Less is More: Reducing Bandwidth to Enable Queueing Control and QoS

Carlo Augusto Grazia, Martin Klapez, and Maurizio Casoni

***Abstract*—The performance of Internet services heavily relies on efficient traffic management mechanisms. This paper investigates the impact of imposing a local bottleneck at the access networks' side compared to the conventional approach of traffic shaping by Internet service providers (ISPs) through first-in-first-out (FIFO) queues. The proposed local bottleneck strategy aims to enhance Internet performance by reducing latency, mitigating congestion, isolating distinct traffic flows, and enabling quality of service (QoS) differentiation. Through extensive experimentation on various home and office network setups, including fiber to the cabinet (FTTC), fiber to the home (FTTH), and fixed wireless access (FWA), we demonstrate the efficacy of the local bottleneck approach in delivering consistent and high-quality Internet performance, especially in congested environments. The results reveal that the traditional ISP bottleneck struggles to maintain a high-performance standard under congested conditions, highlighting the need for innovative traffic management techniques.**

***Index Terms*—Bottlenecks, congestion control, Internet traffic, latency, QoS.**

I. Introduction

THE modern Internet landscape is characterized by a growing demand for high-speed, low-latency, and reliable connectivity. Internet service providers (ISPs) are pivotal in delivering satisfactory user experiences by efficiently managing the vast influx of data traffic. Traditionally, ISPs have employed first-in-first-out (FIFO) queues for traffic shaping, which treats all packets equally, regardless of their type, priority, or content [1], [2]. While FIFO queues are simple and easy to implement, they often fail to address the evolving challenges of network congestion, quality of service (QoS) differentiation, and latency reduction [3]–[5]. This trend followed the increasing demand and deployment of bandwidth first, increasing the backhaul data rate and incrementally deploying higher last-mile bandwidth at the access network point. The current scenario is a worldwide migration from fiber to the cabinet (FTTC) to fiber to the home (FTTH), with some exceptions of fixed wireless access (FWA) in rural areas. ISPs' core business relies on bandwidth allocation with the "higher-is-better" goal, and just a few external companies like *Preseem*[1] in the US offer solutions to the ISPs to include well-known algorithms to shape traffic properly and introduce queueing management, packet scheduling and QoS in the puzzle.

In this paper, we propose an innovative approach to traffic management by imposing a local bottleneck at the access networks' side. Unlike the conventional ISP-centric traffic shaping, the local bottleneck strategy aims to optimize Internet performance by intentionally introducing controlled congestion at a local point within the network. This approach enables improved handling of network congestion, reduced latency, isolation of diverse traffic flows, and efficient QoS differentiation. The idea is simple: no matter how ISPs deal with traffic shaping, forcing a slightly lower bandwidth at the first local step will migrate the bottleneck formation from the remote ISP to the local network. This migration allows moving from a black-box bottleneck to a completely open one, on top of which it is possible to configure state-of-the-art algorithms to deal with queueing managing, packet scheduling, and QoS, assigning network performance control to the end user and increasing the perceived performance of the network, sacrificing just a tiny percentage of the bandwidth. Our tests show that the performance benefit is remarkable both in download and upload, with all the combinations of fixed access network technologies and ISPs tested in our campaign.

The remainder of this paper is organized as follows. Section II provides an overview of related works in the field of Internet traffic management and discusses the limitations of existing approaches. Section III outlines the experimental methodology, while Section IV provides technical details about the testbed configurations used to validate the effectiveness of the proposed local bottleneck strategy. Section V presents the experimental results and performance metrics collected from various network scenarios. Finally, Section VI concludes the paper by summarizing the findings and highlighting the implications of the local bottleneck approach in enhancing Internet performance.

II. Related Works

Several approaches have been proposed in the literature to address the challenges associated with Internet traffic management and QoS provisioning. Most existing studies focus on ISP-centric strategies, such as FIFO queues and bandwidth throttling, aimed at controlling traffic flow and ensuring fair resource allocation. While these methods are effective to some extent, they often need to improve in delivering consistently high performance, especially in congested scenarios.

Manuscript received January 3, 2025; revised May 29, 2025; approved for publication by Paek, Jeongyeup, Division 3 Editor, June 25, 2025.

The authors are with the Department of Engineering *Enzo Ferrari*, University of Modena and Reggio Emilia, via Pietro Vivarelli, 10, 41125, Modena, Italy, email: {carloaugusto.grazia, martin.klapez, maurizio.casoni}@unimore.it.

C. A. Grazia is the corresponding author.

Digital Object Identifier: 10.23919/JCN.2025.000050

[1]Preseem: The QoE Solution for ISPs. https://preseem.com/

Creative Commons Attribution-NonCommercial (CC BY-NC).
This is an Open Access article distributed under the terms of Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided that the original work is properly cited.

1229-2370/25/$10.00 © 2025 KICS

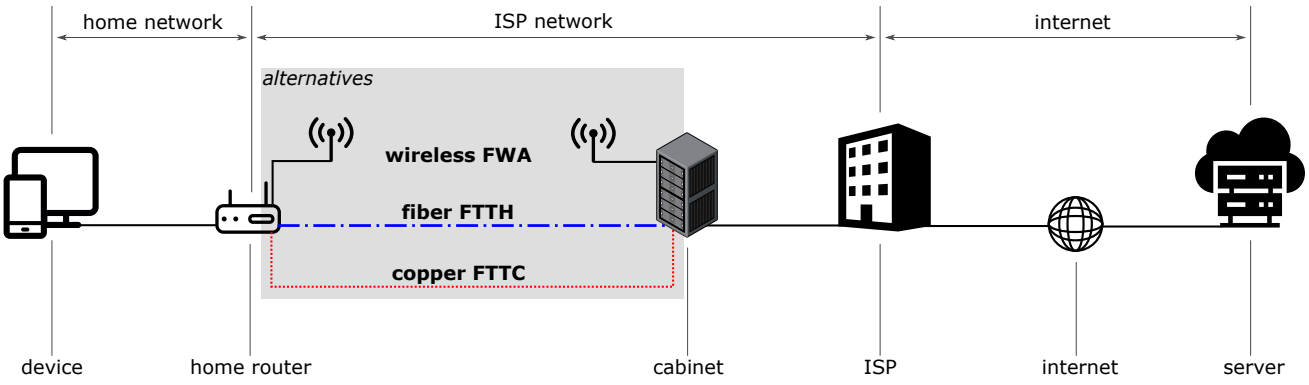


Fig. 1. Existing networks.

Researchers have also explored QoS differentiation techniques to prioritize critical traffic over less important data. Techniques like differentiated services (DiffServ) and traffic classifiers have been introduced to classify and prioritize packets based on their characteristics. However, these techniques still rely on ISP-centric control and may need to address the challenges of emerging applications and services entirely.

In contrast to these traditional approaches, the local bottleneck strategy proposed in this paper offers a novel perspective on traffic management. This approach aims to ensure effective traffic handling, reduced latency, and improved QoS differentiation by intentionally creating a controlled congestion point at the access networks' side. The local bottleneck approach is precious in scenarios where ISP-centric traffic shaping struggles to meet the demands of diverse applications and services.

A possible workaround to network congestion is described in [6]; rather than deal with the bottleneck, the authors proposed to change the path as soon as a congestion point is detected through deep reinforcement learning techniques, changing routing strategies, and reallocating the resources. This contribution finds the perfect place in multi-user mobile edge computing (MEC) networks, facing the challenge of efficiently allocating computational resources to users while considering their diverse computing demands and varying network conditions.

Another option is to deal with the applications at the endpoints of the communication system. This is the case of video streaming, a family of applications that already look for active queue management (AQM) presence at the bottleneck to adjust the rate transmission or other end-to-end parameters like chunk sizes, even if typically is not the ISP to deploy the queueing discipline [7]–[9]. Similarly, the contribution of [10] deserves to be cited; it is the case of quality of experience (QoE) analysis in video streaming, in which the end application shapes and paces the traffic to avoid the ISP bottleneck management.

Other possible circumventions for application-specific services like content delivery is to deploy dedicated functions directly at the smart-router access point, bypassing again the ISP bottleneck with a shaped traffic service [11].

