Understanding the Achieved Rate Multiplication Effect in FlowQueue-based AQM Bottleneck

Jonathan Kua
School of Information Technology
Deakin University, Geelong, Australia
jonathan.kua@deakin.edu.au

***Abstract*—The progressive adoption of Active Queue Management (AQM) and the popularity of Dynamic Adaptive Streaming over HTTP (DASH)-based streaming services have motivated the development of adaptive chunklets. Chunklets significantly improves the Quality of Experience (QoE) of video streaming applications in the presence of cross-traffic, which is known as the Achieved Rate (AR) multiplication effect. However, the detailed behaviour of chunklets and their implications on queuing dynamics have not been well-explored. In this paper, we instrumented a fine-grained, packet-driven FreeBSD queue measurement kernel module, and present an experimentally validated system model to help us better understand the AR multiplication effect in FlowQueue-based AQM bottlenecks. We experimentally demonstrated the accuracy of our system model across a wide range of network settings, along with providing fine-grained insights into queuing behaviours, including the detection of hash collisions. Our approach can provide network operators and researchers with greater network visibility where AQM schemes are deployed.**

***Index Terms*—DASH, TCP, AQM, PIE, CoDel, FQ-CoDel**

I. Introduction

In a typical consumer home broadband environment, various Internet traffic applications compete for a fair share of the last-mile ISP bottleneck bandwidth capacity. According to Sandvine, Netflix accounted for 11% and YouTube accounted for more than 15% of global Internet traffic during the world-wide stay-at-home orders in 2020[1]. Streaming companies use technologies based on the Dynamic Adaptive Streaming over HTTP (DASH)[2] standard to deliver their content to consumers.

The Achieved Rate (AR) multiplication effect is a performance improvement achieved by using *adaptive chunklets* – a novel technique we previously proposed to provide a better QoE for consumers when DASH-based streams have to compete with other greedy/bulk traffic flows in consumer home broadband environments [1]. This effect is most significant when the last-mile bottleneck implements FlowQueue (FQ)-based Active Queue Management (AQM) schemes. FQ-based schemes are one of most effective and widely deployed AQM schemes in modern networks to mitigate the *bufferbloat* phenomenon. Bufferbloat happens when network routers use excessively large drop-tail or first-in-first-out (FIFO) buffers (due to low memory prices, and the conventional wisdom that large buffers improve throughput by absorbing packet bursts and reducing packet drops). Many applications rely on Transmission Control Protocol (TCP) to regulate their data transport, so the use of large FIFO buffers can induce high latencies in congested bottleneck. This is particularly detrimental to latency-sensitive applications such as adaptive video streaming applications.

The IETF has standardised Controlled Delay (CoDel)[3], Proportional Integral controller Enhanced (PIE)[4], and FlowQueue-CoDel (FQ-CoDel)[5] AQM schemes to mitigate the ramifications caused by bufferbloat. These AQM schemes aim to provide low end-to-end delays, by using queuing delays as an indicator of standing queues within the buffer, and drop/mark packets as appropriate. We have previously demonstrated that adaptive chunklets can achieve the AR multiplication effect under a wide range of network settings [1], [2]. However, further research is required to understand this effect in greater detail, so we can more accurately model the intricate interactions between chunklets and the observed performance impact.

In this paper, we present a system model using a fine-grained packet-driven queue statistics logging kernel module in FreeBSD. This model can provide network operators and researchers with greater visibility into the queuing dynamics within network routers. We demonstrated the accuracy of our system model with extensive testbed-based experiments. Section II reviews background information and related work. Section III describes our system model, experimental result and analysis. Section IV concludes and outlines future work.

II. Background and Related Work

In this section, we briefly describe the AR multiplication effect, FlowQueue-based AQM schemes, and related work.

A. The Achieved Rate multiplication effect

The Achieved Rate (AR) multiplication effect is a performance improvement phenomenon we observed when DASH clients implements adaptive chunklets [1], [2]. In DASH systems, video clients retrieve video chunks (which are encoded at multiple different bitrates – with different qualities and segmented into multi-second chunks then stored at the HTTP server). It uses an adaptive bitrate (ABR) algorithm to determine which chunks are the most suitable to be retrieved

[1]Sandvine Global Internet Phenomena Report 2020
[2]https://www.iso.org/standard/57623.html
[3]https://tools.ietf.org/html/rfc8289
[4]https://tools.ietf.org/html/rfc8033
[5]https://tools.ietf.org/html/rfc8290

in order to achieve high QoE. It typically uses recent per-chunk throughput measurements (*a.k.a* achieved rates) and/or playout buffer levels as feedback signals [3].

