New buffer bloat mitigation alternatives: infrastructure versus congestion control

Alejandro D. Popovsky[1]

[1] Universidad de Palermo, Mario Bravo 1050, 1175 Buenos Aires, Argentina
aapopov@palermo.edu

Abstract. This paper studies the use of Palermo receiver side congestion control as an alternative to Active Queue Management (AQM) for end users needing to improve latency and fair sharing in their incoming traffic. Because of the end users lacking administrative access to ISP devices in order to tune their incoming bottleneck queue, this alternative results in a valid option to increase performance.

Key words. Congestion control, buffer bloat, capacity fair sharing, TCP, receiver side congestion control.

1 Introduction

The advantages of using Active Queue Management (AQM) [1] over regular FIFO queues are well known. Getting shorter queues and smaller latencies in the connection path bottleneck, improves total transmission times for short transaction oriented connections, like web navigation, remote console and remote gaming. It also improves fairness among connections sharing the bottleneck. When AQM is not used, these gains can also be obtained by using delay based congestion control in senders of the connections sharing the bottleneck.

Currently most Internet service providers (ISP's) are not implementing AQM in their infrastructure. And it is still less common to find it implemented in devices limiting the end users' capacity in accordance to the service level agreements (SLA's). It is also not common to find content servers using delay based congestion control algorithms because these algorithms generally attain a small portion of the capacity [2] when sharing the bottleneck with regular congestion control algorithms like CUBIC [5].

Delay aware congestion control algorithms had traditionally been implemented as sender side congestion control. End users are most of the time content consumers, so in general their experience comes from their perceived performance as receivers. This prevents end users from improving latency by changing their own congestion control algorithm. And they cannot consider AQM as an option, because they almost never have administrative control over the devices holding the connection bottleneck, which is generally located at the ISP's network.

A *receiver side congestion control* could enable end users to take control of this situation. A way to curb latency from the content receiver would enable end users to improve their performance even when their bottlenecks are shared among many of their neighbors' traffic, or among their own many concurrent connections.

The recently introduced Palermo Receiver Congestion Control [6] can be used by end users to take control of their performance when ISP's and content providers do not implement any solutions for the buffer bloat and capacity sharing fairness problems.

This article brings some comparisons of the results obtained by using Palermo congestion control with FIFO bottleneck versus using AQM with regular receivers, and what the end users can expect in the different sharing situations.

The contents of the paper are as follows: section II includes the sharing dynamics of the receiver side Palermo congestion control and CUBIC congestion control in the contexts of AQM and FIFO queue disciplines. The section III includes the results for tests covering the different queue and sharing situations often faced by end users. Conclusions are included in section IV.

2 Queue sharing

This section shows the bottleneck sharing dynamics of connections using Palermo receiver side control versus connections with regular receivers. The algorithm has been presented in the IETF 95 meeting [6] and the details are coming in a following paper.

A measurement setup was prepared for test connections run between 3 Linux hosts, with a Linux router in the middle running the bottleneck. This router was alternatively configured with AQM CODEL [3] or FIFO queue disciplines in the downstream path, and a NETEM delay of 70ms in the upstream path. The router interface in the downstream path had a 10mbps capacity. The senders used CUBIC in all tests.

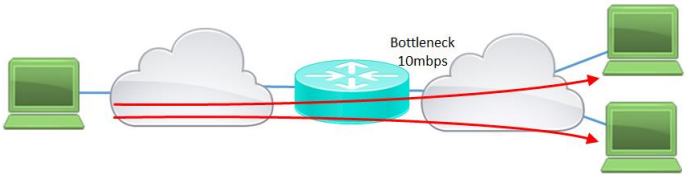

Fig. 1. Test setup with minimum round trip time of 70ms and 10mbps capacity. Bottleneck queue discipline and receiver window adaptation algorithms vary on each test.

Connections were analyzed using the UpPerformanceAnalyzer [7], running at the sender, and profiling all observed connections.

