Revisiting TCP Congestion Control Throughput Models & Fairness Properties At Scale

Adithya Abraham Philip, Ranysha Ware,
Rukshani Athapathu, Justine Sherry, Vyas Sekar
Carnegie Mellon University
United States of America

Abstract

Much of our understanding of congestion control algorithm (CCA) throughput and fairness is derived from models and measurements that (implicitly) assume congestion occurs in the last mile. That is, these studies evaluated CCAs in "small scale" edge settings at the scale of tens of flows and up to a few hundred Mbps bandwidths. However, recent measurements show that congestion can also occur at the core of the Internet on inter-provider links, where thousands of flows share high bandwidth links. Hence, a natural question is: Does our understanding of CCA throughput and fairness continue to hold at the scale found in the core of the Internet, with 1000s of flows and Gbps bandwidths?

Our preliminary experimental study finds that some expectations derived in the edge setting do not hold at scale. For example, using loss rate as a parameter to the Mathis model to estimate TCP NewReno throughput works well in edge settings, but does not provide accurate throughput estimates when thousands of flows compete at high bandwidths. In addition, BBR – which achieves good fairness at the edge when competing solely with other BBR flows – can become very unfair to other BBR flows at the scale of the core of the Internet. In this paper, we discuss these results and others, as well as key implications for future CCA analysis and evaluation.

CCS Concepts

• **Networks** → **Protocol testing and verification**; **Transport protocols**; **Network measurement**.

Keywords

congestion control, computer networks, fairness, throughput, tcp, bbr, reno, cubic

ACM Reference Format:
Adithya Abraham Philip, Ranysha Ware,, Rukshani Athapathu, Justine Sherry, Vyas Sekar. 2021. Revisiting TCP Congestion Control Throughput Models & Fairness Properties At Scale. In *ACM Internet Measurement Conference (IMC '21), November 2–4, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3487552.3487834

IMC '21, November 2–4, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-9129-0/21/11.
https://doi.org/10.1145/3487552.3487834

1 Introduction

Congestion control algorithms (CCAs) are a fundamental building block of the modern Internet, and the networking community has been analyzing and designing CCAs for over three decades [12, 18, 26, 37, 39, 48]. Of specific interest to us is the wide-area setting. In this setting, *throughput* and *fairness* are important CCA properties, as they determine the effectiveness with which data can be transferred across the Internet, and the ability for multiple TCP flows to co-exist. To this end, many past efforts have used systematic models and experimental studies to understand these properties. For instance, the model by Mathis et al. [37] and Padhye et al. [41] predict the throughput of a NewReno flow as a function of packet loss and round-trip time (RTT), and the BBR model by Ware et al. [48] predicts the throughput of BBR when competing with other CCAs. Application developers can use such results to decide which CCA best suits the network conditions they experience or to debug performance issues.

One implicit assumption in many of these results is that in the wide-area setting, congestion almost always occurs close to the *edge* or last mile of the network. To this end, many findings have been derived in contexts that emulate congestion occurring at the edge (e.g., residential link settings). Specifically, they consider a few or tens of flows competing for a shared bottleneck link with a capacity of a few tens or hundreds of Mbps [26, 37, 45, 52].

Many measurements suggest, however, that this assumption of congestion-at-the-edge may not always hold. Indeed, measurements both new [21] and old [11] show that congestion does occur at inter-domain links. These settings are characterized by higher flow counts and a larger network pipe [4, 13]. Indeed, past work on router buffer sizing has shown that CCA properties can change in such a setting [13].

This raises a natural question: do the known findings and models about TCP throughput [37] and fairness [20, 26, 28, 39] derived from edge-link settings, hold at the scale found in the core of the Internet? To this end, we revisit aspects of TCP throughput and fairness at high bandwidths of 10Gbps and with thousands of concurrent flows. Specifically, we ask:

- *Throughput Model:* The commonly accepted Mathis analytical model [37] for TCP throughout prediction says that the throughput depends only on the RTT and loss. Does this model accurately predict TCP NewReno's throughput at scale?
- *Intra-CCA fairness:* NewReno, Cubic, and BBR have shown to be fair at lower flow counts when all the flows have the same CCA and RTT [20, 26, 28, 39]. Does this continue to hold at scale?

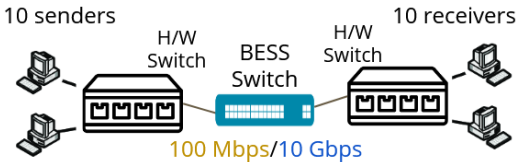


Figure 1: Testbed topology for emulated settings.

- *Inter-CCA fairness:* Does the Inter-CCA unfairness observed in the home link setting, where Cubic takes up to 80% of link bandwidth when competing with NewReno [26], or BBR starves competing NewReno and Cubic flows [28, 47, 48, 52], continue to hold at scale?

We use a simple but effective measurement set up to answer these questions, and re-evaluate past beliefs on a 10 Gbps bottleneck link with thousands of competing flows, representative of the core of the Internet [4, 13]. We see if we obtain the same results as past research derived from measurements and models verified on the edge. Indeed, we find that some edge-derived expectations do not hold at scale:

- The Mathis model [37] for NewReno throughput relies on a parameter p which is commonly interpreted as the network loss rate [44, 46]. While using loss rate for p works well in edge settings, we find that using packet loss rate for p at scale results in more than 45% error in estimating throughput. Instead, operators should use direct measurements of the congestion window halving rate for throughput estimates at this scale.
- BBR surprisingly becomes unfair at scale even when competing with solely other BBR flows at the same RTT, with a Jain's Fairness Index (JFI) as low as 0.4. This is in contrast to the fairness observed by past research in the edge setting or at low flow counts, where the JFI is typically 0.99 [28, 47, 52].

