Who will Save the Internet from the Congestion Control Revolution?

Ferenc Fejes, Gergő Gombos and Sándor Laki
ELTE Eötvös Loránd University
Budapest, Hungary

Szilveszter Nádas
Ericsson
Budapest, Hungary
szilveszter.nadas@ericsson.com

ABSTRACT

Active queue management (AQM) techniques have evolved in the recent years, after defining the bufferbloat problem. In parallel novel congestion control (CC) algorithms have been developed to achieve better data transport performance, often assuming simple tail dropping buffers. On the other hand, AQM algorithms usually assume legacy CC (Cubic). Though all of the novel AQM and CC algorithms improve the performance under these assumptions, their co-existence has not or only partially been tested so far. Similarly, router buffer sizing was studied mainly in the 2000s, also assuming traditional CC and tail dropping buffers. In this paper we show that combining the different AQMs and CCs evolved in the past few years results in poor fairness, because assumptions used during development do not hold in this heterogeneous case. We also show that while a non-traditional AQM, using core-stateless resource sharing control, has the potential to harmonize conflicting CCs, it suffers from deployment issues. We argue that when dimensioning router buffers in networks where connections with different CCs coexist, the right choice of AQM is more important than the size of the buffer itself.

KEYWORDS

Buffer Sizing, Congestion Control, AQM, Fairness

1 INTRODUCTION

Jacobson and Karels proposed congestion avoidance and control in their seminal paper in 1988 [11] and saved the Internet from congestion collapse. Since then the Internet has gone through an enormous evolution and become one of the most complex systems ever built by humans. Today's Internet faces lots of challenging problems to satisfy the various requirements and expectations of end users. One of these being the original design choice of TCP that interprets packet loss as a congestion signal. As the technology evolved, changing NIC capacities from Mbps to Gbps and memory from KBs to GBs, buffers in the routers have become larger and larger to prevent packet losses caused by transient traffic bursts and keep the links fully utilized during congestion. Unfortunately, loss-based congestion control keeps these long buffers full by design, leading to high queueing delay and causing bufferbloat.

The question on how buffers in the routers shall be sized to achieving good utilization with reasonable queuing delay was studied quite in detail in the literature of the 2000s. One of the most comprehensive survey paper [16] from 2009 summarizes existing buffer sizing algorithms, the Bandwidth Delay Product (BPD) rule, the Stanford (or small-buffer model) [1] and the tiny buffer model. Most of the studies assume 1) a Tail Drop buffer in the bottleneck, 2) homogeneous TCP Congestion Control (CC), typically Reno or Cubic, and 3) same Round Trip Time (RTT) for each flow.

In the 2010s, the attention from appropriate sizing of physical buffers shifted towards methods aiming at reducing queueing delay through Active Queue Management (AQM) algorithms. In this decade a renaissance AQMs has been launched, leading to numerous proposals solving the bufferbloat problem. These AQM proposals often assume infinite or Bandwidth-Delay Product (BDP) long buffers and loss-based CC.

In parallel with the development of AQM methods, new CC algorithms have been designed and deployed in the past few years. The goal of the two developments was quite similar: allowing users to exploit the benefits of high speed links with acceptable end-to-end delay. Among the different proposals, one of the most prevalent is BBR [4] that applies a complex congestion model instead of simple packet losses. In contrast to its initial version, BBRv2 contains mechanisms to be fair to Cubic flows which was demonstrated in TailDrop buffers of BDP sizes [5].

With the introduction of 5G for mobile and Fiber To The Home (FTTH) for fixed Internet access, the capacity of the last mile has significantly been increased, resulting in much higher load on the Access-Aggregation Network (AAN) than before and thus moving bottlenecks to routers in the AAN from the edge. In such a network, the CCs used by the flows and the (propagation) RTTs are much more heterogeneous than in data centers and other closed enterprise networks where the whole infrastructure is controlled by a single entity. To handle the increased load on AANs and serve these high-speed bottlenecks, new router equipment is needed where

the buffer sizing is again an important factor in the design because of two reasons: 1) memory still has high impact on the cost of the device, 2) large queuing delay is not tolerated by a number of applications, thus minimizing queueing delay remains important.