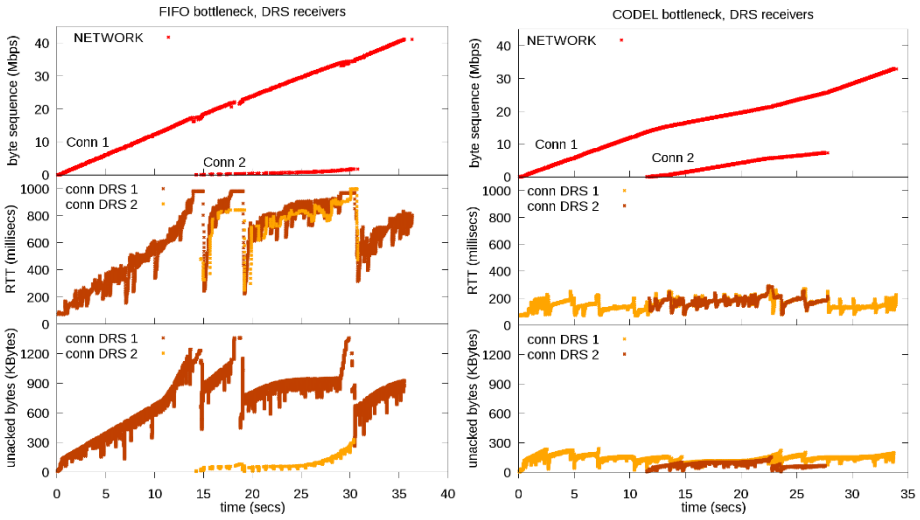

Fig. 2. The left plots show 2 connections sharing a FIFO queue, and the right plots show 2 connections sharing a CODEL queue. All receivers are regular DRS receivers and all senders are CUBIC. The top plots show the relative byte sequence advance. The middle plots show the evolution of the round trip time. And the bottom plots show the evolution of the number of unacknowledged bytes.

Before showing the Palermo receivers behavior, let us first compare FIFO and CODEL for 2 connections using regular DRS [4] receivers. These receivers adapt the receive window in order not to stop the sender by flow control. So the in-flight data length of the connections was only limited by the senders' congestion window and the queue capacity. The plots at the top of Fig. 2 show the advance of the byte sequence numbers. It is easily seen from the sequence advance slope for the first and second connections, that for the FIFO case the late connection gets a very small share of the capacity (it would eventually reach the same share after a long time). But for the CODEL case the late connection quickly gets a similar share of the available capacity.

Buffer bloat is also observed at the middle plots, showing a round trip time that grows to about 1 second for the FIFO case, but stays around 200 millisecs in the CODEL case. The cause for these effects is observed in the bottom plots of the figure, showing the number of in-flight bytes for both connections. In the FIFO case, the first connection CUBIC algorithm does not get packet losses until the queue is already above 1.4 Mbytes, generating the long queue and a corresponding large latency at the bottleneck. But in the CODEL case, the router introduces packet losses when it measures longer than configured queue waits (18milliseconds in this case), causing the sender to decrease its congestion window.

Let us see in the next test the dynamics of a FIFO bottleneck with connections having receivers using a Linux kernel with the Palermo receiver side congestion control.

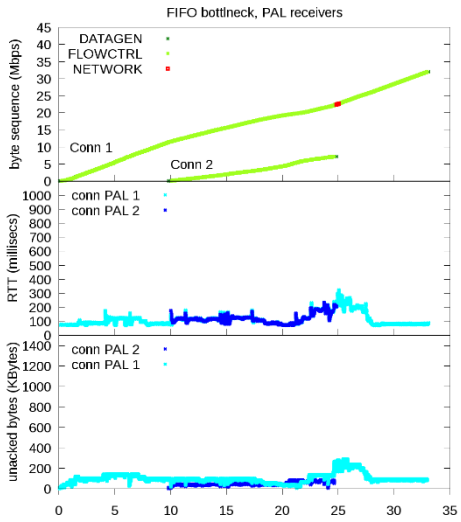

Fig. 3. These plots show 2 connections with Palermo receivers sharing a FIFO queue. Connections are mostly in green at the top plot because they are limited by the receiver window. Round trip time and the number of unacknowledged bytes remain curbed, and both connections get a similar share of capacity during their coexistence.

At the top plot of Fig. 3 it can be seen that both connections get a similar share of the available capacity as in the CODEL case. The middle plot shows that the round trip times are very similar to that obtained with the CODEL case. This is so, because as seen in the bottom plot, the Palermo receivers limited the senders using flow control to curb the length of the unacknowledged in-flight bytes to a number just above de the bandwidth delay product (BDP) generating very short queue lengths at the bottleneck.

And what can be expected when the queue is not exclusive to the end user's host or hosts and has to be shared among connections using regular and Palermo receivers? Fig. 4 shows this case, where a first coming connection receiver is using Palermo congestion control and the late connection receiver uses DRS.

