Packet Trimming to Reduce Buffer Sizes and Improve Round-Trip Times - Extended Abstract

Cedric Westphal, Kiran Makhijani, Richard Li
Futurewei
Santa Clara, CA
cedric.westphal,kiranm,richard.li@futurwei.com

ACM Reference Format:
Cedric Westphal, Kiran Makhijani, Richard Li. 2019. Packet Trimming to Reduce Buffer Sizes and Improve Round-Trip Times - Extended Abstract. In *Proceedings of Buffer Sizing Workshop (BS'19).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The size of the buffers in a wide area network is intrinsically linked to the behavior of the end-to-end transport protocol. Indeed, the buffer's role is dual: reducing the number of packet drops in the network when multiple flows are contending for the same constrained resource, and ensuring that the outgoing link is fully utilized.

Large buffers help with both these objectives, but introduce congestion delays in the network. The larger the buffer, the longer the potential wait of a packet before being sent on the wire.

In order to reduce this congestion latency, shallow buffers are preferred, but they introduce packet drops and retransmission. In particular, the feedback loop of the transport protocol are typically not quick enough to reduce the transmission rate when congestion occurs.

In [9], a mechanism was suggested to selectively drop chunks of a packet in order to reduce congestion in the network by reducing the payload size.

In one instance, the payload may be constituted of several chunks, some with higher priority/importance than others, and the lowest priority chunks can be dropped preferrentially upon congestion. In another instance, the payload may be constituted of chunks drawn from a fountain code so that any number of chunks may be dropped along the way. As long as sufficiently many reach the receiver, the payload can be decoded.

The idea of dropping part of the packet has been explored in the context of the datacenter. Trimming methods have been proposed for data centers before, such as "cut payload" (CP)[4] and NDP [7].

The goal in these prior schemes is to achieve fast re- transmissions for data center workloads with very shallow buffers. They work by trimming the *whole* payload and only keep the header. This work in the Data Center, since the dropped payload may be retransmitted fast enough to be re-sent to the node that dropped the payload before this buffer has emptied. This does not generalize to the case of the WAN.

We contend that selectively trimming the packet rather than the whole payload offers a less drastic mechanism that would work in the wide area network. It is an immediate reaction to congestion that still sends a signal to the end-to-end congestion feedback loop. For this reason, it is better suited to shallow buffers in the WAN.

This is work in progress, and designing an overall end-to-end transport protocol that provides the rate adaptation to minimize the number of packets being trimmed is an open issue.

Related Works: Considerable work has been reported on transport-layer protocols (see, for instance, [10] for a recent survey), and reducing Internet delays has been a major goal in this space [2]. Buffer sizing [1, 5, 11] and reducing the buffer size [6] have been studied previously as well. This prior work established a rule that the buffer depth should be $C \times RTT/\sqrt{N}$, where C is the link capacity, and N is the number of concurrent flows. Large buffers, and bufferbloat in particular, have introduced unwanted latency in the network [3]. We contend that packet trimming, associated with a properly designed transport protocol, can reduce buffer depth to reduce latency due to congestion. Our work is inspired by the work by Li et al. [9], which uses the concept of selectively dropping chunks in a packet based upon some congestion criteria.

2 THEORETICAL FOUNDATIONS

We motivate the benefit of dropping only part of the packets when the network gets congested. In essence, this increases the *per-packet* processing rate at the cost of losing some information in the packet. This is a reaction to congestion that allows the receiver to still receive some data and react accordingly. Receiving some data is better than having an entire packet dropped.

We consider a utilization where the incoming rate is equal or higher than the link capacity. For illustrative purposes, we consider a packet composed of two chunks. This means that, when one chunk is dropped at first, then the processing rate for a packet at the router is doubled. It takes half the time to send a packet half the size. Therefore, twice as many packets can be forwarded on the congested link (of course, with a loss of half the data in the original packet). While we only present analytical models for two-chunk packets, the models are easy to generalize to the case of multiple chunks per packet.

