN multipath transmission using a NUM model; Section V shows the methodology of a delay-based approach that can resolve the NUM model; Section VI presents the detailed design of the delay-based congestion control protocol; Section VII evaluates the performance of DPCCP; Section VIII concludes the paper.

II. Related Works

The conventional traffic control in NDN consists of 1) congestion control and 2) adaptive forwarding. In addition, we consider the 3rd category which denotes the recent traffic control based on path switching.

A. *Congestion Control*

The NDN congestion control falls into two broad categories in: 1) receiver-driven and 2) the hop-by-hop interest shaping.

The receiver-driven NDN congestion control is like TCP in which each consumer treats all paths as an entirety and adjusts the content requesting rate according to congestion signals. The signals include 1) packet loss, 2) delay and 3) explicit congestion notification (ECN). Early studies [11] use a timeout scheme to detect packet losses. However, the latencies on different paths may be different so the timeout estimation is usually unreliable [12]. RAAQM is a RTT-based [6] solution that requires Data packets to record routing labels which enables consumers to distinguish congestion per path. The ECN-based solution [12] requires bottlenecks to explicitly notify congestion using ECNs. Practical congestion control (PCON) [9] is an approach that develops an AQM scheme and the corresponding congestion control protocol. Its AQM scheme is an extension of CoDel, which monitors the congestion (packet stalling time) for both Data and Interest traffic. In addition, it employs ECN to notify consumers of congestion. The core problem of the existing receiver-driven approaches is that the consumer controls the traffic on different paths as an entirety, which results in low bandwidth utilization if one path is congested while the others are not.

The hop-by-hop interest shaping mainly aims to prevent the congestion of Data packets by limiting the Interest forwarding rate. Hop-by-hop interest shaping [14], [15] (HoBHIS) limits the forwarding rate of Interest packets by explicitly allocating fair bandwidths to different flows. CHoPCoP [12] regulates the forwarding rate based on the number of pending (Interest) requests and the packets delayed in the link layer. Dynamic Interest Limiting [16] adjusts the forwarding rate based on the NACKs that are from downstream nodes. Hop-by-hop interest shaping [17] (HIS) pre-calculates the optimal interest forwarding rate based on the local and neighbor bandwidths. Most of these approaches (e.g., HIS, CHoPCoP) requires link layer information such as queue and link capacities, which becomes impractical then NDN is deployed as an overlay.

B. *Adaptive Forwarding*

Adaptive forwarding enables multipath transmission when consumers treat all paths as an entirety. Probability-based Adaptive Forwarding [18] is one solution inspired by ant colony optimization, which selects forwarding interfaces according to RTT. On-demand Multi-Path Interest Forwarding [19] allocates traffic to disjoint paths via the weighted round-robin scheme which also relies on round-trip time. By emulating a liquid piping system, Stochastic Adaptive Forwarding proposed by Daniel et al. [20] provides a robust traffic allocation to tolerate incomplete routing information. Carofiglio et al. [6] proposed a forwarding strategy –Request Forwarding Algorithm (RFA)–, to solve a multi-flow problem. It uses the number of pending requests of each forwarding interface as the approximate congestion metric and equalize it for all interfaces. PCON [9] uses an ECN-based forwarding algorithm to direct excessive traffic to idle links therefore supporting multipath transmission.

Except PCON, other forwarding strategies suffer from low bandwidth issue because their approximate measurements (e.g., RTT and pending requests) cannot accurately reflect the level of congestion. The intuition that PCON shifts traffic from busy links to idle ones is plausible [9]. However, the ECN-based congestion balancing is not always stable and sometimes hard to converge, according to our experiments (Section VII). We interpret this as the “*oscillation of bottleneck*”. Due to receiver-driven congestion control, the bandwidth demands changes over the time. This potentially shifts the bottlenecks and alters the equilibrium of the system. Moreover, ECN is a delayed feedback which causes continuously oscillations of bandwidth demands and leads to low bandwidth utilization.

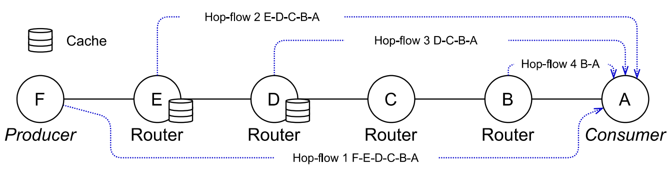


Figure 1 Relationship between Hop-flow and Sub-flow

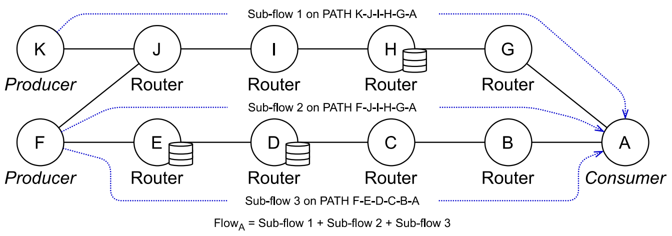

Sub-flow 1 on PATH K-J-I-H-G-A
K
J
I
H
G
Producer
Router
Router
Router
Router
Sub-flow 2 on PATH F-J-I-H-G-A
F
E
D
C
B
A
Producer
Router
Router
Router
Router
Consumer
Sub-flow 3 on PATH F-E-D-C-B-A
Flow_A = Sub-flow 1 + Sub-flow 2 + Sub-flow 3

Figure 2 Relationship between Sub-flow and Flow

C. *Path Switching*

Path switching [10] was developed for high-speed Interest forwarding. It allows each consumer to decide the path to forward each Interest packet. A consumer attached a path label in the Interest message, which is forwarded to a next-hop through the corresponding egress interfaces in conjunction with longest name prefix match (LNPM) FIB lookup. The path switching does not require heavy adaptive forwarding logic at routers, which improves system scalability.

The companion paper [13] of path switching (but published earlier) took advantage of path-switch and developed a RCP-like congestion control approach –MIRCC– to support multi-path transmission. MIRCC is a rate-based approach which allows consumer to control the traffic for each flow separately. It requires routers to periodically estimate the desire Interest forwarding rate for each flow. Then these rates are returned to consumers to adjust requesting speeds. The core limitations of MIRCC include: 1) each router needs to calculate the optimal rate for each flow, which is not scalable 2) it requires accessing underlying link status (queue), which can be infeasible if NDN is an overlay and 3) its RCP-based congestion control cannot support in-network cache.

Inspired by path switching, we presented PTP [5]. It extends the loss-based MPTCP congestion control algorithm (Linked Increases [2]) to support multipath transmission and in-network caches in NDN, without accessing link status. Evaluation results show that PTP offers better bandwidth utilizations while maintaining similar forwarding flexibility. Nevertheless, PTP cannot support the case when content are all provided by caches and it always fully fills bottleneck queues, which results in large delays and bufferbloat [21].

III. Basic Definitions and Assumptions

A. *Content Source and Content Producer*

In this paper, a content source denotes an NDN device that can provide the requested content segment, which may or may not store the whole content object that corresponds to the segment. A content producer denotes the node that can provide the whole content object. A content producer is always a content source, but a content source may not be a content producer. For example, a router may only cache a few segments of a content object.

B. *Hop-flow, Sub-flow and Flow*

To better understand the traffic in NDN, three types of *flows* are defined, namely *hop-flow*, *sub-flow* and *flow*. The *hop-flow* is the traffic from a content source that is on a path in-between the producer and the consumer. An example is shown in Figure 1. Four different sources (B, D, E and F) are concurrently serving different segments of a content object to Consumer A. The traffic from each individual source (B, D, E or F) denotes a hop-flow. The summation of the four *hop-flows* on path F-E-D-C-B-A is defined as a *sub-flow*. The total traffic that is on one consumer-producer path is a *sub-flow*.

In the case of multipath transmission, a consumer may receive content segments from multiple sub-flows via multiple consumer-producer paths. As shown in Figure 2, Consumer A downloads different fractions of a content object from two producers (K and F) via three consumer-producer paths (K-J-I-H-G-A, F-J-I-H-G-A and F-E-D-C-B-A). The three sub-flows are aggregated at Consumer A. The summation of the three *sub-flows* is defined as a *flow* i.e., the total traffic that is received by a consumer from multiple paths.

IV. Problem Formulation

Network Utility Maximization (NUM) [22] is a model to maximize the total utilities of all network flows. In TCP and MPTCP, by defining the utility function to be monotonically increasing and strictly concave, there always exists a solution to maximize the overall utility of all flows [22]. Following this principle, the following section presents a NUM model to formulate the multipath transmission problem in NDN supporting in-network caching. Table I lists the key variables that are used for NUM modelling. The subscripts *s*, *r*, *h* denotes used to index the subsets, if *R* denotes the set of all sub-flows, R_s denotes the sub-flows that belong to flow *s*.

