Buffer Sizing in Wireless Networks: Challenges, Solutions, and Opportunities

Ahmad Showail, Kamran Jamshaid, and Basem Shihada
CEMSE Division
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
{ahmad.showail, kamran.jamshaid, basem.shihada}@kaust.edu.sa

ABSTRACT

Buffer sizing is an important network configuration parameter that impacts the Quality of Service (QoS) characteristics of data traffic. With falling memory costs and the fallacy that 'more is better', network devices are being overprovisioned with large buffers. This may increase queueing delays experienced by a packet and subsequently impact stability of core protocols such as TCP. The problem has been studied extensively for wired networks. However, there is little work addressing the unique challenges of wireless environment such as time-varying channel capacity, variable packet inter-service time, and packet aggregation, among others. In this paper we discuss these challenges, classify the current state-of-the-art solutions, discuss their limitations, and provide directions for future research in the area.

1. INTRODUCTION

Buffers are designed to absorb transient traffic bursts. However, arbitrarily sized buffers can degrade network performance. Large buffers lead to long queuing delays, while very small buffers may result in network under-utilization. Ideally the buffers need to be sized just large enough to keep the link saturated at close to full utilization while minimizing queueing delays.

With declining memory prices and the fallacy that 'more is better', network devices are increasingly being over provisioned with large buffers that aim to improve throughput by limiting packet drops. While throughput is the dominant performance metric, packet forwarding latency also impacts user experience. This includes not only real-time traffic such as VoIP, video conferencing, and networked games, but also web browsing, which is sensitive to latencies on the order of hundreds of milliseconds. Studies have indicated that a one second delay in page load times of e-commerce websites can significantly impact customer conversion. Further, large queueing delays also impact the stability of core internet protocols such as TCP, which rely on timely notification of congestion information to respond effectively.

Buffer sizing for wired networks has been extensively studied (*e.g.* [1], [2], among others). A well-known rule of thumb is to have buffers slightly larger than the Bandwidth Delay Product (BDP) [14] of the network. However, there is limited work in understanding the impact of buffer sizing on wireless networks. Wireless networks have significantly different characteristics from wired networks. For example, the wireless link capacity is not constant and may vary over time due to interference. Moreover, the packet inter-service time may vary due to the random access MAC and frame retransmissions following Automatic Repeat reQuest (ARQ). Also, recent MAC enhancements such as frame aggregation allow transmission of large frame aggregates creating further challenges in efficient managing of buffer sizing techniques in wireless networks.

To illustrate the impact of buffer size on wireless network performance, we performed a number of experiments on a Linux-based Wi-Fi testbed by transferring a large file between two hosts connected via 802.11n radios. We vary the link rate every 50 s: we start at 144.4 Mb/s, then drop it to 65 Mb/s, 6.5 Mb/s, and finally 13 Mb/s. We monitor the growth of the TCP congestion window as well as the Round Trip Time (RTT) between the two hosts. We also measure the queue utilization of the FTP server and the amount of dropped packets by the sender. Our results are shown in Fig. 1. We observe that the TCP congestion window peaks at 1.6 million bytes (window scaling is enabled by default on Linux hosts), with RTT peaking at around 2.6 s. Most of these 'in-flight' TCP segments are queued up at the Linux transmit queue (*txqueue*) interface (default size of 1000 packets), contributing to large queueing delays that lead to the high RTT delays. Most operating systems use some variant of loss-based TCP congestion control algorithms. Having large buffers prevents a timely dropping of a packet that is required for conveying network congestion to the TCP sender, leading the TCP congestion window to shoot up to the high values observed in our experiment. We note that with these large buffers, the queue utilization never drops to 0, despite the TCP sender reducing its congestion window multiple times during the experiment. Slower links lead to the longest queueing delays. The variations in RTT clearly suggest that a uniform static buffer size cannot be used for wireless networks that are fundamentally dynamic in nature. Similar performance degradation due to bloated buffers has also been reported for cellular networks [6]. Fig. 1 also shows that packet drops increases with network load. This is in agreement with what Fu *et al.* found earlier [5].

In this paper, we describe the challenges of buffer sizing in wireless data networks. We then present a summary of buffer sizing solutions available in literature. We classify these solutions for single-hop and multi-hop wireless networks. This delineation is necessary because multi-hop wireless networks introduce a new set of challenges that require rethinking the packet delay paradigm. We also discuss recent Active Queue Management (AQM) techniques. Since these operate at a different control point, they may be used to complement direct manipulation of buffer sizes.