In this paper, we explore a heterogeneous environment of coexisting different CC and AQM solutions under various network conditions (bottleneck capacity and RTTs), examine the efficiency of recent queue management algorithms and show potential issues: mixing fundamentally different CCs using an AQM designed for a specific CC, or similarly, mixing CCs designed to be compatible using a specific AQM. We believe that though it is possible to design a novel AQM algorithm for our specific test cases, that would not solve this general issue as new CCs with different behavior may emerge in the future causing similar incompatibility issues. Finally, we also show that a recent core-stateless AQM proposal called PPV that applies resource sharing control can solve these issues in general by harmonizing CC behavior, but it requires assistance from both end-hosts and network routers, resulting in strong constraints on the Internet-wide deployment.

2 CONGESTION CONTROL ALGORITHMS

We have selected Cubic and BBRv2 as two fundamentally different approaches that can illustrate the effect of heterogeneous CCs.

Cubic [8]. It is the most commonly used TCP CC as of today. It is designed to improve the scalability of TCP over fast and long distance networks. It also improves fairness among connections with different RTTs. Eq. 2 in [13] calculates the "BDP rule" for Cubic TCP, where due to the smaller multiplicative decrease of 0.7 the "BDP rule" is $B_{min} = 0.4 \times C \times RTT$.

BBRv2 [6]. Rather than using events such as loss or buffer occupancy, which are only weakly correlated with congestion, BBR starts from Kleinrock's formal model of congestion and its associated optimal operating point [4]. That operation point is reached when the link utilization is high while it avoids creating (long) queues at the bottleneck. BBR version 1 does not treat packet loss as a congestion signal and it has compatibility problems with Cubic [10]. BBR version 2 aims to provide fair throughput sharing among BBRv2 and Cubic connections, demonstrated using BDP long buffers [6]. Though it reacts to packets lost by keeping packet loss rate below an explicit target, might still have fairness issues, as it filters out rare packet loss events, while for Cubic CC even a single packet loss is treated as a congestion signal. In our experiments BBRv2 is used, as it is designed to be Cubic-friendly.

BBRv2 is likely not the final CC in the current CC evolution. Additionally, with more and more user space CC implementations one can expect connections with highly heterogeneous CCs present in the AAN .

3 AQM ALGORITHMS

Four queue management strategies are considered in this paper: the traditional tail dropping (TailDrop); two modern, delay-based approaches (PIE and GSP); and a recent core-stateless AQM proposal with resource sharing control (PPV). We apply these for packet dropping. ECN is excluded in our study, because the congestion response of Cubic and BBRv2 is incompatible by design when using ECN.

TailDrop is used as reference in our analysis. TailDrop simply drops the incoming packet once the queue is full. Buffer sizing of TailDrop queues has widely been studied in the past decades [16].

PIE (Proportional Integral controller Enhanced) AQM has been proposed in [15] to address the bufferbloat problem. It controls the queueing delay using a PI controller, by periodically comparing the queuing delay to a target and updating a packet drop probability. PIE has several important parameters affecting its performance, including the parameters of PI controller, update period of drop probability and the queuing delay target.

GSP (Global Synchronization Protection) [13] is a minimalist AQM which aims to avoid synchronization among the CCs of different connections for small number of connections. With more connections it acts like an on-off controller aiming to keep a given queueing delay. One of the GSP's advantages is that its parameters can easily be derived from the link capacity and are insensitive to the traffic conditions. GSP assumes classical CCs like Cubic by design and has been selected in this paper since we expect that it has poor interaction with the explicit target loss rate-based congestion detection of BBRv2.

PPV (Per Packet Value) AQM [14] is a recent core-stateless resource sharing and AQM proposal that can provide fairness between connections even with different CCs. In contrast to previous solutions, PPV requires two key mechanisms inside the network: 1) packets are marked with a packet value before the bottleneck (e.g. at the edge of the network) where the value is derived from the per connection resource sharing policy to be applied. 2) At the bottleneck packet dropping is solely based on the carried packet value, connection identification and policy knowledge is not needed. If the queue is full, the node drops packets with minimum packet value considering the packets stored in the queue and the newly arrived one. This is repeated until either the new packet can fit into the queue and then it is stored or it is dropped since all the packets in the queue have at least the same value as

the new packet. PPV by design does not drop packets from connections with throughput below their fair share. Thus even if the packet loss rate is high, none of those losses are affecting new connections in slow start, see e.g. Figure 5 in [14].