A. *NUM model for NDN*

An NDN network can be viewed as a directed graph with a set of links (edges) $l \in L$ with finite capacities $\mathbf{c} = c_l \mid l \in L$. The link capacities are also known as bandwidth, which are shared by a set of flows $s \in S$. For each flow *s*, it is associated with a set of sub-flows $r \in R_s$ which is a subset of the set of all sub-flows R ($R_s \subseteq R$). This relationship between a sub-flow *r* and a flow *s* is given by the matrix **B**, where $b_{s,r} = 1$ if the sub-flow *r* belongs to the flow *s*, and otherwise $b_{s,r} = 0$. Each sub-flow *r* travels through a single producer-consumer path that is a set of links $l \in L_r$, where L_r is a subset of all links ($L_r \subseteq L$). The relationship between sub-flows *R* and links *L* is given by the matrix **A**, where its element $a_{l,r}$ denotes the proportion of a sub-flow *r* that travels through a link *l*. The NUM model of NDN is given in eq. (1).

$$\begin{aligned} \max_{y_s \geq 0} \quad & \sum_{s\in S} U \;\; y_s \\ s.t. \quad & \mathbf{y} = \mathbf{Bx} \qquad b_{s,r} \in \;\; 0,1 \\ & \mathbf{c} - \mathbf{Ax} \geq 0 \quad 0 \leq a_{r,l} \leq 1 \end{aligned} \tag{1}$$

Here, x_r denotes the downloading rate of the sub-flow $r \in R$; $\mathbf{x}$ is a vector contains all x_r; y_s denotes the total downloading rate of the flow $s \in S$; $\mathbf{y}$ denotes a vector contains all $y_s = \sum_{r \in R_S} x_r$. The NUM model [23] in MPTCP is different from that in NDN. In particular, the matrix **A** in MPTCP is a Boolean matrix where each element $a_{l,r}$ denotes whether a sub-flow *r* passes through a link *l whereas* **A** *in NDN is a non-negative fractional parameter matrix*.

The fractions in the matrix A are to model the hop flows caused by caching. An example to interpret $a_{l,r}$ in the NDN model is given in Figure 3. Consumer A attempts to download a content object from Producer C and Router B concurrently, where 40% of the requested content segments are provided by Router B and the remaining 60% are provided by Producer C. This is to say, the proportion of the sub-flow $x_{C\rightarrow B\rightarrow A}$ on the link $l_{B\rightarrow A}$ is 60% + 40% = 100% and the proportion is 60% on the link $l_{C\rightarrow B}$. The two proportion values (100% and 60%) are the elements in the matrix **A**. The new capacity constraint is that the proportion of the sub-flow $x_{C\rightarrow B\rightarrow A}$ on each link (C→B and B→A) should be no larger than the link capacity, as given in eq. (2).

$$\overbrace{\begin{bmatrix} c_{B\rightarrow A} \\ c_{C\rightarrow B} \end{bmatrix}}^{\mathbf{c}} - \overbrace{\begin{bmatrix} 100\% \\ 60\% \end{bmatrix}}^{\mathbf{A}} x_{C\rightarrow B\rightarrow A} \geq 0 \tag{2}$$

In NDN, the distributions of caches change over time. Thus, matrix **A** is time-variant. For modelling, the element $a_{r,l}$ in **A** is treated as a constant value. These non-negative fractional values in matrix **A** will not affect the decomposition of the NUM model and the derivation of the equilibriums [23]. However, these fractions will change the protocol design. In MPTCP, the measurement of matrix **A** is straightforward. For the links l_r that formulate a flow *r*, their corresponding values in **A** are 1. If the path *r* fails, the values corresponding to l_r are set to 0. MPTCP simply removes this flow *r* from its congestion control list. In NDN, consumers need to measure the change of matrix **A** according to the recent hop-flow traffic and react to its changes. If Router B (in Figure 3) no longer provides cached content, Consumer A will need to get content from Producer C. This will result in the change of matrix **A** from $\;1.0 \;\; 0.6\;^{\mathrm{T}}$ to $\;1.0 \;\; 1.0\;^{\mathrm{T}}$. This type of change of matrix **A** is not considered in MPTCP. As a result, MPTCP congestion control cannot be directly used in NDN.

B. *Lagrange-dual Decomposition of NUM*

A standard distributed algorithm to solve eq. (1) is to use dual decomposition. Define the Lagrangian in eq. (3).

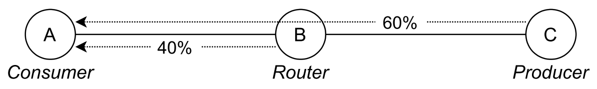


Figure 3 Traffic brought by two hop-flows in a sub-flow

$$\begin{aligned} L \;\; y_s, \lambda_l \;\; &= \sum_{s\in S} U \;\; y_s \;\; + \sum_{l \in L} \lambda_l \left(c_l - \sum_{r\in R} a_{l,r} x_r \right) \\ &= \sum_{s\in S} U \;\; y_s \;\; - \sum_{s\in S} \sum_{r \in R_S} q_r x_r + \sum_{l\in L} \lambda_l c_l \\ &= \sum_{s\in S} \left[U \;\; y_s \;\; - \sum_{r\in R_S} q_r x_r \right] + \sum_{l \in L} \lambda_l c_l \end{aligned} \tag{3}$$

Here, λ_l is the Lagrange multiplier, which can be interpreted as the price or the congestion level (e.g. loss probability or queuing delay) associated with a link *l* [8], [9] and q_r denotes the aggregated price that is received by a consumer for a sub-flow *r* as given in eq.(4).

$$q_r = \sum_{l \in L_r} \lambda_l a_{l,r} \tag{4}$$

The key to understanding the optimal solutions of eq. (3) is to regard x_r as primal variables and λ_l as dual variables [24]. Obviously, the objective function in eq. (1) is concave to x_r; the constraints are all convex sets and Slater's condition holds, i.e. there exists at least one path (a strictly feasible solution) with a positive value of bandwidth to access the content) such that constraints are all satisfied. As a result, the duality gap is 0, which means solving the Lagrange dual problem is equivalent to solving the primal problem. The Lagrangian dual d^* [23] is given in eq. (5).

$$\begin{aligned} d^* &= \min_{\lambda_l} \max_{x_r} L \;\; y_s, \lambda_l \\ &= \min_{\lambda_l} \max_{x_r} \sum_{s \in S} \left[U \;\; y_s \;\; - \sum_{r \in R_S} q_r x_r \right] + \sum_{l\in L} \lambda_l c_l \\ &= \min_{\lambda_l} \left(\sum_{s\in S} \left(\max_{x_r, r\in R_S} L_s \;\; x_r, \lambda \;\; \right) + \sum_{l \in L} \lambda_l c_l \right) \end{aligned} \tag{5}$$

Solving d^* requires solving $\max_{x_r, r\in R_S} L_s \;\; x_r$ for each flow *s* individually, as shown in eq.(6) via congestion control.

$$L_s \;\; x_r \;\; = U \left(\sum_{r \in R_S} x_r \right) - \sum_{r\in R_S} q_r^* x_r \tag{6}$$

Here, q_r^* is the current price by solving the master dual problem $\min_{\lambda_l} D \;\; \lambda_l$ at routers based on Active Queue Management, as shown in eq.(7).

$$D \;\; \lambda_l \;\; = \sum_{s\in S} \left(U \left(\sum_{r\in R_S} x_r^* \right) - \sum_{r \in R_S} q_r x_r^* \right) + \sum_{l\in L} \lambda_l c_l \tag{7}$$

Here, x_r^* is the rate by solving each sub-problem. λ_l is the congestion level; λ_l will remain at 0 if the link *l* is not overloaded because the capacity constraints are satisfied. Once the link is overloaded, λ_l will be updated using a gradient method. By assigning a different meaning to q (e.g., queue length, packet loss probability or delay) and developing a proper utility function U, it will derive a family of multipath congestion control protocols for NDN. Because solving this NUM model needs to distinguish sub-flows r on different paths, it needs the support of path switching [10] which will be discussed in Section VI.A.

V. DELAY-BASED NETWORK UTILITY MAXIMIZATION IN NDN

Delay-based congestion control have been studied in MPTCP literatures e.g., wVegas [23]. This section will show when a network has cache presents, how to measure/aggregate delays in NDN and how to use it to solve the NUM problem with the objective of maximizing logarithmic utility. Table I depicts the variables for solving the two problems.

A. Master Dual Problem and Gradient Descent Algorithm

For the master problem $\min_{\lambda_l} D(\lambda_l)$, each router updates the price λ_l for each link l using a projected gradient descent method [23] based on the current rates x_r^*, using eq. (8).

$$\lambda_l(t+\Delta t) = \left[\lambda_l(t) + \gamma\left(\sum_r a_{l,r}x_r^* - c_l\right)\Delta t\right]^+ \qquad (8)$$

Here, in a delay-based approach, $\sum_r a_{l,r}x_r^* - c_l$ denotes the excessive bandwidth on the link l; γ denotes a normalization factor that maps the excessive bandwidth to the queuing delay, i.e. $\gamma = 1/c_l$; $(\cdot)^+$ guarantees every λ_l to be nonnegative ($(x)^+ = 0$ if $x \le 0$ and $(x)^+ = x$ if $x > 0$), meaning the queuing delay is never negative. The advantage of using a delay-based approach is that, with such configurations, the price λ_l can be interpreted as the packet queuing delay of link l and routers do not need to explicitly solve eq. (8). Instead, eq. (8) can be viewed as the *native* behavior of the queue of link l.

B. Congestion Equality Principle

According to the Karush-Kuhn-Tucker conditions eq. (3), a ***necessary*** condition to achieve the optimal bandwidth utilization in eq. (1) is the *Congestion Equality Principle*:

"*if every flow strives to equalize the **extent of congestion** that it perceives on all its available paths by means of shifting traffic, then network resources will be fairly and efficiently shared by all the flows*" wVegas [23]

This principle has been either implicitly or explicitly applied to multipath congestion control. In wVegas, the ***extent of congestion*** denotes the end-to-end queuing delay q_r per sub-flow. This principle shows that when a flow s tends to use only the paths with the minimum queuing delay while giving up others, thereby maximizing its utility.

