Fairness Measurement Procedure for the Evaluation of Congestion Control Algorithms

Romuald Corbel, Arnaud Braud, Xavier Marjou, Gwendal Simon, Annie Gravey

Fairness Measurement Procedure for the Evaluation of Congestion Control Algorithms

Romuald Corbel Arnaud Braud Xavier Marjou
Orange
Lannion, France
firstname.lastname@orange.com

Gwendal Simon Annie Gravey
IMT Atlantique
Bretagne, France
firstname.lastname@imt-atlantique.fr

***Abstract*—Congestion is one of the most critical issues impacting the performance of Internet networks, hence the need of Congestion Control Algorithms (CCAs) to either prevent or remove it. Nevertheless, CCAs may affect the network fairness given that the transport behavior can drastically change in function of the CCA (e.g., Performance-oriented Congestion Control (PCC), Bottleneck Bandwidth and Round-trip propagation time (BBR), Reno) used in the endpoints.**

New transport protocols such as Quick UDP Internet Connections (QUIC) allow to easily customize their CCA, which increases the types of CCAs on the Internet. Evaluating the fairness among competing CCAs is thus essential. In this work, we focus on determining an impartial Fairness Measurement Procedure (FMP) capable of assessing *equity* when several CCAs compete in a bottleneck during a given period of time. In order to validate the proposed mechanism, we perform an exhaustive evaluation of various competing CCAs within multiple network configurations while varying latency, packet loss, queuing policy and buffer size. Results show that the grade of fairness achieved by two different CCAs depends on the network configuration. This claim confirms the relevance of the proposed fairness measurement procedure.

***Index Terms*—fairness, congestion control, QUIC.**

I. Introduction

Since the rise of the Internet, researchers have developed protocols and algorithms to increase the overall network efficiency while preserving fairness among concurrent users or applications and preventing network collapses. In this work we are especially interested in the Congestion Control Algorithms (CCAs), which are implemented in the transport layer of the Internet protocol stack. The most popular CCAs, namely RENO [1] and CUBIC [2], identify congestions by means of packet loss detection. These algorithms enable data rate adjustment among concurrent network flows; however packet-loss-based CCAs do not meet the latency requirements of the incoming network services such as real-time applications, low latency slices in 5G mobile, virtual reality, among others.

Recent research efforts have yielded new CCAs, in particular Bottleneck Bandwidth and Round-trip propagation time (BBR) [3], [4] and Performance-oriented Congestion Control (PCC) [5]. Both proposals aim to improve the network performance by maximizing both network data rate and user *satisfaction* BBR and PCC differ from RENO and CUBIC by no longer relying on loss detection. BBR is based on latency evolution whereas PCC is based on flow prediction.

Another factor of changes in the implementation of CCAs is the development of new transport protocols, especially Quick UDP Internet Connections (QUIC) [6], which has been recently deployed by Google. In QUIC (based on User Datagram Protocol (UDP)), the CCA is implemented in the user space within applications, for instance in Internet browsers as Chrome or in web servers as those of YouTube. This fact facilitates the diversification of CCAs on Internet and enables their continuous updating unlike traditional CCAs, which are implemented in the kernel of the Operating System (OS) at both client and server sides. Thus, updating and upgrading kernel-based CCAs as TCP CUBIC and TCP RENO require patching a significant amount of devices.

When the network deals with a wide variety of transport protocols and notably CCAs, the risk is that the fair sharing of network resources is jeopardized due to insufficiently tested CCAs. This claim is a major concern for network operators. Moreover, recent studies such as those presented in [7] reveal that the performance of CCAs does not always match what is announced by their designers. For instance, a content provider that implements a CCA tailored to its particular needs cannot be certain of the obtained results. It is then required to develop appropriate testing and validation mechanisms of incoming CCAs, notably when evaluating their behavior in heterogeneous environments where various CCAs run concurrently.

When implementing new CCAs, the best practice is to check whether new algorithms compete fairly with traditional standardized CCAs. The challenge is that there exists a wide range of Internet configurations which need to be taken into account during evaluations. Yan et al. in [7] have recently proposed a worldwide platform (namely Pantheon) for testing the performance of CCAs and protocols over the Internet. Pantheon employs various client and server machines deployed around the world to measure the performance of various CCAs (BBR, CUBIC, PCC, among others) while capturing measurements for more than one year. Results show that the performance of CCAs can *dramatically* vary as a function of the network configuration, i.e., network path, bottleneck buffers sizes (notably when considering the queuing strategy), and traffic variations.

In the interest of validating new CCAs a first estimation

of their *friendliness*[1] need to be obtained by means of local test platforms. In this line of research, the Real-time Transport Protocol Media Congestion Avoidance Techniques (RMCAT) initiative [8], [9] proposes several methods for network fairness assessment through a limited number of representative network scenarios. However, this initiative is limited to video streaming use-cases. The main drawback is that RMCAT measures the network equity at a given instant of time although the fairness must be evaluated over the entire session. Indeed, the beginning of sessions can easily become more relevant than the end of them notably when considering web services and video streaming use-cases where the Quality of user Experience (QoE) is mainly influenced by the first elements [10], [11].