On the other hand, our findings validate at scale prior claims about CCAs which were derived from analyses evaluating the edge:

- A single BBR flow takes up 40% of link capacity even when competing with thousands of NewReno or Cubic flows at scale. Prior work had only measured this phenomenon at up to 16 competing flows [42, 47, 48], and our measurements illustrate that this phenomenon persists even at scale. This confirms the prediction from the model by Ware et al. [48].
- The intra-CCA fairness of NewReno and Cubic and the inter-CCA unfairness of Cubic competing with NewReno, continue to hold at scale. The extreme inter-CCA unfairness when multiple BBR flows compete with multiple Cubic or NewReno flows also persists at scale.

Our results, though preliminary, have key implications for CCA design and analysis and suggest the need for future analysis. First, applying the Mathis model over the Internet precisely will require end-host TCP instrumentation to obtain the congestion window values as one cannot rely on just measured packet loss. Second, BBR's unexpectedly high unfairness when competing with just other BBR flows at scale highlights the importance of explicitly including evaluations with thousands of flows and Gbps bandwidths as part of future CCA design and evaluation roadmaps.

2 Related Work & Motivation

We begin this section with an introduction to CCAs, followed by past work on throughput models and fairness. Lastly, we discuss congestion at the core of the Internet and CCA results in data centers and high bandwidth settings.

CCA Background: Today there are many CCAs on the Internet, including NewReno [25], Cubic [44], Vegas [17], Copa [14], and BBRv1 [18] (hereafter referred to as 'BBR') as well as BBRv2 [2] (which remains a work in progress). Developers and network administrators evaluate CCAs for many important properties, including (1) *throughput*, or the rate at which a connection transfers data [26, 37, 41] and (2) *fairness*, or how equitably multiple connections share throughput when competing over a bottleneck link [20, 39, 44].

Throughput Models: To help us understand how well a CCA performs in a given network setting, analytical models predict the throughput of a connection as a function of key network properties (e.g., loss, delay). For example, the NewReno models by Mathis et al., [37] and Padhye et al., [41] predict the throughput of a NewReno flow given the network RTT and loss rate. Researchers have derived other models with similar goals for Cubic [26] and BBR [48]. In this paper, we revisit the simpler model for NewReno throughput by Mathis et al. [37] and investigate the fairness implications of the BBR model by Ware et al. [48].

Fairness: Fairness determines how deployable a CCA is. Say, for example, that Cubic flows completely starve NewReno flows when competing for bandwidth over a shared link. This would result in Netflix streams (which use NewReno) seeing degraded performance every time they share a bottleneck link with large downloads using Cubic. Fairness is typically evaluated in two settings: (1) Intra-CCA fairness, where all competing flows have the same CCA; and (2) Inter-CCA fairness, where the competing flows have different CCAs.

Past research has found that in the wide-area context, NewReno, Cubic, and BBR all exhibit high intra-CCA fairness when all flows have the same RTT, with most flows getting the same throughput [18, 20, 26, 28]. There is also work on intra-CCA fairness when flows have different RTTs [26, 32, 35, 39]. In this paper, as a simpler starting point, we specifically evaluate the same-RTT setting.

In the inter-CCA setting, prior work shows that Cubic flows compete unfairly with NewReno, with Cubic obtaining up to 80% of total bandwidth [26]. Past research also finds that BBR competes unfairly with both Cubic and NewReno, with a single BBR flow taking up 40% of link capacity irrespective of the number of competing Cubic and NewReno flows [47, 48]. Multiple BBR flows competing with an equal number of Cubic flows also result in the BBR flows obtaining 90% to 95% of link bandwidth with large buffers [45] and up to 99% with small buffers [28]. We re-evaluate all of these properties at scale.

Congestion in the core: Many prior CCA efforts implicitly assume that Internet congestion occurs mostly at the network edge, evaluating only tens of flows at the scale of a hundred Mbps [18, 26, 37]. However, both older and more recent work [11, 21] show that there is persistent congestion on inter-provider links in the Internet core. This is significant in the light of analysis that the properties of CCAs can change as network parameters scale; e.g., the work

p	*EdgeScale*	*CoreScale* Flow Count		
		1000	3000	5000
Packet Loss	**1.78**	**3.95**	**3.64**	**3.24**
CWND Halving	1.47	1.36	1.36	1.34

Table 1: Deriving the Mathis constant C using the packet loss rate results in different flow count-dependent constants in *CoreScale* vs *EdgeScale*, while using the CWND halving rate results in closer and more consistent values across settings and flow counts.

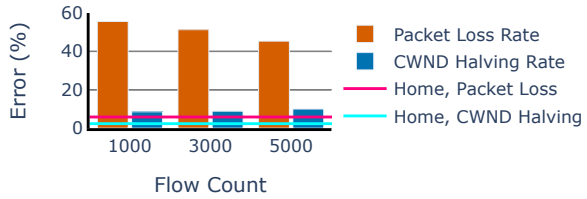


Figure 2: The median prediction error for the Mathis model in *CoreScale* is $\leq$ 10% using CWND halving rate, but 45% to 55% with packet loss rate. In *EdgeScale* both packet loss rate and CWND halving rate result in <10% error.

of Appenzeller et al. [13] finds that when thousands, rather than tens, of NewReno flows compete over a "core" bottleneck link, they desynchronize, allowing the use of smaller router buffers compared to recommendations in the edge setting.

