Congestion Safety Audits for Internet Services

Adithya Abraham Philip, Srinivasan Seshan, Justine Sherry, Isabel Suizo and Ranysha Ware

Carnegie Mellon University

Extended Abstract

When two family members sharing a home Internet connection simultaneously log in to their online applications of choice, how well do their services perform? Will one videogaming child experience timeouts and stalls because their sibling is also streaming a popular TV show? Will a teleconferencing mother have her calls drop because her spouse chooses just this moment to install a software update? When multiple Internet users share a bottleneck network link and overload it, the link is said to become congested. Bandwidth – the volume of data that can be transmitted per second – must be shared between users. If the link is overloaded, then latency – the amount of time it takes one bit of information to travel from sender to receiver – will increase, and packets of data can be dropped entirely by the network.

Most Internet service providers do little to police the impact of congestion between users, simply forwarding packets in the order that they arrive, "first come, first serve." This policy leaves the responsibility of mitigating the effects of congestion to the senders themselves. Internet applications use algorithms called Congestion Control Algorithms (CCAs) to carefully speed up and slow down their transmission to avoid congestion. CCAs aim to share available bandwidth equally between connections, keep latencies low, and avoid packet loss. Unfortunately, recent research shows that the success of applications in equitably mitigating congestion is mixed at best. In a recent study of widely deployed services, it was found that one file-distribution application consumed well over its fair share of bandwidth, causing other concurrently running applications to obtain as little as 16% of their fair share of bandwidth. In the same study, it was found that a popular video streaming service could lead to stalls for competing teleconferencing applications (even though other video streaming services did not cause this problem).

In this paper, we explore how to evaluate whether Internet applications are friendly or fair to other applications sharing the same bottleneck link. Taking inspiration from the security community, we propose congestion safety audits, a methodology for analyzing an Internet service to identify common-case pitfalls in congested environments. Much like it is impossible to prove that an Internet application is "definitely secure", there exists no approach to guarantee that an application will always share bandwidth with other applications or never lead to packet loss or stalls for competing traffic. Security audits allow inspectors to perform a suite of tests and evaluations to ensure that due diligence has been carried out to attempt to secure an application. We propose that service operators can similarly use a standardized suite of tests to verify that their applications meet common-sense guidelines for performance under competition.

With a standardized process for congestion safety auditing, companies can evaluate their own *applications to avoid bad publicity. Or, in a world in which regulators take an interest in behavior under competition, congestion safety audits might provide validation that operators have followed due diligence in avoiding overly aggressive behavior on the Internet.*

1 Introduction

Because the Internet is so critical to modern commerce, communication, and daily life, policy researchers have paid close attention to how to ensure that the Internet is equitably and universally accessible to users and service operators. One common concern among policymakers is how *bandwidth* – the number of bits per second that a network link can transfer – is shared among competing services. Every network link has some total bandwidth capacity, and when multiple Internet connections share the same link (as in a home connection with multiple family members), this bandwidth must be shared between the competing connections. Policymakers have primarily explored bandwidth sharing in the context of *network neutrality*: the argument that network operators (such as AT&T or Comcast) should not prioritize the data of one service over another (*e.g.*, granting more bits per second to Domino's rather than Pizza Hut) [54].

This paper explores a related, but different challenge: that even in so-called *neutral* networks, where the network performs no explicit prioritization of one service over another, it still remains challenging to ensure that bandwidth is shared equitably between competing services. As we will discuss in §2, in the absence of explicit prioritization mechanisms (or explicit mechanisms to enforce fairness) the allocation of bits to applications is driven by decisions made at senders regarding how fast to transmit. Services (such as Facebook or Netflix) use algorithms called *congestion control algorithms* (CCAs) to identify the best rate to transmit at; well-designed algorithms aim to share bandwidth *fairly* (see §3 for definitions of fairness) between competing connections when multiple connections share the same network link.

Unfortunately, the technical community faces several challenges in actually delivering equitable bandwidth sharing. First, within the community, there is significant debate over how an ideal – or even a 'good enough' – algorithm should behave, with competing definitions of fairness each drawing in crowds of supporters. Second, even when developers agree upon how algorithms *should* behave, it remains near-impossible to develop algorithms which guarantee this behavior in the dynamic and unpredictable environment of the Internet.

The goal of this white paper is to bring these challenges to light to the tech policy community; to solicit feedback, ideas, and debate from experts in law, standardization, economics, and social factors. Networking researchers will find our introductory material (§§2–3) to be old news and should skip to §4.

At the heart of our concern is a hypothesized future conflict in which the policy community may find itself without roadmap or precedent. Poorly designed CCAs can dominate a network link, consuming a majority of network bandwidth for themselves and leaving little for the competition, leading to a condition called *starvation*, where a connection is unable to succeed due to lack of bandwidth. If, someday, we find that Pepsi accuses Coca-Cola of deploying a

CCA that is too-aggressive, advantaging Coca-Cola and disadvantaging Pepsi, how should the policy community proceed?

Our own research has illustrated the challenges and pitfalls of deploying services that share Internet bandwidth equitably. Over the past eight years, our lab has performed both mathematical and experimental analyses of deployed Internet services to investigate how competing services share network bandwidth [41, 52, 53, 51, 36, 42]. In 2019, we provided the first mathematical model of a new CCA from Google called BBR [52] which illustrated that BBR was fundamentally unable to cede bandwidth to competing connections using legacy algorithms as the number of services sharing the same network link increased. This led to some backlash in the popular press [49, 3, 48], but to Google's credit, they quickly patched the problem with a new BBR 'v2' and invested engineering effort [9, 7] and research dollars [23] into ensuring that future versions of BBR play nicely with competing traffic. Google's example highlights how challenging it is to guarantee equitable outcomes, even among well-intended actors.

At the same time, in 2024, we showed empirically that the file-sharing service Mega dominates the network links it uses, leaving little leftover bandwidth for the competition; this has yet to be patched [41]. We diagnose part of the root cause as Mega using multiple parallel Internet connections to transfer a single file, a tactic that has been long-known to lead to advantageous results for one service at the expense of others [5]. Hence, it is clear that some operators are not performing even a minimum threshold of diligence towards bandwidth sharing.