We propose in this paper a generic impartial Fairness Measurement Procedure (FMP), in order to evaluate the *session-level fairness* between two network endpoints while considering a broad range of wire-line network configurations representing real scenarios. The proposed FMP is based on the Jain's index (a quantitative measure of *fairness* proposed in [12]) and takes into account the data rate evolution over time for a given session. To be more specific, the proposed fairness assessment metric meets the need of determining whether incoming CCAs preserve the fair network performance while paying special attention at the beginning of sessions. The proposed FMP is validated by emulation while performing an exhaustive evaluation of network configurations. We evaluate three CCAs: RENO and the two (supposedly more aggressive) new CCAs BBR and PCC.

This paper is organized as follows: Section II presents a general background and related works. Section III provides a formal definition of fairness. Section IV introduces the proposed metric in order to evaluate CCA fairness. Section V describes the test-bed platform used for the emulation of the various network configurations. The fairness evaluation of the aforementioned CCAs (i.e., RENO, BBR, PCC) is discussed in Section VI. Conclusions are finally presented in Section VII.

II. Background and Related Work

A. On Fairness

Fairness has strong importance for network operators, who aim at ensuring that each user receives a fair share of network resources, in particular a fair share of the bandwidth. In an unfair flows, one application in particular gets more data to the detriment of other applications. This detrimental behavior can result in unsatisfaction for the users. While being a key requirement to meet, fairness has also been a complex object of research for years. Thus, multiple definitions have been proposed by researchers to capture a different characteristic of the problem [13]. The *max-min* fairness in particular compares multiple competing flows sharing a bottleneck in the network under three principles: (i) increasing the data-rate of one flow results in a decrease in the data-rate of another flow; (ii) each flow has a saturation point in its path; and (iii) each flow obtains the best performance with respect to its requirements.

To the best of our knowledge, the existing definitions of fairness do not capture one element that has become an essential development driver for service provider: the notion of session. Today's connections between a client and a server are long sessions where not all data have the same importance. Thus, service providers can develop strategies where a flow (session) preempts more resources at certain time of the connection, resulting in transient unfairness. As we will discuss in Section III, the fairness should neither be measured at the end of a session, not be measured on a small fraction of its duration. The researchers miss a definition that could take into account that the competition between flows should be considered at the level of sessions.

B. Fairness Metrics

Metrics determine whether CCA results in a fair share of network resources. The Internet Engineering Task Force (IETF) introduces three such metrics [14], for n competing flows, each flow i receiving T_i data.

Jain Index [12]. Let O_i be the max-min optimal bit-rate (defined for example with the algorithm of progressive filling [13]). The Jain's index is defined as

$$J(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2} \tag{1}$$

with $x_i = T_i/O_i$. Jain's index ranges from $\frac{1}{n}$ to 1, which allows comparison between network management policies regardless of the bottleneck bandwidth.

Product measure [15]. The product of the data-rates is maximized when all connections receive the same bit-rate.

$$P = \prod_{i \in \{1,\ldots,n\}} T_i \tag{2}$$

The maximum value, $\left(\frac{B}{n}\right)^n$ depends on the link bandwidth B (bottleneck). Hence, the fairness comparison is only possible when network links have similar bandwidth configurations.

Epsilon-fairness [16]. It enables defining a binary metric, whether a network management policy is fair or is not. The boundary between fair and unfair is determined by a variable ϵ.

$$\frac{\min_{i \in \{1,\ldots,n\}} T_i}{\max_{i \in \{1,\ldots,n\}} T_i} \leq 1 - \epsilon \tag{3}$$

The higher is ϵ, the stricter to fair is the ratio. This metric focuses on extreme flows.

These metrics do not respect two key fairness requirements: (i) they do not evaluate fairness evolution over the duration of a session and (ii) they do not take into account the data-rate fluctuations within the session.

[1]The term *friendliness* refers to the capacity of new CCAs of coexisting with others without disrupting them notably in terms of fairness.

C. Fairness Evaluation for CCAs

CCA fairness is usually evaluated either by testing it on the Internet, by using a testbed for emulating the real network behavior, or even by simulation. A recent solution called Pantheon [7] presents a new performance evaluation framework of CCAs which is able to run on both Internet and a pre-configured testbed. Pantheon aims to replace platforms like Planet-Lab [17], Emulab [18], and ORBIT [19] for CCAs performance evaluation.

In general, Pantheon enables the performance evaluation of protocol and CCAs under multiple network configurations. In [7], authors reveals that the performance of transport protocols notably depends on (i) the network path and (ii) the time. Hence, evaluating CCA performance on Internet does not enable reliable results. In fact, the use of testbeds and simulations with well-defined path features is mandatory to obtain valuable fairness measurements [20].

III. Fairness Analysis in Network Sessions

Online services are no longer restricted to only data download. Instead, the interactions with users require complex network exchanges throughout the duration of sessions. A sudden drop of performance during a session can have harmful repercussions for the user (for example service interruptions, video re-buffering, and misbehaviors). Meanwhile, service providers try to get more resources at some time of a session because a sudden increase of data-rate can be suitable during the session (for example fast-video playing delay and key web resources download). In this context, network operators aims at guarantying a fair access to the network resources with respect to all competing flows at any time.

In order to evaluate the session-level fairness during the time, we introduce the notion of *slots* to split the session into small temporal periods of time (typically 100 ms). The idea of the *session-level fairness* is that every flow gets a fair sharing of the resources throughout the full duration of the sessions.