All the previous solutions have the huge limitation to control only a specific scenario or application; they can not be scaled to the general group of services that access the same access network, which can potentially trigger the congestion in the ISP bottleneck anyway. Moreover, managing different services simultaneously requires the introduction of QoS in the puzzle.

Some of the inefficiencies of the ISPs identified in our manuscript are also reported in [12], where the authors focus on the best routing strategy to adopt in order to mitigate them. Even the approach adopted by the authors in [13] points towards the same direction: they implement multi-path TCP solutions to mitigate the ISP bottleneck formation rather than dealing directly with the bottleneck itself, literally a bypass.

At the same time, a vast area of research has been active in the last decades, designing Linux-based algorithms to deal with queueing management. It is the case of CoDel [14] now deployed as FQ-CoDel [15], which has been one of the main outputs in fighting bufferbloat [4], [16], also in case of multiple bottleneck formation [17]. Together with practical software contributions, also theoretical contributions about modeling of wired and wireless bottlenecks have been proposed: it is the case of Wi-Fi bottlenecks in [18] and hybrid wired/wireless ones in [19].

Since an end-user can not apply the products of the last decade of scientific research (e.g., FQ-CoDel) on top of ISPs' bottlenecks, we propose to migrate the bottleneck to a place where the end-user can deal with and where these solutions can be applied: the local access network.

Finally, considering that the access network often corresponds to a Wi-Fi hop, additional research has focused on optimizing these environments. Works such as [20]–[24] explore methods to enhance QoS and overall network performance. Moreover, recent studies address challenges like TCP small queues (TSQ) and their mitigation [25], [26], further reinforcing the feasibility and benefits of managing congestion locally.

III. Methodology and Experimental Setup

To evaluate the effectiveness of the local bottleneck approach, a comprehensive set of experiments was conducted on various home and office network configurations. Our test campaign included seven ISPs: Telecom Italia Mobile (TIM), Vodafone, Wind, Fastweb, Sky-WiFi, Eolo and StarLink. Each ISP has been tested at least on one connection typology between FTTC, FTTH, and FWA. The topology of each test is

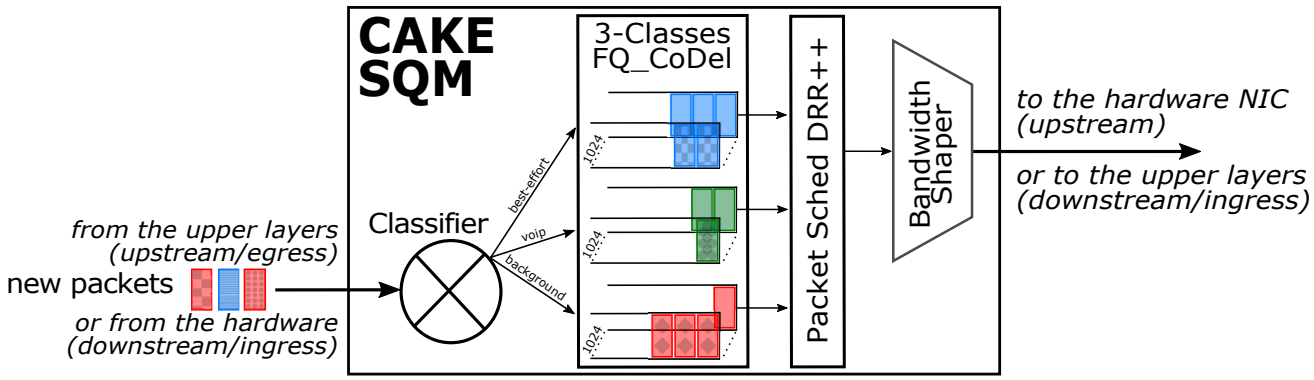


Fig. 2. CAKE.

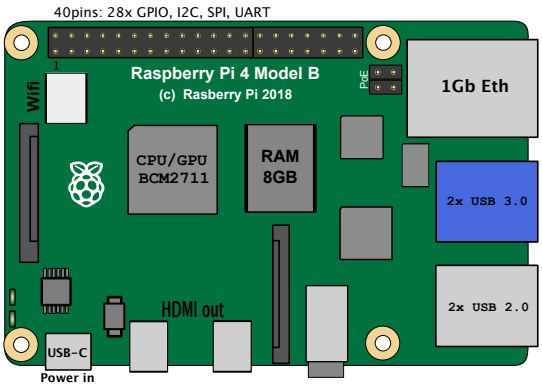


Fig. 3. Testbed device.

summarized in Fig. 1 and, to the best of our knowledge, all the tests performed in our campaign stick with that topology. The total number of home networks tested is over 50, while the hours of tests are over 100. We collected all the data, the logs, and the post-processed results in a repository [27], including in it only consistent data coming from the Italian campaign, where multiple repetitions of multiple combinations of ISPs and technologies have been collected according to Table I. To clarify, we obtained identical results in smaller datasets resulted from separate test campaigns in Norway, Sweden, and the US. These last results are not reported here due to lower statistical rigor, yet they confirm that what we find in the paper can be transferred from Italian ISPs to ISPs of other countries as well. Since this paper aims to highlight the possible performance improvements with a local bottleneck environment, we hide the ISP names by referring hereafter to them generically with ISP_i, where $i \in [1, 7]$.

Every network has been tested in the same way, comparing the performance of the local bottleneck strategy against the traditional ISP-centric traffic shaping that we typically see served by FIFO queues. Performance metrics such as latency, throughput, and QoS differentiation were measured and analyzed to assess the impact of the local bottleneck approach.

TABLE I
NETWORKS' TESTED.

ISP	Technology	Networks tested	Test's hours
ISP_1	FTTH, FTTC	21	39
ISP_2	FTTH, FTTC	14	31
ISP_3	FTTH, FTTC	8	18
ISP_4	FTTH, FTTC	7	16
ISP_5	FTTC	15	33
ISP_6	FWA	6	10
ISP_7	FWA	2	3

IV. LOCAL BOTTLENECK & TESTBED

Our idea to move the bottleneck at the access network stage allows each user to configure it using the best available networking solutions from the state-of-the-art. We decided to configure a bottleneck using the CAKE [28] (Common Applications Kept Enhanced) algorithm, which the bufferbloat community of Linux has proposed to simultaneously implement traffic shaping, packet scheduling, AQM, and QoS at the Linux kernel Internet layer. Fig. 2 reports a picture that summarizes the CAKE's schema. The CAKE key features can be summarized as follows:

- **Shaper**: CAKE deploys a deficit-mode shaper, which does not exhibit the initial burst typical of token-bucket shapers, and the parameters can be adjusted or changed at round-time without losing any packets.
- **Classification & QoS**: CAKE uses a default of three classes (Background, Best Effort, and Voice) to differentiate traffic in different queues. Each class accommodates up to 1024 queues, and the three classes are treated differently according to default QoS parameters.
- **AQM & Scheduling**: Each CAKE class is served by an FQ-CoDel scheduler, which combines CoDel (or Cobalt is most recent kernels) for AQM and a variant of Deficit Round Robin for schedules among the flows of the same class.

Our testbed is composed of a device under test (DUT), which is a Raspberry Pi Model 4 (RP4) with 8 GB of RAM and a 1 Gbit Ethernet interface as reported in Fig. 3. The RP4 runs an Ubuntu-Server Linux distribution and has been our end-node device in any network configuration depicted in the left side of Fig. 1, while a desktop Arch-Linux node at

the University of Modena and Reggio Emilia hosted the server side of our tests, connected in fiber to the Internet without ISP in place.