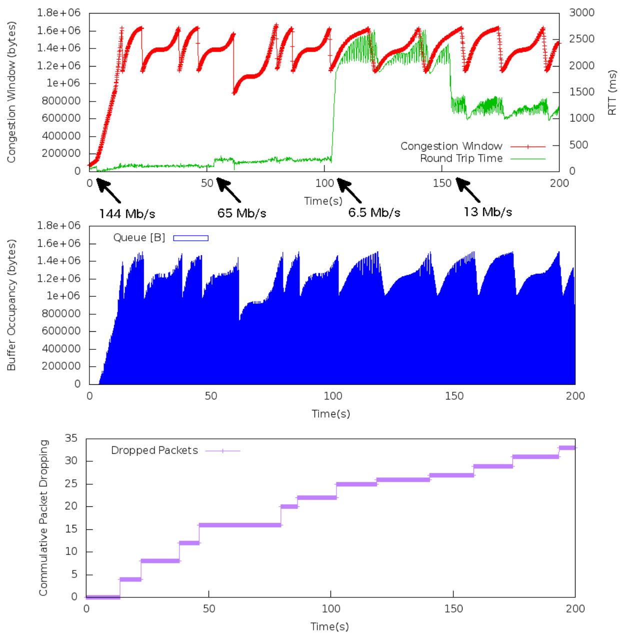


Figure 1: TCP congestion window, RTT, egress queue utilization and the number of dropped packets for a TCP flow in a 802.11n wireless testbed with varying link rates over time. Buffer size values correspond to values in the stock Linux kernel.

To ground our discussion, we also present some performance measurements from our wireless network testbed. We conclude the paper by presenting our view on future directions to address the buffer sizing problem in the wireless domain.

2. BUFFER SIZING CHALLENGES

Buffer sizing techniques and their impact on performance of wired networks is well-understood [1] [2] [14]. However, these techniques cannot be directly applied to the wireless domain because of several unique challenges described below.

2.1 Link scheduling

The wireless spectrum is a shared resource between a set of neighboring nodes. Interference considerations may require that only one of these nodes transmit at a time. The number of transmit opportunities available to a node is partly dependent on the number of neighboring nodes that are also actively contending for channel access. Thus, unlike a wired link, a wireless link cannot be scheduled independently of its neighboring nodes. This limits the available, usable capacity of a wireless link, as it now varies depending on the network topology and the number of competing flows. Therefore, while the physical wireless link rates may connect at 300 Mb/s, the actual rate achievable by a flow may be significantly less and would further vary over time based on the link scheduling constraints.

2.2 Adaptive link rates

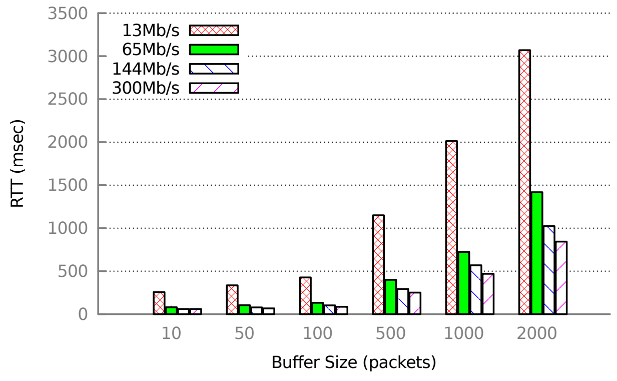
Wired link rates are constant and often known a priori. In contrast, link rate adaptation algorithms dynamically set the wireless link rate in response to changing network conditions. These link rates may exhibit significant variations over time, *e.g.*, the link rate for a 802.11n radio may vary from 6.5 Mb/s to 600 Mb/s. Depending on the link rate adaptation algorithm, these link rates may vary on time scales ranging from milliseconds to minutes. This has implications on the network BDP and the resulting buffer size required for saturating the link.

We have performed various experiments to study the impact of variable link rates on wireless network dynamics. Our testbed uses Atheros 802.11n wireless cards on Linux machines with ath9k drivers. The default txqueue size on current Linux kernels is 1000 packets. We use a radio channel that does not interfere with our campus production network. We transfer a large file between two wireless nodes and simultaneously monitor the goodput as well as other TCP statistics. We repeated this experiment at multiple link rates and txqueue buffer sizes while enabling and disabling wireless frame aggregation. Fig. 2 shows the end-to-end delay and network goodput over a single-hop wireless network. We observe that there is no optimal buffer size that works across the four link rates used in our experiments. Large buffers work well with fast links where they can saturate the link capacity while maintaining acceptable delays. Small buffers are better suited for slow links, where they limit the queueing delays while giving similar goodput as large buffers at the price of packet drop. For example, in Fig. 2(b) and 2(d) we observe that changing the buffer size from 10 to 50 packets shows only a minor goodput improvement for 13 Mb/s to 144.4 Mb/s link, yet shows a 30% increase in goodput for the 300 Mb/s link. However, this buffer size cannot be used across all link rates as the RTT with 13 Mb/s link already exceeds 250 ms over a single wireless hop. Such delays are unacceptable when these queues are shared with real-time traffic. Fig. 2(f) shows the packet drop % for each buffer size. We observe that shrinking the buffer size increases the number of dropped packets. This is in agreement with the results of Dhamdhere and Dovrolis [2], who show that extremely small buffers lead to high loss rates.