Fairness Definition. We consider that the network behavior is fair at the session-level of a given flow when it is friendly with all other competing flows while the totality of slots are separately analyzed. In other words, a session that would obtain much more resources than other flows for some slots but compensate them by getting less resources for some other slots is not fair since this behavior may jeopardize the access to the services for the competing flows (even if, overall, the equity is respected). The type of session is also important when evaluating fairness. In fact, the type of session determines the network requirements in order to optimize the QoE of users. Some examples of network services according to the network features are given below [21], [22]:

- **Constant data-rate over time**: real-time applications
- **More importance at the beginning of a session**: video streaming, web pages
- **Low latency and reduced packet loss**: autonomous cars, critical virtual reality applications (medicine)
- **Non-critical**: IoT, download software, OS update

In this paper, we focus on the services where the QoE is highly impacted by the amount of data received at the beginning of the session. This is typically the case for online video services and web pages, which represent respectively 73 % and 18 % of the overall Internet traffic [23]. For the traffic related to web pages, several studies have shown that 40 % of people leave a website if they have to wait more than 3 s to get the first information [24]. To reduce the display time, browsers can use multiple Transmission Control Protocol (TCP) connections to make multiple Hypertext Transfer Protocol (HTTP) requests simultaneously (for example, Firefox and Chrome use up to maximum 6 TCP connections simultaneously). For the traffic related to video streaming, other studies have revealed that the abandonment rate rises quickly after more than two seconds spent at waiting for start of the video (5.8 % more abandon per second after the first two seconds). Then, after these critical first slots, the network flows for web pages and for online video services do not require as much bandwidth as in the earliest seconds. Typically in online video, once the buffer is filled, the video rate adaptation mechanisms can accommodate slight reduction of throughput [25].

Due to the need for more data in the earliest time, the service providers could implement a CCA, which is especially aggressive at the earliest instant of the session, and which is thus more aggressive than the other competing flows. On the contrary, network operators are concerned by maintaining a fair sharing of the bottleneck throughout the session and especially in the first slots of the session. In this paper, we address this problem by introducing a novel fairness metric, as well as a full procedure for assessing the fairness when several independent sessions sharing the same bottleneck start concurrently.

IV. Fairness Measurement Procedure

We introduce a new metric called Fairness Measurement Procedure (FMP) to evaluate the fairness at the level of a whole session. The main idea behind FMP is splitting sessions into multiple slots, to measure the fairness during each slot, and to aggregate all measures at the end of the session in order to compute a global session-level fairness. We concretely aim at assessing a session-level fairness for a set of competing sessions. We represent in Figure 1 an illustration of our proposal for two competing flows sharing the same bottleneck.

A. FMP definition

Lets consider two competing flows: A and B, where Δ is the time during which both flows compete for the network resources. We split the competition time (namely, the session) into S temporal slots. In our experiments, we set S so that the duration of a slot $\frac{\Delta}{S}$ is 100 ms. For each time slot $i, 1 \leq i \leq S$, we collect two measures:

- The fairness J_i during i-th slot, using any of the aforementioned metric. In our case, we use the Jain index j_i because it is a well-adopted bounded measure. Hence,

we have J_i is equal to $1 - j_i$ so that 0 represents a fair network and $\frac{1}{2}$ is the worst case (unfair).

- The dominant flow d_i, which indicates which flow receives the greatest amount of data during slot i. We have:

$$d_i = \begin{cases} -1, & \text{if } A \text{ is dominant} \\ 1, & \text{if } B \text{ is dominant} \end{cases} \quad (4)$$

Then, based on the collected measures $J = \{J_1, \ldots, J_S\}$ and $D = \{d_1, \ldots, d_S\}$, the generic definition of a FMP is:

$$K = f(J, D)$$

In this work, we focus on the fact that all slots are not equal, i.e., some slots have a greater importance than others during a competition between two given flows. As discussed in Section III, various scenarios can be studied. Here, we are interested in the case where both flows compete at the beginning of the session by an overly aggressive policy. Hence, we study in this paper a particular function $f(\cdot)$ where more importance is given to the beginning of the session:

$$K = \sum_{i \in \{1,\ldots,S\}} d_i \alpha_i J_i$$

where α_i is the weight of the i-th slot. Since the earliest slots are more important than the latest slots, we define α_i as a strictly decreasing function of i. Thus, the FMP is a weighted sum of fairness measurements, where, depending on the variation of dominant flow, the effect of transient unfairness can reinforce or diminish (with a decreasing impact) the overall result.

B. FMP analysis

To illustrate the behavior of the FMP, we discuss in the following three typical scenarios of competition between two flows sharing a bottleneck of 1 Mbps in the network. These scenarios are shown in Figure 1. For each typical flow competition, we represent three sub-figures: the measured data-rate, the variation of K over time (new procedure), and finally the averaged value of the Jain's index collected over whole the session (existing procedure). The cases are:

Figure 1a. A fair share of the bottleneck. The value of K is 0 over the all session. The values of the averaged Jain's index converge toward 1.

Figure 1b. An unfair share of the bottleneck. The K values quickly decrease toward 0.5. At the end, we obtain $K = 0.57$, which indicates the domination of a flow. The cumulative average of Jain's index behaves similarly. The final Jain's index is 0.56 (which is close to the final K value).