To distinguish good-faith actors from negligent or bad-faith actors, we propose the idea of *congestion safety audits* in §5. Congestion safety audits are a process that Internet providers might pursue in order to (a) strive to develop CCAs which more equitably share bandwidth with the competition, and (b) provide evidence to external parties of best-effort intent to cede bandwidth for competing connections.

Our discussion is merely a starting point and an invitation to the policy community to innovate with us. To further facilitate policy researchers to engage with the end-to-end bandwidth sharing problem, we present a set of questions for discussion in §6 and finally provide a list of recommended reading in §7).

2 The End-to-End Sharing Challenge

In this section, we first introduce the end-to-end bandwidth sharing challenge. We then describe why this challenge arises in so-called 'neutral' networks (which do not prioritize traffic from one user over another's) and why even well-designed systems struggle to guarantee equitable bandwidth sharing We then discuss some history of end-to-end rate sharing and how the problem differs from Internet fairness issues in the popular press such as network neutrality and peer-to-peer file sharing.

Terminology and Problem Statement: When an application transmits a file over the Internet, the file is broken into smaller chunks called *packets* and these packets are transmitted one-by-one through the network. Using packets allows network links to serve multiple users at once, *e.g.*, serving a packet from user A, then one from user B, and then one from user A again.

The end-to-end bandwidth sharing challenge concerns how much data is transferred for user A versus user B when sharing in this way. If user A receives service at a rate of 10 packets per second, while user B receives service at a rate of 1 packet per second, we might consider the result *unfair* (for definitions of fairness we will defer to §3).

In practice, every link (wire, cable, wireless connection, *etc.*) has some *bandwidth* or *capacity* and it is computed not in terms of the number of packets-per-second but instead in bits-per-second (bps), the number of 1's and 0's that can traverse the link in one second. A typical home broadband link in the United States has capacity of several million bits (megabits) per second (Mbps). When an application transmits over a link, data will be transferred at some rate referred to as the *throughput*; throughput will be less than or equal to the link capacity depending on, *e.g.* if there are other senders in the network (necessitating that the bandwidth be divided between users) or if the sending application is even transmitting at a rate high enough to use all of the available bandwidth (*e.g.*, high definition video streaming requires somewhere between 8-50Mbps and no more [25, 47, 39, 32]).

Routers and *switches* are responsible for steering packets across numerous network links. When multiple packets simultaneously arrive for transport on the same link, the packets are placed into a *queue* to wait to be served. Each queue has some capacity and once it fills, it can no longer accept new packets and any arriving new packets will be dropped so long as the queue reimains full.

Sharing Issues Arise Even in Neutral Networks: Many policymakers have weighed in on *network neutrality*: the idea that routers and switches should not prioritize the packets of one sender over another in anti-competitive ways [14]. Nonetheless, the end-to-end bandwidth sharing challenge occurs explicitly in *neutral* networks, which do nothing to prioritize or balance traffic between users. Hence, when some senders transmit at an overly aggressive rate, the network does nothing to stop them from obtaining an outsized proportion of network capacity.

The default behavior for routers and switches receiving packets from multiple users is to serve arriving packets in *first-in first-out* (FIFO) order. This scheme does nothing to prioritize one sender over another. At the same time, the scheme also does nothing to *enforce* any form of equal sharing between users. This means that if user A *transmits* at two packets per second and user B transmits at one packet per second, A will simply obtain twice as much throughput as B.

Sender-Based Algorithms Determine Rate Allocations: At first glance, one might expect senders to then transmit at the maximum rate that they possibly can, to maximize their sending rate in the presence of competition. This behavior, however, is also non-ideal due to the properties of queues we discussed above. If data arrives at a router or switch faster than it can serve the data, packets will build up in the router's queue leading to two undesirable outcomes:

1. Packet Loss: packets will be dropped as queues are unable to accept new packets for lack of incoming space. Packet loss significantly degrades application performance: the receiver must signal to the sender that the packet was never received, and it needs to be re-transmitted, all of which takes time and increases resources usage.

2. High Latency: packets that are not dropped must spend extra time waiting their turn to reach the front of the queue and be transmitted. This increases the amount of time it takes between when the packet is sent by the sender and received at the client; increases in latency are most problematic for real-time applications like video conferencing.

To avoid the negative repercussions of overload, senders instead carefully try to adapt their sending rates to achieve an ideal throughput, generally trying to maximize throughput, to avoid excessive packet loss, and to avoid high latency.

Congestion Control Algorithms (CCAs) are algorithms that run at Internet senders to adapt sending rates. CCAs implement some strategy of reducing and increasing sending rates based on indirect signals they observe in the network. For example, when a packet is dropped, many CCAs interpret this packet loss as an indicator that the network is overloaded and hence slow down [24]. There are a wide range of CCAs in use today in part because these indirect signals make it challenging to infer the 'perfect' sending rate. A packet loss, as just mentioned, may indicate a network overload – or, it might indicate that the network is wireless and there is some form of transitory radio interference at play which has nothing to do with congestion at all. Hence, some algorithms choose *not* to interpret packet loss as a sign of congestion at all, and instead rely on other signals [8]. No algorithm in the literature today is without a well-studied list of shortcomings that engineers and researchers are constantly aiming to improve.

At the same time that engineers try to develop CCAs that avoid overload, they also carefully analyze their CCAs to try to ensure that CCAs share network bandwidth according to some notion of fairness; *e.g.* to avoid the example above with one sender transmitting two packets per second and another transmitting one packet per second. This is not entirely out of generosity to 'the competition': a single Internet service may open multiple connection simultaneously and hence there is internal incentive to design algorithms such that bandwidth is shared evenly between its own connections.

Foundations of Fairness in Congestion Control: In 1989, Chiu and Jain [13] published a foundational proof that senders who followed an *additive-increase multiplicative-decrease* (AIMD) approach to loss-based congestion control would, under certain conditions, eventually arrive at equal sending rates over a shared link. The conditions were impractical to achieve in reality: for example, the proof assumed that both senders were the same distance from their receiver and thus had the same *latency*, it also assumed that both senders would experience packet losses simultaneously. However, even under common conditions that violated these assumptions the AIMD algorithm could be shown (using Chiu and Jain's approach to analysis) to arrive at predictable sharing outcomes that rarely led to outright *starvation* (where one sender is unable to transmit data altogether). The AIMD approach had several other benefits and hence it was encoded into operating systems first using the Tahoe [33] and later the Reno and NewReno [26] algorithms which became the de facto standard on the Internet for many years.