Note on DualQ AQM We are not considering DualQ AQM [3] in this work. We are aware that it can help harmonizing the Data Center TCP-like ECN response of BBRv2 and the Cubic CC in the dual queue case. Even with DualQ AQM, connections of different CCs may share the same queue, resulting in problems similar to the ones discussed in this paper.

4 TESTBED DESCRIPTION

Our testbed consists of 3 servers, all of them being Xeon E5-1660 v4 @ 3.20GHz, 64Gb RAM with Intel XL710 40GbE Ethernet NIC. They run Ubuntu 16.04 and are connected in a chain topology. The two ends are the sender and the receiver that use BBR2 alpha kernel (5.2.0-rc3). To generate persistent Cubic and BBRv2 connections (flows) we use the iperf3 tool (v3.1.3).The middle machine acting as a router uses Linux kernel 4.4 and it executes the DPDK-based AQM and bottleneck emulator. For TailDrop and PPV we use our own implementation [12] based on DPDK 19.02. For GSP and PIE the implementation in [2] with small changes are used(relying on DPDK 16.11). The propagation delay is emulated with Linux tc netem [9] by delaying ACKs on the receiver side. The goodput values of each of the connections in the 3 minute long test cases are measured by the iperf3 tool. We are working on making the source codes publicly available at [7].

5 EXPERIMENTAL ANALYSIS

Our experiments were performed with various bottleneck capacities: 100 Mbps, 1 Gbps (and 5 Gbps); and (propagation) RTTs: 10 and 100 ms. The physical buffer lengths were defined as a factor times the RTT where factors we set are 0.05, 0.1, 0.2, 0.5, 1 and 2 (e.g. 0.5 meaning that the buffer length in time is half of the RTT). Where applicable (PIE and GSP) we configured the AQMs' target delay to 75% of the buffer length[1]. The number of TCP connections (N) in different test cases were 2, 10, 20 and 100. With respect to applied CC two tests were introduced: 1) *mono-CC tests* where all the connections are using the same CC and 2) *multi-CC tests* in which half of the connections are BBRv2 and the other half are Cubic. *Multi-RTT* setups are also considered with propagation RTTs of 10 ms and 100 ms simultaneously.

In all the experiments the Jain's fairness index (among all connections), the total goodput and the relative goodput of the connection classes are analyzed. A *connection class* contains all connections with the same CC and RTT. *Relative goodput* is defined as the ratio of the average goodput within a connection class, and the ideal per-connection fair-share (the bottleneck capacity per the total number of TCP connections N). For example, relative goodput 1.0 means that the connections of the given class experience the fair-share of the total capacity in average.

Due to space limitation we only present selected results, all further results are available at [7].

[1]We run tests with several settings [7], this setting had the best fairness.

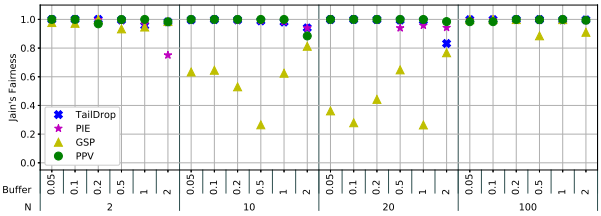

Figure 1: Mono-BBR fairness (1Gbps, 10ms).

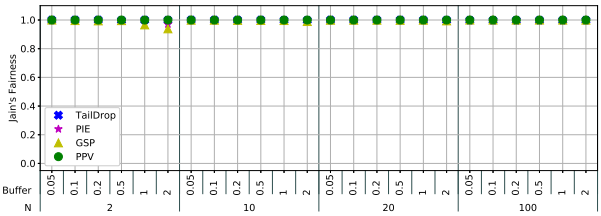

Figure 2: Mono-Cubic fairness (1Gbps, 10ms).

5.1 Mono-CC experiments with single RTT

Figures 1 and 2 depict the results of our mono-CC tests for BBR and Cubic CC methods, respectively, using a 1 Gbps bottleneck and homogeneous 10 ms RTT. Using Cubic CC almost perfect fairness can be achieved, slight deviations can only be observed when only 2 connections share a large buffer ($\geq$ 1 BDP). This is in align with the fact that most AQM methods were designed for Cubic-like CCs. In contrast, using BBR CC the results are more diverse: 1) GSP shows very bad fairness when N is between 10 and 20. This can be improved by increasing the buffer size. 2) PIE and TailDrops only show slight degradation compared to the Cubic scenario. The most significant unfairness can be seen when the buffer size is large (2 BDP) and N is small. This phenomenon also affects PPV (esp. with $N = 10$).