Figure 1c. An alternate domination of flows. At the beginning, the flow A is dominant, so the values of K after 40 s show a very unfair network share, with a value close to –0.5. Then, the flow B becomes dominant. Here, the K values increase toward 0, which reflects that, after 80 s, both flows have received a fairly good amount of data (with a slight advantage for flow A because it was the first one to be served).

On the contrary, the averaged Jain's index notices only unfair sharing measures but, due to its blindness of the dominant flow, it fails in reflecting the dominance variation between A and B.

The proposed FMP is thus able to identify when one flow dominates over another, and particularly is able to detect the alternation of domination. Furthermore, it is able to put priority to some slots such as in the first seconds of the session by means of weights. In this proposal, FMP is designed for a competition between two flows. In the case of competition between multiple flows, in particular three flows as we will see later, we compute the FMP for every pair of flows and we take the worst K value.

V. Test-Bed Settings

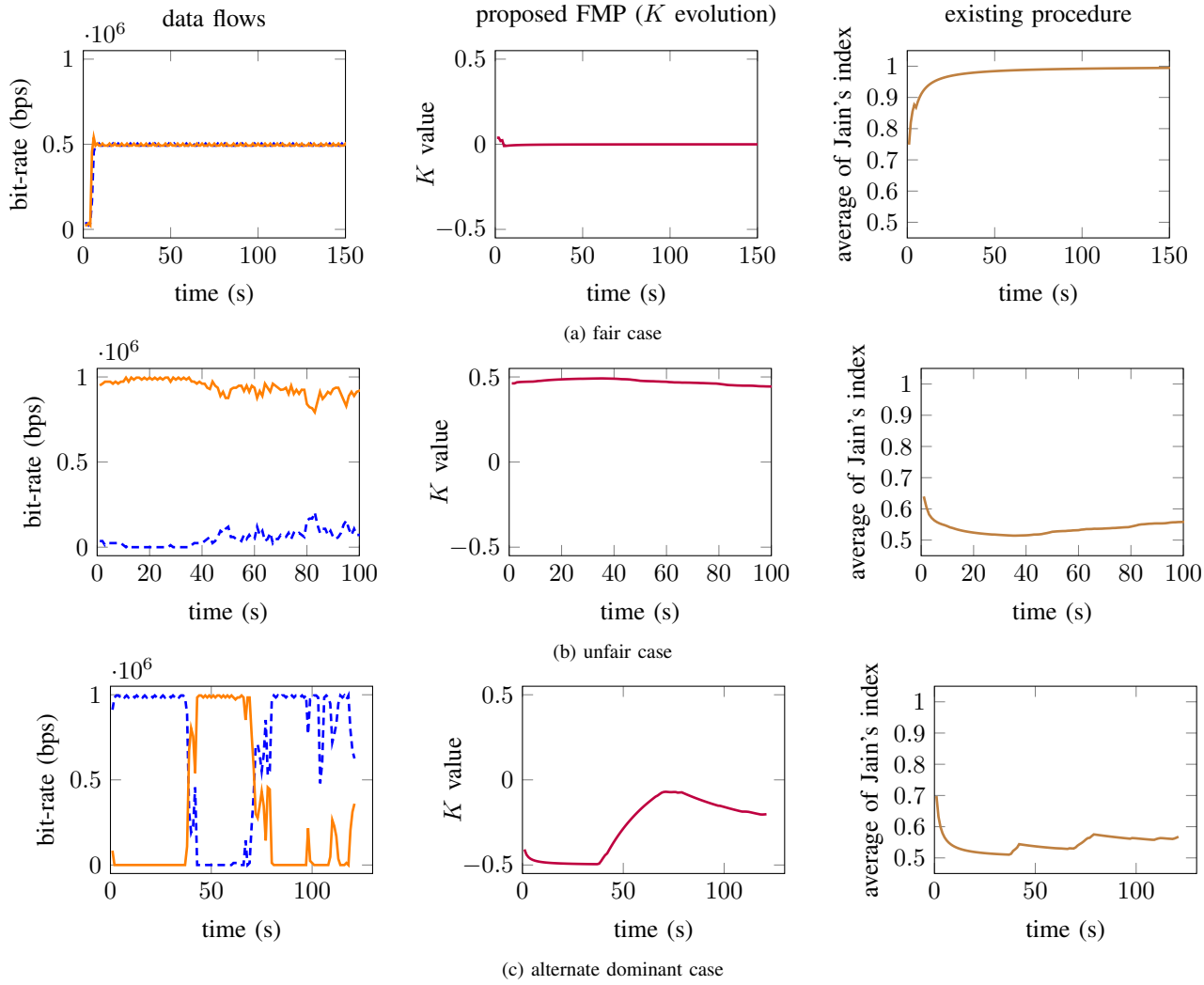
Testing the fairness of competing CCAs in a laboratory requires the implementation of a test-bed for emulating real client-server communications under various network scenarios. For this purpose, we have implemented a test-bed whose architecture is illustrated in Figure 2.