Chunklet-enabled clients experienced a larger perceived bandwidth share when competing with cross-traffic across the same bottleneck. This influences the ABR algorithm to select video chunks encoded at higher bitrates or Representation Rates (RR) – leading to the AR multiplication effect. This is most noticeable when FQ-based AQM schemes are deployed at network bottlenecks. The AR multiplication effect can be explained simplistically as follows. When a DASH client uses *N* chunklets and competes with *M* bulk/greedy TCP flows across a FQ-CoDel bottleneck, each chunklet flow is perceived as a unique flow to FQ-CoDel. DASH's AR then sees a multiplication effect with an approximate increased rate of $AR_N = (\frac{N}{N+M}) * C$, allowing the client to retrieve RR sustainable up to AR_N [1].

B. FlowQueue-based AQM schemes

CoDel drops/marks packets depending on packet sojourn time within the queue. It measures queuing delays by timestamping packets upon ingress and egress. If the minimum queuing delay is lower than the target delay (5ms by default), than the packets are preserved in the queue. If it is higher for at least one interval (starting at 100ms), CoDel will start dropping packets and adjust its next drop time using a control law. PIE drops/marks packets depending on the calculated drop probability during packet ingress. At regular time intervals (30ms by default), a background process re-calculates the probability based on queue delay deviations from the target delay (15ms by default) and queue delay trends.

FQ-CoDel and FQ-PIE are hybrid schemes that aims to control queuing delays while sharing bottleneck capacity relatively evenly among competing flows. Both schemes use the modified Deficit Round Robin (DRR) scheduler to manage two lists of queues (old and new queues) to provide brief periods of priority to lightweight/short burst flows. It first classify incoming traffic flows using a flow hashing function, then dynamically create the sub-queues for each flow, with each queue being controlled by separate instances of CoDel or PIE. FQ-CoDel was first implemented in Linux kernel v3.14[6], and implemented in FreeBSD-11 along with FQ-PIE[7]. Since v5.9, Linux kernel supports FQ-PIE as default queuing discipline[8].

C. Related work

Many studies have demonstrated the benefits of deploying AQM schemes [4], [5], [6]. However, many of these studies rely on end-hosts' congestion control mechanisms to infer queuing behaviours. There is currently limited literature on queue measurement in AQM bottlenecks, especially in FQ-based AQM bottlenecks. In [7] the authors presented *ConQuest*, which is a mechanism for fine-grained queue measurement in a Software Defined Networking (SDN) environments. *ConQuest* (in the data plane) can identify flows that contribute significantly to the queuing delays. Simulation and experimental results confirmed the efficacy of their method, achieving up to 90% precision on a 1ms timescale. In [8] the author presented a mathematical formulation to accurately model the latency of sparse flows in FQ-CoDel. The paper seek to understand the exact conditions for when a flow is considered sparse and prioritised by FQ-CoDel.

Our work complements these studies, and provide an approach to derive ground truth directly from within the AQM-managed queue itself, recording fine-grained, packet-driven queue statistics and dynamics in real-time.

III. Evaluation Methodology and Results

In this section, we first present our experimental testbed setup and describe our system model. We then demonstrate our experimental results and analysis.

A. Experimental testbed setup

We use the same experimental testbed setup, network settings and performance metrics as described in [1], based on [9], [10]. In this work, we instrumented an in-house FreeBSD kernel module to help us better understand the queuing dynamics in FreeBSD-based AQM schemes. This module provides fine-grained, per-packet granularity queue statistics, such as a flow's source/destination IP address/port number, total number of packets/bytes in queue, cumulative packet drops, scheduler ID, sub-queue index, sub-queue queue length, and the number of packets dropped by each sub-queue.

This information provides insights at a per-packet granularity. This has enabled us to reconstruct the measured AR values in an effort to understand the performance impact of chunklets. This kernel module also enabled us to detect hash collisions (when the same hash value is generated for multiple flows, causing them share the same sub-queue). Performance benchmarking in our testbed has showed that loading/enabling the kernel module has minimal impact on system performance.