Note that the mono-CC experiments with other bottleneck capacity and RTT settings have shown similar Jain's fairness values. A difference worth mentioning is that with 100 ms RTT and BBR CC, we observed a slightly worse fairness than the one presented in this section (see [7]).

Observations: In general, all the investigated AQMs provides good fairness among Cubic connections. Using BBR,

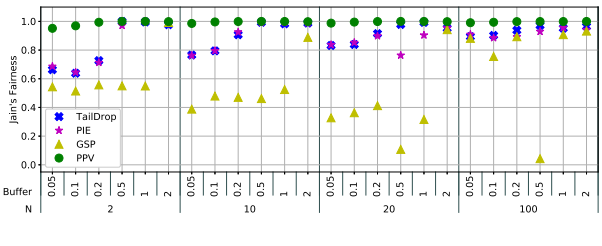

Figure 3: Multi-CC fairness (1Gbps, 10ms).

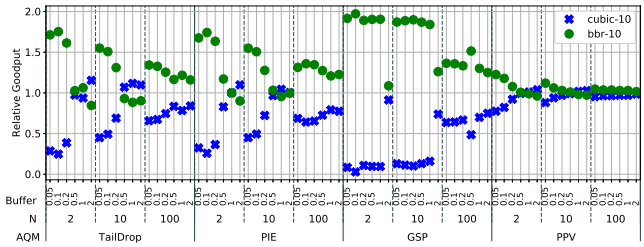

Figure 4: Multi-CC relative goodput (1Gbps, 10ms).

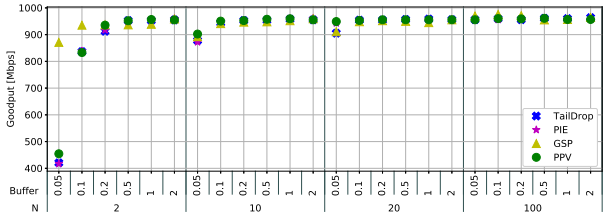

Figure 5: Multi-CC total goodput (1Gbps, 10ms).

the fairness is still reasonable for TailDrop, PIE and PPV, but GSP shows high degradation caused by the non-Cubic-like behavior of BBR. For all AQMs, large buffer sizes (2 BDP) can degrade the fairness among BBR connections if the number of connections is small.

5.2 Multi-CC experiments with single RTT

Figures 3-5 depict Jain's fairness index, relative goodput and the average goodput for the multi-CC test with the 1 Gbps bottleneck and 10 ms RTT settings. One can observe that for TailDrop the fairness is reasonable in most cases, though BBR connections might reach as much as 7 times higher goodput with small buffer sizes ($\leq$ 0.2 BDP representing less than 166 packets of 1500 bytes). The fairness is improved as N increases. Utilization issues can only be observed with small N (2 or 10) and small buffer sizes ($\leq$ 0.1 BDP). Note that 0.5 BDP is enough to achieve full link utilization and good fairness (6:4 relative goodput ratio between BBRv2 and Cubic connection groups) in general. PIE shows comparable behavior in terms of fairness and utilization to TailDrop, with slight degradation in some cases (e.g. 0.5 BDP queue size and N=20 or 100). In contrast, GSP results in significant

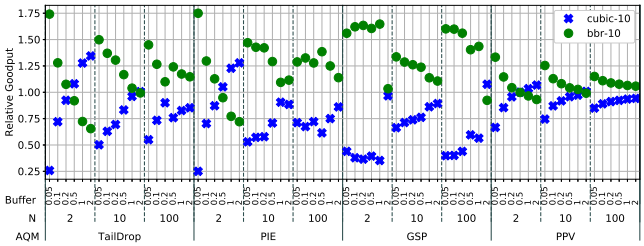

Figure 6: Multi-CC relative goodput (100Mbps, 10ms).