Both client and server OSs are based on Linux (Debian for the client and Ubuntu for the server). The configuration of the network settings is based on the Linux traffic control *tc* program, which enables changing the traffic control configuration. In our case, *tc* is used to set a customized latency, the error rate of the link, the bottleneck queuing policy, and the buffer size.

The *flow configuration* enables changing the characteristics of each data flow (latency, error rate, and bit rate) before the bottleneck. The network configuration is done before the traffic generation. Then, the traffic generation (file delivery) concretely modifies the size of files, the current number of connections on the network link, and the type of CCAs. Since, each CCA is configured in the OS and we evaluate two CCAs, we need two different endpoints. We then use two virtual machines for the traffic generation. Note that, they are not part of the bottleneck emulation, which is done on a real network card.

This platform evaluates different CCAs, which are meant to avoid congestion collapse on the network. Most CCAs fall into three main categories: loss-based, delay-based and bit rate prediction-based. In order to obtain valuable and no skewed results, we have selected one representative CCA per category, namely RENO, BBR, and PCC and tested them on a wide range of network configurations.

- *Loss based CCA*: the CCA detects a congestion on the network by the observation of a packet loss. The popular RENO algorithm manages congestion by two different mechanisms. The first one is based on detecting isolated packet loss. Upon detecting a missing packet, the receiver notifies it with three Acknowledgment messages (ACKs) to the transmitter. In turn, the transmitter resends the packet and halves the bit-rate, thereby reduces the network load. The second mechanism detects network saturation based on TCP timeout. The transmitter then



(a) fair case

(b) unfair case

(c) alternate dominant case

Fig. 1: Three typical scenarios of competition between two flows sharing a bottleneck. On the left, the evolution of data-rates, on the middle, intermediate K values of FMP (proposed procedure), and on the right average values of the Jain's index (existing procedure).

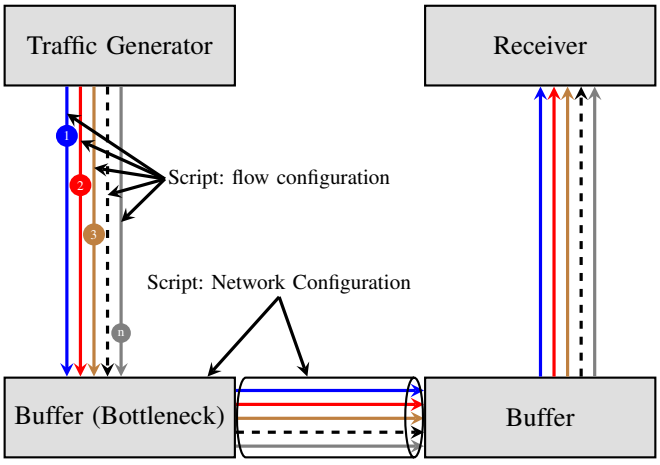


Fig. 2: Test-Bed architecture.

begins a new period of slow-start, which reduces the bit-rate.

- *Delay based CCA*: the CCA is based on latency variation to determine a congestion on the network. An example is BBR [4] recently deployed by Google. It observes latency variance to determine when the network is saturated (i.e. the latency increases when the buffer is full). It adapts the bit-rate accordingly, to prevent congestion.
- *Bit-rate prediction CCA*: the CCA analyzes the current data exchange to estimate the next bit-rate. An example is PCC [5], which is currently a work in progress at the IETF. PCC then tests the performance under various data-rate and chooses the best one for the user (either a constant bit-rate while avoiding packet loss, or a higher bit-rate without looking for reducing the packet loss).

The main network settings are detailed below.

Queuing Policy. Different queuing policies are implemented

by default on Linux: (i) *First In First Out (FIFO)* is the default buffer implementation: the first arrived packet is the first one to come out. When the buffer is full, packets are dropped. (ii) *fair queuing controlled delay (fq_codel)* is a buffer algorithm that aims to fix inequity problems on the network. It emulates a dedicated FIFO buffer per flow which is identified by source address, destination address, source port, destination port and type of protocol. *Controlled Delay (CoDel)* avoids the accumulation of packets in the buffer and removes the inequity between the different flows. (iii) *Random Early Detection (RED)* drops packets with a probability that gradually increases as the buffer fills in.

Link Characteristics For latency, loss rate, and buffer size, we use well-adopted values [9] (see Table I).

Latency on the link (ms)	0, 1, 50, 150, 300
Loss rate on the link (%)	0, 1, 5, 10, 20
Buffer size (bytes)	8750, 37 500, 62 500, 125 000, 250 000

TABLE I: Network characteristics.

VI. Performance analysis

We analyze in this section the result of the fairness measurement procedure when evaluating three CCAs (RENO, PCC, BBR) by using the above presented testbed. First we study the behavior of CCA as a function of latency, loss rate, buffer size and queuing policy and then we determine the number of tests that must be performed for achieving accurate results.

A. Competition between BBR and PCC

To evaluate fairness, files are simultaneously sent between the client and the server with all network configurations applied on the link after the bottleneck (see Section V). We limit the bit rate of each flow to 1 Mbits/s at the generator traffic level and the bandwidth of the link after the bottleneck is 1 Mbits/s, this configuration prevents burst problems. During the execution of the tests, we monitor the link after the bottleneck using *tcpdump*. For all captures using *tcpdump* we apply the metric detailed in Section IV to measure the fairness. Based on these measurements, we study the fairness among flows for the three CCAs.