Although NewReno algorithm had well-known fairness flaws (for example, shorter connections would typically fail to attain their 'fair share' of bandwidth competing with longer-lived connections), its behavior was well-understood and since almost every sender on the Internet used it, there was no concern about 'taking advantage' of NewReno to unfair or anti-competitive ends.

A New Generation of CCAs Leads to Fairness Uncertainty: Today, engineers and researchers are constantly innovating, attempting to develop new CCAs which improve upon the status quo. NewReno, for example, provided poor latency for services such as real time video communication; a wide range of new algorithms have been designed to overcome this limitation. Today, there are 10 Internet-usable [1] CCAs that deploy automatically with Linux; studies find that algorithms BBR and Cubic dominate the landscape of web servers and that Reno no longer reigns [52, 38]. At the same time, many major tech companies are developing in-house and proprietary algorithms, especially in the context of real-time communication and video game streaming [11, 2, 1, 43]. The arrival of new CCAs leads to new challenges in understanding how network bandwidth is divided between senders and the history is littered with pitfalls and failures.

Early proposals for new CCAs were often stymied by issues of co-existence with Reno. A 1994 proposal, called Vegas, promised lower latency than the Tahoe and Reno family of algorithms. But, running alongside these algorithms, Vegas could not attain high throughputs: although Vegas performed well and shared bandwidth equitably when multiple connections using the Vegas algorithm shared a link, Vegas struggled when competing with connections using the Tahoe/Reno/NewReno family of algorithms. When sharing links with these algorithms, Vegas failed to attain good throughput at all. Hence, Vegas was abandoned.

In 2016, Google released a new algorithm called BBR which renewed interest in how heterogeneous CCAs can co-exist on the Internet in part because BBR's earliest version was believed by many to be overly aggressive. Although Google's internal testing showed that BBR, when sharing a network link with a single legacy Reno or Cubic, would often be *more than fair* to the legacy traffic [8], subsequent work discovered that BBR's bandwidth consumption would inevitably become *unfair* as more legacy connections arrived on the shared link. In a published experiment from a Google report to the IETF, one of the Internet standards bodies, Google illustrated a scenario in which a single BBR connection shared a link with a single Cubic connection; BBR attained 40% of the

[1] Yet another set of CCAs are specifically designed for custom settings, such as datacenters.

bandwidth and left 60% to Cubic. Follow-on investigations of the same testing scenario, however, showed that BBR's 40% was inflexible: in the same testbed, with one BBR connection and 16 Cubic connections, the BBR connection continued to consume 40% of the bandwidth, leaving each of the remaining 16 connections with under 4% of the bandwidth each [53, 17]. Google patched BBR and deployed 'BBR 2.0' (and today they are on '3.0') with modifications designed to make BBR friendlier to co-resident connections using other algorithms [7, 10].

Questions Technologists Can't Answer Alone: Since the discovery of BBR's fairness problems, the technical community continues to debate questions of CCA co-existence and fairness. Was BBR 1.0's behavior truly unacceptable? Who decides that a CCA is unacceptable? And, although it seems clear that BBR's fairness outcome was accidental, what should happen if a service provider were to *deliberately* deploy an algorithm that took over more than its fair share of bandwidth in order to have an advantage over their competitors?

In the next few sections, we explore three key questions that we believe researchers in Internet policy should weigh in on:

- What sharing outcomes are ideal in a novel CCA? What sharing outcomes are unacceptable in a novel CCA? (§3)
- Who might care about monitoring and enforcing acceptable CCA deployments? (§4)
- How does an organization demonstrate best-effort intent to avoid unacceptable CCA behavior? (§5)

3 What constitutes unacceptable CCA behavior?

In the previous section, we discussed how bandwidth sharing on the Internet is typically derived from the behavior of algorithms running at senders. Nothing stops a sender from deliberately deploying an algorithm that tries to consume more bandwidth at the expense of other competing senders. However, most operators at least state an intended goal to 'play nicely' on the Internet. Nonetheless, the idea of what construes 'playing nicely' remains unsettled in the community.

There is substantial disagreement about how an 'ideal' algorithm should behave.: There are many definitions of an ideal, 'fair' outcome for bandwidth sharing [35]. The two largest camps in the research community center around *max-min fairness* and *proportional fairness*.

Max-min fairness takes the perspective of a *single* congested link, *i.e.* a link where there are multiple senders competing over bandwidth, with bandwidth c. Each sender has some *demand* d with the first sender having demand d_1, the second, d_2, and so on. The demand represents how much bandwidth each sender needs to consume. In a large file transfer, a sender mind try to consume all of the link capacity c. However, as mentioned previously, many applications do not need to use a lot of bandwidth: video streaming typically tops out under 50Mbps, and hence a sender may only have demand $d = 50Mbps$ and no more. If the senders all have demands $d \geq c$, then max-min fairness allocates each sender $\frac{c}{n}$ throughput, for each of n senders. However if any of the senders use less than $\frac{c}{n}$ throughput, then, the sender x which needs less than its 'fair share' would get its full demand d_x. The remaining bandwidth would then be divided among the other senders who would each get $\frac{c-d_x}{n-1}$ throughput. Another way of thinking of this is that each of the remaining senders would get their original $\frac{c}{n}$, with the leftover bandwidth that x doesn't use being divided evenly among them. This process can be repeated when multiple senders have demand that is less than their 'fair share', with each low-demand sender receiving all the bandwidth they need, and the remainder continuing to be divided equally among higher demand senders.

Max-min fairness is typically the model of fairness that is applied by routers and switches when operators choose not to implement FIFO scheduling and instead enforce some form of fair sharing within the network [46]. TCP Cubic attempts to achieve max-min fairness as its sharing goal [24].

Another camp of researchers however, argues against Max-min fairness and instead supports a model called *proportional fairness* [35]. Unlike max-min fairness, proportional fairness takes a 'network wide' view. It assumes that processing packets at a switch has some implicit *cost* (energy, time, cycles) and hence that connections where packets travel long distances consume more energy/time/compute cycles than connections that travel short distances. Instead of thinking about sharing bandwidth evenly at a bottleneck link, we should think instead about equalizing the overall amount of energy/cycles/time devoted to a given connection over time. Hence, a connection traversing four routers and links should attain roughly $\frac{1}{4}$ the bandwidth as a connection traversing just one router/link.