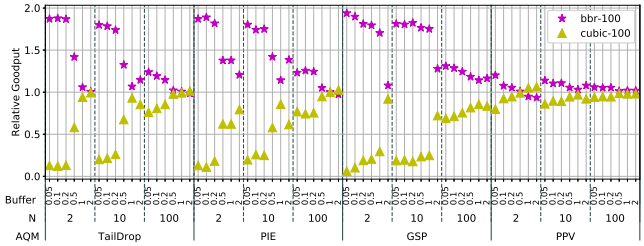

Figure 7: Multi-CC relative goodput (100Mbps, 100ms).

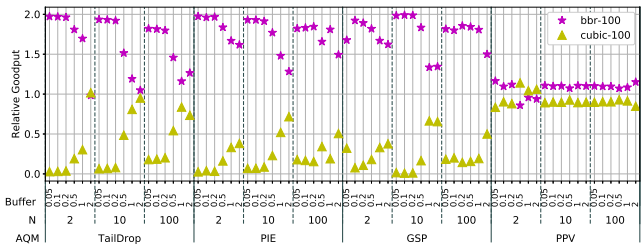

Figure 8: Multi-CC relative goodput (1Gbps, 100ms).

degradation in fairness and slightly lower link utilization. One can only observe good performance comparable to PIE and TailDrop when the buffer is 2 BDP long. In all the other cases, BBR connections get much (16 times or more) higher goodput share than Cubic ones. The number of connections also have an effect on fairness, but it is less visible than with PIE and TailDrop. The total goodput is comparable in all cases except that GSP shows a quite high goodput for 2 connections even with small buffer sizes (though fairness is quite bad in this case). Though PPV's resource sharing control helps in achieving high link utilization and better fairness than the other approaches, the relative goodput ratio between BBR and Cubic connections is also affected by the buffer size; longer buffers can help Cubic connections to compensate the overshoot of BBR ones. One can observe in cases with $N \leq 20$ that queue sizes $\geq$ 1 BDP result in slightly higher share for Cubic connections than for BBR ones in the total traffic, while if it is below the BDP, BBR connections get higher goodput.

Figures 6-8 depict similar multi-CC cases with different bottleneck capacities (100Mbps and 1Gbps) and RTT values

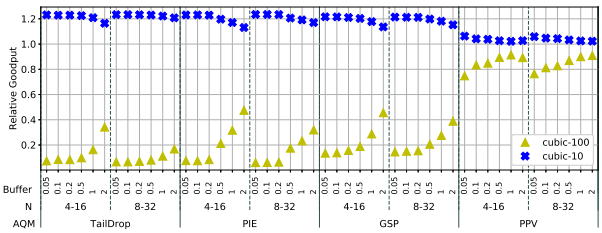

Figure 9: Cubic relative goodput (1Gbps, 10&100ms).

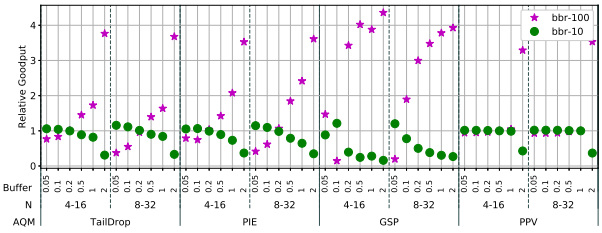

Figure 10: BBR relative goodput (1Gbps, 10&100ms).

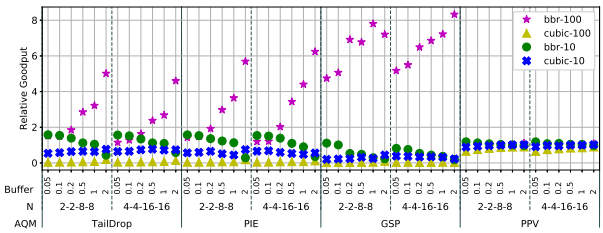

Figure 11: Multi-CC, multi-RTT rel. goodput (1Gbps).

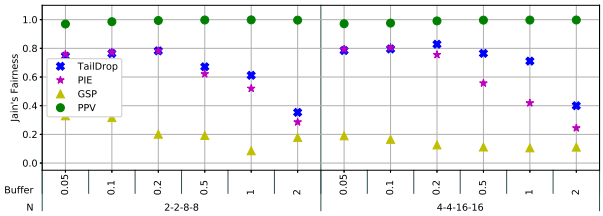

Figure 12: Multi-CC, multi-RTT fairness (1Gbps).