We show an example of such a study in Figure 3 with a competition between PCC- and BBR- based flows. We represent the latency, the loss rate, and the queuing policies on the x-axis and the number of occurrences of K according to these values on the y-axis. The x-axis is divided into two parts: the left part represents the values of K in the configuration where PCC-based flow dominates and the right part represents the values of K when BBR-based flow is dominant. Note that, the conclusions made in the following apply to various other CCA competitions

Figure 3a represents the fairness evaluation according to the latency. For K values between 0.5 and 0.8 the number of occurrences is constant, i.e., the latency does not influence fairness.
The main reason is that BBR considers the latency fluctuation, then, when it is constant BBR does not react (i.e., whether the fixed latency value is 0 ms or 300 ms has no impact on the congestion detection).

Figure 3b shows the fairness evaluation according to the loss rate. Results reveal that for higher values of loss rate the unfairness increases and harms BBR.
The main reason is that PCC does not react to the loss rate and seeks to maximize the flow data rate. Therefore, when the loss rate increases, PCC does not change its behavior while BBR has more difficulty to estimate the available bandwidth due that the Round-Trip Time (RTT) increases.

Figure 3C shows the different queuing policies. Queuing policies have a significant impact on the fairness indicators. The FIFO queuing policy is the least fair one among the three tested queuing policies. Since FIFO delivers the oldest stored packet first, it drops the newest packets without regards to the others flows. The RED queuing policy is more equitable than FIFO due that the dropping strategy is not based on random choices but on probability. It chooses with higher probability the dominant flow. This solution is only efficient for the CCAs that are based on packet loss. Since PCC and BBR ignore packet loss, the fq_codel queuing policy is the fairest. *CoDel* avoids the accumulation of packets in the buffer and removes the inequity between the different flows.

In general, the above fairness evaluation reveals that an exhaustive number of tests are required to compare two CCAs. This fact is due to the variability of the behavior of CCAs in function of network configurations. For example, for two competing CCAs that are based on different principles, the competition is fair under some network configurations and unfair under some others. An important message that we convey in this paper is the necessity to define the required number of tests for achieving fairness metric accuracy.

B. Required Number of Tests for FMP Accuracy

We are now interested in determining the number of tests (different network configurations) that are necessary to assess the fairness between competing flows. For a configuration between two flows, we have tested 374 network configurations corresponding to each latency, loss rate, buffer size values and queuing policies. The runtime of each test takes on average 10 min. Hence, 2 days and 14 h are required to compare two CCA. Since we have six different competitions between the three considered CCAs, the whole runtime is more than 41 days.

We take the 90 th percentile to represent a unique final result of all obtained K values. Indeed, it is a good way to express that, in most of the configurations (except the 10 th percentile of outliers), the competition has a certain level of fairness. We provide the results of the six competitions in Table II.

Since the runtime is too long, even for two competing flows, we are then interested in obtaining a representative fairness

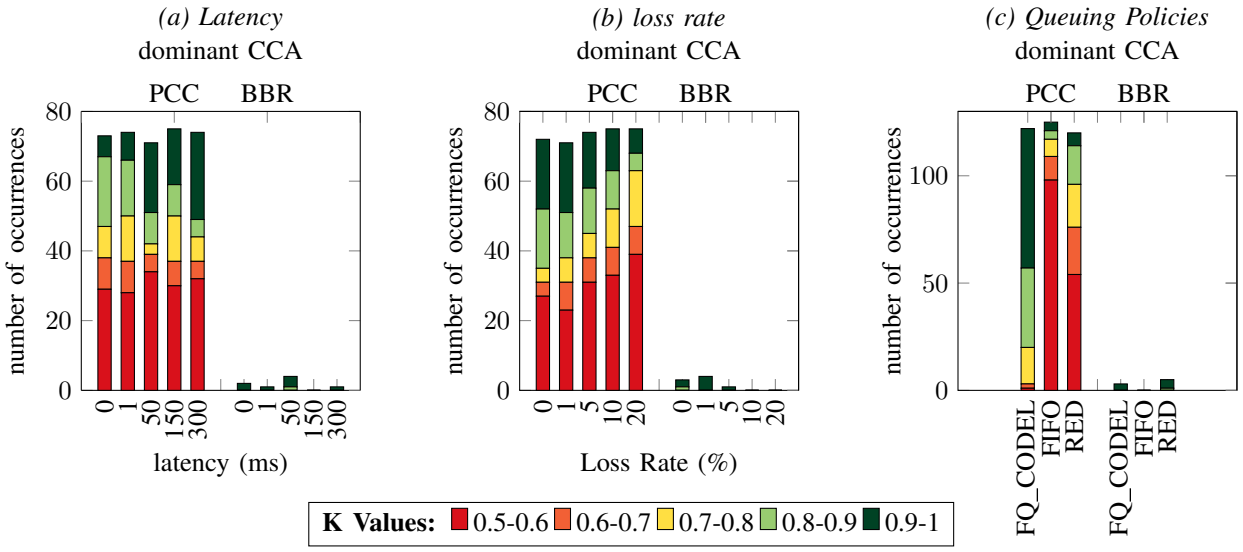


Fig. 3: Comparison of two CCAs according to latency, loss rate and queuing policies.