Proportional fairness has desirable efficiency and game-theoretic properties [35], and TCP Reno and NewReno achieve fairness outcomes that are more similar to proportional fairness [26].

There is general agreement that, even if operators agree upon an ideal, it is unlikely that any algorithm will perfectly achieve ideal fairness.: At the same time that engineers and researchers argue about fairness ideals, there is general agreement that *no congestion control algorithm will ever deliver on these ideals*, especially in an environment with heterogeneous algorithm deployments. For example, although Reno and Cubic, as mentioned above, internally strive towards proportional and max-min fairness respectively, they are overly aggressive when running alongside Vegas (mentioned previously) which ultimately failed to be widely deployed due to its inability to perform well in the presence of competing Reno or Cubic connections.

There exist a few proposals that part from the idea of ideal outcomes and instead focus on minimum standards for deployability.: A few proposals in recent years have instead focused on *minimum standards* for deployment. Rather than focusing on how well new CCAs meet an ideal, we might instead simply ask if they are 'safe enough' for the Internet. The IETF's working group on congestion control has argued in recent years that so long as a new CCA has been tested and demonstrated not to lead to *starvation* in cross-traffic, that this is sufficient for a new algorithm to be considered deployable [18].

Other researchers (many of whom are authors of this work) have argued for a model based on *harm* [51]: that a developers of a new algorithm can measure the slowdown they cause for competing connections, and compare that slowdown to what legacy algorithms already do. So long as the new algorithm is not *more harmful* than what already exists, it can be considered as maintaining the status queue and therefore acceptable to deploy.

At the same time, even these minimum standards can be difficult to meet. If a new algorithm is sometimes more harmful than the status quo – but no more than 10% so – and sometimes less harmful than the status quo – by about 10% or so – does this effect balance out? Or, in a world where we simply expect starvation freedom, is it acceptable if a new algorithm causes starvation, but only in contrived, laboratory experiments that are unlikely to be reflective of any real-world scenario?

Our Conclusions:: Given the challenges of achieving any fairness ideal, and the difficulty of strictly meeting even a minimum deployability standard, the community should look towards *best-effort* attempts by developers to develop deployable algorithms. Best-effort attempts might constitute regular testing with cross-traffic to evaluate harm, starvation, and/or fairness, and dedicating a quantifiable portion of engineering effort to improving these metrics with an understanding that 'edge cases' and error scenarios are likely to occur. An organization should not be judged harshly when a problematic fairness outcome occurs, but might be judged harshly for failure to 'patch' the problem once made aware of it.

4 Who monitors CCA behavior?

Until this point, we have discussed the technical history of CCA co-existence and ongoing debates among engineers for what standard (max-min fairness, proportional fairness, starvation-avoidance, harm-avoidance) to consider in developing CCAs for the Internet. The purpose of this draft, however, is to introduce the challenge to a wider audience of stakeholders beyond CCA developers and engineers. In this section, we explore a few Internet stakeholders who are impacted by the design of CCAs and why they might care about CCA co-existence. We start by discussing groups which already weigh in on the CCA co-existence debate (developers, the Internet operator community) and then discuss groups with increasing distance from the details of CCA development themselves (Competitors, Internet Advocates, Regulators, and Users).

Developers and Service Providers: Since the inception of CCAs in the 1980s, developers of CCAs have been the primary agents involved in the CCA co-existence debate. Developers typically also represent the interests of their employers; the original CCA was developed by Van Jacobsen who was then the employee of a US National Lab, but today CCA developers are more often employees of corporations offering Internet services such as Google, Netflix, or Amazon. Many developers are interested in CCA co-existence out of a pure sense of fairness or equity. Corporate interests are nonetheless aligned with CCA co-existence in many regards. First, corporations may seek CCA fairness out of a desire to *avoid scandal*. Google's BBR deployment brought with it negative press, including in the popular media [49, 3, 48] and others reasonably want to avoid this negative attention. Second, corporations value the detente between competitors over Internet bandwidth. A frank way to say this is, 'I promise to play nicely with my competitor's Internet traffic because it ensures that my competitors will continue to play nicely with my Internet traffic.' Presumably due to these motivations, we often see operators from large companies investing engineering effort and resources into maintaining or improving the Internet status quo, with engineers at Google [10, 22] and Netflix [45] standing out as exemplars in publishing and deploying new initiatives for CCA coexistence.

At the same time, there are significant corporate pressures to ignore or defy CCA co-existence. Ignoring the problem outright is easy to do in an environment in which developers are rewarded for improvements in quality metrics for a service's clients (e.g., improving page load times or video streaming quality) while ignoring co-existence metrics (e.g., the throughput for cross-traffic). In this environment, developers may easily deploy a patch with negative effects for cross-traffic without ever noticing the problem. Furthermore, some operators may simply not care about maintaining any detente with their competitors: the file sharing service Mega, which has its roots in online piracy, was shown to be extremely aggressive (often leaving competing services with less than 10% of their max-min fair share) in experiments performed by our research group [41].

Standards Bodies: Much of the debate about CCA co-existence already takes place in the community of operators at *standards bodies*. Internet standards bodies are organizations with members representing Internet services (e.g. Facebook, Netflix), network infrastructure (e.g., AT&T), hardware manufacturers (e.g. Cisco), researchers (e.g. National labs, universities) and others. These organizations set out standards, rules, and procedures for operating on the Internet, for example, defining the maximum size of an Internet packet or how error messages are transmitted. For the Internet Engineering Task Force [30], the CCA-coexistence debate goes back decades [20, 21, 18]. Other bodies that may have interest in the CCA fairness debate include the International Telecommunications Union (ITU) [29] or the IEEE [27]. For these organizations, their raison d'etre is setting rules and standards for the Internet and hence CCA coexistence naturally falls under their scope.

Competitors: While corporations have interest in ensuring that *their own CCAs* that they develop are capable of co-existence, they perhaps have an even larger interest in knowing that *their competitors* are also developing friendly CCAs. This interest derives from market competition. If Coke and Pepsi each use a proprietary CCA, Coke might perceive it to be unjust that Pepsi users achieve higher quality service than Coke users because Pepsi uses a CCA which is, from

Coke's perspective, overly aggressive. Users might flock to Pepsi because it 'performs better' while Coke's service, which is friendly and built using more responsible design principles, appears to 'perform worse.' More damaging, is if Coke users often share congested networks with Pepsi users: Coke may reasonably be upset if Pepsi's aggressive connections consume more than their fair share of bandwidth, leaving less bandwidth for Coke, perhaps leading to outright accusations that Pepsi's behavior is anti-competitive.