CCAs in data centers and high-bandwidth settings: While past research has investigated CCA properties in the data center setting [12, 19, 30, 33, 43], we are interested in the wide-area setting, which sees higher RTTs and has routers with larger buffers [13, 38]. There is also work on CCA fairness at Gbps bandwidths [10, 28, 32, 39], but they typically evaluate tens to a few hundred flows, not thousands. To the best of our knowledge, the Mathis model and fairness properties of CCAs when thousands of flows compete on Gbps links have not been rigorously studied in the wide-area setting.

3 Problem Scope and Methodology

In this section, we define the scope and methodology of our analysis, its relevance, and its limitations.

3.1 Problem Scope

Before we begin, we concretely define the two settings of interest for our study:

- *EdgeScale*: This represents the edge-link setting with a bottleneck bandwidth of 100 Mbps with 2 to 50 competing flows and a 3MB buffer.
- *CoreScale*: The "at scale" setting with a bottleneck bandwidth of 10 Gbps [4], 1000 to 5000 competing flows, and a 375MB buffer.

In both cases, a drop-tail queue is used at the bottleneck link, and the buffer size is approximately 1 BDP (bandwidth-delay product) based on the bandwidth of the bottleneck link and assuming a maximum RTT of 200ms. We choose this size based on the rule of thumb used to size router buffers [13]. It is the smallest buffer that would allow a single NewReno flow to saturate the link. While past work has shown that smaller buffers equal to a fraction of the BDP are sufficient to ensure upto 99% link utilization at scale [13, 16], recent work [38] has found that in practice ISPs still use extremely large buffers.

CCAs Analyzed: We focus our evaluation on three popular CCAs: NewReno, Cubic, and BBR. These CCAs are chosen based on both the depth of their research literature and their widespread usage on the Internet today [5, 40, 40]:

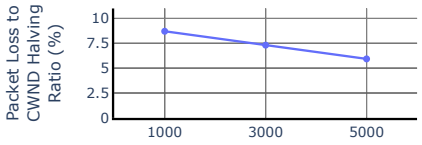
1. NewReno is a classic example of a loss-based CCA. It is widely used today, most notably by Netflix [40], which is believed to make up 13% of all traffic on the Internet [5].
2. Cubic is another loss-based CCA [26]. It is currently the default CCA on Linux and Windows Server and is the standard baseline almost every new CCA is compared with [18, 35, 39, 48].
3. BBR is a comparatively new CCA proposed by Google [18]. However, it is used by YouTube [40], which accounts for 6% of all Internet traffic [5]. While a new version 'BBRv2' [2] exists, it is currently a work in progress. We, therefore, focus on the well-studied BBRv1 [28, 45, 47, 48].

3.2 Setup and Methodology

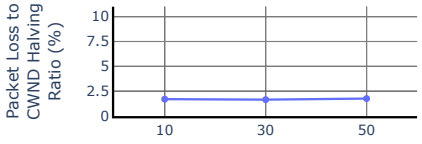
Studying TCP properties at scale is challenging; e.g., traditional packet-level simulators such as ns-3 [27] take several days for a simple Gbps-scale experiment [31], and past work on data-center networking that uses such simulators at scale typically run experiments modeling just a few seconds [34]. Approximations (e.g., flow or fluid model simulations [1]) may not accurately capture fine-grained dynamics. To achieve both fidelity (e.g., running actual TCP stacks) and scale, we use a simple testbed setup described below.

Our testbed uses a physical network with a dumbbell topology, with ten sender-receiver node pairs connected to a BESS software switch [3], as seen in Figure 1. We choose this topology as it is a common topology used to evaluate throughput models and fairness, and has been used to model a wide variety of scenarios [26, 37, 39, 45]. The bottleneck bandwidth for the experiments is varied between *EdgeScale* and *CoreScale* by changing the bandwidth and buffer size on the BESS software switch. We use a software switch as it allows greater control over the queue size and bottleneck bandwidth than the physical switches available to us, while still being closer to using physical network elements than a simulator like ns-2 [6] or ns-3 [27]. The edge link bandwidths between the sender/receiver nodes and bottleneck link at the BESS switch is always 25 Gbps, which guarantees that congestion occurs at the BESS switch. The base RTT of flows is set using `netem` [8] to add the appropriate delay at the receiver, similar to past work [39, 45, 51]. We calculate the packet loss rate by logging packet drops at the bottleneck queue in the software switch, and use the Linux tool tcpprobe [9] to measure the congestion window halving rate to validate the Mathis throughput model. The testbed was hosted on CloudLab [22].

All TCP flows are distributed equally across each of the sender-receiver pairs and send infinite data, as common in past experiments [26, 28, 39, 45, 51]. The flows run for a maximum duration of 3 hours, significantly longer than past studies [26, 28, 45, 48], or until the metric being evaluated changes by less than 1% over 20 minutes. When an experiment starts, each flow waits a random

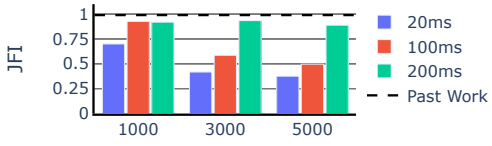


(a) *CoreScale*

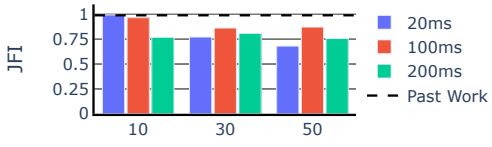


(b) *EdgeScale*

Figure 3: The ratio between packet losses and congestion events (i.e., CWND halvings) changes between *CoreScale* and *EdgeScale*, and across across different flow counts within *CoreScale*.