Each test has been performed using Flent [29] (flexible network tester), which is a network benchmarking tool. In particular, we used Flent to initially identify the ISP bottleneck parameters of upload and download bandwidth limitation, namely Th_{up} and Th_{down}. After the ISP bottleneck has been identified in terms of bandwidth, the main test has been executed, and it includes the following:

1) Four suites of flows transmissions between the RP4 and the Server, according to Fig. 4. A bulk upload, a bulk download, and two particular variants of the real-time response under load (RRUL) test designed by the bufferbloat community. These four tests will be described in detail.
2) Two different TCP variant configuration: TCP Cubic [30] and TCP BBR [31]. The former is the most used loss-based TCP variant, while the latter is a Google model-based variant, significantly reducing the queue occupation at the bottleneck stage, allowing us to extract information on the black-box ISP remote bottleneck when it is under test.
3) Two different bottleneck configurations. In one case, the RS4 does not impose any local limit, allowing the ISP to act as a bottleneck, which is the default configuration of each network under test. In the other case, the same tests of points 1 and 2 are performed under a local bottleneck configuration. With the local bottleneck, the RS4 deploys two instances of CAKE both in upload and in download (through the Linux *ingress* policy, which gives the possibility to apply traffic control tools to the incoming packets) with respectively a bandwidth shaping which is 5% less than Th_{up} and Th_{down}, to guarantee that the local bottleneck is stricter in bandwidth with respect of the ISP remote one, and the packets are accumulated locally.

All tests have been configured to run three instances of the test family reported in Fig. 4, stressing the network in very different ways:

- **Bulk Download**. A TCP Download from the server to the DUT, in parallel to an `ICMP` flow. The test runs for 30 seconds: 5 initial seconds of `ICMP` alone, 20 seconds with `ICMP` and TCP in parallel, and 5 final seconds of `ICMP` alone. This test is able to capture the network response to a single TCP download that congests the network, which is supposed to hit a single buffer in the network, and identify possible misbehavior due to the absence of queue management policies.
- **Bulk Upload**. A TCP Upload from the DUT to the server, in parallel to an `ICMP` flow. Despite the test being identical to the previous one regarding time segmentation, it can also capture the TCP RTT since the TCP flow control is in the hands of the DUT.
- **RRUL Best-Effort**. Four TCP flows in upload and four TCP flows in download congest the network, together with four UDP flows and an `ICMP` one. This test creates a severe congestion event stressing the network with multiple flows of the same priority (i.e., not marked in any possible different ways). This test is able to capture the presence of a packet scheduling algorithm in the network, verifying if these flows are treated in the same manner.
- **RRUL QoS version**. This test is identical to the previous one, but, in this case, the four different TCP flows (both in download and upload) are marked according to four different classes of traffic: CS5, EF, BE, and BK, listed from the highest to the lowest priority. They are real-time classes, expedited forward classes, best-effort classes, and background classes, respectively.

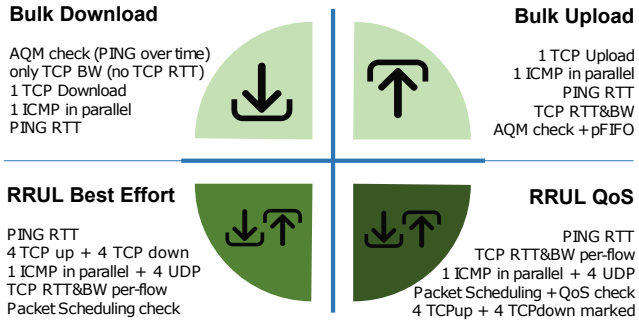
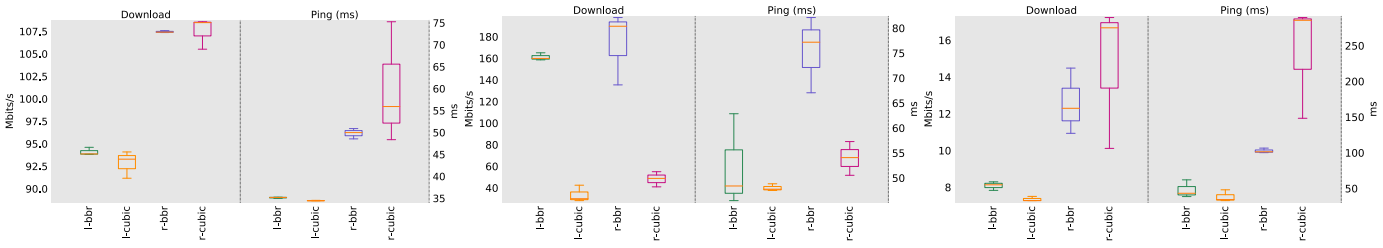


Fig. 4. Flent tests methodology.

V. Experimental Results

The experimental results demonstrated significant improvements in Internet performance through the local bottleneck strategy compared to the conventional ISP-centric traffic shaping. In congested scenarios, the local bottleneck approach effectively reduced latency and maintained consistent throughput levels. QoS differentiation was also more pronounced, enabling critical traffic to be prioritized without compromising overall network performance. All experiments used CAKE in its default configuration, including three DiffServ-based priority tiers and FQ-CoDel as the internal scheduler. This setup reflects a typical user deployment without manual tuning. A more detailed exploration of CAKE's configuration space (including variations in quantum size, shaper behavior, and DiffServ handling) has already been investigated in [28], offering complementary insights into how parameter tuning affects performance across different scenarios. Specifically, in FTTC, FTTH, and FWA setups, the local bottleneck approach consistently outperformed the ISP-centric FIFO queues in terms of latency reduction and QoS differentiation. These findings highlight the potential of the local bottleneck strategy to address the limitations of existing traffic management techniques and enhance Internet performance in diverse network environments.

In the following subsections, we showcase the numerical results reporting the more interesting in terms of assessable performance without mentioning the specific ISP; indeed, as we already mentioned, our paper aims to provide an ISP-independent solution that increases Internet performance regardless of the brand. We divide the subsections following



(a) FTTC download. (b) FWA satellite download. (c) FWA terrestrial download.

Fig. 5. One TCP stream download: throughput and latency with different technologies.

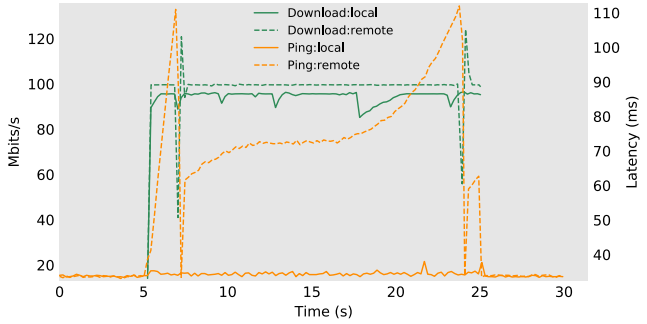


Fig. 6. FTTC download: time series.

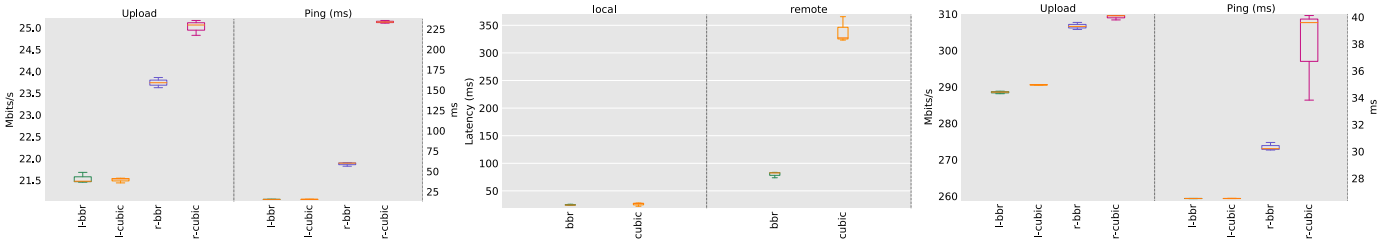
the main tests performed: bulk upload, bulk download, mixed up/down unmarked (RRUL best effort), and mixed up/down marked (RRUL QoS).