This sort of interaction is purely hypothetical at the moment: we have not seen contention between service providers over CCA co-existence in any public setting. This perhaps stems from the fact that a few large-scale operators dominate Internet usage, and each of the major operators so far has demonstrated some commitment to friendly algorithms. Furthermore, in the US, there is significantly less congestion than in other countries, perhaps leading to fewer opportunities for a contentious environment overall.

Internet Advocacy Organizations: Internet Advocacy Organizations include groups like the Electronic Frontier Foundation in the US [19], Derechos Digitales in Latin American [15], or the Internet Society worldwide [31]. These organizations lobby, sponsor, and advocate for issues of Internet free speech, Internet access, and network neutrality. Through the network neutrality debate, many Internet Advocacy Organizations have made arguments that the Internet should remain a 'fair playing field' for new entrants and that competing services should have equal opportunity to provide good performance to users. These arguments apply equally to the CCA co-existence problem: operators with overly aggressive CCAs should not be able to take advantage of operator with friendly services. As a result, we expect advocacy organizations may also have an interest in understanding and weighing in on the CCA co-existence debate but we hope to hear more from the TPRC community at this workshop.

Regulators: It is unclear to us whether regulators would have an interest in, or should weigh in, on CCA co-existence. This uncertainty is part of what brings us to TPRC. European regulators did weigh in on co-existence during the COVID pandemic, when they asked operators to deliberately reduce sending rates to avoid excess contention during the rapid shift of Internet use from in-office to telework [55, 12]. However, in general, regulators tend to weigh in on issues once they have already become a noticeable problem for users or operators, and as many of the worst-case outcomes (outright deployment of aggressive CCAs, conflict between competing Internet services over bandwidth) remain hypothetical, it may not be the case that CCA co-existence merits intervention at this point in time.

Users: As a final consideration, we reflect on the impact of friendly co-existence for users. At the end of the day, this problem defines whether the Internet's promise of sharing capacity for multiple users, multiple applications, and multiple services actually works in practice. If a home Internet connection is effectively 'used up' by an overly aggressive service, leaving roommates or family members unable to access their services, the Internet is not delivering on this promise. Or, more subtly, if a partner has to ask their spouse to stop watching high-definition television because there are too many stalls and glitches in their simultaneous voice call, the Internet has once again failed to deliver true sharing: one family member is getting perfect Internet service and the lion's share of bandwidth, while the other has an experience of poor quality. It is in every user's interest that the services they access divide resources equitably.

5 How can a developer demonstrate intent to deploy safe CCAs?

We now find ourselves looking towards the horizon, and a day that might bring conflict between interested parties over Internet bandwidth sharing. Should, some day, a regulator, competitor, or advocacy group, accuse an operator of deploying an overly-aggressive CCA, which consumes too much bandwidth at the expense of competing traffic, what should happen? In the civil world, a regulator might pass legislation enabling them to fine the perceived bad actor. An advocacy group might call for a boycott of the offending service. And a competitor might pursue a lawsuit against the aggressive operator. As we discussed in §4, all of these groups have a stake in seeing successful CCA co-existence on the Internet.

However, as we discussed in §3, *establishing that an operator has engaged in wrongdoing for deploying an aggressive CCA is hard.* It is hard for two reasons. First, it is unclear what standard the community should adopt with regard to CCA friendliness: is max-min fairness, proportional fairness, starvation-freedom, or hard-avoidance our goal? Or something else? Second, even with a clear standard and observations of an Internet service which violates this standard, it is unclear what constitutes 'wrongdoing'. Is any violation of the expected behavior wrongdoing? This seems hard to enforce as the outcomes of competing, heterogeneous CCAs are poorly understood and there are likely many scenarios where unfair or harmful outcomes occur even with well-intended designs. Is it only wrongdoing if the operator *intended* to violate the standard, deliberately deploying an algorithm that consumes excess bandwidth to the detriment of others? Or, can an operator be accused of *negligence* – failure to test and evaluate for friendly sharing outcomes, and therefore still responsible when bad but unexpected outcomes occur?

We submit all of the above questions to the TPRC community for discussion. At the same time, we provide a few suggestions to operators who might brace themselves for accusations of aggressive CCA deployment in the future.

We propose **Congestion Safety Audits** as a mechanism for operators to internally manage congestion safety evaluations, and also as a mechanism to *produce evidence* to external entities that the operator is committed to safe congestion control deployments. Congestion safety audits are inspired by computer security standards which do not force operators to guarantee that a security breach will never happen – instead, they require operators to describe in detail their security practices and procedures and to ensure that these practices and procedures are up to date with state of the art expectations [34, 28]. A congestion security audit should contain three core components.

(1) Targeted Ideal and Worst-Case Standards: Developers should choose *two* standards representing the goal or ideal sharing behavior of their CCA, as well as a worst-case lower bound whose violation merits an immediate patch. For example, an operator might develop a CCA which targets implementing proportional fairness. Recognizing that guaranteeing proportional fairness under all conditions, with arbitrary competing traffic, is likely infeasible, the operator specifies simply that this is their *goal* and that they will document efforts towards improving upon it.

The also specify a *lower-bound* for bandwidth sharing, for example, that cross traffic using any of the top-5 published CCAs will not experience less than 25% of their proportional fair share of bandwidth when competing with the new CCA. Once again, the operator does not commit to always guaranteeing that their new CCA will never violate this lower bound: instead, they guarantee that if reproducible conditions are discovered such that this lower-bound is violated, the operator will 'patch' the problem within a specified timeline.

(2) Testing Strategy, Iteration Pace, and Documentation: Operators of Internet services are constantly innovating and fine-tuning their deployments. The code running an individual website may change dozens or hundreds of times a day as developers roll out new content, features, patches, and performance tweaks. This rate of change can introduce unintended consequences for bandwidth sharing and hence we argue that operators should integrate tests for friendly CCA coexistence into their development testing pipelines.

A congestion safety audit should document how this testing occurs. Is the testing in emulation, or on the live Internet? What network capacities, regions of the world, amount of competing cross-traffic, legacy competing CCAs or services are considered? Is the testing integrated into continuous integration services? Is it part of A/B testing on the live Internet? Is it reviewed hourly, daily, weekly?