(a) *CoreScale*



(b) *EdgeScale*

Figure 4: BBR shows intra-CCA unfairness in *CoreScale*, with JFIs as low as 0.4. Milder unfairness can also be seen beyond 10 flows in *EdgeScale*, with JFIs as low as 0.7.

period of time between 0 and 2 minutes before it establishes a connection with the receiver, and the throughput obtained by all flows in the first 5 minutes of the experiment is ignored.

Limitations: As observed by many others, capturing the dynamics of Internet links with high fidelity — including random loss, arrival, and departures of new flows, application-level sending behaviors, etc — is perhaps impossible to achieve perfectly [15, 49, 50]. Furthermore, understanding the behaviors of CCAs can be challenging in "real" settings where many uncontrolled variables combine to influence CCA behavior. We instead opt to focus directly on *only two key variables*: the number of concurrent flows (which increases by two orders of magnitude between *EdgeScale* and *CoreScale*) and the link capacity (which increases similarly between *EdgeScale* and *CoreScale*). Therefore, when we say 'at scale', we refer to the setting where the bottleneck bandwidth is 10 Gbps and the flow count ranges from 1000 to 5000 flows. By controlling all other aspects of the experiment (all flows have the same, lengthy duration; there is no random loss; buffer sizes are approximately 1 BDP in both settings; all flows have the same RTT etc.) we can more easily inspect the impact of these two variables on CCA behavior.

4 Revisiting the Mathis Throughput Model

Background: The Mathis model [37] predicts the throughput of a NewReno flow as a function of loss (p) and round-trip time (RTT). It depends on two constants: C, which may be different for different CCAs, and MSS (maximum segment size), which in our case is fixed to 1448 bytes. The Mathis model equation can be expressed as:

$$Throughput = \frac{MSS * C}{RTT * \sqrt{p}} \quad (1)$$

The original paper by Mathis et al. [37] states that p refers to the congestion event rate. This can be interpreted in one of two ways: (a) the congestion window (CWND) halving rate or (b) the packet loss rate. While the original paper states that the CWND halving rate should be used for TCP with selective ACKs, subsequent research has often applied the packet loss rate instead [44, 46]. We, therefore, evaluate the Mathis equation with both the packet loss rate and the CWND halving rate.

The original paper derives a constant $C = 0.94$ for NewReno with delayed and selective ACKs [37]. The paper also demonstrates how to derive C empirically for varying NewReno configurations. For our modern NewReno [7, 36] stack we derive C empirically following the methodology described by Mathis: we calculate the C which minimizes the least squared prediction error of the Mathis equation at a given flow count and setting. For the following results, all flows run NewReno and have a 20ms RTT.

Finding 1: ***Deriving C using packet loss rate results in flow-count dependent values and different values in *CoreScale* vs. *EdgeScale*. Using CWND halving rate produces consistent C values across both settings and flow counts. (Table 1)***

Table 1 shows the empirically derived "best-fit" constant C for NewReno in a few example settings. We see two main observations here. First, when using the packet loss rate the C value is quite different between *EdgeScale* and *CoreScale* and also changes between different flow counts in *CoreScale*. This violates the Mathis model which states that C depends only on the CCA being used, and should not change with the number of competing flows or bottleneck bandwidth. However, using the CWND halving rate produces a more consistent constant that changes only slightly between the *EdgeScale* and *CoreScale*, and does not change significantly between flow counts within *CoreScale*.

Finding 2: ***Using the CWND halving rate results in accurate predictions ($\leq$ 10% median error) in *CoreScale*; using packet loss rate results in 45%-55% median error. In *EdgeScale*, however, both are accurate. (Fig 2)***

Fig 2 shows the median Mathis prediction error at different flow counts in *CoreScale*, while the two horizontal lines represent the median prediction error obtained in *EdgeScale*. These results show that the Mathis model does indeed hold at scale, as long as we use the CWND halving rate for p. The 45%-55% median error when using the packet loss rate implies it cannot be used to accurately predict NewReno throughput at scale, even though the packet loss rate works well in *EdgeScale*. The error at scale is foreshadowed by the significantly different C values derived across settings and flow counts when using the packet loss rate.

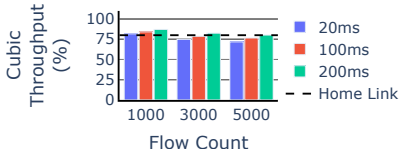


Figure 5: Cubic takes 70% to 80% of total throughput when competing with an equal number of NewReno flows in *CoreScale*.

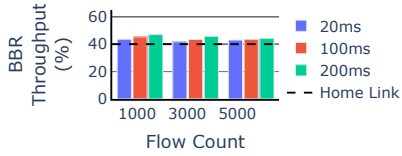


Figure 6: 1 BBR flow takes 40% of total throughput when competing with thousands of NewReno flows in *CoreScale*.

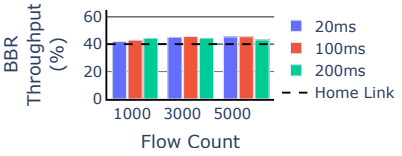


Figure 7: 1 BBR flow takes 40% of total throughput when competing with thousands of Cubic flows in *CoreScale*.