(10ms and 100ms). One can observe that fairness between BBR and Cubic connections is clearly worst when there is a high speed bottleneck (1 Gbps) and the RTT is large (100 ms) (Figure 8). Both TailDrop, PIE and GSP shows significant fairness degradation, totally suppressing Cubic connections in cases where buffer size is small. Though the fairness can be improved by increasing the queue size, TailDrop shows better relative goodput ratio than PIE and GSP. In cases with 100 Mbps bottleneck capacity, much better fairness can be observed, but the phenomenon is similar, showing the relationship between fairness, buffer size and N. In these cases PIE and GSP have slight fairness degradation compared to TailDrop. As seen in the previous scenario, PPV shows very good fairness even with small buffer sizes.

Observations: Utilization is less affected by the different investigated parameters. Significant degradation can only be seen if limited number of connections share a small buffer ($0.05 - 0.1$ BDP). However, the fairness between BBR and Cubic connections is strongly affected by the buffer size. Cubic connections need longer buffers to get higher goodput, approaching their ideal fair-share. In these scenarios, ≥ 0.5 BDP has proved to be enough for PIE and TailDrop to reach the 6:4 BBR to Cubic relative goodput ratio. The worst performance is shown by GSP, reflecting that it has originally been designed for Cubic-like CCs. One can also see that PPV with its in-network resource sharing mechanism can help connections with different CCs in achieving better fairness with much smaller buffer sizes.

5.3 Heterogeneous RTTs

In contrast to previous sections where homogeneous RTT for each connection has been assumed, in this section we consider connections with multiple propagation RTTs: either 10ms or 100ms. N is 4-16 (4 with 100 ms RTT and 16 with 10 ms) or 8-32, respectively. Note that the buffer size is set according to the smaller RTT (10 ms). Figures 9 and 10 depict the relative goodput with mono-CC tests using either Cubic or BBR CCs, respectively. One can observe that in case of Cubic CC, connections with 10 ms RTT reach much higher goodput than with 100 ms as expected. The fairness with TailDrop is quite bad, though it improves slightly as the buffer size increases. Compared to TailDrop there is a slight improvement with PIE and a more significant improvement with GSP even though the fairness is still pretty bad. PPV achieves reasonable fairness, which further improves as the buffer size increases.

In the BBR case, the results are less straightforward. For small buffers low RTT connections experience slightly higher goodput, while for larger buffers the high RTT connections dominate. GSP degrades fairness (compared to TailDrop) significantly, while PIE behaves similarly to TailDrop. With PPV fairness is good, except the 2 BDP case, when the connections with 100 ms RTT get much higher goodput share. The 2 BDP case is problematic with all AQMs, we guess that the 10 ms RTT connections react on delays larger than 10 ms with throughput decrease, while the 100 ms connections do not.

Figures 11 and 12 show the results of multi-CC and multi-RTT cases. N is 2-2-8-8, indicating the number of connections in CC-RTT groups: BBR-100ms, Cubic-100ms, BBR-10ms and Cubic-10ms, respectively. Even in this highly heterogeneous environment, PPV provides reasonable fairness, especially with longer buffers. Fairness with the other three AQMs (TailDrop, PIE and GSP) is quite bad, it is the worst

with GSP. In general, BBR connections get higher relative goodput than Cubic ones. Within connections of the same CC, the relations are similar to the mono-CC, multi-RTT cases. In case of buffer sizes above 0.5 BDP, BBR connections with 100 ms RTT contribute to the unfairness significantly.
Observations: The heterogeneity in RTTs solely results in significant unfairness, especially if the buffers have improper sizes. The performance degrades even more, resulting in more chaotic behavior when multiple CCs are used. Unfortunately, the requirements of Cubic and BBR are quite the opposite: for Cubic, fairness is best with long buffers; while for BBR, it is best with medium sized buffers. PPV AQM can harmonize connections with different RTTs for mono-CC cases and keeps performing well even in multi-RTT, multi-CC cases.

6 DEPLOYMENT ISSUES OF PPV

Even though the performance of PPV is very promising in all the test cases, it suffers from deployment issues. The other investigated AQMs are reasonably simple to implement and do not require additional standardization or coordination. PPV, however, requires identification of traffic aggregates, policy configuration and packet marking. The marked Packet Value requires a header field and it must be a trusted value.