In addition, operators should maintain records showing the results of these tests over time. In general, one would expect to see improvement – or at least stability – with regard to the target/goal standard over time, rather than movement in the opposite direction. An operation who can demonstrate a neutral or positive trend is unlikely to be accused either of deliberate aggression or of negligence.

(3) Mitigation SLOs: A Service Level Objective (SLO) is a quantifiable goal set out by an operator that can be measured and guaranteed to external users. With a Congestion Safety Audit, operators should define an SLO providing a guarantee for how quickly they will respond should their CCA be found to violate their lower-bound target. For example, an operator might guarantee that the discovery of a scenario where, in the presence of an operator's service, some third-party service using a top-5 popular CCA achieves less than 25% of its proportionally-fair share of bandwidth, would result in the operator patching the problem within 180 days. Once again, the operator might keep logs or records of how long it takes them, on average, to patch these problems when they arise as a demonstration of intent.

Role of Standards Bodies, Advocacy Organizations, and Regulators: The above sketch is just that – a sketch. It does not lay out any community guidelines as to what specific standards are acceptable and which are too lax; it does not provide guidelines about what timeline for mitigation is reasonable; nor about the quality of testing strategies. We invite conversation from the TPRC community and members of these policy and standards-oriented groups towards establishing best practices and norms for congestion safety testing.

6 Discussion and Conclusion

For further discussion, we put forth the following questions:

What standards for fairness, and minimum standards for deployment, should be acceptable? While the research community has invested significant effort in arguing over fairness ideals [51, 56, 5, 6], the debate over what consitutites a minimum standard is more recent and remains under sharp debate. We believe that this latter question of minimum standards is the most important to clarify from a policy perspective, as violating a minimum standard is likely to be the origin of conflict between human entities.

How do we monitor and detect bandwidth sharing issues? Gaining insight into whether and how severely bandwidth sharing challenges arise is key to understanding the scope of the problem. Many operators complain of hotly-contended, congested network links in the 'core' of the Internet [16] (for example, ongoing conflict between Deutsche Telekom and Meta [44]) and yet there are no measurements of how many connections traverse these links or how equitably bandwidth is shared in practice, in part because outside research agencies have no ability to measure them. Home networks are another place where congestion is hypothesized to occur, but in wealthy countries where broadband access is high these networks may simply have enough capacity to go around (if a home network connection offers 300Mbps and a 4K streaming video requires only 50Mbps, a family of four may never observe any problems). To develop robust measurements, we likely need a coalition of service operators to provide access to the necessary data and infrastructure.

How should the community respond in the presence of unacceptable behavior? Even with data and standards, it is unclear what recourse stakeholders have to demand improvements in the face of poorly-designed CCAs and Internet services. Is enforcement of responsible CCA deployment simply a game of 'naming and shaming' bad actors? Should legislation be involved? Or, should network operators (e.g., AT&T or Verizon) step in and implement routers and switches that explicitly force the network to implement max-min fair sharing universally?

7 Further Reading

The technical literature on bandwidth sharing is vast. Because this whitepaper is intended for new audiences, we highlight introductory material below and forgo any attempt to be comprehensive.

For Discussion of Congestion Control Deployability: The Internet Engineering Task Force (IETF) is the standards body where most of the debate over congestion control deployment plays out. The document RFC 9743 documents existing guidelines for congestion control deployment [18]. An older document by congestion control luminary Sally Floyd provides an easy to read discussion of the principles behind congestion control deployment [21].

For a Textbook on the Computer Science of Congestion Control: Peterson, Brakmo, and Davie provide a textbook giving a general introduction to congestion control in their book 'TCP Congestion Control: A Systems Approach' and it is available online for free [40]. The book targets students of computer science with a basic understanding of systems and algorithm design.

For a Tutorial on the Mathematics of Congestion Control: Le Boudec provides a very nice and brief tutorial focusing on fairness outcomes titled 'Rate adaptation, Congestion Control and Fairness: A Tutorial'. It is also available online for free, and presumes some understanding of queueing theory and game theory [4].

For a Deep Dive into Recent Research: Ranysha Ware and Ayush Mishra are two recent PhDs whose dissertations both received awards for contributions to congestion control deployability. Their dissertations document their own research, while also including introductory material and related work sections that nicely illuminate the history of the space. [50, 37]

References

[1] Amazon Web Services. *What is Amazon IVS Low-Latency Streaming?* `https://docs.aws.amazon.com/ivs/latest/LowLatencyUserGuide/what-is.html`. 2025.

[2] Apple Inc. *Enabling Low-Latency HTTP Live Streaming (HLS)*. `https://developer.apple.com/documentation/http-live-streaming/enabling-low-latency-http-live-streaming-hls`. 2020.

[3] Karl Bode. "Google's Network Congestion Algorithm Isn't Fair, Researchers Say". In: *Vice* (Oct. 2019). URL: `https://www.vice.com/en/article/googles-network-congestion-algorithm-isnt-fair-researchers-say`.

[4] Jean-Yves Le Boudec. *Rate Adaptation, Congestion Control and Fairness: A Tutorial*. Tech. rep. École Polytechnique Fédérale de Lausanne (EPFL), Dec. 2000. URL: `https://personal.ie.cuhk.edu.hk/~dmchiu/leb_tutorial.pdf`.

[5] Bob Briscoe. *Flow Rate Fairness: Dismantling a Religion*. Internet-Draft (Informational) draft-briscoe-tsvarea-fair-02. Internet Engineering Task Force, July 2007. URL: `https://www.bobbriscoe.net/projects/2020comms/refb/draft-briscoe-tsvarea-fair-02.pdf`.

[6] Lloyd Brown et al. "On the Future of Congestion Control for the Public Internet". In: *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*. HotNets '20. Virtual Event, USA: Association for Computing Machinery, 2020, pp. 30–37. ISBN: 9781450381451. DOI: `10.1145/3422604.3425939`. URL: `https://doi.org/10.1145/3422604.3425939`.

[7] Neal Cardwell, Ian Swett, and Joseph Beshay. *BBR Congestion Control (BBRv3)*. Internet-Draft (Experimental). IETF CCWG (IETF 119 meeting), Mar. 2024. URL: `https://datatracker.ietf.org/doc/draft-cardwell-ccwg-bbr-00/`.