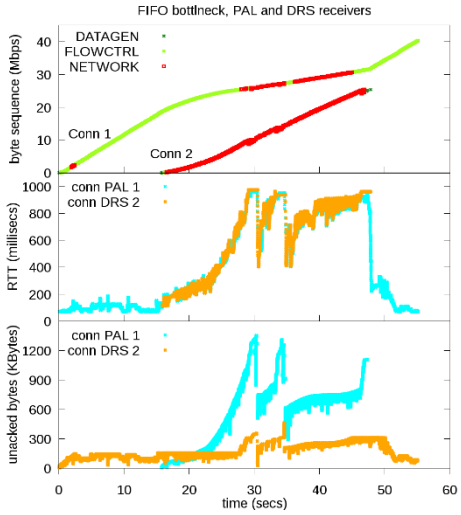

Fig. 4. These plots show 2 connections with Palermo and regular DRS receivers sharing a FIFO queue. Palermo connection switch to network limitation (red) and regular congestion control during its coexistence with the DRS connection, although it takes a long time to regain fairness.

Before the arrival of the second connection, the first connection achieves a near optimal round trip time while getting all the available capacity. When the second connection arrives (at t=17secs), the Palermo receiver senses its share decrease, but cannot tell if it is because of a single bad behaved connection or because of several well behaved connections. So it waits until it reaches a (kernel) configured threshold, and reverts to a regular DRS behavior preventing the receive window to refrain the sender. From this point onwards (at t=28secs) the capacity sharing evolves in the same way as in a case of 2 regular CUBIC connections, taking a long time to achieve a fair sharing. At this point any action by the first connection receiver to prevent buffer bloat would be useless because the second connection is already flooding the queue.

In this example, the second connection leaves the bottleneck before the first connection CUBIC sender grows its congestion window to achieve the same rate as the second connection. At that point the Palermo receiver regains its buffer bloat preventing behavior and decreases the receive window to regain the optimum round trip time.

3 Performance tests

The measurements included in this section illustrate about the effects of introducing Palermo congestion control in the receivers, and how it compares against the performance of having AQM in the bottleneck.

For the tests we used the same test setup as in the previous section. During the tests, a long background connection was flowing through the bottleneck while late arriving connections of various lengths came in, alternatively having DRS and Palermo receivers.

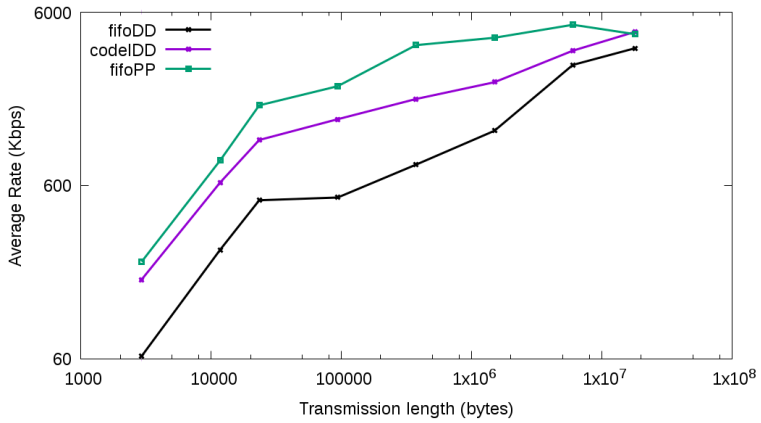

Fig. 5. Average rate obtained by late arriving connections sharing the bottleneck with a very long background transfer. Black curve represents the FIFO case with regular DRS receivers. The magenta curve represents the CODEL case with regular DRS receivers. And the green curve represents the FIFO case with Palermo receivers.

In Fig. 5 the curves for the average rate obtained by a second arriving connection is plotted for transmission lengths varying from 3Kbytes to 18Mbytes. The first curve (black) represents the case with a FIFO bottleneck and regular receivers both for the background and short connections. The second curve (magenta) represents the case for a CODEL bottleneck and regular receivers. And the third curve (green) represents the case with the FIFO bottleneck, but with Palermo receivers.

The rates for short connections are very low for all three curves because the round trip time plays a larger influence in the total connection time. For this same cause, the curve for the FIFO bottleneck and regular receivers (black), having very long queues at the bottleneck caused by the background connection, has the smaller rates for all short to medium lengths late arriving connections.

It can be seen that Palermo receivers with FIFO bottleneck (green) obtained better rates for all connection lengths, even better than those obtained by regular receivers with CODEL bottleneck (magenta).

At long connection lengths all rates became very similar because the queue delay has a smaller influence in the total connection time.

In the tests of Fig. 5 the background and late arriving connections sharing the bottleneck used the same receiver types. What could be expected of the mixing of receivers types? In the tests plotted in Fig. 6 the background connection has a regular DRS sender, and the late arriving connections have Palermo receivers. The bottleneck is FIFO.