B. System model

As introduced in Section II, the conceptualisation of chunklets is primarily based on the idea of leveraging FQ-based AQM's per-flow fair capacity sharing capabilities. The AR multiplication effect is mainly influenced by how much chunklets within a cluster overlap in time. On a finer-grained time scale, the higher the number of packets from different chunklet flows (within the same cluster) are interleaved in a FQ-based bottleneck, the greater the AR multiplication effect.

To better understand the AR multiplication effect, we developed a model for reconstructing the measured AR values based on our knowledge of (i) the number of active queues at the AQM-enabled bottleneck when a video packet is dequeued from any DASH sub-queues, (ii) the number of non-DASH sub-queues; and (iii) the bottleneck bandwidth capacity. Our work establishes ground truth and provides a solid foundation for future work in predictive modeling of chunklet-enabled

[6]https://elixir.bootlin.com/linux/v3.14/source/net/sched/sch_fq_codel.c
[7]https://reviews.freebsd.org/rS300779
[8]https://elixir.bootlin.com/linux/v5.9/source/net/sched/sch_fq_pie.c

streaming. In the following section, we illustrate this model using the notations described in Table I.

TABLE I
NOTATIONS USED IN OUR SYSTEM MODEL

Notation	Description
N	Number of chunklets
M	Number of competing bulk flows
C	Rate-limited bottleneck bandwidth
C'	Theoretical bandwidth share for DASH flow
D	Degree of overlap
AR	Achieved Rates as measured by the client
AR'	Reconstructed Achieved Rates

The degree of overlap *D* is calculated by averaging the number of active DASH sub-queues as observed within the chunk transfer window (chunk's first byte denoted as *start time* and last byte as denoted by *end time*). A normalised *D* of 1 signifies all chunklets within a cluster totally overlap in time whereas 0 signifies no overlap.

$$D = \frac{\sum_{k=starttime}^{k=endtime} N_k}{N} \tag{1}$$

The theoretical Internet Protocol (IP) layer bottleneck bandwidth share *C'* is an approximation of the bandwidth available, as a fraction of the total bandwidth capacity *C*, to a DASH client should it use *N* chunklets to compete with *M* bulk flows.

$$C' = \frac{N}{N+M} * C \tag{2}$$

After *D* and *C'* is calculated, AR values can be reconstructed by simply scaling the theoretical bandwidth share *C'* with the overlap factor *D*. Since AR is an application-layer metric, we took into account the overhead of TCP/IP headers and scale it by a factor of 1460/1500 (bytes). The reconstructed AR values are then smoothed over four chunks to match the *dash.js* streaming client's AR measurements.

$$AR' = D * C' \tag{3}$$

$$AR' = D * (\frac{N}{N+M} * C) \tag{4}$$

We experimentally test and validate our system model across all experiment trials described in [1]. Due to space constraints, we only present a selected set of experiment scenarios in the following section.

C. Results and analysis

In this section, we present experiment scenarios where the DASH client is using $N = \{1, ..., 10\}$ chunklets, competing with $M = 4$ downstream bulk TCP flows for bottleneck capacity $C = 12\,Mbps$ across an FQ-CoDel bottleneck. Figure 1 illustrates in the time domain how the reconstructed AR values closely follow the measured AR values (AR values as measured at the application-level by the DASH client) in an experimental trial where $N = 6$ chunklets.

As we look closely at the 10-second time window $t = 200s$ to $t = 210s$ of the same trial, we can understand how the number of active queues can influence the AR multiplication effect. Figure 2 shows the number of active DASH sub-queues within FQ-CoDel, where five complete chunks are delivered. This number is measured when a video packet is dequeued. In this example, a value of 6 indicates all six FQ-CoDel sub-queues are backlogged with packets, hence bandwidth will be shared evenly among all sub-queues. Sub-queues belonging to bulk TCP flows are always backlogged with packets. The AR multiplication is at its maximum when all DASH sub-queues are backlogged with packets, allowing *AR'* to approximate *C'*.

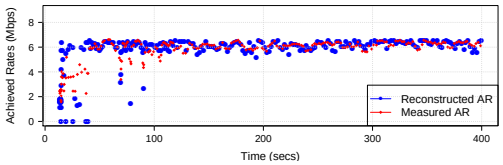


Fig. 1. Reconstructed and measured Achieved Rates vs time, $N = 6$, $M = 4$.