A. TCP Bulk Download

The download test is the most critical in terms of extractable features. In a download stream, the device under test receives the main TCP packets and replies with the ACKs; this inhibits the RTT measurement for the TCP flow. The network latency is anyway monitored with the Flent test through the parallel `ICMP` flow. By monitoring `ICMP PING` before, during, and after the TCP download stream, Flent measures the induced latency, a strong indicator of AQM or packet scheduling policy at the bottleneck. Fig. 5 reports an FTTC and two FWA connections tested; all the plots show the throughput of the TCP flow and the induced latency measured through the `ICMP` flow. The names l-bbr and l-cubic represent the results of a local bottleneck configuration with TCP BBR and TCP CUBIC as congestion control, respectively. Similarly, r-bbr and r-cubic represent the result of the remote bottleneck (imposed by the ISP) with TCP BBR and TCP CUBIC as congestion control, respectively. Figs. 5(a) and 5(c) show very similar trends where the slight bandwidth reduction imposed by the local bottleneck provides sensible latency reduction; the former plot refers to a FTTC connection while the latter plot to a terrestrial FWA one. Continuing the discussion of these two plots, when TCP CUBIC acts as congestion control, the latency reduction is even more significant; this is because it is known from the literature that TCP CUBIC is a loss-based congestion control that fills the queues compared with TCP BBR that tries to completely avoid queueing thanks to its model-based engine. Similar but still effective results hold for Fig. 5(b), a satellite-based FWA where the two congestion controls behave differently. In this scenario, TCP BBR reaches higher throughput with higher latency in the remote configuration, while moving to a local bottleneck again helps maintain the network's minimum RTT. In this specific scenario, TCP CUBIC seems unable to fulfill the available bandwidth due to its inefficiency in satellite hops. To give extra insight into the benefit of migrating the bottleneck locally, we also report Fig. 6 showing how TCP CUBIC behaves differently between a remote FTTC FIFO bottleneck imposed by the ISP and a local CoDel-based one that avoids bufferbloat. The dashed lines of the figure refer to the remote bottleneck, while the solid lines refer to the local bottleneck. Monitoring the `ICMP PING` values, it is clear how `ICMP` and TCP packets fall into the same queue, showing the typical cubic curve of the TCP CUBIC congestion window. Also, the spikes in the throughput measurements are indeed aligned with the maximum latency higher than 100 ms, where packet drops occur. The solid lines, instead, show the importance of AQM CoDel that blocks the TCP CUBIC filling-queue behavior, maintaining the minimum latency, which does not change in the central part of the test where `ICMP` and TCP CUBIC share the same bottleneck, and the throughput is steady and high close to the maximum. From this simple result, it is clear that the remote bottleneck does not use packet scheduling techniques, mixing TCP streams and `ICMP` streams in the same queue. It also does not deploy AQM disciplines, treating the bottleneck with a simple FIFO queue. This result is confirmed in almost all the tests gathered in our repository [27] no matter the access technology deployed (FTTH, FTTC, or FWA) and, more importantly, the ISP brand.

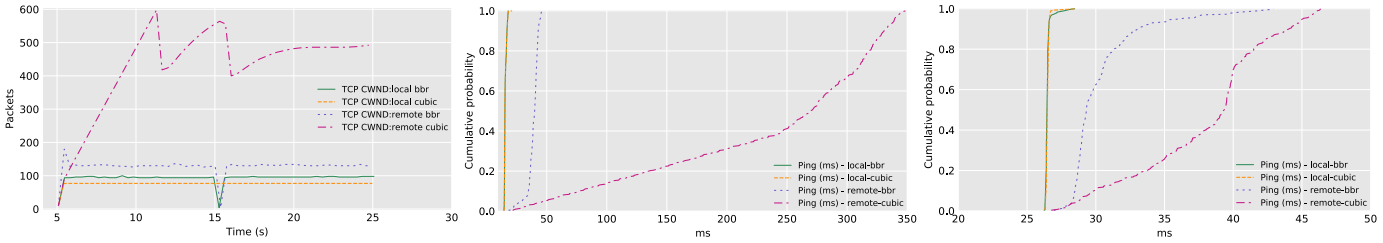
B. TCP Bulk Upload

The upload test introduces more information than the download test. Since the device under test is uploading the TCP stream, it has control over the primary TCP data, allowing to measure the TCP congestion window (CWND) as well as the actual TCP RTT, giving the possibility to compare it with the `ICMP PING` and distinguish between `ICMP` latency and TCP latency if needed. Moreover, ISPs are typically unbalanced between upload and download streams, applying more severe traffic shaping in upload, i.e., restricting the bottleneck, which, if not managed properly, affect negatively the Internet upload performance even more than the download. Fig. 7 reports similar plots to the previous subsections, with upload throughput and `ICMP` induced latency in Figs. 7(a) and 7(c) for FTTC and FTTH, respectively. As reported for



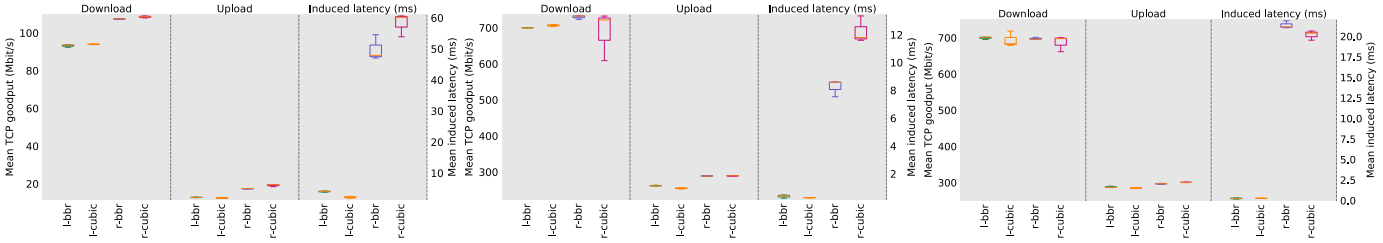
(a) FTTC upload. (b) FTTC TCP RTT. (c) FTTH upload.

Fig. 7. One TCP stream upload: throughput and latency with different technologies.



(a) FTTC TCP CWND. (b) FTTC `ICMP PING` CPF. (c) FTTH `ICMP PING` CPF.

Fig. 8. One TCP stream upload: throughput and latency with different technologies.



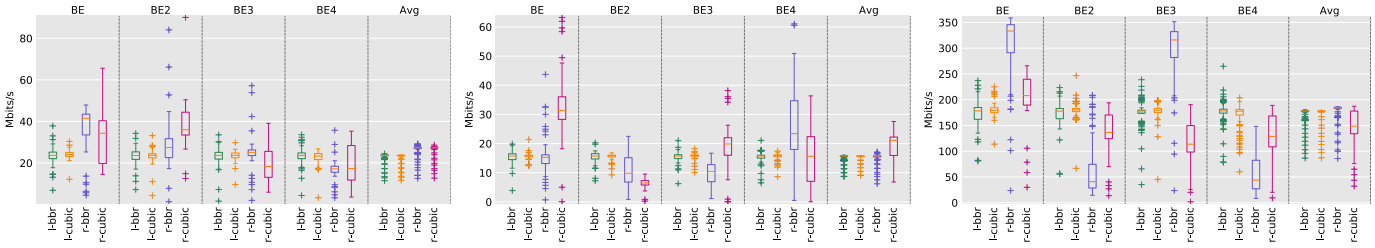
(a) FTTC. (b) FTTH. (c) FTTH (64 flows).

Fig. 9. Four TCP stream upload and download (a, b), 64 flows (c): throughput and latency with different technologies.

the download tests, also in this case, a minor reduction of the throughput, necessary for migrating the bottleneck from the remote ISP place to the local one, is associated with a remarkable decrease of the induced latency in the network. Both FTTC and FTTH also show that TCP BBR is less stressed by the remote bottleneck configuration despite obtaining benefits with bottleneck migration anyway. This trend is confirmed in Fig. 7(b) where only TCP RTT is reported for an FTTC test; TCP RTT of BBR halves moving from a remote to a local bottleneck, while TCP RTT of CUBIC decreases by almost one order of magnitude. Continuing, Fig. 8 reports different plots, showing the TCP CWND evolution in a FTTC network in Fig. 8(a) and a comparison between the cumulative probability functions (CPFs) of `ICMP PING` between FTTC and FTTH networks, in Figs. 8(b) and 8(c) respectively. In Fig. 8(a) is visible how TCP CUBIC CWND grows in the remote configuration, confirming the presence of a FIFO queue without an AQM early-dropping system at the ISP, which is the reason why TCP CUBIC manifests high latency. TCP BBR, instead, is less affected by the bottleneck migration in terms of CWND development since it is model-based and is more queue-size independent; it can also be observed in the plot the negative spike at the second 15, so after 10 seconds from the TCP upload start, corresponding to the draining stage of the finite state machine ruling the TCP BBR behavior, regardless the bottleneck position: remote and local TCP BBR spikes are aligned, indeed. Figs. 8(b) and 8(c) report the CPF of `ICMP PING` showing a similar pattern between FTTC and FTTH, what changes is the absolute scale, which is shorter in FTTH networks due to the higher bandwidth that is mapped to reduced queueing delay. In any case, also in FTTH network, the migration of the bottleneck from the remote ISP to the local access network reduces the latency substantially.