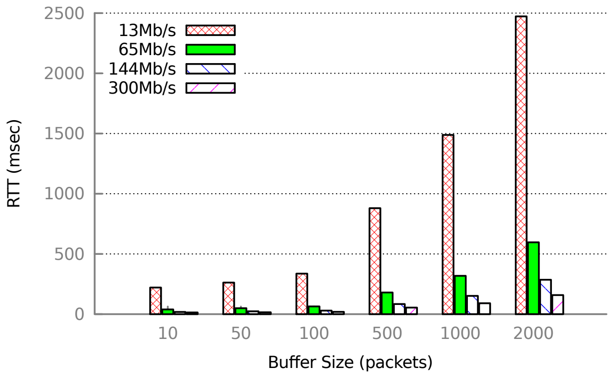
2.3 Frame aggregation

While these challenges are common across wireless networks in general, standard-specific enhancements introduce additional complexity. For example, 802.11n standard specifications include various enhancements to improve channel capacity utilization, including frame aggregation. Aggregate MAC Protocol Data Unit (A-MPDU) aggregates multiple IP packets back-to-back into a single frame. A-MPDU is limited in size to 65,535 B (bound by the 16-bit length field in the HT-SIG headers) and can carry a maximum of 64 subframes (limited by the Block Acknowledgement frame). Fig. 2 shows that A-MPDU aggregation increases the network goodput by 5× for 300 Mb/s link with large buffers, and up to 3× with small buffers. It also shows that big buffers lead to slightly higher packet drop rate when A-MPDU frame aggregation is enabled. Both of these observations are attributed to the fact that big buffers allows large aggregates, as shown in Fig. 3. The only exception is at 6.5 Mb/s link rate as frame aggregation is disabled at this rate in our hardware (transmitting a large A-MPDU frame at this link rate violates the 4 ms frame transmit duration regulatory requirement).

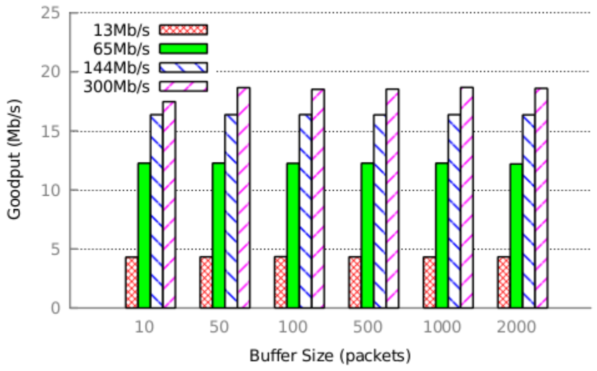
UDP flows are used in real-time communication, such as