One option is to perform packet marking at the edge of a domain, e.g. of the Access-Aggregation Network. AANs often have Traffic Identification Function anyway, which could also perform this task. Even in this case all typical bottlenecks within that domain must be PPV-capable. In this case the implementation of the PPV AQM in the routers is quite simple [12]. Another option is to make PPV router internal, however then the router must have all the information needed for packet marking which makes the router implementation much more complex. Internet-wide agreement in packet marking is also a possible option, but it seems rather unlikely to reach it.

7 SUMMARY

Our evaluation results confirm that the investigated modern Active Queue Management algorithms, PIE and GSP, are not prepared for modern Congestion Controls and CC co-existence. The modern AQMs rarely provide any improvement over TailDrop buffers and often their performance is worse than with TailDrop in the heterogeneous cases presented in this paper. Even though BBRv2 is designed to be fair to connections using Cubic CC, the achieved fairness is highly dependent on the buffer size and on other system parameters like the bottleneck capacity, the RTTs, and the number of connections. This even holds with TailDrop buffers that have been the design target for BBRv2. Using an AQM, which also controls resource sharing and harmonizes congestion control behavior (PPV), solves these problems, but it has deployment issues, which are not trivial to solve.

Based on our results we highlight that choosing the right scenario and the right AQM mechanism is equally, if not more, important than the actual buffer sizing method. We have demonstrated that if the buffer size is dimensioned for good performance for the homogeneous scenarios (RTT and CC), the performance for the heterogeneous cases might not be acceptable. In some scenarios a reasonable fairness for the heterogeneous cases cannot be achieved even with much larger buffers, when not using the PPV AQM or other resource sharing control mechanisms.

ACKNOWLEDGMENTS

The authors thank the support of Ericsson Hungary Ltd. This research was supported by Thematic Excellence Programme, Industry and Digitization Subprogramme, NRDI Office, 2019. The research of S. Laki was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

REFERENCES

[1] Guido Appenzeller et al. 2004. Sizing Router Buffers. In *ACM SIGCOMM '04*. 281–292.
[2] Roland Bless et al. 2018. Policy-oriented AQM Steering. *IFIP Networking Conference* (2018).
[3] Bob Briscoe et al. 2019. *Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture.* Internet-Draft draft-ietf-tsvwg-l4s-arch-04. Internet Engineering Task Force. Work in Progress.
[4] Neal Cardwell et al. 2016. BBR: Congestion-Based Congestion Control. *ACM Queue* 14, 5, Article 50 (Oct. 2016), 34 pages.
[5] Neal Cardwell et. al. 2018-07. BBR Congestion Control Work at Google. slides-102-iccrg-an-update-on-bbr-work-at-google-00. (2018-07).
[6] Neal Cardwell et. al. 2019-03. An update on BBR. slides-104-iccrg-an-update-on-bbr-00. (2019-03).
[7] Ferenc Fejes et al. 2019-10. Buffer sizing measurement results. In *http://ppv.elte.hu/buffer-sizing*.
[8] Sangtae Ha et al. 2008. CUBIC: A New TCP-friendly High-speed TCP Variant. *SIGOPS Oper. Syst. Rev.* 42, 5 (July 2008), 64–74.
[9] Stephen Hemminger et al. 2005. Network emulation with NetEm. In *Linux conf au*. 18–23.
[10] Mario Hock et al. 2017. Experimental evaluation of BBR congestion control. In *IEEE ICNP*. IEEE, 1–10.
[11] Van Jacobson et al. [n. d.]. Congestion avoidance and control.(1988). In *Proceedings of the SIGCOMM*, Vol. 88.
[12] Sándor Laki et al. 2018. Scalable Per Subscriber QoS with Core-Stateless Scheduling. *ACM SIGCOMM Industrial Demos* (2018).
[13] Wolfram Lautenschlaeger et al. 2015. Global synchronization protection for bandwidth sharing TCP flows in high-speed links. In *IEEE HPSR 2015*.
[14] Szilveszter Nádas et al. 2018. Towards a Congestion Control-Independent Core-Stateless AQM. In *ANRW '18*. 84–90.
[15] Rong Pan et al. 2013. PIE: A lightweight control scheme to address the bufferbloat problem. In *IEEE HPSR*. 148–155.
[16] Arun Vishwanath et al. 2009. Perspectives on router buffer sizing: Recent results and open problems. *ACM SIGCOMM Computer Communication Review* 39, 2 (2009), 34–39.