[8] Neal Cardwell et al. "BBR: Congestion-Based Congestion Control: Measuring bottleneck bandwidth and round-trip propagation time". In: 14.5 (Oct. 2016), pp. 20–53. ISSN: 1542-7730. DOI: `10.1145/3012426.3022184`. URL: `https://doi.org/10.1145/3012426.3022184`.

[9] Neal Cardwell et al. *BBRv2: A Model-based Congestion Control*. Internet-Draft (Expired). IETF ICCRG (IETF 104 meeting), Apr. 2019. URL: `https://datatracker.ietf.org/doc/draft-cardwell-iccrg-bbr-congestion-control-02/`.

[10] Neal Cardwell et al. *BBRv3: Algorithm Bug Fixes and Public Internet Deployment*. `https://datatracker.ietf.org/meeting/117/materials/slides-117-ccwg-bbrv3-algorithm-bug-fixes-and-public-internet-deployment-00`. July 2023.

[11] Gaetano Carlucci et al. "Analysis and design of the google congestion control for web real-time communication (WebRTC)". In: *Proceedings of the 7th International Conference on Multimedia Systems*. MMSys '16. Klagenfurt, Austria: Association for Computing Machinery, 2016. ISBN: 9781450342971. DOI: `10.1145/2910017.2910605`. URL: `https://doi.org/10.1145/2910017.2910605`.

[12] Foo Yun Chee. "EU wants streaming platforms to ease internet gridlock amid virus crisis". In: (Mar. 18, 2020). URL: `https://www.reuters.com/article/technology/eu-wants-streaming-platforms-to-ease-internet-gridlock-amid-virus-crisis-idUSKBN2153RV/` (visited on 08/04/2025).

[13] Dah-Ming Chiu and Raj Jain. "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks". In: *Computer Networks and ISDN systems* 17.1 (1989), pp. 1–14.

[14] Jon Crowcroft. "Net neutrality: the technical side of the debate: a white paper". In: *SIGCOMM Comput. Commun. Rev.* 37.1 (Jan. 2007), pp. 49–56. ISSN: 0146-4833. DOI: `10.1145/1198255.1198263`. URL: `https://doi.org/10.1145/1198255.1198263`.

[15] Derechos Digitales. *Derechos Digitales: Defending Human Rights in the Digital Environment*. https://www.derechosdigitales.org/. 2025.

[16] Amogh Dhamdhere et al. "Inferring persistent interdomain congestion". In: *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. SIGCOMM '18. Budapest, Hungary: Association for Computing Machinery, 2018, pp. 1–15. ISBN: 9781450355674. DOI: 10.1145/3230543.3230549. URL: https://doi.org/10.1145/3230543.3230549.

[17] Mo Dong et al. "PCC Vivace: Online-Learning Congestion Control". In: *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. Renton, WA: USENIX Association, 2018, pp. 343–356. ISBN: 978-1-931971-43-0. URL: https://www.usenix.org/conference/nsdi18/presentation/dong.

[18] Martin Duke and Gorry Fairhurst. *Specifying New Congestion Control Algorithms*. Best Current Practice RFC 9743. Internet Engineering Task Force, Mar. 2025. URL: https://datatracker.ietf.org/doc/rfc9743/.

[19] Electronic Frontier Foundation. *Electonic Frontier Foundation (EFF)*. https://www.eff.org/. 2025.

[20] Sally Floyd. *Metrics for the Evaluation of Congestion Control Mechanisms*. RFC 5166. Mar. 2008. DOI: 10.17487/RFC5166. URL: https://www.rfc-editor.org/info/rfc5166.

[21] Sally Floyd and Van Jacobson. *Congestion Control Principles*. Informational RFC 2914. Internet Engineering Task Force, Sept. 2000. URL: https://datatracker.ietf.org/doc/rfc2914/.

[22] Monia Ghobadi et al. "Trickle: Rate Limiting YouTube Video Streaming". In: *Proceedings of the 2012 USENIX Annual Technical Conference (ATC '12)*. USENIX Association, 2012, pp. 191–196. URL: https://www.usenix.org/conference/atc12/technical-sessions/presentation/ghobadi.

[23] Google Research. *Google Academic Research Awards*. 2025. URL: https://research.google/programs-and-events/google-academic-research-awards/.

[24] Sangtae Ha, Injong Rhee, and Lisong Xu. "CUBIC: A New TCP-friendly High-speed TCP Variant". In: *SIGOPS Oper. Syst. Rev.* 42.5 (July 2008), pp. 64–74. ISSN: 0163-5980. DOI: 10.1145/1400097.1400105. URL: http://doi.acm.org/10.1145/1400097.1400105.

[25] *HBO Max loads slow, buffers, or pauses*. Max Help Center. URL: https://help.hbomax.com/do-en/answer/detail/000002519 (visited on 08/04/2025).

[26] Janey C. Hoe. "Improving the start-up behavior of a congestion control scheme for TCP". In: *Conference Proceedings on Applications, Technologies, Architectures, and Protocols for Computer Communications*. SIGCOMM '96. Palo Alto, California, USA: Association for Computing Machinery, 1996, pp. 270–280. DOI: 10.1145/248156.248180. URL: https://doi.org/10.1145/248156.248180.

[27] Institute of Electrical and Electronics Engineers. *Institute of Electrical and Electronics Engineers (IEEE)*. `https://www.ieee.org/`. 2025.

[28] International Organization for Standardization. *International Organization for Standardization (ISO)*. `https://www.iso.org/`. 2025.

[29] International Telecommunications Union. *International Telecommunications Union (ITU)*. `https://www.itu.int/`. 2025.

[30] Internet Engineering Task Force. *Internet Engineering Task Force (IETF)*. `https://www.ietf.org/`. 2025.

[31] Internet Society. *Internet Society: The Internet Is for Everyone*. `https://www.internetsociety.org/`. 2025.

[32] *Internet speed recommendations*. Disney+ Help Center. URL: `https://help.disneyplus.com/article/disneyplus-recommended-speeds` (visited on 08/04/2025).

[33] Van Jacobson. "Congestion avoidance and control". In: *Proceedings of the ACM SIGCOMM '88 Conference on Communications Architectures and Protocols*. ACM, 1988, pp. 314–329. DOI: `10.1145/52324.52356`. URL: `https://doi.org/10.1145/52324.52356`.