In NDN, content segments may be cached in the intermediate routers i.e., **A** is no longer a Boolean matrix therefore q_r can be viewed as the weighted summation of queuing delays q_h for hop-flows $h \in H_r$. Following the *Congestion Equality Principle*, each consumer will equalize the aggregated queuing delays q_r for each sub-flow r belonging to the flow s. The estimation is given in eq. (9) derived from eq.(4), referring to Appendix I for proof.

Table I Variables and Symbols for NUM and DPCCP

Variable	Definition / Explanation
S, R, H	The set of a flow, sub-flow and hop-flow
s, r, h	A flow, sub-flow and hop-flow
x_r	The equilibrium rate of a sub-flow *r*
y_s	The equilibrium rate of a flow *s*
λ_l	The price of a link *l*
q_r	The aggregated price of a sub-flow *r*
$\Delta_s, \Delta_r, \Delta_h$	The number of backlogged packets of *s, r, h*
α_s, α_r	The setpoint of the backlogged packets of *s, r, h*
q_r, q_h	The queuing delay for a sub-flow *r* or a hop-flow *h*
q_s	The desired queuing delay for all the sub-flows *r*
u_r	The requesting rate of a sub-flow *r*

$$q_r = \sum_{l\in L_r} \lambda_l a_{l,r} = \sum_{h\in H_r} w_h q_h \qquad (9)$$

Here, the weight w_h denotes the split ratio of the hop-flow h to the corresponding sub-flow r; q_h denotes the queuing delay of a hop-flow $h \in H_r$. The practical ways to obtain q_h will be discussed in Section VI.C.

C. Sub-problem and Utility Function

Sub-problems L_s can be solved at the consumer-side via congestion control. The optimal utility is defined based on the steady state. For each sub-problem $\max_{x_r} L_s(x_r)$, at the steady state, the optimal solution, eq. (10) can be derived from the KKT conditions [23].

$$\begin{aligned} &U'(y_s) - \sum_{r\in R_s} \omega_r q_r = 0 \\ &U'(y_s) = U'\left(\sum_{r\in R_s} x_r^*\right) = \sum_{r\in R_s} \omega_r q_r \end{aligned} \qquad (10)$$

Here, ω_r denotes the proportion of the sub-flow $r \in R_s$ to a flow s, where the summation of ω_r is 1 for all the sub-flows $r \in R_s$. At the steady-state, all q_r will have the same value equal q_s.Therefore eq. (10) can be re-written as eq. (11).

$$U'(y_s) = q_s\left(\sum_{r\in R_s} \omega_r\right) = q_s \qquad (11)$$

By defining the setpoint of the steady-state $\alpha_s = y_s q_s$ (the number of packets backlogged in different paths at the steady-state), the utility function can be resolved as follows:

$$U'(y_s) = \frac{\alpha_s}{y_s}, \quad U(y_s) = \alpha_s \log(y_s) \qquad (12)$$

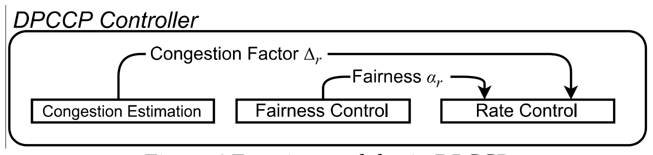


Figure 4 Function modules in DPCCP

Clearly, this utility function is increasing, twice continuously differentiable and strictly concave with respect of y_s. In the case that each consumer controls its requesting rate so that the number of backlogged packets converges towards α_s, each consumer will achieve the rate-proportional fairness, as the result of the logarithm utility function.

In conclusion, the optimization of the master dual problem is the natural behavior of links, and the bandwidth resources can be fairly and efficiently shared by all flows (consumers) if the following two conditions are satisfied by consumers.

Condition 1: for every sub-flow $r \in R_s$, the queuing delay q_r is equalized to the same value q_s.

Condition 2: for every flow *s*, the number of backlogged packets of all sub-flows is converged to a constant α_s.

VI. Delay-based Path-specified Congestion Control Protocol

A. *Assumption: Path Switching*

The proposed protocol requires using path switching [10] as the underlying forwarding plane so that consumers can decide the forwarding path of each Interest packets, i.e., each consumer is capable to control each sub-flow independently. To preserve the transmission resilience provided by NDN, our work [5] re-activates the default (name-based) forwarding scheme if the specified path is not valid.

In order to guarantee scalability, path switching only allows consumers to specify consumer-producer paths but not include the paths from cache. In other words, the consumer controls a sub-flow as an entirety instead of controlling each hop-flow. The rationale is that the different caching policies will affect the content distribution in routers. For example, some cached segments may be scattered but the others may be continuous. Enabling a consumer to request certain content segments from a specified router requires the synchronization of segment-level routing information amongst all nodes, which is impractical due to the high communication overheads.

B. *Protocol Overview*

DPCCP contains three modules: 1) congestion estimation, 2) fairness control and 3) rate control, as shown in Figure 4. The design principle of DPCCP follows MPTCP wVegas and TCP BBR. The equilibrium of DPCCP is measured and adjusted based on the number of backlogged packets [23]. The congestion estimation module measures the number of backlogged packets Δ_r for each sub-flow *r* based on RTT and baseRTT. The fairness control module calculates the target number of backlogged packets α_r for each sub-flow *r* such that the aggregated queuing delay q_r is equalized [23]. Based on the measurement Δ_r and the calculated setpoint α_r, the rate control module adjusts the requesting rate to converge Δ_r to α_r. The RTT probing and bandwidth probing methods in DPCCP borrows the same concept in TCP BBR but are implemented in a different way. This is because BBR cannot achieve multi-flow fairness in the case of multipath transmission. The BBR consumer with more sub-flows will receive more bandwidth than the consumer with fewer sub-flows. Moreover, the bandwidth probing in BBR is based on packet-pair probing which assumes a train of packets travel through the same path. This is no longer be valid in NDN because some packets can be provided by the cache while others are provided by the producer. To this end, we switch to probe bottleneck bandwidth to check if its queue is empty. This design is more generable in NDN and can better support complicated transmission scenarios. The following sections will present the detailed implementations of these modules.

C. *Congestion Estimation*

The measured number of backlogged packets for a sub-flow Δ_r is the product of the sub-flow queuing delay q_r and the sub-flow rate x_r. As shown in the previous section, q_r equals the weighted summation of the hop-flow queuing delays q_h. This indicates Δ_r can be interpreted as the total number of backlogged packets of this sub-flow *r*, which is equivalent to the summation of the number of backlogged packets Δ_h of the hop-flow *h* as shown in (13).

$$\begin{aligned}\Delta_r = q_r x_r &= \textstyle\sum_{h\in H_r} q_h w_h \;\; x_r = \sum_{h\in H_r} q_h \;\; w_h x_r \\ &= \textstyle\sum_{h\in H_r} q_h x_h = \sum_{h\in H_r} \Delta_h\end{aligned} \tag{13}$$

For each hop-flow *h*, Δ_h is calculated as the product of the queuing delay q_h and the rate x_h, as shown in eq. (14).

$$\Delta_h = x_h q_h = \bar{x}_h \;\; \bar{d}_h - \tilde{d}_h \tag{14}$$

Here, the queuing delay q_h is the difference between the recent averaged RTT $\bar{d}_h$ and the baseRTT $\tilde{d}_h$ of a hop-flow *h*; $\bar{x}_h$ denotes the average value of the recent data receiving rate. To accurately calculate the queuing delay q_h for each hop-flow, it is necessary to separate the traffic from different hop-flows. The path-specified approaches inherently enable a consumer to separate traffic for each sub-flow by checking the path tag recorded for each forwarded Interest packet. However, it does not allow a consumer to distinguish the hop-flows because a consumer cannot predict which node will return the requested Data packet. A 6-bit hop-count tag (to support at most 63 hops) is added to the Data packet to enable counting the number of hops that the Data packet has travelled. For example, if a consumer recorded an Interest packet to be forwarded along path A-B-C-D, and the returned Data packet carries a tag with 2 hops, the consumer will know that the Data packet is returned from C via C-B and B-A.

Another critical challenge is to measure the baseRTT for each hop-flow, which significantly affects the accuracy of estimating the number of backlogged packets. In this design, if

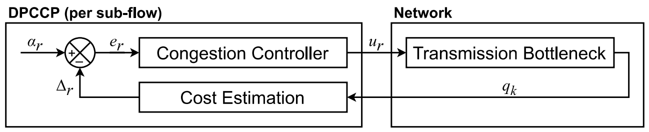


Figure 5 Control Diagram of DPCCP

a new sub-flow joins the network, it triggers the other sub-flows to reduce the requesting rates, so that the queue at the bottleneck is emptied, which allows the consumer to measure its hop-flows' baseRTTs. The detailed implementation is discussed in Section VI.E.7). In addition, if a consumer detects a significant change of recent RTTs for a hop-flow [25], the consumer will assume that the baseRTTs of certain hop-flows are inaccurate (e.g. the routing paths of the lower layer devices have changed). To re-measure baseRTTs, the corresponding consumers reduce the requesting rates of the sub-flow to empty the bottleneck's queue. The implementation to drain bottleneck queues is further presented in Section VI.E.6).