2.1 Fluid Model Approximation

Consider a fluid system in which requests arrive as a process of rate λ packets per second and packets are processed at the server with a rate μ packets per second. Each packet is composed of two chunks of equal size. If $\lambda > \mu$ then the request process builds up in the server's buffer. Assume that the system drops one chunk of all new incoming packet when the system reaches a threshold T (say, when the buffer is half full). This can be viewed as a *per-packet* processing rate increasing to 2μ.

In this system, it is easy to see that the system will stay empty if $\lambda < \mu$. The system will overflow if $\lambda > 2\mu$, given that 2μ is the maximum processing rate. The system will converge to the threshold and keep the buffer half full if $\mu \leq \lambda \leq 2\mu$. Consequently, if the incoming rate is between μ and 2μ, the system operates with a buffer of constant size B, where B is the buffer utilization threshold, and the wait time for a task is simply B/C, where $C = 1/\mu$ is the link capacity. The three modes of operation of such a system are therefore

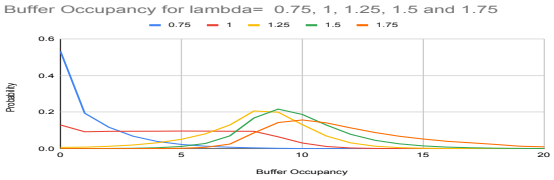

Figure 1: Buffer occupancy probability for a system with 2 chunks per packet and buffer depth of 20 packets, threshold at 50% occupancy

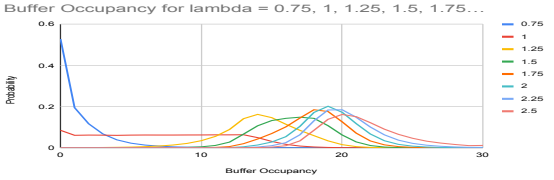

Figure 2: Buffer occupancy probability for a system with 3 chunks per packet and buffer depth of 30 packets, thresholds at 50% and 66.7% occupancy

to have: (a) an empty buffer ($\lambda < \mu$), (b) a buffer at the threshold T ($\mu \leq \lambda \leq 2\mu$), or (c) an overflowing buffer.

The waiting time for a stable system is either 0 or B/C. Because the system has constant buffer size, the outgoing packet rate is equal to the link capacity after filling up the buffer. This implies that the fraction α of packets that are shrunk is equal to: $(1-\alpha)\lambda + \alpha/2\lambda = \mu$ or $\alpha = 2(1-\mu/\lambda)$.

A fraction α of the packet is shrunk, and the $(1-\alpha)$ remaining goes through whole.

2.2 M/D/1 Model Approximation

Instead of a fluid model, we consider that packet arrivals occur according to a Poisson process with rate λ. For simplicity, packet lengths are assumed to be identical (prior to any trimming operation), which results in processing times that are deterministic. Packets are served immediately if they arrive in an empty system, or put in a queue if the server is busy processing a packet. We assume the following policy: if the queue holds fewer than or equal to B tasks, then the packet is processed with rate μ (and mean service time $1/\mu$). If the queue holds more than B tasks, then the service rate is 2μ. The arrival rate for any state is simply λ for any state, while the service rate is μ for states 0 to B, and 2μ for states $B+1$ to k. Define by p_k the probability that the buffer holds k packets and let $\rho = \lambda/\mu$.