Finding 3: ***The ratio between packet losses and congestion events (i.e., CWND halvings) changes between** CoreScale **and** EdgeScale, **and across across different flow counts within** CoreScale. **(Fig 3)***

While investigating why the packet loss rate results in different constants in *EdgeScale* vs *CoreScale*, we discovered the ratio of packet loss rate to CWND halving rate is different in the two settings. As seen in Fig 3, in *EdgeScale*, the ratio of packet losses to CWND halvings is approximately 1.7 regardless of the number of concurrent flows. But in *CoreScale* the ratio varies between 6 and 9 and depends on the flow count. This explains why using packet loss rate results in different constants between *EdgeScale* and *CoreScale*, and different constants within *CoreScale* at different flow counts. While the idea that packet loss rate diverges from CWND halving rate is not new [23, 37, 41], we believe the drastic increase in divergence as we move from *EdgeScale* to *CoreScale* is a new finding.

Since the ratio is stable for *EdgeScale*, there is no reason to doubt past research that uses packet loss rate for p when evaluating links with tens of flows and only tens or hundreds of Mbps [44, 46]. *However, our results show one should not use the packet loss rate for estimating throughput over the Internet core.*

We hypothesize that the reason for different packet loss rate to CWND halving rate ratios is that losses are burstier at scale, causing multiple losses in the same burst or RTT which result in only one congestion window halving. We corroborate this hypothesis by measuring the burstiness of losses at the queue using the Goh-Barabasi burstiness score [24] which ranges from -1 to 1, where a higher score means the drops are burstier. We obtain median values close to 0.2 in *EdgeScale* and closer to 0.35 in *CoreScale*, implying that losses are indeed burstier at scale (Figure not shown).

Implications: Overall, we find that the Mathis model for throughput still holds in *CoreScale*, if C is calculated using CWND halving rate and not the more commonly used packet loss rate for the variable p. Unfortunately, this makes applying the Mathis model in practice more challenging, as obtaining the CWND halving rate requires end-host state reconstruction, where packet loss rate can be measured more easily via network-measurable loss. Furthermore, our findings also change our expectations regarding NewReno's performance with respect to loss: a flow on a congested core link can tolerate four times the packet loss rate of a flow on a congested home link, and still obtain the same bandwidth because the CWND halving rate is the same.

5 Revisiting Fairness

In this section, we measure how fairly competing flows share bandwidth in our *CoreScale* setting.

5.1 Intra-CCA Fairness

Background: The classic metric used for measuring fairness is Jain's Fairness Index (JFI) [29], which ranges from 0 to 1 with a higher value indicating greater fairness. Past research in the edge setting has found Cubic, NewReno, and BBR to be intra-CCA fair – i.e., fair when competing only with other flows of the same CCA and RTT – with a JFI of 0.9 or more [18, 20, 26, 45, 52].

Finding 4: ***NewReno & Cubic continue to show high intra-CCA fairness in** CoreScale **with a JFI > 0.99, as expected from past research. (Figure not shown)***

Both theoretical [20] and empirical studies [26, 39] have shown that when NewReno flows compete with other NewReno flows, or Cubic flows compete with other Cubic flows, throughput is shared almost equally when all flows have the same RTT. Our experiments confirm this in the *CoreScale* setting: NewReno and Cubic show high fairness with a JFI > 0.99.

Finding 5: ***BBR surprisingly shows intra-CCA unfairness in** CoreScale, **with JFIs as low as 0.4, which is not expected from past research. Milder unfairness also occurs when more than 10 flows compete in** EdgeScale, **with JFI's as low as 0.7. (Fig 4)***

Fig 4 shows the JFI for BBR flows with the same RTT when they compete amongst themselves at different flow counts. It also shows the JFI based on results from past work (0.99) which finds BBR to be intra-CCA fair when all flows have the same RTT [18, 28, 47, 52].

We see that at scale BBR surprisingly becomes unfair at 20ms and 100ms RTTs, with the JFI going as low as 0.4. We investigate further and discover that BBR shows signs of unfairness even in *EdgeScale*, but at relatively higher flow counts (greater than 10) not examined by past research. This unfairness is exacerbated at scale.

Cardwell et al. [18] argue that BBR flows, share bandwidth fairly amongst each other at lower flow counts due to flow synchronization. While we have not verified it, we hypothesize that the unfairness in *CoreScale* might be due to BBR flows desynchronizing at scale, similar to NewReno [13].

Implications: Prior work showed that BBR is unfair when competing with other CCAs (e.g. Cubic, NewReno) – however, it was assumed that if the entire Internet adopted BBR users could expect fair outcomes. Our *CoreScale* experiments show that this is not the case when thousands of flows compete in wide-area like settings;

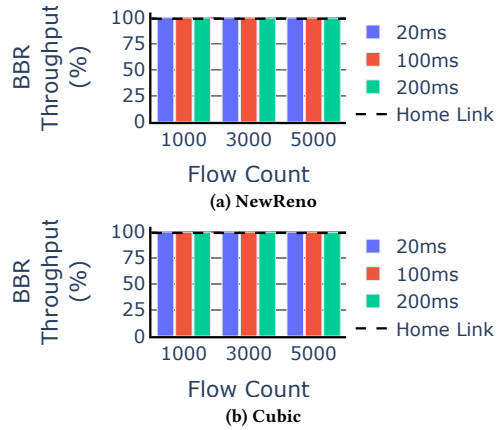


Figure 8: BBR takes 99.9% of total throughput when competing with an equal number of NewReno or Cubic flows.

this emphasizes the need for CCA testing and evaluation at scale to understand whether a new algorithm is acceptable for deployment.

5.2 Inter-CCA Fairness

In this section, we evaluate how flows from different CCAs with the same RTT compete with each other.

Background: Past research in the edge link setting found that Cubic competes unfairly with NewReno, taking up to 80% of total throughput [26, 39] and that BBR is unfair to both Cubic and NewReno [28, 45, 48]. We revisit these properties at scale.