D. *Fairness Control*

Since q_r and Δ_s are always interacting with each other, it is challenging to satisfy the two conditions by controlling q_r and Δ_s separately. An alternative method [23] assigns a target number of backlogged packets α_r to each sub-flow *r* such that q_r is equalized within a flow *s* with the constraints that the summation of α_r $(r \in R_s)$ is fixed at α_s. To determine the suitable α_r, an iterative algorithm [23] is employed. In the steady-state, Δ_r at each sub-flow will converge to the target number of backlogging packets α_r while q_r is equalized to q_s, the relationship between α_r and q_s satisfies eq. (15).

$$\alpha_r = q_s x_r \quad \Rightarrow \quad \alpha_r \propto x_r \tag{15}$$

This indicates, to equalize q_r, a necessary condition is that α_r is proportional to x_r, i.e. α_r is adjusted according to eq. (16). The proof can be found in [23].

$$\alpha_r = \alpha_s \frac{x_r}{y_s} \tag{16}$$

Here, the target backlogged number of packets of a sub-flow *r*; α_r is considered as the *set point* for congestion control. Each sub-flow adjusts its requesting rate such that Δ_r is converged to the set point α_r.

E. *Rate Control*

1) *Control Diagram*

In DPCCP, efficiency and fairness are achieved by adjusting the sub-flow requesting rate u_r such that the number of backlogged packets of a sub-flow Δ_r converges to the setpoint α_r. The diagram to control the sub-flow requesting rate u_r is shown in Figure 5 . Note that u_r may not satisfy the capacity constraint and it can cause congestion at certain links

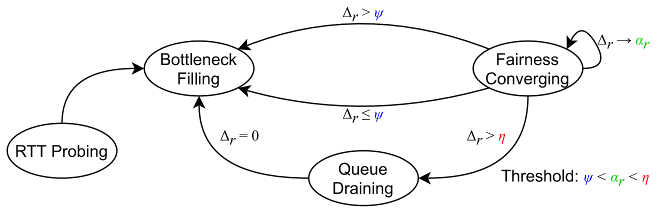


Figure 6 State-transfer Diagram of DPCCP

and increase the queuing delay q_r. For each sub-flow, the setpoint is α_r. The difference between the setpoint α_r and Δ_r is defined as an error e_r. The congestion controller takes the error as an input to generate the new requesting rate u_r as the stimulation to the bottleneck. The bottleneck maps requesting rates to queuing delays, which are detected by consumers. As shown in previous works [23], [24], the fairness in a Vegas-like scheme is controlled by the setpoint α_r, which means that any stable control law (e.g. AIAD or AIMD) that can converge Δ_r to α_r is suitable.

2) *Rate-based Congestion Control and Control Cycle*

As multiple hop-flows with different RTTs may exist in one sub-flow, it is improper to define one single congestion window for all the hop-flows as these hop-flows are within distinctive RTT periods and need different control cycles. However, it is infeasible to set a congestion window for each hop-flow because the consumer cannot specify a router as the source of a content segment. As a result, DPCCP avoids using congestion windows for traffic control. Instead, a rate-based controller is employed. In DPCCP, the rate-based approach allows consumers to match the bottleneck capacities using the requesting rates without being affected by the different RTTs of hop-flows. The control cycle to adjust the requesting rate is set to the maximum RTT value of the Data packets that are received within the previous cycle. This allows a consumer to measure the congestion level of the previous control cycle.

3) *State-transfer Graph*

For each sub-flow *r*, different congestion control states are defined based on Δ_r. The transfer diagram is shown in Figure 6. They are Bottleneck-Filling (BF), Fairness-Converging (FC) and Queue-Draining (QD). In addition, an RTT-Probing (RP) state is introduced to measure the baseRTT when a new sub-flow joins the network. The transfers between states are triggered based on thresholds ψ, α_r, η. Specifically, ψ is a very small value, i.e. $\Delta_r < \psi$ indicates that the bottleneck is underutilized; α_r is the control setpoint, i.e. it is calculated according to eq. (16) and η is the largest threshold, i.e. $\Delta_r > \eta$ means that the bottleneck is heavily congested.

During the BF state, the sub-flow increases the bandwidth utilization of the underloaded bottleneck. During the FC state, the consumer converges Δ_r to the setpoint α_r, which enables fair and efficient bandwidth sharing among consumers. During the QD state, the consumer minimizes the requesting rate of the sub-flow to empty the bottleneck's queue. During the RP state, a consumer enables probing baseRTTs for a new sub-flow (and its hop-flows) by notifying (see Section VI.E.7)) other sub-flows to enter the QD state to drain the queue in the bottleneck. This enables the consumer to measure baseRTT for each hop-flow.

4) *Bottleneck-Filling State*

When a bottleneck is underutilized, the sub-flows that travel through this bottleneck are required to rapidly increase their requesting rate to improve the bandwidth utilization. A Zigzag Multiplicative Increase (Z-MI) algorithm is proposed for the bottleneck filling state, which is shown in eq. (17).

$$\text{Z-MI:}\begin{cases} u_r = \ 1+\beta \ \ u_r^* & \text{Stage I} \\ u_r = u_r^* & \text{Stage II} \\ u_r = \ 1-\beta \ \ u_r^* & \text{Stage III} \end{cases} \tag{17}$$

Here, u_r^* is the tolerable requesting rate, which means the network can tolerant this rate without backlogged packets. Three stages are involved in a Z-MI round, where each stage lasts for a control cycle. During Stage I, the requesting rate is increased to $1+\beta \ \ u_r^*$ thus to probe if the bottleneck can tolerate the increased rate. During Stage II, the requesting rate is kept as u_r^* to stabilize the queue. During Stage III, the requesting rate is decreased to $1-\beta \ \ u_r^*$ thus to cancel the congestion that may be caused by Stage I. If a few number of packets ($\Delta_r < \psi$) is detected during the current Z-MI round, the tolerable requesting rate u_r^* of the next Z-MI round is increased to the current Stage I requesting rate, eq. (18).

$$u_r^* := \ 1+\beta \ \ u_r^* \quad \text{if } \Delta_r < \psi \tag{18}$$

This Zigzag function can detect if the increased requesting rate $1+\beta \ \ u_r^*$ (Stage I) exceeds the bottleneck's capacity. Particularly, if the requesting rate is larger than the capacity, the excessive bandwidth causes packets to be backlogged in the bottleneck. These packets lead to queuing delays of the hop-flows, which will be detected in Stage II or Stage III. When congestion is detected, the consumer records the current receiving rate of the sub-flow and assigns it to the requesting rate. The rationale for using the receiving rate as the requesting rate is that the receiving rate is the saturated rate that can critically fill the bottleneck. Then, it quits the BF state and enters the FC state. If no congestion is detected, the sub-flow can safely increase its stable requesting rate u_r^* to $1+\beta \ \ u_r^*$ and start the next Z-MI round. Note that, as the back-off stage (Stage III) reduces the requesting rate, which cancels the backlogged packets introduced by Stage I, the algorithm introduces zero backlogged packets each round.

5) *Fairness-Converging State*

When the bottleneck is fully utilized, the requesting rate of each sub-flow that travels through the same bottleneck adjusts its number of backlogged packets Δ_r to the setpoint α_r to achieve multi-flow fairness. In the original Vegas, an AIAD algorithm is used to adjust Δ_r. A significant drawback of the original AIAD algorithm employed in Vegas is its low convergence speed [26]. To tackle this issue, an Adaptive Additive Increase Additive Decrease (A-AIAD) algorithm is proposed for the FC state. Note that A-AIAD is a rate-based control law that is derived from the original window-based AIAD control law with adaptive factors. The rate-based A-AIAD is shown in eq. (19).

$$\text{A-AIAD:}\begin{cases} u_r = x_r + \rho & \Delta_r < \alpha_r \\ u_r = x_r - \rho & \Delta_r > \alpha_r \end{cases} \tag{19}$$

Here, ρ is the adaptive factor, which is adjusted according to the expected convergence time and the actual convergence speed. The factor is changed according to eq. (20).

$$\begin{array}{lll} \rho := \rho + 1 & \mathit{if} & \left| \Delta_r \ n \ - \Delta_r \ n - K \ \right| / K < 1 \\ \rho := \rho - 1 & \mathit{if} & \left| \Delta_r \ n \ - \Delta_r \ n - K \ \right| / K > 1 \\ \rho := 1 & \mathit{if} & \begin{array}{l} \Delta_r \ n \ \geq \alpha_r \geq \Delta_r \ n-1 \\ \Delta_r \ n \ \leq \alpha_r \leq \Delta_r \ n-1 \end{array} \end{array} \tag{20}$$

The average convergence speed $\left| \Delta_r \ n \ - \Delta_r \ n - K \ \right| / K$ is calculated every K non-overlapping baseRTT periods (K > 2, K = 3 in our implementation). Accordingly, the adaptive factor ρ is updated every K non-overlapping baseRTT periods. Supposing the expected time to change Δ_r by 3 towards α_r is 3 baseRTT periods. This means the changing speed of Δ_r (the number of backlogged packets) is 1 per baseRTT, if the current convergence speed is slower than the expectation, ρ is increased. Otherwise, ρ is reduced. Note that if ρ passes through the setpoint α_r, it is always reset to 1 to attenuate oscillation. Compared to the original AIAD with a fixed factor, A-AIAD that utilizes an adaptive factor to adjust the requesting rate accelerates the convergence towards the setpoint α_r and enables sub-flows to fairly and efficiently share the bandwidth.