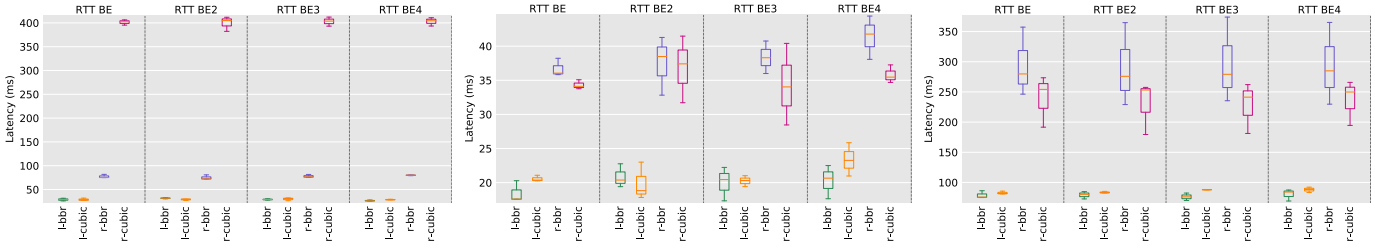
C. *Real-time Response Under Load: Best Effort*

This subsection describes the first Real-Time Response Under Load test (RRUL [32]), in which the device under test performs a mix of four TCP uploads and four TCP downloads simultaneously, together with ICMP PING and UDP traffic. The characteristic of this test is that all TCP flows are equally marked as best effort (BE), meaning they are expected to be treated identically by the bottleneck. The desired outcome is an equal bandwidth distribution among flows, which is achievable only if the bottleneck employs a fair packet scheduler. This test is referred to as RRUL_BE. Fig. 9 presents results similar to previous subsections, including upload and download throughput as well as latency induced by ICMP PING. Fig. 9(a) shows results for an



(a) FTTC Four TCP streams. (b) FTTC Four TCP streams (another ISP). (c) FTTH Four TCP streams.

Fig. 10. Four TCP streams download: throughput fairness with different technologies.



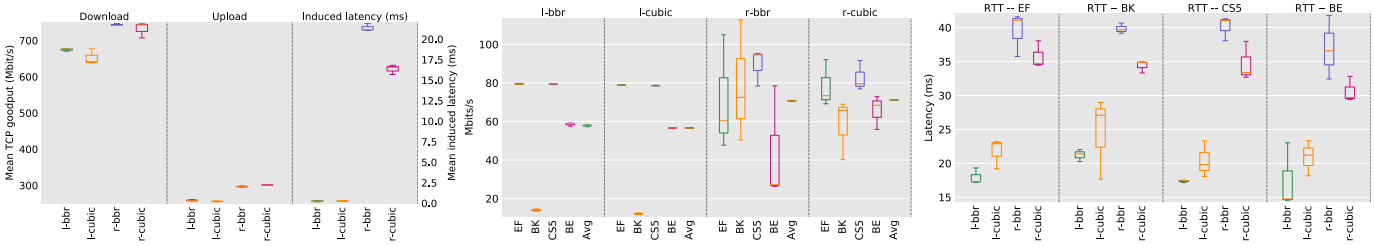
(a) FTTC Four TCP streams. (b) FTTH Four TCP streams. (c) FWA Four TCP streams.

Fig. 11. Four TCP streams upload: RTT with different technologies.

FTTC connection, while Figs. 9(b) and 9(c) refer to two congestion levels on the same FTTH ISP, to incorporate a scalability analysis. The key takeaway from these plots is that, regardless of access technology or congestion severity, the trend remains consistent: a minor sacrifice in available bandwidth results in a substantial reduction in latency, with absolute values varying according to the technology used. Moreover, increasing the number of active flows from four to sixty-four slightly enhances the benefits of the local bottleneck strategy. We report the 64-flow RRUL test only in this scenario to avoid redundancy, as results across other configurations show similar trends and are available in our repository [27]. As observed in earlier tests, the local bottleneck consistently improves latency, regardless of the TCP congestion control algorithm used. Among different technologies and congestion conditions, TCP BBR generally outperforms TCP CUBIC.

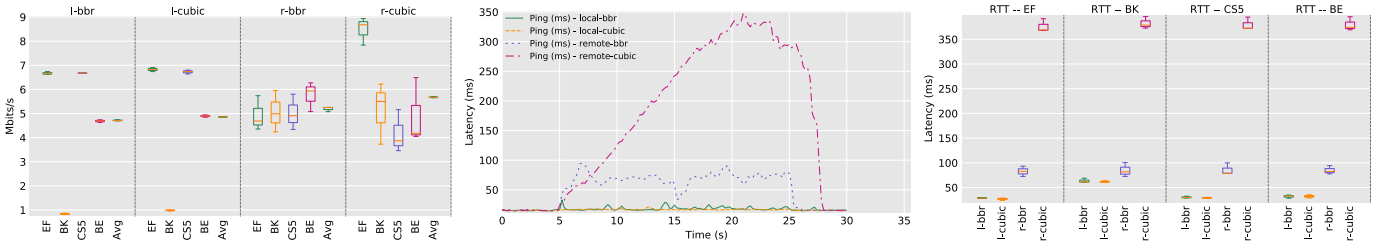
This first RRUL test aims to highlight the presence of a packet scheduler at the bottleneck that treats equal flows in an equal way. Fig. 10 helps discover this behavior by reporting the download performance of each of the four TCP streams, together with the average value. The ideal result is to have the same download value for the four BE flows and, so, equal to the average. In this case, we report two different FTTC ISPs in Figs. 10(a) and 10(b), while Fig. 10(c) report an FTTH result. In all the plots, as well as in all the tests collected in our repository, the results are very similar: with our local bottleneck solution, all four TCP downloads share the bandwidth equally, with almost identical throughput values, reaching high fairness regardless of the TCP congestion control adopted. This result is remarkable also comparing the average throughput between local and remote bottleneck; the former is always close to the latter, in some cases (like Fig. 10(c)) even better with a specific TCP congestion control (TCP CUBIC in this particular case). In case of remote bottleneck, none of the tested ISP networks can guarantee fairness among the TCP flows, with global throughput that is largely unbalanced between flows, regardless of the TCP congestion control. For instance, in Fig. 10(b), BE, and BE2 CUBIC flows with the remote bottleneck are more than 30 Mbps and less than 10 Mbps, respectively, with a factor of three throughput ratio. In another case, in Fig. 10(c), BE2 and BE3 BBR flows with the remote bottleneck have a factor of six throughput ratio instead.

While the download side is more interesting in terms of bandwidth, the upload side also offers the actual RTT measurements of the TCP flows, as discussed separately in the first two subsections. The bandwidth analysis would be identical in terms of fairness moving from the four TCP downloads to the four TCP uploads, and it is not reported for redundancy; we remember the availability of all the results in our repository [27]. On the other hand, the four TCP uploads offer the comparison of the four RTTs of the BE TCP flows, which is meaningful, and Fig. 11 includes an example for FTTC, FTTH and FWA in Fig. 11(a), Fig. 11(b), and Fig. 11(c), respectively. A characteristic of Fig. 11, and the RRUL test in general, is the difficulty in evaluating fairness through RTT; despite the significant values of RTT, also the remote configurations report similar RTTs between the four TCP streams, between flows of the same TCP congestion control family. This phenomenon is justified by the FIFO of the ISP, i.e., all the flows are mixed together. Once the queue is almost full, all the flows experience the same maximum RTT. At the same time, the throughput is given by the number of packets they have, which is hard to control and fairly distributed within a FIFO queue, resulting in very different throughput despite the similar RTT. What clearly emerges is, again, a massive gap between the RTTs under the local or remote bottleneck configuration: the former reaches the minimum RTT in all the technologies, stable for the four flows, and independent by the TCP congestion control adopted, while the latter depends



(a) FTTH RRUL up/down aggregate. (b) FTTH RRUL upload throughput. (c) FTTH RRUL upload RTT.