For the following results, we measure the aggregate throughput obtained by the flows of one CCA as a fraction of the total throughput obtained by all flows.

Finding 6: ***A single BBR flow takes 40% of total throughput when competing with thousands of NewReno or Cubic flows in*** *CoreScale,* ***as predicted by past research in the edge setting. (Figs 6, 7)***

The BBR model by Ware et al. [48] shows that a single BBR flow could take 40% of total throughput irrespective of the number of competing NewReno or Cubic flows. We show that this result holds at scale and that a single BBR flow takes 40% of total throughput even when competing with thousands of NewReno or Cubic flows, as seen in Figs 5 and 6.

Finding 7: ***BBR takes 99.9% of total throughput when competing with an equal number of NewReno (or Cubic) flows in*** *CoreScale,* ***confirming past research in the edge setting. (Fig 8)***

Past research in the edge setting has shown that BBR can take up to 99% of total throughput when competing with an equal number of Cubic flows [28, 45, 52]. Our results show that this inter-CCA unfairness persists even in *CoreScale*, with BBR obtaining up to 99.9% of total throughput when competing with an equal number of Cubic or NewReno flows, as seen in Fig 8.

Finding 8: ***Cubic takes 70% to 80% of total throughput when competing with an equal number of NewReno flows in*** *CoreScale,* ***confirming the unfairness results of past research in the edge setting. (Fig 5)***

Past research [26, 39] expects Cubic flows to get around 80% of total throughput when competing with an equal number of NewReno flows in the edge setting. Our experiments show this holds true even in *CoreScale*, as seen in Figure 5.

Implications: Our results confirm that the inter-CCA unfairness displayed by Cubic to NewReno and BBR to both Cubic and NewReno persist at higher flow counts at scale. In these settings, the disparities between flows can be even more extreme than at the edge setting – with a single BBR flow attaining 4 Gbps and 5000 competing flows Reno or Cubic flows obtaining just 1.2 Mbps each. While past work shows that a few 'bad player' flows can impact fairness between a small number of users sharing an edge link (e.g. roommates in a shared house, co-workers in an office), the fact that prior unfairness findings extend to *CoreScale* suggests severely unfair outcomes where a single sender can impact thousands of physical neighbors with whom he or she shares a large inter-domain link.

6 Conclusions

Conventional wisdom about congestion control adopted by application and systems designers has been evaluated in settings implicitly assuming congestion at the edge. When congestion occurs in the core, as shown by many measurements, it is not clear if these accepted norms about throughput and fairness still hold. We revisit these and find that the widely accepted Mathis model can be applied using either loss or congestion window halving in edge settings, but these metrics diverge at scale. Similarly, we find that when BBR competes with other BBR flows, it goes from being completely fair in the edge setting to completely unfair at scale. We hope that our work, though preliminary, serves as a cautionary tale for naïvely extrapolating CCA properties at scale and inspires further modeling and analysis.

7 Acknowledgements

We would like to thank our shepherd Balakrishnan Chandrasekaran and the anonymous reviewers for their comments and help revising the paper. We would also like to thank Hugo Sadok, Christopher Canel, Nirav Atre, Anup Agarwal, and Srinivasan Seshan for their valuable feedback. This research is supported in part by NSF Grant No. 1850384 and the vmWare Systems Research Award.

References

[1] 2020. htsim. https://nrg.cs.ucl.ac.uk/mptcp/implementation.html. Accessed: 2020-03-12.
[2] 2021. BBR v2: A Model-based Congestion Control. https://datatracker.ietf.org/meeting/104/materials/slides-104-iccrg-an-update-on-bbr-00. Accessed: 2021-03-04.
[3] 2021. BESS: A Software Switch. https://github.com/NetSys/bess. Accessed: 2021-03-04.
[4] 2021. Google LLC Peering Data. https://www.peeringdb.com/net/433. Accessed: 2021-03-04.
[5] 2021. Netflix falls to second place in global internet traffic share as other streaming services grow. https://www.sandvine.com/inthenews/netflix-falls-to-second-place-in-global-internet-traffic-share. Accessed: 2021-03-04.
[6] 2021. The Network Simulator - ns-2. https://www.isi.edu/nsnam/ns/. Accessed: 2021-03-04.
[7] 2021. NewReno RFC History. https://www.rfc-editor.org/search/rfc_search_detail.php?title=NewReno. Accessed: 2021-03-04.
[8] 2021. tc-netem(8) - Linux Manual Page. https://man7.org/linux/man-pages/man8/tc-netem.8.html. Accessed: 2021-03-04.
[9] 2021. tcpprobe. https://wiki.linuxfoundation.org/networking/tcpprobe Accessed 2021-09-10.

[10] I. Abdeljaouad, H. Rachidi, S. Fernandes, and A. Karmouch. 2010. Performance analysis of modern TCP variants: A comparison of Cubic, Compound and New Reno. In *2010 25th Biennial Symposium on Communications*. 80–83. https://doi.org/10.1109/BSC.2010.5472999

[11] Aditya Akella, Srinivasan Seshan, and Anees Shaikh. 2003. An Empirical Evaluation of Wide-Area Internet Bottlenecks. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement* (Miami Beach, FL, USA) *(IMC '03)*. Association for Computing Machinery, New York, NY, USA, 101–114. https://doi.org/10.1145/948205.948219

[12] Mohammad Alizadeh, Adel Javanmard, and Balaji Prabhakar. 2011. Analysis of DCTCP: Stability, Convergence, and Fairness. 39, 1 (June 2011), 73–84. https://doi.org/10.1145/2007116.2007125

[13] Guido Appenzeller, Isaac Keslassy, and Nick McKeown. 2004. Sizing Router Buffers. *ACM SIGCOMM* (2004).