CCAs	K values
RENO-RENO	0.97
BBR-BBR	0.90
PCC-PCC	0.88
RENO-BBR	0.60
RENO-PCC	0.50
BBR-PCC	0.51

TABLE II: 90 th percentile of the K values obtained from 374 network configurations.

value while using a limited number of tests. We thus randomly pick a smaller number of configurations, ranging from 20 to 200, and we compute the obtained 90 th percentile from the collected results. In order to obtain a representative K value, we repeat 1000 times the fairness evaluation for each group of randomly picked network configuration. Hence, the fairness accuracy can be easily perceived according to the number of tests. Accuracy results are shown in Figure 4.

Competition between Two Flows. From Figure 4 we obtain the following conclusions: *(i)* A small number of values (notably 20) can be insufficient to achieve accuracy, typically for BBR-BBR, PCC-PCC and RENO-BBR. The difference between the exhaustive measure and estimated one (i.e., when using randomly picked network configurations) can be up to 0.2, which is a significant difference. In the worst case, the estimated measurement for the group twenty configurations provide a 90 th percentile equal to 0.5 although the exact value is 0.88. *(ii)* Experiments show that more than 100 tests (which is less than one third of the exhaustive analysis) is enough to provide an accurate and reliable measure of the fairness between two competing flows. Executing more tests (e.g., 200 tests) does not enable improving the accuracy.

Thus, we compared three CCAs competing in one-to-one session and we obtained an accurate fairness measure when executing 600 tests (during 4 days) with a precision that

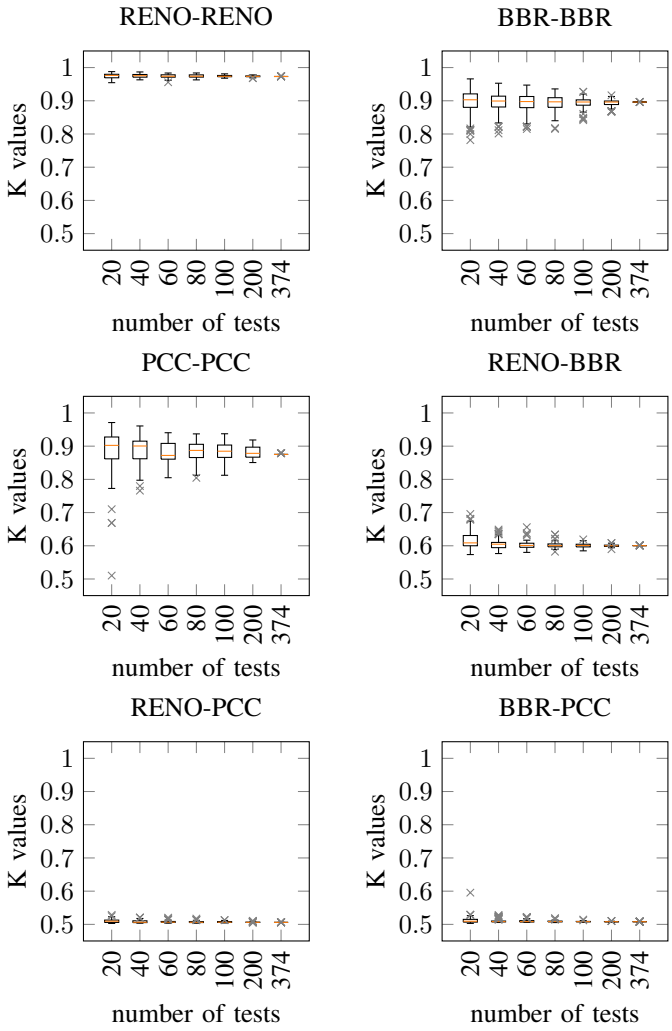


Fig. 4: FMP's accuracy evaluation as a function of the number of tests.

would have required 2274 tests and more than 41 days of computations.

Competition between Three Flows. To observe the behavior of three CCAs, we repeat the same experiment. When the three flows use the same CCA, results are similar to the case of two competing flows, so we do not represent these cases.

The number of tests that are necessary to obtain fairness accuracy is less important than when evaluating two flows. Hence, accurate values can be obtained when using only 60 different tests.

VII. Conclusions

The implementation of CCAs in the Internet has recently received a growing attention from researchers. New CCAs such as BBR and PCC have been designed to improve the network performance in terms of data rate for the satisfaction of service providers. Furthermore, the implementation of QUIC is a promise for faster updates and deployment for new CCAs. This line of research can however become a problem if new CCAs are insufficiently tested when considering the fair usage of network resources. The question of validating that a CCA behaves well for other competing flows in the network is open.

In this paper, we introduce a generic impartial fairness measurement procedure called FMP, which evaluates equity among the various CCAs competing in the network. The proposed metric returns a quantitative value of *fairness* which ranges from 0 (unfair) to 1 (fair). The main contribution of this work is that the fairness is evaluated at the session level and it takes into account the fact that not all instants of a session have the same importance. In fact, the beginning of a network session is a critical time, where over-aggressive CCA policies can harm the fair sharing of network resources. We evaluate the accuracy of the proposed metric by using an emulation platform. We illustrate the interest and accuracy of our proposal by studying the behavior of RENO, BBR and PCC under multiple network configurations.