Fig. 12. RRUL QoS: Throughput and RTT FTTH.



(a) FTTC RRUL upload throughput. (b) FTTC RRUL ICMP PING. (c) FTTC RRUL upload RTT.

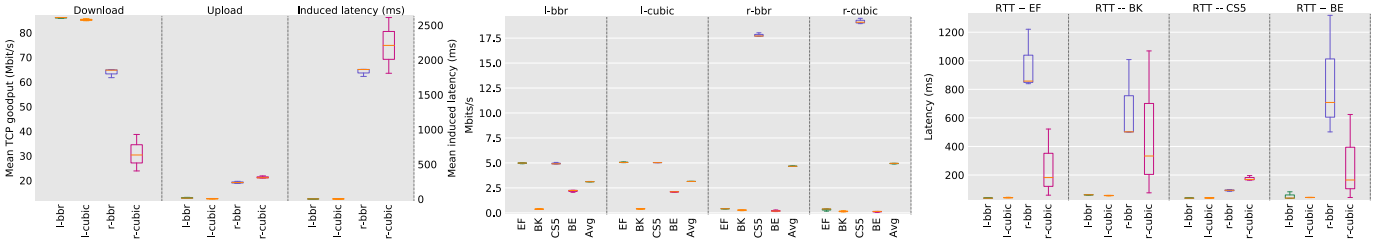
Fig. 13. RRUL QoS: Upload Throughput, ICMP PING and TCP RTT on FTTC.

both by the access technology and the congestion control. Fig. 11(a) refers to an FTTC technology; the local bottleneck configuration allows all the flows to oscillate stably around 30 ms of RTT, while the remote bottleneck configuration introduces a large gap between TCP BBR, with values of RTT around 80 ms, and TCP CUBIC, with values of RTT higher than 400 ms. The reason is probably given by the large buffers of the ISP, which are not fully exploited with TCP BBR, a characteristic of Google's congestion control, while they are filled by TCP CUBIC. A large buffer, indeed, introduces a significative queueing delay in a network with a limited upload bandwidth, circa 10 Mbps in the related test. Fig. 11(b) refers to an FTTH technology, with very similar results for what concerns the local bottleneck. In contrast, in the remote bottleneck configuration, the absolute values are lower than the FTTC one but still close to a double latency with respect to the local bottleneck. In the specific FTTH reported, there are no significative differences between TCP BBR and TCP CUBIC, probably because of a reduced FIFO buffer size of the particular ISP under test. Moving from Fig 11(b) to Fig. 11(c), the trend is conserved, while the absolute values change due to the very different technology nature: the former has already been discussed, while the latter is a FWA, and the maximum RTT grows up to 275 ms in the remote bottleneck configuration. The reason for this gap lies in the lower bandwidth and characteristics of the wireless medium; in particular, TCP BBR struggles a bit more for this reason since it is still not optimized for wireless bottlenecks.

D. Real-time Response Under Load: QoS

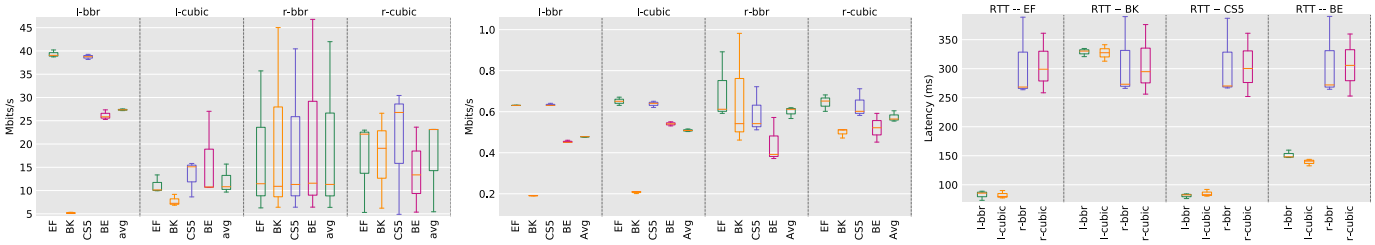
The final subsection of our results describes the second RRUL test named RRUL_QoS, which introduces the QoS marks in the TCP flows. The four TCP flows in download and upload are marked as BE, Background (BK), VoIP (CS5), and Expedited Forward (EF), respectively. As in the previous tests, also in the RRUL_QoS, the TCP flows are transmitted together with ICMP PING and UDP traffic. The characteristic of this test is that the four categories, BE, BK, CS5, and EF, are not supposed to be treated equally by the ISP bottleneck. The desired result is not an equal bandwidth division, which could be obtained only if the bottleneck is served with a packet scheduler with an active QoS policy. The literature suggests guidelines to treat flows of different categories [33]–[35], all of them agree in protecting VoIP (CS5 in our tests) flows, assigning more bandwidth and reducing the latency while doing the opposite for background traffic (BK in our tests) which does not require strict QoS performance and could receive less bandwidth with relaxed latency constraints. In other words, the goal of the RRUL_QoS test is to realize if the ISPs under test are able to deal with QoS, treating different QoS flows in different ways, and to realize if the migration to local bottleneck can enable active QoS treatment regardless of the ISP under test.

We start from Fig. 12, with an ISP that does not present active QoS capabilities over an FTTH technology. It is essential to clarify that Fig. 12(a) reports the overall download and upload performance of the four aggregate flows, and so it gives identical insight between RRUL_BE and RRUL_QoS tests; again, the point is that the local bottleneck with a small percentage of bandwidth reduction allows to reduce remarkably the network latency. Figs. 12(b) and 12(c) report the throughput and the TCP RTT of the four upload flows, divided into the four categories EF, BK, CS5, and BE. The local bottleneck solution manifests a clear pattern, regardless of the TCP congestion control deployed, imposed by the CAKE algorithm, with more bandwidth for CS5 and EF applications, penalizing BK traffic both in terms of bandwidth (Fig. 12(b)) and in terms of TCP RTT (Fig. 12(c)). A similar and stable differentiation is not visible in the remote configuration; the four flows oscillate as in the previous RRUL_BE test, as



(a) FTTC RRUL upload throughput. (b) FTTC RRUL `ICMP PING`. (c) FTTC RRUL upload RTT.

Fig. 14. RRUL QoS: Throughput and RTT of an ISP with QoS.



(a) FWA RRUL download throughput (satellite). (b) FWA RRUL upload throughput. (c) FWA RRUL upload RTT.

Fig. 15. RRUL QoS: Throughput and RTT of FWA.

shown in Figs. 10 and 11. The absence of QoS treatment at this specific ISP remote bottleneck is confirmed in Fig. 12(c) with an RTT for CS5 equal or higher with respect to the other flows.

The same QoS pattern for the local bottleneck is maintained moving from FTTH to FTTC, as can be observed moving from Fig. 12 to Fig. 13. This comparison allows us to see the stability of the local bottleneck benefits regardless of the technology under test, while the performance of the remote ISP bottleneck changes from case to case. Fig. 13(a), for instance, has very similar average throughput with local and remote bottleneck, while the remote bottleneck does not differentiate at all between the flows, with very similar throughput of TCP BBR and a massive unfairness with CUBIC, which penalizes, contrarily from the desired treatment, the CS5 flow. The latency performance for a non-QoS ISP (FTTC) is summarized in Figs. 13(b) and 13(c), showing `ICMP PING` RTT and TCP RTT, respectively. The `PING` RTT shows TCP CUBIC's inefficiency in containing the latency in a network with a large remote bottleneck unserved by AQM policies. In contrast, TCP BBR contains a bit more latency induction. Instead, Fig. 13(c) differentiates the four TCP flows, reporting again the good behavior of the local bottleneck, which penalizes BK traffic. At the same time, the remote configuration cannot differentiate, and generally, it provides higher results. In particular, TCP CUBIC reports RTTs order of magnitudes higher than the local bottleneck, while TCP BBR has RTTs consistently higher than the worst local BK flow.