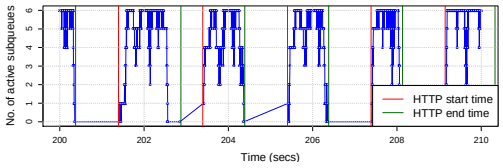


Fig. 2. Number of active DASH queues for $t = 200s$ to $t = 210s$ ($N = 6$, $M = 4$), measured when a video packet is dequeued.

Figure 3 consolidates the experimental results with a box-plot. The distribution of measured AR values is overlaid with the theoretical AR (*C'*) value and the median values of the reconstructed AR values (*AR'*). The medians of reconstructed AR closely track the medians of the measured AR. As N increases, both medians deviate further from the theoretical AR curve, due to the decrease in overlapping probabilities (the likelihood of TCP dynamics from each chunklet flow synchronising in time decreases, as indicated in Figure 4).

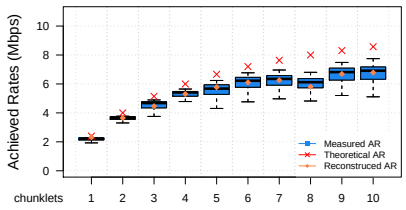
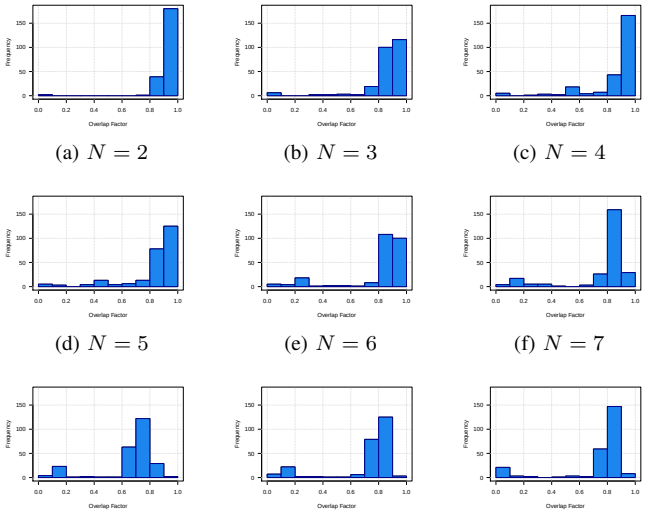


Fig. 3. Achieved Rates (measured, theoretical, reconstructed medians) vs N across $12/1Mbps$ FQ-CoDel bottleneck, $20ms$ base RTT, competing with 4 downstream bulk TCP flows.

Figure 4 presents the histogram of overlap factors (D) for $N = \{2, ..., 10\}$. Note that an overlap factor of 1 indicates that all chunklets within a cluster totally overlap in time whereas 0 indicates they do not overlap. As N increases, we observe that the probability of D approximating 1 decreases. This is because the likelihood of all chunklets overlapping in time decreases (due to increasingly unsynchronised TCP dynamics across multiple connections). Consequently, both measured and reconstructed AR deviates further from the theoretical AR.

Note that in N=8, the measured and reconstructed AR matches nicely but they deviate quite significantly from the theoretical AR (and has a value lower than N=7). Further investigation using our model has showed that hash collisions have occurred in this scenario, causing a reduction in achievable throughput.



(a) $N = 2$ (b) $N = 3$ (c) $N = 4$

(d) $N = 5$ (e) $N = 6$ (f) $N = 7$

(g) $N = 8$ (h) $N = 9$ (i) $N = 10$

Fig. 4. Overlap factors histogram for $N = \{2, ..., 10\}$, $M = 4$. A value of 1 means total overlap, whereas 0 means no overlap. Partial overlap is indicated by value between 0 and 1.

Figure 5 presents the cumulative distribution function (CDF) distribution plots of reconstructed and measured AR values for the corresponding scenarios. The results showed a very similar distribution curve for both reconstructed and measured AR values, thus proving the accuracy of our system model.