6) *Queue-Draining State*

When the network is stable, sub-flows will never move into the QD state. We consider two specific situations that cause a sub-flow to enter the QD state. First, if the consumer perceives that many backlogged packets are queued in the bottleneck $\Delta_r > \eta$, the sub-flows enter the QD state. For instance, a consumer is downloading the Data packets from a router. However, after a certain time, the router no longer caches the requesting Data packets. Thus, the consumer retrieves the following Data packets from the producer. If the bottleneck bandwidth between the producer and the consumer is much lower than that between the router and the producer, many Data packets will be backlogged at the bottleneck. The QD state enables rapidly emptying backlogged packets to prevent bufferbloat. Second, if the measured queuing delay changes dramatically (e.g., lower-layer connectivity are changed [25]), the sub-flow enters the QD state.

During the QD state, consumers drain the queue until they detect that the RTT no longer decreases anymore. This enables consumers to re-measure the baseRTTs for hop-flows. The QD state is composed of two stages.

$$\text{QD:}\begin{cases}\begin{cases}u_r^* = x_r \\ u_r = 2\end{cases} & \text{Stage I} \\ u_r = \zeta u_r^* & \text{Stage II}\end{cases} \quad (21)$$

During the first stage, the sub-flow records its current receiving rate x_r as a backup rate u_r^*. Then it reduces its requesting rate u_r to a minimum value (2 packets per second in our implementation) to drain the backlogged packets at the bottleneck until the number of backlogged packets reaches zero. After the queue is emptied, the QD enters the second stage, which sets the requesting rate to a fraction ζ of the backup rate u_r^* (ζ = 0.8 in our implementation) to reserve bandwidth to new sub-flows. At the same time, the sub-flow is transferred to the BF state to re-fill the bottleneck.

7) RTT-Probing State

Because DPCCP causes consistent queues, a newly added consumer may not measure the accurate baseRTTs. To tackle this problem, a cooperative method is developed. It lets a new sub-flow notify others to drain the queue. The explicit notification is realized by adding a 1-bit "*RTT-probing*" flag on each outgoing Interest packet of the new sub-flow. Once this Interest packet is received by a router through an interface, the interface is labelled "*RTT-Probing*". For all Data packet that travels through this interface, they will be tagged with a 1-bit flag. This allows spreading queuing draining signals to the sub-flows which are using the interface. As a result, these sub-flows will enter QD to empty queues.

The rationale is that the receiving interfaces of "*RTT-probing*" Interest packets are also the sending interfaces of the Data packets of the same sub-flow. To facilitates a newly added consumer to get the correct baseRTTs, all queues of these interfaces should be emptied i.e., the sub-flows who are using these interfaces should cooperatively empty the queue.

VII. Evaluation

A. Simulation Settings

In this section, simulations are conducted to evaluate DPCCP in comparison with conventional and SOTA congestion control solutions including 1) Optimal Multipath Congestion Control with Request Forwarding (OMCC-RF) [6], 2) Multipath-aware ICN Rate-based Congestion Control (MIRCC) [13] and 3) Practical Congestion Control (PCON) [9]. The reason for choosing OMCC-RF and PCON is because they are the well-known window-based approaches with the supports of multipath transmission and in-network caching while the reason for comparing DPCCP with MIRCC because MIRCC is a SOTA rate-based approach that also considers multipath transmission. Some transport protocols have been also proposed for NDN, most of them consider neither congestion control nor adaptive forwarding. This presents in difficulties in evaluating them. The three approaches for comparison are implemented using the NS-3 based NDN simulator – ndnSIM [27]. For PCON, we directly use the published source code [28]. OMCC-RF and MIRCC are implemented according to the original papers [6], [13].

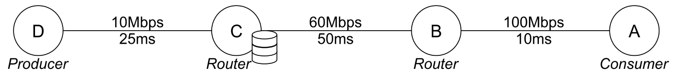


Figure 7 Linear Topology for Single-path (SP) Scenario 1

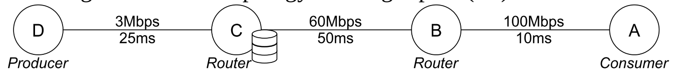


Figure 8 Linear Topology for Single-path (SP) Scenario 2

Each requested content object is split into segments with a payload of 1024Bytes, and each segment is encapsulated in a single Data packet. The target number of backlogged packets α_s controls the fairness amongst flows so they are set to a constant value (20 packets) based on empirical studies. If flows use different α_s, the bottleneck bandwidth cannot be fairly shared amongst flows. A low α_s value can reduce the number of backlogged packets at bottlenecks however, it also increases the chance of underutilizing bottleneck bandwidths, and vice versa. In order to evaluate the performance with in-network caching or without caches, some routers are deployed with simulated caches. Simulated caches enable a router to provide Data packets (to emulate cache hit) without caching then in advance. A parameter κ is used to control the cache hit probability at router.

B. Single-Path (SP) Transmission

1) Scenario 1: Fixed Caching Hit Rates

In this scenario, the performances of the three protocols are compared via a linear topology (Figure 7). The first experiment tests the feasibility of DPCCP: a consumer retrieves a proportion of the requested content from the cache and the remaining part from the producer. Router C is cache enabled. In this topology, Producer D and Consumer A are the nodes located at the two ends of the graph. When caches do not exist in Router C, a single flow travels through bottleneck D–C (10Mbps). The sum of the propagation latencies in the consumer-producer path is $170ms = 2\times\ 25+50+10\ ms$. As Consumer A only maintains a single consumer-producer path, the multipath forwarding strategy is not engaged. Producer D provides the content to Consumer A via path D-C-B-A. Different time-invariant caching hit probabilities κ (0%, 20%, 40%, 60%, 80% and 100%) are set for Router C. Every experiment is run 5 times, where each run lasts 200 seconds.

Comments on the results: Figure 9 presents the averaged downloading rate for each protocol and for each cache-hit probability. DPCCP achieves a near-perfect link utilization for the first 5 cases (with 0% – 80% cache hit probabilities). This is because the latency-based approach always requires queuing packets at a bottleneck (link D-C). If a bottleneck queue is not emptied, its utilization is always 100%. OMCC-RF and MIRCC achieve good performances in the first and the last cases (with 0% and 100% cache hit probabilities) but worse performances in the other cases. For OMCC-RF on the first

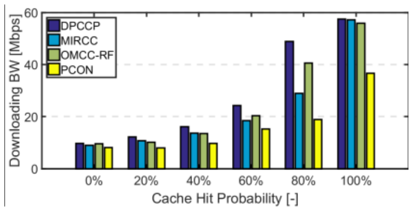


Figure 9 [SP-Scenario 1]
Downloading Rates of four protocols

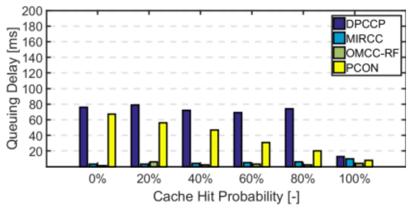


Figure 10 [SP-Scenario 1]
Queuing Delays of four protocols

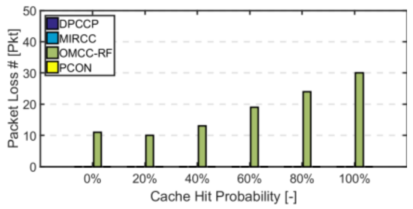


Figure 11 [SP-Scenario 1]
Numbers of Packet Loss of four protocols

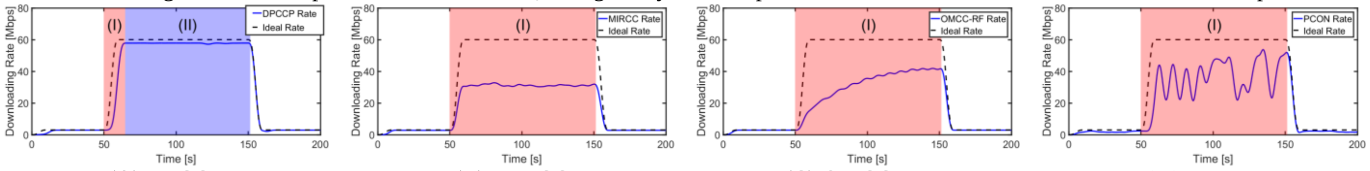

(A) DPCCP (B) MIRCC (C) OMCC-RF (D) PCON

Figure 12 [SP-Scenario 2] Bandwidth convergences of four protocols; Note: The downloading speeds are divided into different areas (I, II) according to the convergence status (I: adaptive status and II: steady status)

and the last cases, the content is provided by a single content source (a producer in the first case and a router in the second case); while in the other cases, the content is concurrently provided by two content sources. The Data packets from them may arrive at the bottleneck (C–B) around the same time and cause unexpected queuing delays (RTT jitter). In OMCC-RF, as capturing a congestion event is sensitive to RTT variations, the jitter causes reducing Consumer A's requesting rate incorrectly. Although MIRCC enables separating traffic control for different paths, however, it cannot distinguish multiple sources (C and D) on the same path. As the result, Consumer A adjusts its requesting rate for bottlenecks D–C and C–B. This results in the oscillation of requesting rates and low bandwidth utilization.

Figure 10 shows the averaged queuing delays for the four protocols. Not surprisingly, the averaged queuing delay of OMCC-RF and MIRCC is nearly 0 as it operates at the point that the queue is nearly empty. As DPCCP requires a small number of backlogged packets for each sub-flow to balance throughputs and fairness, some queuing delays are inevitably introduced. The queuing delay also validates the correctness of A-AIAD. As the target number of backlogged packets α_s (i.e. the number of packets that are expected to be backlogged at the bottleneck) is set to 20 packets (i.e. the queue length is about 20Kbyte as the payload of each Data packet is 1Kbyte) and the bandwidth of the only bottleneck (D–C) is 10Mbps in the first five cases, the expected queuing delay is 20Kbyte / 10Mbps * 8 ≈ 16ms. For the last case, the only bottleneck is link C–B and its bandwidth is 60Mbps. As a result, the queuing delay is further reduced to 20Kbyte / 60Mbps * 8 ≈ 2.7ms. The theoretical results match the experimental results as shown in Figure 10. Like DPCCP, PCON introduces queuing delays because its congestion control scheme requires measuring the stalling time of queuing packets.