A separate analysis must be reported for an FTTH/FTTC ISP that implements QoS guarantees among flows, differentiating between CS5 and the others. These ISP results are reported in Fig. 14 for an FTTC network. The central Fig. 14(b) clearly shows that this ISP prioritizes the CS5 traffic, with higher throughput with respect to the other flows. While the implementation of QoS policies at the remote ISP bottleneck is undoubted, it requires further investigation on how the QoS policy is deployed. Fig. 14(c) reports very high TCP RTT in the remote configuration, and even if the CS5 flow is protected with respect to EF, BK, and BE, it is still close to 100 ms with TCP BBR and 200 ms with TCP CUBIC, which is again drastically more than the TCP RTT reported with the local bottleneck configuration with any TCP congestion control and any TCP flow, even BK. By looking at the overall performance of Fig. 14(a), it is also clear that with the remote configuration, the induced latency is extremely high, and the ability to maximize upload and download throughput is compromised, with the local bottleneck that performs better than the remote one also in terms of bandwidth. To the best of our knowledge, these plots can be translated in a pfifo_fast queueing discipline; At the same time, we have seen only FIFO queues in the ISPs; in this case, there is the differentiation between CS5 and the other traffic, with clear prioritization of CS5; in both the queues, there is no AQM allowing to reduce the latency and the bufferbloat effect.

We conclude our analysis with Fig. 15 presenting different FWA examples: a satellite communication download in Fig. 15(a), a WiMAX upload in Fig. 15(b), and its RTTs counterpart in Fig. 15(c). The satellite-based FWA connection of Fig. 15(a) allows us to see two important things: (i) the local bottleneck also works in download, shaping traffic through the ingress interface once it arrives in the local network, and (ii) in this specific case, TCP BBR performs better than TCP CUBIC, with a much more clear QoS pattern. The reason why TCP BBR outperforms TCP CUBIC is in its model-based behavior. Indeed, the packets accumulated in the ingress interface traveled the entire network from the server to DUT once with an FWA wireless hop. TCP BBR is more likely to maintain the desired bandwidth allocation stably due to its nature of ignoring packet losses: this avoids the

TABLE II
NETWORKS' BOTTLENECKS OVERALL RESULTS.

Network	Tecnology	AQM	Packet Scheduling	QoS	Queueing Discipline
ISP_1	FTTH/C	no	no	yes	pfifo_fast
ISP_2	FTTH/C	no	no	no	FIFO
ISP_3	FTTH/C	no	no	no	FIFO
ISP_4	FTTH/C	no	no	no	FIFO
ISP_5	FTTC	no	no	no	FIFO
ISP_6	FWA	no	no	no	FIFO
ISP_7	FWA	no	no	no	FIFO
Local Bottleneck (ISP independent)	ALL	yes	yes	yes	CAKE

problem of misunderstanding a channel loss (e.g., a wireless medium error) with a packet drop caused by congestion. Continuing, all three plots report a clear QoS pattern in the local bottleneck configurations, regardless of the actual FWA-specific technology (satellite or terrestrial), with RTTs of the remote bottleneck that can be compared only with the penalized BK flow of the local one. In conclusion, Table II summarizes the results obtained for each ISP, reminding that all the ISP data and technologies not reported here for space constraints are available in [27].

E. Discussion of Limitations

While the local bottleneck strategy shows clear benefits in latency reduction, QoS enablement, and congestion handling, it is important to recognize potential limitations and contextual dependencies of this approach.

First, the effectiveness of migrating the bottleneck to the local access network depends on the ability of the end-user to configure and maintain traffic shaping and queue management mechanisms, such as CAKE. This may pose challenges in deployment for non-technical users, especially in unmanaged or low-cost consumer hardware environments. It is worth noting that the CS5-marked flows used in the RRUL QoS tests are representative of real-time applications such as video conferencing, VoIP, and online gaming. The consistent latency improvements observed for these flows under the local bottleneck strategy suggest that end-user applications relying on low-latency performance would directly benefit from this approach. Second, while we demonstrate significant improvements across various technologies (FTTH, FTTC, FWA), our approach currently assumes that the bandwidth available at the ISP-side bottleneck remains relatively stable over time. This assumption allowed us to configure a fixed local shaping threshold just below the measured bandwidth limit. However, in real-world scenarios where ISPs dynamically adjust bandwidth—due to traffic engineering, congestion, or policy enforcement—the remote bottleneck may shift, potentially undermining the effectiveness of the fixed local bottleneck. In such cases, the local shaping may no longer be sufficient to guarantee control over congestion, requiring adaptive mechanisms to track and respond to changing upstream conditions. Third, local bottleneck strategies may introduce unintended side effects when multiple competing users share the same access point. For instance, traffic shaping might penalize certain users disproportionately if classification mechanisms are misconfigured or if dynamic traffic patterns are not well accounted for. In addition, although the Raspberry Pi was used as a controlled, Linux-based testbed, the CAKE algorithm is lightweight and increasingly available on consumer routers (e.g., via OpenWrt). Comparable improvements can be expected in real-world deployments, provided the router supports traffic shaping and has sufficient CPU capacity to manage access-line rates. Finally, emerging access technologies such as 5G fixed wireless or low-latency satellite links might present new bottleneck characteristics not fully addressed by the current implementation. Future studies should explore how these environments interact with local queue management and whether additional adaptations are needed. Despite these considerations, we believe that the ability to control the bottleneck locally remains a significant step forward for enabling advanced QoS techniques in consumer networks, provided proper configuration and support tools are in place.

VI. CONCLUSION

This paper presented a novel approach to Internet traffic management by imposing a local bottleneck at the access networks' side of FTTC, FTTH, and FWA deployments. The local bottleneck strategy offers improved latency, congestion handling, traffic isolation, and QoS differentiation by intentionally introducing controlled congestion. Extensive experiments on various home and office network configurations demonstrated the superior performance of the local bottleneck approach compared to traditional ISP-centric traffic shaping using FIFO queues. The results underscore the importance of rethinking conventional traffic management techniques and exploring innovative approaches to meet the evolving demands of modern Internet services. The local bottleneck strategy holds significant promise in enhancing Internet performance, especially in congested environments, and provides a valuable direction for future research and network optimization efforts. While this paper demonstrates the effectiveness of local bottleneck strategies across multiple technologies and configurations, several avenues remain open for further investigation. First, adaptive shaping mechanisms should be explored to address scenarios where ISP bandwidth allocation fluctuates dynamically over time. Incorporating real-time monitoring and automatic adjustment of shaping thresholds would enhance robustness. Second, broader usability studies could assess how non-technical users interact with such systems and identify requirements for making deployment and configuration more

accessible. Additionally, the impact of the local bottleneck in highly dynamic or mobile environments, such as 5G or satellite networks, warrants dedicated research. Finally, extending the solution to support multi-user fairness and cross-device coordination in shared access networks could significantly improve its scalability and practicality. Moreover, while this work focuses on user-managed local queueing, collaboration with ISPs through standardized mechanisms such as DSCP-based QoS enforcement holds significant potential to further enhance latency and fairness outcomes. Future research should investigate hybrid approaches that combine local bottleneck strategies with ISP-coordinated QoS policies to leverage the strengths of both ends of the access network. Such cooperation could enable more granular traffic differentiation and dynamic resource allocation, ultimately delivering better user experiences across diverse network conditions.

REFERENCES

[1] K. S. Kim, "On guaranteeing the quality of service of conformant traffic in excess bandwidth allocation for shared access networks," in *Proc. IEEE Sarnoff Symposium*, 2015. .