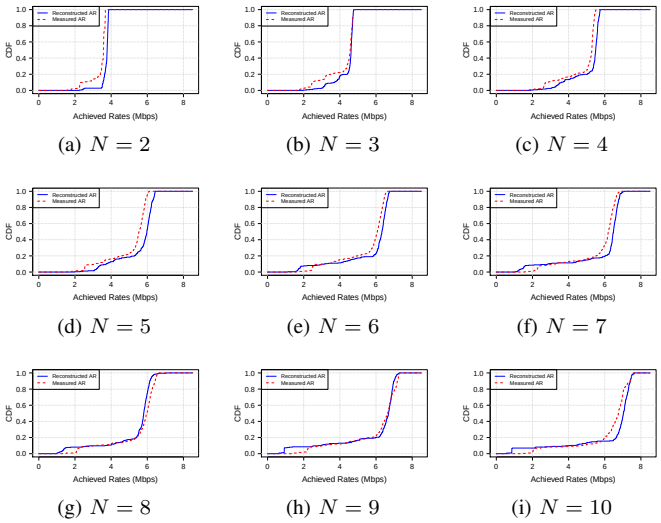
IV. Conclusions and Future Work

In this paper, we have presented a system model that seeks to understand the Achieved Rate multiplication effect in FlowQueue-based AQM bottleneck. We instrumented a fine-grained packet-driven FreeBSD kernel module that monitors queuing statistics directly from within the queue. Our system model has helped us better understand the interactions between video streaming applications, transport protocols behaviours and queuing dynamics at network bottlenecks, including the detection of hash collisions. Using a controlled testbed, we experimentally evaluated and demonstrated the accuracy of our model across a wide range of network settings.

Future work includes using machine learning models to interpolate queuing data to inform decision-making processes, such as improving ABR algorithms and/or adaptive chunklets. Other work includes generalising our model to include emerging AQM schemes in environments where there is a greater mix of application traffic sources, and analysing the sparseness of latency-sensitive flows in such environments.

Acknowledgments

The author would like to sincerely thank Grenville Armitage (Netflix, Inc and Swinburne University of Technology) and Philip Branch (Swinburne University of Technology) for providing valuable guidance on this work, and acknowledge Rasool Al-Saadi (Al-Nahrain University) for developing the FreeBSD kernel modules used in this research.



(a) $N = 2$ (b) $N = 3$ (c) $N = 4$

(d) $N = 5$ (e) $N = 6$ (f) $N = 7$

(g) $N = 8$ (h) $N = 9$ (i) $N = 10$

Fig. 5. CDF distribution plots for $N = \{2, ..., 10\}$, $M = 4$. Both measured and reconstructed Achieved Rates follow a very similar distribution, proving the accuracy of our system model.

References

[1] 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, Dec. 2019.

[2] J. Kua and G. Armitage, "Optimising DASH over AQM-enabled Gateways Using Intra-Chunk Parallel Retrieval (Chunklets)," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, 2017, pp. 1–9.

[3] J. Kua, G. Armitage, and P. Branch, "A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1842–1866, 2017.

[4] J. Kua, G. Armitage, and P. Branch, "The Impact of Active Queue Management on DASH-based Content Delivery," in *2016 IEEE 41st Conference on Local Computer Networks*, Nov 2016, pp. 121–128.

[5] J. Kua, S. H. Nguyen, G. Armitage, and P. Branch, "Using Active Queue Management to Assist IoT Application Flows in Home Broadband Networks," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1399–1407, Oct 2017.

[6] J. Kua, P. Branch, and G. Armitage, "Detecting Bottleneck Use of PIE or FQ-CoDel Active Queue Management During DASH-like Content Streaming," in *2020 IEEE 45th Conference on Local Computer Networks (LCN)*, 2020, pp. 445–448.

[7] X. Chen, S. L. Feibish, Y. Koral, J. Rexford, O. Rottenstreich, S. A. Monetti, and T.-Y. Wang, "Fine-Grained Queue Measurement in the Data Plane," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, ser. CoNEXT '19. New York, NY, USA: ACM, 2019, p. 15–29.

[8] T. Høiland-Jørgensen, "Analyzing the Latency of Sparse Flows in the FQ-CoDel Queue Management Algorithm," *IEEE Communications Letters*, vol. 22, no. 11, pp. 2266–2269, 2018.

[9] J. Kua, R. Al-Saadi, and G. Armitage, "Using Dummynet AQM - FreeBSD's CoDel, PIE, FQ-CoDel and FQ-PIE with TEACUP v1.0 testbed," Centre for Advanced Internet Architectures, Swinburne University of Technology, Melbourne, Australia, Tech. Rep. 160708A, 08 July 2016.

[10] J. Kua and G. Armitage, "Generating Dynamic Adaptive Streaming over HTTP Traffic Flows with TEACUP Testbed," Centre for Advanced Internet Architectures, Swinburne University of Technology, Melbourne, Australia, Tech. Rep. 161216A, 16 December 2016.