Figure 11 plots the total number of dropped packets for the four protocols. All the algorithms achieve near-zero packet drop. This is because none of these protocols are loss-based, i.e., they do not rely on packet losses as congestion signals. Nevertheless, the initial slow-start procedure (to probe the bandwidth) of OMCC-RF is aggressive and unbounded. This causes a small number of packet losses.

2) *Scenario 2: Time-variant Fixed Caching Hit Rates*

In this scenario, the topology is still linear, like Scenario 1 but using different bottleneck settings (Figure 8). It aims to test the reaction time to time-varying caches.

Particularly, Router C is cache enabled. The experiment aims to test if the protocol can quickly track the bottleneck bandwidth changes caused by the sudden cache appearances. This is very common in NDN. For instance, a consumer firstly downloads content from a certain producer as no requested Data packets are cached in the intermediate router. Later, nearly all the Interest packets can find matched Data packets in an intermediate router. This means the consumer can immediately download Data packets from this router. The cache hit usually provides the consumer with more bandwidth than that can be provided from the producer. To simulate this scenario, the cache hitting probability changes over time. It is set to 0% from 0s to 50s, 95% from 50s to 150s and 0% from 150s to 200s.

Comments on the results: Figure 12 (A), (B), (C) and (D) show the downloading rate variations of the four protocols. DPCCP outperforms the others significantly. Although PCON and OMCC-RF support slow-start to probe the available bandwidth, the unbounded bandwidth probing is only executed when a flow joins the network. This indicates that, when additional bandwidth appears, OMCC-RF can only use Additive Increase (AI) to probe the unused bandwidth thus it takes longer time to fully fill the new bottleneck (C-B). As MIRCC does not separate the traffic from different sources in the same path, it results in continuously triggering congestion control for different hop-flows thus causes bandwidth oscillations and low utilization. OMCC-RF shows a worse performance. As mentioned in the previous scenario, the two hop-flows in Figure 8 introduce some RTT jitter that causes the consumer to mistakenly reduce the requesting rate. As a result, OMCC-RF barely converges to the optimal bandwidth utilization. In DPCCP, when a consumer detects that its sub-flow is not fully utilizing the bandwidth of the bottleneck, it applies Z-MI to probe and occupy the available bandwidth. As the increasing speed of the requesting rate in DPCCP is exponential with time, the time to arrive at the optimal

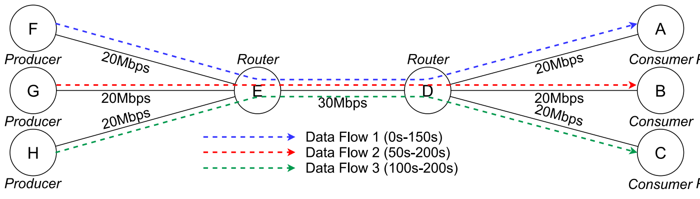


Figure 13 Dumbbell Topology for Single-path (SP) Scenario 3

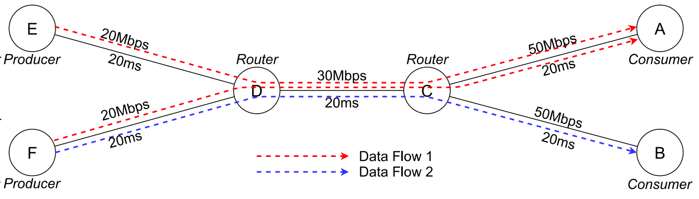
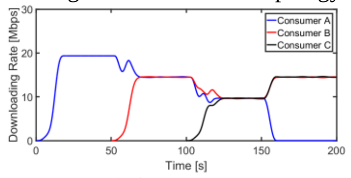
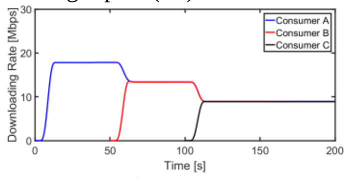


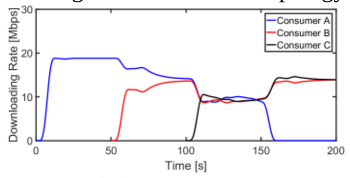
Figure 14 Dumbbell Topology for Multipath (MP) Scenario 1



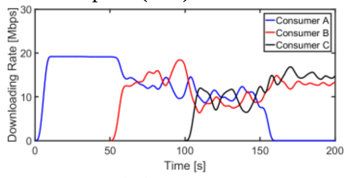
(A) DPCCP



(B) MIRCC

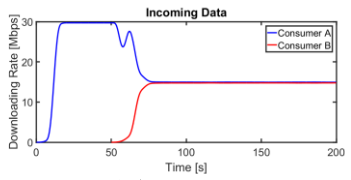


(C) OMCC-RF

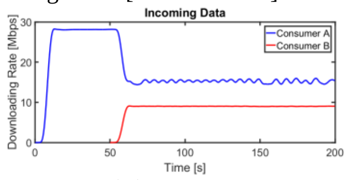


(D) PCON

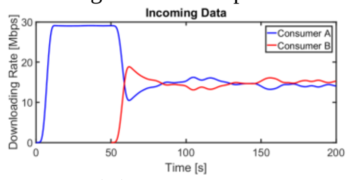
Figure 15 [SP-Scenario 3] Fairness convergences of four protocols



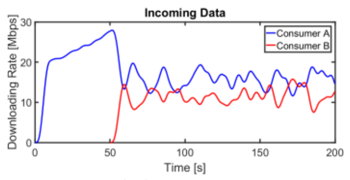
(A) DPCCP



(B) MIRCC



(C) OMCC-RF



(D) PCON

Figure 16 [MP-Scenario 1] Multi-flow fairness of four protocols

utilization is much shortened compared with those using AI.

3) *Scenario 3: Multi-user Fairness Convergence*

In this scenario, a dumbbell topology as shown in Figure 13is used to verify the fairness convergence among different flows. In this topology, the transmission latency of each link is set to 20ms. Producer F serves content to Consumer A (starts from 0s and ends in 150s, Data Flow 1). Producer G serves content to Consumer B (starts from 50s and ends in 200s, Data Flow 2). Producer H serves content to Consumer C (starts from 100s and ends in 200s, Data Flow 3). All the three flows travel through link E-D. In the case that only Consumer A is downloading content, the bottleneck is at F-E (20Mbps). In the case that more than one flow is downloading content, the bottleneck becomes E-D (30Mbps).

Comments on the results: Ideally, the congestion control scheme should achieve 20Mbps downloading rate when only a single flow is activated; 15Mbps downloading rate when two flows are activated; and 10Mbps downloading rate when three flows are activated. Figure 15 shows that all four protocols can fairly and efficiently share the bottleneck bandwidth between the three flows. However, convergence speeds are different. As a result of employing A-AIAD in DPCCP, the number of backlogged packets of each flow converges to the setpoint quickly thus facilitate bandwidth convergences. For MIRCC, the router explicitly works out then feeds back the ideal bandwidth for each flow. The explicit calculation and congestion notification endow MIRCC a quick convergence toward fair bandwidth share. For the same reason in the previous scenarios, as OMCC-RF use AI to increase the congestion window, each flow takes more time to approach the optimal point. The situation is different for PCON, we could see that the downloading bandwidths of the three flows continuously oscillate when they are competing for the bottleneck. This is because PCON uses a BIC-like control law.

C. *Multipath (MP) Transmission*

1) *Scenario 1: Inter-flow Fairness Convergence*

In this scenario, a dumbbell topology (Figure 14) is used to verify the multi-flow fairness among different flows. Producer E and Producer F provide content to Consumer A concurrently (starts at 0s, Data Flow 1). The downloading task starts at 0s. This denotes the first flow contains two sub-flows E-D-C-A and F-D-C-A. Producer F provides the content to Consumer B (starts at 50s, Data Flow 2). This denotes the second flow. Considering multi-flow fairness, Consumer A and Consumer B should receive the same downloading rate as the two flows are travelling through the same bottleneck (D-C).