[2] A. Kesselman, Y. Mansour, and R. Stee, "Improved competitive guarantees for QoS buffering," vol. 43, no. 1–2, 2005. [Online]. Available: https://doi.org/10.1007/s00453-005-1158-x

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

[4] Y. Gong, D. Rossi, C. Testa, S. Valenti, and M. Taht, "Fighting the bufferbloat: On the coexistence of aqm and low priority congestion control," in *Proc. IEEE Computer Communications Workshops*, 2013.

[5] Y. Guo, F. Qian, Q. Chen, Z. Morley Mao, and S. Sen, "Understanding on-device bufferbloat for cellular upload," in *Proc. ACM IMC*, 2016.

[6] Q. He *et al.*, "Routing optimization with deep reinforcement learning in knowledge defined networking," *IEEE Trans. Mobile Comput.*, vol. 23, no. 2, pp. 1444–1455, 2023.

[7] J. Kua, P. Branch, and G. Armitage, "Detecting bottleneck use of PIE or FQ-CoDel active queue management during DASH-like content streaming," in *Proc. IEEE LCN*, 2020.

[8] J. Kua, G. Armitage, and P. Branch, "The impact of active queue management on DASH-based content delivery," in *Proc. IEEE LCN*, 2016.

[9] J. Kua, G. Armitage, P. Branch, and J. But, "Adaptive chunklets and AQM for higher-performance content streaming," *ACM Trans. Multimedia Comput., Commun. Appl.*, vol. 15, no. 4, pp. 1–24, 2019.

[10] J. Jiang *et al.*, "Q-FDBA: Improving QoE fairness for video streaming," *Multimedia Tools Appl.*, vol. 77, no. 9, pp. 10787–10806, 2018.

[11] H. Xue, J. You, and J. Wang, "An intelligent and decentralized content diffusion system in smart-router networks," *IEICE Trans. Commun.*, vol. E102B, no. 8, pp. 1595–1606, 2019.

[12] P. Kumar *et al.*, "Semi-oblivious traffic engineering: The road not taken," in *Proc. USENIX NSDI*, 2018.

[13] M. Z. Shafiq, F. Le, M. Srivatsa, and A. X. Liu, "Cross-path inference attacks on multipath TCP," in *Proc. ACM HotNets-XII*, 2013.

[14] K. Nichols and V. Jacobson, "Controlling queue delay," *Commun. ACM*, vol. 55, no. 7, pp. 42–50, 2012.

[15] T. Hoeiland-Joergensen, P. McKenney, D. Taht, J. Gettys, and E. Dumazet, "FlowQueue-CoDel," https://tools.ietf.org/html/rfc8290, 2018.

[16] T. Høiland-Jørgensen, P. Hurtig, and A. Brunstrom, "The good, the bad and the WiFi: Modern AQMs in a residential setting," *Comput. Netw.*, vol. 89, pp. 90–106, 2015.

[17] J. Ye, K.-C. Leung, and S. H. Low, "Combating bufferbloat in multi-bottleneck networks: Theory and algorithms," *IEEE/ACM Trans. Netw.*, vol. 29, no. 4, pp. 1477–1493, 2021.

[18] C. Grazia, "A performance model for Wi-Fi frame aggregation considering throughput and latency," *IEEE Commun. Lett.*, vol. 24, no. 7, pp. 1577–1580, 2020.

[19] C. A. Grazia, "Future of TCP on Wi-Fi 6," *IEEE Access*, vol. 9, pp. 107929–107940, 2021.

[20] S. Manzoor, Z. Chen, Y. Gao, X. Hei, and W. Cheng, "Towards QoS-aware load balancing for high density software defined Wi-Fi networks," *IEEE Access*, vol. 8, pp. 117623–117638, 2020.

[21] S. Manzoor, Y. Yin, Y. Gao, X. Hei, and W. Cheng, "A systematic study of IEEE 802.11 DCF network optimization from theory to testbed," *IEEE Access*, vol. 8, pp. 154114–154132, 2020.

[22] S. Manzoor, S. Azam, R. Wójcik, and J. Domżał, "Swif: Smart Wi-Fi integration framework for mobility and resource optimization," *IEEE Access*, vol. 12, pp. 177609–177620, 2024.

[23] S. Manzoor, N. I. Ratyal, and H. G. Mohamed, "Achieving QoS in smart cities using software defined Wi-Fi networks," *IEEE Access*, vol. 11, pp. 98256–98268, 2023.

[24] S. Manzoor, M. A. Kayani, N. Ali, N. I. Ratyal, and H. G. Mohamed, "Tiwa: Achieving tetra indicator Wi-Fi associations in software defined Wi-Fi networks," *IEEE Access*, vol. 11, pp. 89520–89534, 2023.

[25] C. A. Grazia *et al.*, "Adapting TCP Small Queues for IEEE 802.11 Networks," in *Proc. IEEE PIMRC*, 2018.

[26] C. A. Grazia, N. Patriciello, T. Høiland-Jørgensen, M. Klapez, and M. Casoni, "Aggregating without bloating: Hard times for TCP on Wi-Fi," *IEEE/ACM Trans. Netw.*, vol. 30, no. 5, pp. 2359–2373, 2022.

[27] "Less Is More bottleneck repository, source scripts and tests results," https://netlab.unimore.it/sw, March 2024.

[28] T. Høiland-Jørgensen, D. Täht, and J. Morton, "Piece of cake: A comprehensive queue management solution for home gateways," in *Proc. IEEE LANMAN*, 2018.

[29] T. Høiland-Jørgensen, C. A. Grazia, P. Hurtig, and A. Brunstrom, "Flent: The flexible network tester," in *Proc. ACM ValueTools*, 2017.

[30] S. Ha, I. Rhee, and L. Xu, "Cubic: A new TCP-friendly high-speed TCP variant," *ACM SIGOPS*, vol. 42, no. 5, pp. 64–74, 2008.

[31] N. Cardwell, Y. Cheng, C. Stephen Gunn, S. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Commun. ACM*, vol. 60, no. 2, pp. 58–66, 2017.

[32] D. Taht, "Realtime Response Under Load (RRUL) test," 2012. [Online]. Available: https://www.bufferbloat.net/projects/bloat/wiki/RRUL_Spec/

[33] D. Z. Rodriguez, J. Abrahao, D. C. Begazo, R. L. Rosa, and G. Bressan, "Quality metric to assess video streaming service over TCP considering temporal location of pauses," *IEEE Trans. Consumer Electron.*, vol. 58, no. 3, pp. 985–992, 2012.

[34] A. Bouch, A. Kuchinsky, and N. Bhatti, "Quality is in the eye of the beholder: meeting users' requirements for Internet quality of service," in *Proc. ACM CHI*, 2000.

[35] V. Firoiu, J.-Y. Le Boudec, D. Towsley, and Z.-L. Zhang, "Theories and models for Internet quality of service," *Proc. IEEE*, vol. 90, no. 9, pp. 1565–1591, 2002.

Carlo Augusto Grazia is a tenure-track Assistant Professor in Telecommunications at the Department of Engineering "Enzo Ferrari," University of Modena and Reggio Emilia (UNIMORE), Italy. He earned an M.S. in Theoretical Computer Science from the University of Bologna (2012) and a Ph.D. in ICT from UNIMORE (2016), where his dissertation won the ICT Doctorate School's Best Thesis award. His work focuses on vehicular communications (V2X): cooperative awareness, VANET protocol design, GPS-based messaging, and QoS strategies for connected and autonomous vehicles. Beyond V2X, he researches Linux-based TCP/IP networking, AQM, packet scheduling, and SDN optimization.

Martin Klapez received his Ph.D. in 2017 from DIEF at UNIMORE, where he is currently a Post-Doctoral Research Fellow. He has collaborated with the Italian nanoscience National Research Center S3, and he has been involved in the EU FP7 Project PPDR-TC. His research interests verge around network softwarization, public safety networks, and safety-related V2X systems.

Maurizio Casoni is an Associate Professor of Telecommunications at DIEF in UNIMORE, Italy. He received his M.S. with honours and his Ph.D. in Electrical Engineering from the University of Bologna, Italy, in 1991 and 1995, respectively. In 1995 he was with the Computer Science Department at Washington University in St. Louis, MO, as a Research Fellow. He has been responsible at UNIMORE for the EU FP7 Projects E-SPONDER and PPDR-TC.