References

[1] A. Gurtov, T. Henderson, S. Floyd, and Y. Nishida, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 6582, Apr. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6582.txt

[2] I. Rhee, L. Xu, and S. Ha, "CUBIC for Fast Long-Distance Networks," Internet Engineering Task Force, Internet-Draft draft-rhee-tcpm-cubic-02, Aug. 2008, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-rhee-tcpm-cubic-02

[3] N. Cardwell, Y. Cheng, S. Gunn, S. H. Yeganeh, and V. Jacobson. (2016) Bbr congestion control. [Online]. Available: https://www.ietf.org/proceedings/97/slides/slides-97-iccrg-bbr-congestion-control-02.pdf

[4] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: congestion-based congestion control," *ACM Queue*, vol. 14, no. 5, pp. 20–53, 2016.

[5] M. Dong, Q. Li, D. Zarchy, P. B. Godfrey, and M. Schapira, "Pcc: Re-architecting congestion control for consistent high performance," in *12th USENIX Symposium NSDI*, 2015.

[6] J. Iyengar and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport," Internet Engineering Task Force, Internet-Draft draft-ietf-quic-transport-04, Jun. 2017, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-quic-transport-04

[7] "Pantheon: the training ground for internet congestion-control research," in *2018 USENIX Annual Technical Conference (USENIX ATC 18)*. Boston, MA: USENIX Association, 2018. [Online]. Available: https://www.usenix.org/conference/atc18/presentation/yang-francis

[8] Z. Sarker, V. Singh, X. Zhu, and D. M. A. Ramalho, "Test Cases for Evaluating RMCAT Proposals," Internet Engineering Task Force, Internet-Draft draft-ietf-rmcat-eval-test-05, Apr. 2017, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-rmcat-eval-test-05

[9] V. Singh and J. Ott, "Evaluating Congestion Control for Interactive Real-time Media," Internet Engineering Task Force, Internet-Draft draft-singh-rmcat-cc-eval-04, Oct. 2013, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-singh-rmcat-cc-eval-04

[10] N. Islam, V. J. D. Elepe, J. Shaikh, and M. Fiedler, "In small chunks or all at once? user preferences of network delays in web browsing sessions," in *2014 IEEE International Conference on Communications Workshops (ICC)*, June 2014, pp. 575–580.

[11] H. Nam and H. Schulzrinne, "Youslow: What influences user abandonment behavior for internet video?" 2017.

[12] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems," DEC-TR-301, Digital Equipment Corporation, Tech. Rep., Sep. 1984. [Online]. Available: http://arxiv.org/abs/cs.NI/9809099

[13] J.-Y. Boudec, "Rate adaptation, congestion control and fairness: A tutorial," 2000.

[14] S. Floyd, "Metrics for the Evaluation of Congestion Control Mechanisms," RFC 5166, Mar. 2008. [Online]. Available: https://rfc-editor.org/rfc/rfc5166.txt

[15] D. Mitra and J. B. Seery, "Dynamic adaptive windows for high speed data networks with multiple paths and propagation delays," *Computer Networks and ISDN Systems*, vol. 25, no. 6, pp. 663 – 679, 1993, high Speed Networks. [Online]. Available: http://www.sciencedirect.com/science/article/pii/016975529390060H

[16] Y. Zhang, S.-R. Kang, and D. Loguinov, "Delayed stability and performance of distributed congestion control," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, pp. 307–318, Aug. 2004. [Online]. Available: http://doi.acm.org/10.1145/1030194.1015501

[17] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "PlanetLab: An Overlay Testbed for Broad-coverage Services," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 3, pp. 3–12, Jul. 2003. [Online]. Available: http://doi.acm.org/10.1145/956993.956995

[18] B. White, J. Lepreau, and S. Guruprasad, "Lowering the barrier to wireless and mobile experimentation," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 47–52, Jan. 2003. [Online]. Available: http://doi.acm.org/10.1145/774763.774770

[19] M. Ott, I. Seskar, R. Siraccusa, and M. Singh, "Orbit testbed software architecture: supporting experiments as a service," in *First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*, Feb 2005, pp. 136–145.

[20] K. Fall and S. Floyd, "Simulation-based comparisons of tahoe, reno and sack tcp," *SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 3, pp. 5–21, Jul. 1996. [Online]. Available: http://doi.acm.org/10.1145/235160.235162

[21] ITU-R. (2015) Imt vision framework and overall objectives of the future development of imt for 2020 and beyond. [Online]. Available: https://www.itu.int/rec/R-REC-M.2083-0-201509-I/fr

[22] M. Arumaithurai, X. Fu, and K. K. Ramakrishnan, "Nf-tcp: A network friendly tcp variant for background delay-insensitive applications," in *NETWORKING 2011*, J. Domingo-Pascual, P. Manzoni, S. Palazzo, A. Pont, and C. Scoglio, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 342–355.

[23] CISCO. (2017) The zettabyte era: Trends and analysis. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

[24] M. Varela, L. Skorin-Kapov, T. Mäki, and T. Hoßfeld, "Qoe in the web: A dance of design and performance," in *2015 Seventh International Workshop on Quality of Multimedia Experience (QoMEX)*, May 2015, pp. 1–7.

[25] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," 08 2014.