Comments on the results: As shown in Figure 16, DPCCP enables fair sharing of the bandwidth (with a split ratio nearly 1:1) between the two flows. This is because DPCCP follows the Congestion Equality Principle, which allows the two flows to backlog the same numbers of Data packets in the bottleneck D-C, results in bottleneck bandwidths being equally shared by two flows thus guarantees multi-flow fairness. OMCC-RF achieves near-perfect multi-flow fairness. However, the convergence of OMCC-RF is slower than DPCCP, and its control law results in noticeable bandwidth jitters. As for PCON, Consumer A first uses its slow start to fully fill one path, say E-D-C-A. Then Consumer A will receive congestion signal from path E-D-C-A, this results in Router D re-directs requests to D-F thus enables Consumer A to further utilize the bandwidth of path F-D-C-A. As slow-start is stopped by a congestion signal, the bandwidth probing for path F-D-C-A is based on AI. This is reflected as a linear increase (10s to 50s) after the exponential increase (0s to 10s). Moreover, the fairness of PCON is worse than it is of DPCCP. As we can see from Figure 16 (B), MIRCC cannot guarantee multi-flow fairness. According to our observations, MIRCC separates the

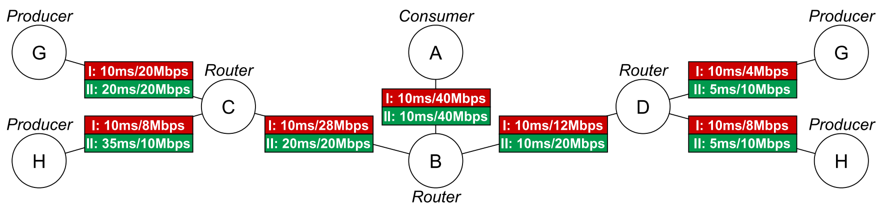


Figure 17 Tree-structured Topology for Multipath Scenario 2

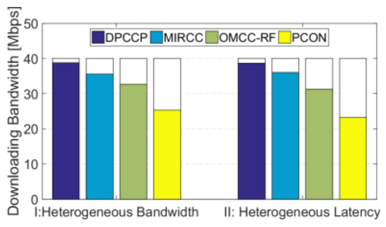


Figure 18 [MP-Scenario 2] Downloading rate of four protocols

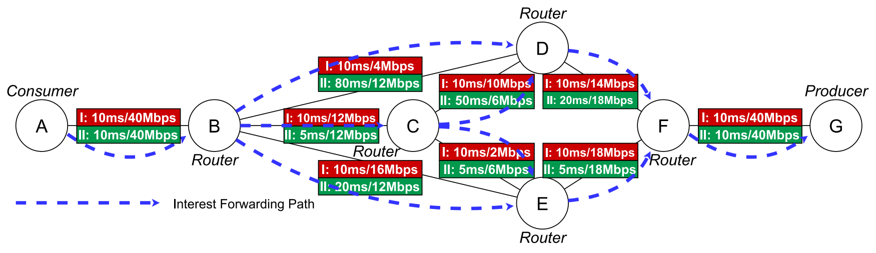


Figure 19 Diamond-structured Topology for Multipath Scenario 3

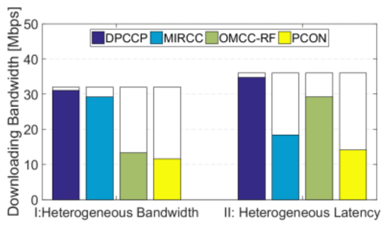


Figure 20 [MP-Scenario 3] Downloading rate of four protocols

traffic control for each path. One sub-flow of Consumer A (via F-D-C-A) shares the bandwidth with the flow of Consumer B (via F-D-C-B), which results in Consumer B only receives 50% bandwidth (around 10Mbps) and another sub-flow of Consumer A (via E-D-C-A) soaks the remaining bandwidth. This causes the unfair bandwidth sharing.

2) *Scenario 2: Heterogeneous Bandwidths / Latencies (Tree-structured Topology)*

The topology is a tree structure as shown in Figure 17. In this experiment, Producer E, F, G and H serves content to Consumer A concurrently. The purpose of this scenario is to evaluate the multipath transmission performances of the four protocols using a simple tree structure topology.

Two types of link settings are considered. In Case I, the latencies of different transmission paths are identical, but the bandwidths are different. This group of settings validate if the protocols can entirely utilize the heterogeneous bandwidths of all paths. In Case II, the bandwidths of different transmission paths are the same, but the latencies are set to different values. This setting verifies if the protocols have a bias on latency.

Comments on the results: Figure 18 reports the averaged downloading rate of Consumer A. The white box denotes the theoretical bandwidth. As Consumer A is connected to the content via a bottleneck link B-A, where all traffic is aggregated in this link, the effective downloading rate is equal to the throughput of link B-A. In Case I, the link utilization of DPCCP is approximately 96.91% and that of MIRCC is 90.01%; that of OMCC-RF is 81.56%; that of PCON is 63.30% respectively. All solutions can shift congestion from the congested paths to the congestion-free paths to further enhance link utilization. For OMCC-RF, as the congestion metric, Pending Interest cannot accurately reflect congestion. Its forwarding overloads the paths with lower bandwidths earlier. For MIRCC, its downloading rate is slightly lower than that in DPCCP. This is because of the parameter setting (η in the original paper [13]) of link utilization in MIRCC is set to η=0.95 to attenuate packet queuing and congestion overshoot which results in bottlenecks remaining underutilized. For PCON, the downloading bandwidth is quite low due to oscillation of its adaptive forwarding. Although ECN triggers its adaptive strategy to re-direct traffic to from congested links to others with lower utilization, these congestion signals also trigger consumers to lower down requesting rate thus causes slow convergences. To reduce transmission latency in addition to improving bandwidth, PCON periodically shifts some traffic from the longer paths to shorter paths. In certain cases, the longer paths may hold more bandwidth than the shorter paths. This conflict design results in the oscillation of adaptive forwarding and the downloading bandwidth. As DPCCP enable controlling congestion for each path individually, which enables them to manage the traffic for each path separately, the bandwidth utilization is near-perfect.

In Case II, the link utilization of DPCCP is approximately 96.55% and that of MIRCC is about 90.03%; that of OMCC-RF is about 78.33%; that of PCON is about 58.13% respectively. As the solution OMCC-RF balances the load based on PI, it suffers from latency bias [8], which means it will always overuse the low latency paths. Specifically, the path A-B-D-E and the path A-B-D-F with the smallest latency (25ms) are always fully filled first. As these two paths reach congestion earlier than the others, the congestion control scheme RAAQM (in OMCC-RF) decreases the congestion window when the other paths are underutilized, which causes in low bandwidth utilization. For MIRCC, the link utilization is slightly lower because of the parameter setting. For PCON, the low link utilization is because of the convergence issue of adaptive forwarding. In DPCCP, because a consumer controls sub-flows independently, the congestion of a certain path will not cause reducing the requesting rates for other sub-flows. Specifically, the conservative A-AIAD control law will always keep some backlogged packets at bottleneck but never cause continuous packet loss. As a result, DPCCP achieves the highest bandwidth utilization.

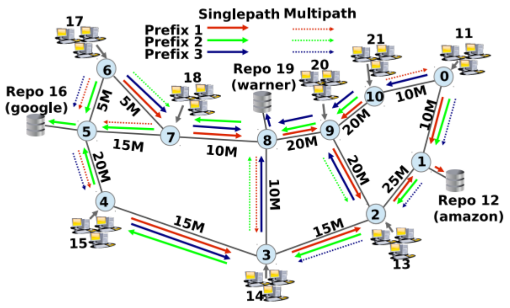


Figure 21 Abilene Topology [16]

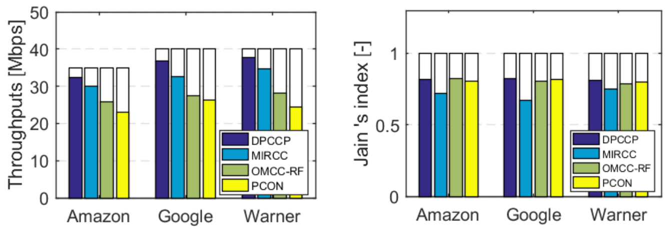
3) Scenario 3: Heterogeneous Bandwidths / Latencies (Diamond-structured Topology)

This experiment verifies the multipath transmission performances of the four protocols using a complex diamond topology [29] (Figure 19). The diamond topology contains looped links, where the traffic is aggregated and split at certain nodes. This consequently causes the congestion levels of different sub-flows to be coupled. The topologies with looped links are typical scenarios [29] to verify the effectiveness of multipath congestion control protocols. In this experiment, Consumer A is connected to the content via the joint consumer-producer paths calculated by the shortest path (A-B-C-D-F-G, A-B-C-E-F-G, A-B-D-F-G and A-B-E-F-G).

Two different link settings are considered: 1) the same latencies and different bandwidths (Case I) and 2) with the same bandwidths and different latencies (Case II). The bottleneck links (B-C, C-E, C-D, B-E, B-D, D-F and E-F) need to be fully utilized to achieve optimal throughputs.

Comments on the results: Figure 20 reports the averaged throughput in both cases in comparison with the theoretically optimal throughput. In Case I, the link utilization of DPCCP is approximately 97.12%, and that of MIRCC is 91.25%; that of OMCC-RF is 41.51%; that of PCON is 36.25% respectively. For OMCC-RF, as the bandwidth differences are relatively large, congestion always happens at the bottlenecks with much lower bandwidths (C-E and B-D) first. Therefore, Consumer A reduces the congestion window immediately when it detects RTT increases from the congested bottlenecks (C-E and B-D). Even through path B-E-F-G does not suffer from congestion, the utilization of path B-E-F-G remains at a very low level. The performance of MIRCC follows the same pattern as previous for the same reason. For PCON, because congestion signals will be aggregated from and disseminated to shared paths, this results in more difficulties for routers to perceive the real congestion situation of downstream links. The difficulty of balance the congestion on different paths results in low downloading bandwidth. Because DPCCP inherits the congestion balancing from MPTCP, which enables shifting the traffic from more congested paths to less congested paths.