[14] Venkat Arun and Hari Balakrishnan. 2018. Copa: Practical Delay-Based Congestion Control for the Internet. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Renton, WA, 329–342. https://www.usenix.org/conference/nsdi18/presentation/arun

[15] Sachin Ashok, Sai Surya Duvvuri, Nagarajan Natarajan, Venkata N. Padmanabhan, Sundararajan Sellamanickam, and Johannes Gehrke. 2020. IBox: Internet in a Box. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks* (Virtual Event, USA) *(HotNets '20)*. Association for Computing Machinery, New York, NY, USA, 23–29. https://doi.org/10.1145/3422604.3425935

[16] Neda Beheshti, Yashar Ganjali, Monia Ghobadi, Nick McKeown, and Geoff Salmon. 2008. Experimental Study of Router Buffer Sizing. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement* (Vouliagmeni, Greece) *(IMC '08)*. Association for Computing Machinery, New York, NY, USA, 197–210. https://doi.org/10.1145/1452520.1452545

[17] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. 1994. TCP Vegas: New Techniques for Congestion Detection and Avoidance. *SIGCOMM Comput. Commun. Rev.* 24, 4 (Oct. 1994), 24–35. https://doi.org/10.1145/190809.190317

[18] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. 2016. BBR: Congestion-Based Congestion Control. *ACM Queue* 14, September-October (2016), 20 – 53. http://queue.acm.org/detail.cfm?id=3022184

[19] Yanpei Chen, Rean Griffith, Junda Liu, Randy H. Katz, and Anthony D. Joseph. 2009. Understanding TCP Incast Throughput Collapse in Datacenter Networks. In *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking* (Barcelona, Spain) *(WREN '09)*. Association for Computing Machinery, New York, NY, USA, 73–82. https://doi.org/10.1145/1592681.1592693

[20] Dah-Ming Chiu and Raj Jain. 1989. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Comput. Netw. ISDN Syst.* 17, 1 (June 1989), 1–14. https://doi.org/10.1016/0169-7552(89)90019-6

[21] A. Dhamdhere, D. Clark, A. Gamero-Garrido, M. Luckie, R. Mok, G. Akiwate, K. Gogia, V. Bajpai, A. Snoeren, and k. claffy. 2018. Inferring Persistent Interdomain Congestion. In *ACM SIGCOMM*.

[22] Dmitry Duplyakin, Robert Ricci, Aleksander Maricq, Gary Wong, Jonathon Duerig, Eric Eide, Leigh Stoller, Mike Hibler, David Johnson, Kirk Webb, Aditya Akella, Kuangching Wang, Glenn Ricart, Larry Landweber, Chip Elliott, Michael Zink, Emmanuel Cecchet, Snigdhaswin Kar, and Prabodh Mishra. 2019. The Design and Operation of CloudLab. In *Proceedings of the USENIX Annual Technical Conference (ATC)*. 1–14. https://www.flux.utah.edu/paper/duplyakin-atc19

[23] Sally Floyd, Mark Handley, Jitendra Padhye, and Jörg Widmer. 2000. Equation-Based Congestion Control for Unicast Applications. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (Stockholm, Sweden) *(SIGCOMM '00)*. Association for Computing Machinery, New York, NY, USA, 43–56. https://doi.org/10.1145/347059.347397

[24] K.-I. Goh and A.-L. Barabási. 2008. Burstiness and memory in complex systems. *EPL (Europhysics Letters)* 81, 4 (jan 2008), 48002. https://doi.org/10.1209/0295-5075/81/48002

[25] Andrei Gurtov, Tom Henderson, Sally Floyd, and Yoshifumi Nishida. 2012. The NewReno Modification to TCP's Fast Recovery Algorithm. RFC 6582. https://doi.org/10.17487/RFC6582

[26] Sangtae Ha, Injong Rhee, and Lisong Xu. 2008. CUBIC: a new TCP-friendly high-speed TCP variant. *Operating Systems Review* 42 (07 2008), 64–74. https://doi.org/10.1145/1400097.1400105

[27] Thomas R Henderson, Mathieu Lacage, George F Riley, Craig Dowell, and Joseph Kopena. 2008. Network simulations with the ns-3 simulator. *SIGCOMM demonstration* 14, 14 (2008), 527.

[28] Mario Hock, Roland Bless, and Martina Zitterbart. 2017. Experimental evaluation of BBR congestion control. 1–10. https://doi.org/10.1109/ICNP.2017.8117540

[29] Raj Jain, Dah Ming Chiu, and Hawe WR. 1998. A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems. *CoRR* cs.NI/9809099 (01 1998).