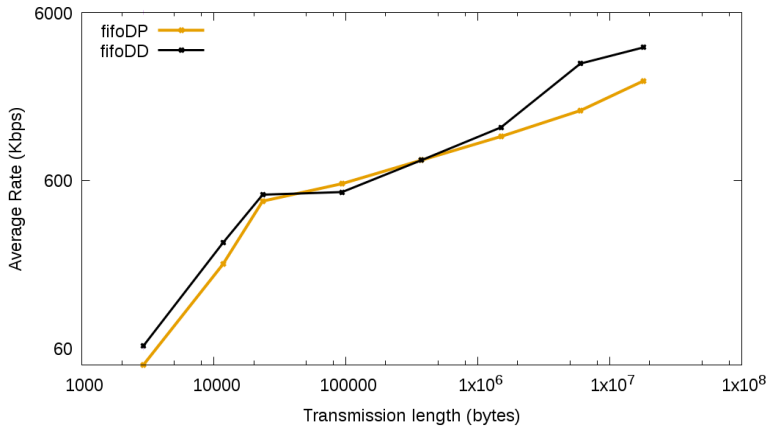

Fig. 6. Average rate obtained by late arriving connections sharing a FIFO bottleneck with a very long background transfer with a regular DRS receiver. The black curve is for regular DRS receivers on the late coming connections. The brown curve is for Palermo late coming receivers.

For short and medium lengths late arriving connections, both receiver types obtain almost the same rates. Although low rates because of the background regular receiver connection flooding the bottleneck. At long transmission lengths, Palermo receivers get about 30% smaller rates, because of their fairness oriented behavior.

The case of a background Palermo receiver connection with regular late coming DRS receivers was not plotted because is very similar to the case with Palermo late coming receivers observed in Fig. 5.

These tests are in accordance to the real Internet traffic experience mentioned in [6]. The Linux kernel on the web proxy for one of the buildings of the Palermo University in Argentina, was changed for the modified version implementing the Palermo receive window adaptation, obtaining a 54% improvement in total download time for popular newspapers web pages, while sustaining very long downloads in parallel from several sites. Similar results were obtained for most sites having many small objects (such as graphics), each one needing an individual transaction affected by the bottleneck queue latency, significantly reduced by the Palermo receiver.

In our experience analyzing traffic for ISPs connected to the Internet Exchange Points of the Argentine Chamber of Internet (Cabase) [7] we have found that a large portion of the providers generate customers' bottlenecks that are (most of the time) located at the service level agreement (SLA) enforcing devices, and these bottlenecks are not shared with other customers. Of course there are also many (overbooked) providers generating bottlenecks shared among many customers, different from the SLA enforcer, that limit their throughput to smaller levels than those stated in the SLA.

For the first group of providers, customers can switch to Palermo receiver side congestion control and obtain the performance improvements of the case of the FIFO queue with all Palermo receivers, equal or better than that obtained with AQM bottlenecks. For the second group, customers can still obtain a significant improvement depending on the level of overbooking of their particular provider. Because a bottleneck already flooded with other customers' traffic leaves no options to well-behaved connections for any improvements.

3 Conclusions

Receiver side congestion control is a valid option for end users willing to take control of their performance and the fairness of their capacity sharing, when their ISPs and content providers have not taken measures to prevent buffer bloat.

Palermo receiver side congestion control can obtain the same performance improvement levels as those achieved when using AQM at the bottleneck, when this bottleneck is mostly shared by Palermo receivers or other well behaved connections. And can also obtain an acceptable performance when sharing the bottleneck with other queue-flooding algorithms.

A sender side version of the Palermo congestion control algorithm is currently being studied in order to aim for the same performance goals from the content provider side.

References

1. Athuraliya, S., Low, S. H., Li, V. H., Yin, Q.: REM: active queue management. In: IEEE Network, vol. 15, no. 3, pp. 48-53. (2001).
2. Welzl, M., Ros, D.: A Survey of Lower-than-Best-Effort Transport Protocols. In RFC 6297, IETF. (2011).
3. Nichols, K., V. Jacobson, V.: Controlling queue delay. In: Communications of the ACM, vol. 55, no. 7, pp. 42-50. (2012).
4. Fisk, M., Feng, W.: Dynamic Right-Sizing in TCP. In: Los Alamos Computer Science Institute Symposium (LACSI'01). Santa Fe, New Mexico, USA, (2001).
5. Ha, S., Rhee, I., Xu, L.: CUBIC: a new TCP-friendly high-speed TCP variant. In: ACM SIGOPS Operating Systems Review, vol. 42, no. 5, pp. 64-74. ACM. (2008).
6. Popovsky, A.: Peeking at the bottleneck, opportunities for buffer bloat prevention. In: Internet Congestion Control Research Group, IETF 95, Buenos Aires (2016).
7. Popovsky, A., Fritz, P.: Traffic Performance Analysis at the NAP. In: LACNIC 18 / LACNOG 2012. Montevideo, Uruguay (2012).