In Case II, the link utilization of DPCCP is 96.58% and that of MIRCC is 51.11%; that of OMCC-RF is 81.19%; that of PCON is 39.08% respectively. For OMCC-RF, as the latency of path A-B-C-E-F-G is relatively smaller than others (the number of PIs at Router B and Router C is less), it is congested earlier, and it triggers the congestion control at the consumers. In general, OMCC-RF achieves a better utilization



(a) Producer Throughputs (b) Jain's Fairness

Figure 22 [MP-Scenario 4]

in Case II than that in Case I. This can be explained because the loop links average the RTT estimations for different forwarding interfaces. As the bandwidths and latency of these paths are similar, the numbers of PIs reflect the congestion levels on different paths therefore MOCC-RF enables reasonable traffic allocations. For MIRCC, we could observe that its dual-class sub-flow scheme fails to estimate the requesting rate for all paths. We consider that the heterogeneous latencies cause the rate estimation algorithm not working well for the second-class sub-flows. The result of PCON is like Case I for the same reason. For DPCCP, the traffic shifting behavior guarantees the bandwidth utilization of all paths therefore it achieves near-perfect utilization.

4) Scenario 4: Abilene Network

In this scenario, we consider the performance in the Abilene topology in Figure 21 used in [13]. This scenario features a more realistic set of nodes, dynamic consumer arrival time and varying flow size. Three repositories, at nodes 12, 16, 19, each store content under a given name prefix (/Amazon, /Google, /Warner, respectively). Four consumers per prefix are randomly assigned to nodes 11, 13, 14, 15, 17, 18, 20, 21.

Comments on the results: Figure 22 (a) demonstrates the throughputs of three repositories. It shows that DPCCP beats the other three protocols in terms of throughputs to all consumers. This further verifies that our delay-based traffic allocation does a better job than the OMCC-RC, MIRCC and PCON. In Figure 22 (b), it presents the Jain's fairness indices for consumers requesting different content packets. Because some consumers are farther away from repositories than others, it is not like to have the perfect fairness. DPCCP outperforms others in terms of fairness because the downloading bandwidths of consumers explicitly converge to the proportional fairness brought by the log utility function.

VIII. Conclusion

The multi-source and multipath transmission nature of NDN brings unique challenges to the transport protocol design. This paper formulates the multi-source and multipath transmission problem in NDN using a Network Utility Maximization (NUM) model that enables deriving a family of congestion control protocols. Delay-based Path-specified Congestion Control Protocol (DPCCP) is an implementation under the context of path-specified transport. In contrast to loss-based congestion control, the delay-based approach lets consumers estimate congestion more accurately. Additionally, a set of congestion control laws is developed. As the result, DPCCP brings immediate benefits: 1) it enjoys high bandwidths provided by caches; 2) it allows consumers react to congestion accurately; 3) the proposed control law enables fast convergence of the bandwidth utilization and fairness. The complete protocol is implemented and evaluated using ndnSIM. In future work, we plan to model the behavior of TCP-BBR [30] and extend it to further support multipath transmission and in-network cache in NDN.

REFERENCES

[1] L. Zhang et al., 'Named Data Networking', ACM SIGCOMM Comput. Commun. Rev., vol. 44, no. 3, pp. 66–73, Jul. 2014, doi: 10.1145/2656877.2656887.

[2] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, 'Design, implementation and evaluation of congestion control for multipath TCP', in Proceedings of the 8th USENIX conference on Networked systems design and implementation, 2011, pp. 99–112. [Online]. Available: http://dl.acm.org/citation.cfm?id=1972468

[3] J. R. Iyengar, P. D. Amer, and R. Stewart, 'Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths', IEEEACM Trans. Netw., vol. 14, no. 5, pp. 951–964, 2006.

[4] Q. Chen, R. Xie, F. R. Yu, J. Liu, T. Huang, and Y. Liu, 'Transport control strategies in named data networking: A survey', IEEE Commun. Surv. Tutor., vol. 18, no. 3, pp. 2052–2083, 2016.

[5] Y. Ye et al., 'PTP: Path-specified transport protocol for concurrent multipath transmission in named data networks', Comput. Netw., vol. 144, pp. 280–296, Oct. 2018, doi: 10.1016/j.comnet.2018.08.002.

[6] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, and S. Wang, 'Optimal multipath congestion control and request forwarding in information-centric networks', in Network Protocols (ICNP), 2013 21st IEEE International Conference on, 2013, pp. 1–10. Accessed: Apr. 07, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6733576

[7] A. Hall, S. Hippler, and M. Skutella, 'Multicommodity flows over time: Efficient algorithms and complexity', Theor. Comput. Sci., vol. 379, no. 3, pp. 387–404, 2007.

[8] D. Nguyen, M. Fukushima, K. Sugiyama, and A. Tagami, 'Efficient multipath forwarding and congestion control without route-labeling in CCN', London, UK, 2015, pp. 1533–1538. Accessed: Apr. 07, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=7247397

[9] K. Schneider, C. Yi, B. Zhang, and L. Zhang, 'A practical congestion control scheme for named data networking', in Proceedings of the 3rd ACM Conference on Information-Centric Networking, 2016, pp. 21–30.

[10] I. Moiseenko and D. Oran, 'Path switching in content centric and named data networks', in Proceedings of the 4th ACM Conference on Information-Centric Networking, 2017, pp. 66–76.

[11] G. Carofiglio, M. Gallo, and L. Muscariello, 'ICP: Design and evaluation of an interest control protocol for content-centric networking', in Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on, 2012, pp. 304–309. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6193510/

[12] F. Zhang, Y. Zhang, A. Reznik, H. Liu, C. Qian, and C. Xu, 'A transport protocol for content-centric networking with explicit congestion control', in 2014 23rd International Conference on Computer Communication and Networks (ICCCN), 2014, pp. 1–8. Accessed: Sep. 27, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6911765

[13] M. Mahdian, S. Arianfar, J. Gibson, and D. Oran, 'MIRCC: Multipath-aware ICN rate-based congestion control', in Proceedings of the 3rd ACM Conference on Information-Centric Networking, 2016, pp. 1–10.

[14] N. Rozhnova and S. Fdida, 'An effective hop-by-hop interest shaping mechanism for ccn communications', in Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on, 2012, pp. 322–327. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6193514/

[15] N. Rozhnova and S. Fdida, 'An extended Hop-by-hop Interest shaping mechanism for Content-Centric Networking', in Global Communications Conference (GLOBECOM), 2014 IEEE, 2014, pp. 1–7. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7389766/

[16] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, 'Adaptive forwarding in named data networking', ACM SIGCOMM Comput. Commun. Rev., vol. 42, no. 3, pp. 62–67, 2012, doi: 10.1145/2317307.2317319.

[17] Y. Wang, N. Rozhnova, A. Narayanan, D. Oran, and I. Rhee, 'An improved hop-by-hop interest shaper for congestion control in named data networking', in ACM SIGCOMM Computer Communication Review, 2013, vol. 43, pp. 55–60. Accessed: Sep. 27, 2016. [Online]. Available: http://dl.acm.org/citation.cfm?id=2491233

[18] H. Qian, R. Ravindran, G. Wang, and D. Medhi, 'Probability-based adaptive forwarding strategy in named data networking', Ghent, Belgium, 2013, pp. 1094–1101. Accessed: Apr. 07, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6573145

[19] A. Udugama, X. Zhang, K. Kuladinithi, and C. Goerg, 'An On-demand Multi-Path Interest Forwarding strategy for content retrievals in CCN', in Network Operations and Management Symposium (NOMS), 2014 IEEE, 2014, pp. 1–6. Accessed: Apr. 07, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6838389

[20] D. Posch, B. Rainer, and H. Hellwagner, 'SAF: Stochastic Adaptive Forwarding in Named Data Networking', *ArXiv Prepr. ArXiv150505259*, 2015, Accessed: Apr. 07, 2016. [Online]. Available: http://arxiv.org/abs/1505.05259

[21] J. Gettys and K. Nichols, 'Bufferbloat: Dark buffers in the internet', *Queue*, vol. 9, no. 11, p. 40, 2011.

[22] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, 'Layering as optimization decomposition: A mathematical theory of network architectures', *Proc. IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[23] Y. Cao, M. Xu, and X. Fu, 'Delay-based congestion control for multipath TCP', in *Network Protocols (ICNP), 2012 20th IEEE International Conference on*, 2012, pp. 1–10.

[24] S. H. Low, 'A duality model of TCP and queue management algorithms', *IEEEACM Trans. Netw. ToN*, vol. 11, no. 4, pp. 525–536, 2003.

[25] D. J. Leith, R. N. Shorten, G. McCullagh, L. Dunn, and F. Baker, 'Making available base-RTT for use in congestion control applications', *IEEE Commun. Lett.*, vol. 12, no. 6, 2008.

[26] A. Afanasyev, N. Tilley, P. Reiher, and L. Kleinrock, 'Host-to-Host Congestion Control for TCP', *IEEE Commun. Surv. Tutor.*, vol. 12, no. 3, pp. 304–342, Third 2010, doi: 10.1109/SURV.2010.042710.00114.

[27] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, 'ndnSIM 2: An updated NDN simulator for NS-3', Technical Report NDN-0028, Revision 2, NDN, 2016. [Online]. Available: https://named-data.net/wp-content/uploads/2016/11/ndn-0028-2-ndnsim-v2.pdf

[28] *PCON Source Code*. https://github.com/Shydandan/ndnSIM-1: Klaus Schneider. [Online]. Available: https://github.com/Shydandan/ndnSIM-1

[29] J. Cao *et al.*, 'Improving the freshness of NDN forwarding states', in *IFIP Networking Conference (IFIP Networking) and Workshops, 2016*, 2016, pp. 189–197. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7497213/

[30] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, 'BBR: Congestion-based congestion control', *Queue*, vol. 14, no. 5, p. 50, 2016.