[30] Glenn Judd. 2015. Attaining the Promise and Avoiding the Pitfalls of TCP in the Datacenter. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. USENIX Association, Oakland, CA, 145–157. https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/judd

[31] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. 2020. Exploring the "Internet from Space" with Hypatia. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 214–229. https://doi.org/10.1145/3419394.3423635

[32] Tomoki Kozu, Yuria Akiyama, and Saneyasu Yamaguchi. 2013. Improving RTT Fairness on CUBIC TCP. In *2013 First International Symposium on Computing and Networking*. 162–167. https://doi.org/10.1109/CANDAR.2013.30

[33] Soojeon Lee, Myungjin Lee, Dongman Lee, Hyungsoo Jung, and Byoung-Sun Lee. 2015. TCPRand: Randomizing TCP payload size for TCP fairness in data center networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. 1697–1705. https://doi.org/10.1109/INFOCOM.2015.7218550

[34] Yuliang Li, Rui Miao, Hongqiang Harry Liu, Yan Zhuang, Fei Feng, Lingbo Tang, Zheng Cao, Ming Zhang, Frank Kelly, Mohammad Alizadeh, and Minlan Yu. 2019. HPCC: High Precision Congestion Control. In *Proceedings of the ACM Special Interest Group on Data Communication* (Beijing, China) *(SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 44–58. https://doi.org/10.1145/3341302.3342085

[35] Gustavo Marfia, Claudio E. Palazzi, Giovanni Pau, Mario Gerla, M. Y. Sanadidi, and Marco Roccetti. 2007. TCP *Libra* : Exploring RTT-Fairness for TCP. In *NETWORKING 2007. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet, 6th International IFIP-TC6 Networking Conference, Atlanta, GA, USA, May 14-18, 2007, Proceedings*, Ian F. Akyildiz, Raghupathy Sivakumar, Eylem Ekici, Jaudelice Cavalcante de Oliveira, and Janise McNair (Eds.). Springer. https://doi.org/10.1007/978-3-540-72606-7_86

[36] Matt Mathis, Nandita Dukkipati, and Yuchung Cheng. 2013. Proportional Rate Reduction for TCP. RFC 6937. https://doi.org/10.17487/RFC6937

[37] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott. 1997. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. *SIGCOMM Comput. Commun. Rev.* 27, 3 (July 1997), 67–82. https://doi.org/10.1145/263932.264023

[38] Nick McKeown, Guido Appenzeller, and Isaac Keslassy. 2019. Sizing Router Buffers (Redux). *SIGCOMM Comput. Commun. Rev.* 49, 5 (Nov. 2019), 69–74. https://doi.org/10.1145/3371934.3371957

[39] Dimitrios Miras, Martin Bateman, and Saleem Bhatti. 2008. Fairness of High-Speed TCP Stacks. 84–92. https://doi.org/10.1109/AINA.2008.143

[40] Ayush Mishra, Xiangpeng Sun, Atishya Jain, Sameer Pande, Raj Joshi, and Ben Leong. 2019. The Great Internet TCP Congestion Control Census. *Proc. ACM Meas. Anal. Comput. Syst.* 3, 3, Article 45 (Dec. 2019), 24 pages. https://doi.org/10.1145/3366693

[41] J. Padhye, V. Firoiu, Towsley DF, and J. Kurose. 2000. Modeling TCP through-put: A simple model and its empirical validation. *Computer Communication Review* 28 (01 2000).

[42] J. Sherry R. Ware, M. K. Mukerjee and S. Seshan. 2018. The Battle for Bandwidth: Fairness and Heterogeneous Congestion Control. In Poster at NSDI 2018.

[43] Costin Raiciu, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. 2011. Improving Datacenter Performance and Robustness with Multipath TCP. 41, 4 (Aug. 2011), 266–277. https://doi.org/10.1145/2043164.2018467

[44] Injong Rhee, Lisong Xu, Sangtae Ha, Alexander Zimmermann, Lars Eggert, and Richard Scheffenegger. 2018. CUBIC for Fast Long-Distance Networks. RFC 8312. https://doi.org/10.17487/RFC8312

[45] Kanon Sasaki, Masato Hanai, Kouto Miyazawa, Aki Kobayashi, Naoki Oda, and Saneyasu Yamaguchi. 2018. TCP Fairness Among Modern TCP Congestion Control Algorithms Including TCP BBR. 1–4. https://doi.org/10.1109/CloudNet.2018.8549505

[46] Stefan Savage, Tom Anderson, Amit Aggarwal, David Becker, Neal Cardwell, Andy Collins, Eric Hoffman, John Snell, Amin Vahdat, Geoff Voelker, and John Zahorjan. 1999. Detour: a case for informed internet routing and transport. *IEEE Micro* 19 (1999), 50–59.

[47] Dominik Scholz, Benedikt Jaeger, Lukas Schwaighofer, Daniel Raumer, Fabien Geyer, and G. Carle. 2018. Towards a Deeper Understanding of TCP BBR Congestion Control. *2018 IFIP Networking Conference (IFIP Networking) and Workshops* (2018), 1–9.

[48] Ranysha Ware, Matthew K. Mukerjee, Srinivasan Seshan, and Justine Sherry. 2019. Modeling BBR's Interactions with Loss-Based Congestion Control. *IMC* (2019).

[49] Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan. 2013. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. USENIX Association, Lombard, IL, 459–471. https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/winstein

[50] Francis Y. Yan, Jestin Ma, Greg D. Hill, Deepti Raghavan, Riad S. Wahby, Philip Levis, and Keith Winstein. 2018. Pantheon: the training ground for Internet congestion-control research. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*. USENIX Association, Boston, MA, 731–743. https://www.usenix.org/conference/atc18/presentation/yan-francis

[51] Zhaojuan Yue, Xiaodan Zhang, Yongmao Ren, Jun Li, and Qianli Zhong. 2012. The performance evaluation and comparison of TCP-based high-speed transport protocols. In *2012 IEEE International Conference on Computer Science and Automation Engineering*. 509–512. https://doi.org/10.1109/ICSESS.2012.6269516

[52] Yuxiang Zhang, Lin Cui, and Posco Tso. 2018. Modest BBR: Enabling Better Fairness for BBR Congestion Control. 00646–00651. https://doi.org/10.1109/ISCC.2018.8538521

A Ethics

To the best of our knowledge, this work raises no ethical concerns.