Define by π_k^μ to be the stationary distribution of the M/D/1 queue with arrival rate λ and service rate μ. This can be recursively computed, cf for instance [8]. A coupling argument shows that the system is stable for $\lambda < 2\mu$ and that for $\lambda > \mu$, once the buffer occupancy goes beyond B, the system behaves exactly as the M/D/1 queue with rate 2μ. This means that we can compute p_k for $k \geq B$ as $\pi_{(k-B)}^{2\mu} P(L > B)$ where L is the buffer occupancy. An upper bound on the mean buffer occupancy is therefore:

$$L = B + \frac{1}{2}\left(\frac{\rho^2/4}{(1-\rho/2)}\right) \quad (1)$$

Because the system is stable, the computation of α still applies for $\mu < \lambda < 2\mu$. Figs. (1) and (2) show the probability distribution of the buffer occupancy, with a clear peak at the target threshold value. We can observe a fast drop-off in the probability of a buffer occupancy over the highest threshold.

Fig. 3 displays the evolution of the trimming process as the buffer size grows. It can be observed that for very small buffer, there is

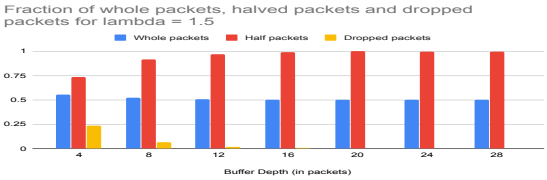

Figure 3: Rate of Packets Transmitted Intact, Halved or Dropped, for $\lambda = 1.5$

obviously a lot of packets that get dropped (namely at a rate of 0.25 for a total rate of 1.5 packets per unit of time).

However, for relatively small packet size (with depth 8 or 12 packets), we see that the behavior converges to a trimming of 2 packets out of 3 and no packet drops.

This is the relationship that needs to be investigated in relation to the end-to-end control loop of the transport protocol.

3 NEXT STEPS

We have considered a simple method to stabilize buffer size near a target value. This method is similar in principle to NDP

The interaction with the queue behavior and the end-to-end control loop for congestion management is for further study as well: dropping chunks of a packet indicates the congestion to the end point. How to respond to it is an open challenge.

REFERENCES

[1] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '04, pages 281–292. ACM, 2004.

[2] B. Briscoe, A. Brunstrom, A. Petlund, D. Hayes, D. Ros, I. Tsang, S. Gjessing, G. Fairhurst, C. Griwodz, and M. Welzl. Reducing internet latency: A survey of techniques and their merits. *IEEE Communications Surveys Tutorials*, 18(3):2149–2196, 2016.

[3] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson. Bbr: Congestion-based congestion control. *ACM Queue*, 14:20–53, 2016.

[4] P. Cheng, F. Ren, R. Shu, and C. Lin. Catch the Whole Lot in an Action: Rapid Precise Packet Loss Notification in Data Centers. In *11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*, pages 17–28, 2014.

[5] A. Dhamdhere and C. Dovrolis. Open issues in router buffer sizing. *SIGCOMM Comput. Commun. Rev.*, 36(1), Jan. 2006.

[6] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden. Routers with very small buffers. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, April 2006.

[7] M. Handley, C. Raiciu, A. Agache, A andVoinescu, A. Moore, G. Antichi, and M. Wójcik. Re-architecting Datacenter Networks and Stacks for Low Latency and High Performance. In *ACM SIGCOMM*, 2017.

[8] L. Kleinrock. *Theory, Volume 1, Queueing Systems*. Wiley-Interscience, New York, NY, USA, 1975.

[9] R. Li, K. Makhijani, H. Yousefi, C. Westphal, L. Dong, T. Wauters, and F. D. Turck. A framework for qualitative communications using big packet protocol. *ACM SIGCOMM Workshop on Networking for Emerging Applications and Technologies (NEAT'19)*, 2019. https://arxiv.org/abs/1906.10766.

[10] M. Polese, F. Chiariotti, E. Bonetto, F. Rigotto, A. Zanella, and M. Zorzi. A survey on recent advances in transport layer protocols. *CoRR*, abs/1810.03884, 2018.

[11] A. Vishwanath, V. Sivaraman, and M. Thottan. Perspectives on router buffer sizing: Recent results and open problems. *Computer Communication Review*, 39:34–39, 04 2009.