[34] Joint Task Force Transformation Initiative. *Security and Privacy Controls for Information Systems and Organizations*. NIST Special Publication 800-53, Revision 5. National Institute of Standards and Technology, Sept. 2020. DOI: `10.6028/NIST.SP.800-53r5`. URL: `https://doi.org/10.6028/NIST.SP.800-53r5`.

[35] Jean-Yves Leboudec, Patrick Thiran, and Network Calculus. *A theory of deterministic queuing systems for the Internet*. LNCS, 2001.

[36] Zili Meng et al. *Confucius Queue Management: Be Fair But Not Too Fast*. arXiv ePrint 231.18030. 2023. arXiv: `2310.18030 [cs.NI]`.

[37] Ayush Mishra. "Understanding the Modern Internet's Heterogeneous Congestion Control Landscape". PhD thesis. National University of Singapore, Sept. 2024. URL: `https://www.comp.nus.edu.sg/~bleong/publications/phd-ayush.pdf`.

[38] Ayush Mishra et al. "Keeping an Eye on Congestion Control in the Wild with Nebby". In: *Proceedings of the ACM SIGCOMM 2024 Conference*. ACM SIGCOMM '24. Sydney, NSW, Australia: Association for Computing Machinery, 2024, pp. 136–150. DOI: `10.1145/3651890.3672223`. URL: `https://doi.org/10.1145/3651890.3672223`.

[39] *Netflix-recommended internet speeds*. Netflix Help Center. URL: `https://help.netflix.com/en/node/306` (visited on 08/04/2025).

[40] Larry L. Peterson, Lawrence Brakmo, and Bruce Davie. *TCP Congestion Control: A Systems Approach*. Systems Approach Series, 2021. URL: `https://tcpcc.systemsapproach.org`.

[41] Adithya Abraham Philip et al. "Prudentia: Findings of an Internet Fairness Watchdog". In: *Proceedings of the 2024 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*. Sydney, Australia: ACM, 2024.

[42] Adithya Abraham Philip et al. "Revisiting TCP Congestion Control Throughput Models & Fairness Properties At Scale". In: *Proceedings of the 2021 ACM Internet Measurement Conference (IMC)*. IMC '21. Virtual: ACM, 2021.

[43] Devdeep Ray et al. *SQP: Congestion Control for Low-Latency Interactive Video Streaming*. 2022. arXiv: `2207.11857 [cs.NI]`. URL: `https://arxiv.org/abs/2207.11857`.

[44] Dan Rayburn. *What Many Are Getting Wrong in the Meta and Deutsche Telekom Peering Dispute*. Streaming Media Blog. Sept. 26, 2024. URL: `https://www.streamingmediablog.com/2024/09/meta-dt-peering-dispute.html` (visited on 08/05/2025).

[45] Bruce Spang et al. "Sammy: smoothing video traffic to be a friendly internet neighbor". In: *Proceedings of the 2023 ACM SIGCOMM Conference*. ACM, 2023, XX–YY. DOI: `10.1145/3603269.3604839`. URL: `https://doi.org/10.1145/3603269.3604839`.

[46] Ion Stoica, Scott Shenker, and Hui Zhang. "Core-stateless fair queueing: achieving approximately fair bandwidth allocations in high speed networks". In: *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*. SIGCOMM '98. Vancouver, British Columbia, Canada: Association for Computing Machinery, 1998, pp. 118–130. ISBN: 1581130031. DOI: `10.1145/285237.285273`. URL: `https://doi.org/10.1145/285237.285273`.

[47] *System requirements & supported devices for YouTube*. YouTube Help. URL: `https://support.google.com/youtube/answer/78358?hl=en` (visited on 08/04/2025).

[48] James Titcomb. "Google Algorithm 'Hogs' Internet Traffic, Researchers Show". In: *The Telegraph* (Oct. 2019). URL: `https://www.telegraph.co.uk/technology/2019/10/27/google-algorithm-hogs-internet-traffic-researchers-show/`.

[49] Daniel Tkacik. "CMU Researchers Find Google's New Congestion Control Algorithm Treats Data Unfairly". In: *Carnegie Mellon University News* (Oct. 2019). URL: `https://www.cmu.edu/news/2019/cmu-researchers-find-googles-new-congestion-control-algorithm-treats-data-unfairly`.

[50] Ranysha Ware. "Battle for Bandwidth: On the Deployability of New Congestion Control Algorithms". PhD thesis. Carnegie Mellon University, Feb. 2025. URL: `https://kilthub.cmu.edu/articles/thesis/Battle_for_Bandwidth_On_the_Deployability_of_New_Congestion_Control_Algorithms/29241941`.

[51] Ranysha Ware et al. "Beyond Jain's Fairness Index: Setting the Bar For The Deployment of Congestion Control Algorithms". In: *Proceedings of the Eighteenth ACM Workshop on Hot Topics in Networks* (*HotNets*). Princeton, New Jersey: ACM, 2019.

[52] Ranysha Ware et al. "CCAnalyzer: An Efficient and Nearly-Passive Congestion Control Classifier". In: *Proceedings of the 2024 Conference of the ACM Special Interest Group on Data Communication* (*SIGCOMM*). Sydney, Australia: ACM, 2024.

[53] Ranysha Ware et al. "Modeling BBR's Interactions with Loss-Based Congestion Control". In: *Proceedings of the 2019 ACM Internet Measurement Conference* (*IMC*). IMC '19. Amsterdam, Netherlands: ACM, 2019.

[54] Wikipedia contributors. *Network Neutrality — Wikipedia, The Free Encyclopedia*. 2025. URL: `https://en.wikipedia.org/wiki/Net_neutrality`.

[55] "YouTube, Amazon and Netflix cut picture quality in Europe". In: *Financial Times* (Mar. 20, 2020). URL: `https://www.ft.com/content/70333747-f180-4887-8a26-27ab6b230299` (visited on 08/04/2025).

[56] Adrian Zapletal and Fernando Kuipers. "Slowdown as a Metric for Congestion Control Fairness". In: *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks*. HotNets '23. Cambridge, MA, USA: Association for Computing Machinery, 2023, pp. 205–212. DOI: `10.1145/3626111.3628185`. URL: `https://doi.org/10.1145/3626111.3628185`.