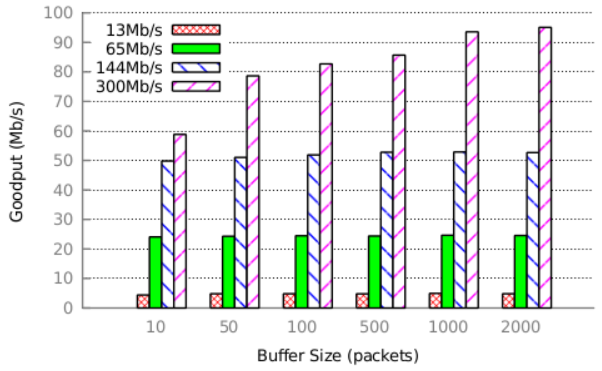
(a) Latency without A-MPDU aggregation



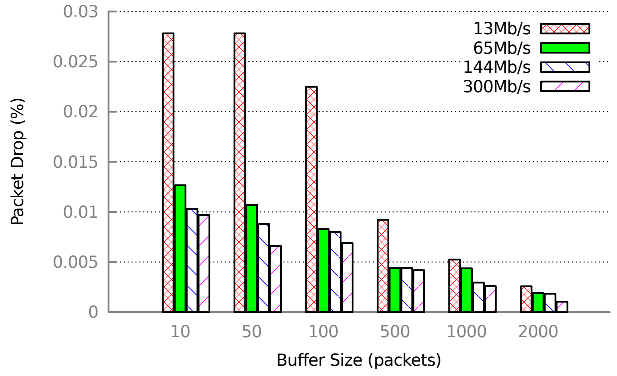
(b) Latency with A-MPDU aggregation



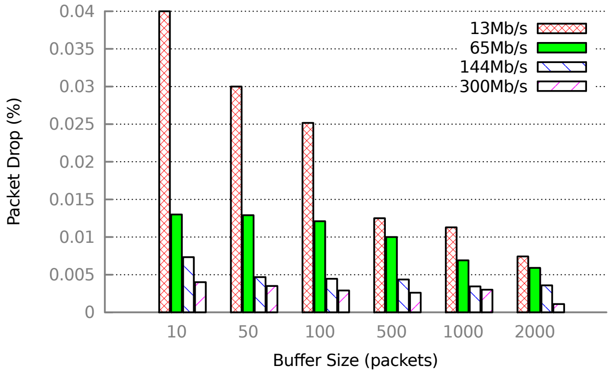
(c) Goodput without A-MPDU aggregation



(d) Goodput with A-MPDU aggregation



(e) Drop without A-MPDU aggregation



(f) Drop with A-MPDU aggregation

Figure 2: Latency, goodput and packet drop of a TCP large file transfer over various link rates and buffer sizes

online games, IPTV, and VoIP. Hence, it is important to compare such flows to other flows that favor reliable delivery over timely delivery. We repeated the same experiments with UDP instead of TCP to evaluate the interaction of frame aggregation with UDP flows. The only difference in the experiment setup is enabling the default rate control algorithm in Linux (Minstrel) instead of fixed link rates. Latency, goodput, and packet drop results are shown in Fig. 4. UDP consistently achieves higher goodput compared to TCP. This is due to multiple factors: (1) UDP does not incur the overhead of transmitting TCP ACK segments, and thus the capacity spared can be used to send additional data packets. (2) TCP employs congestion control algorithms, while UDP can saturate the medium with a sustained traffic rate. In our experiments, the only case when TCP goodput outperforms UDP is with the 10 packets buffer; we attribute this to the high UDP drop rate (around 9%) which limits the performance of A-MPDU frame aggregation. Fig. 4(b) shows that UDP goodput stabilizes when the aggregation is disabled, though we observe variations in results with aggregation for buffers larger than 10 packets. This is because large buffers allow longer aggregates. For

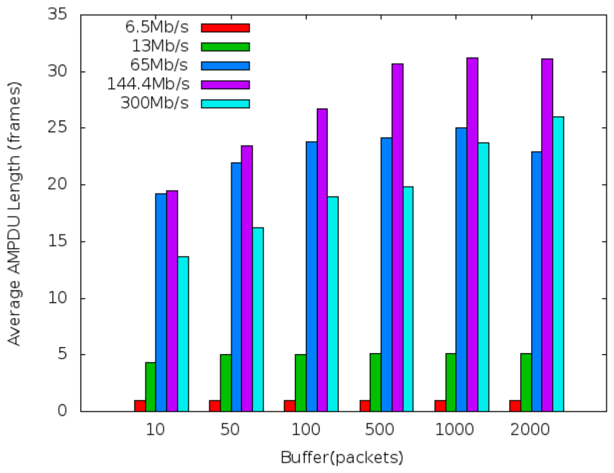


Figure 3: Average A-MPDU length of a TCP large file transfer for various link rates and buffer sizes

example, the average number of frames per aggregate increases from 16.1 for the 50 packets buffer to 17.6 for a 2000 packets buffer. Fig. 4(a) shows that UDP delays are always smaller than TCP; this is because UDP does not incur extra delays for connection management and reliability.

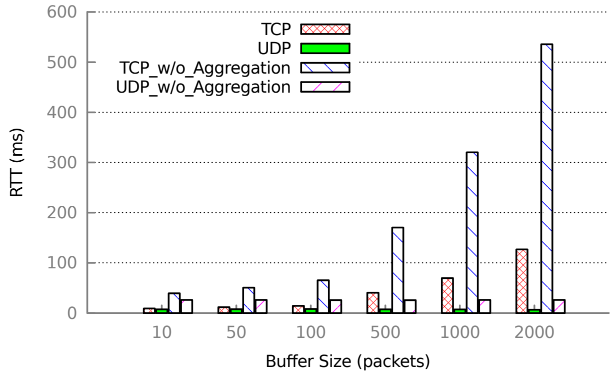
2.4 Variable packet inter-service time

The packet inter-service rate for any wired link is deterministic for a given packet size. In contrast, the packet inter-service rate for a wireless link is variable due to several reason. First, MAC protocols such as CSMA/CA use random backoffs to reduce the probability of a collision. Second, the wireless link Bit Error Rate (BER) is typically orders of magnitude higher than that of a wired link (BER of 10^{-5} to 10^{-3} for a wireless link vs. 10^{-15} to 10^{-12} for a wired link). Wireless MAC protocols use ARQ to provide reliability. As a result, a packet may be transmitted multiple times (*e.g.*, up to 7 retries per IEEE 802.11 standard specifications) before it is successfully received, contributing to variations in inter-service delays.

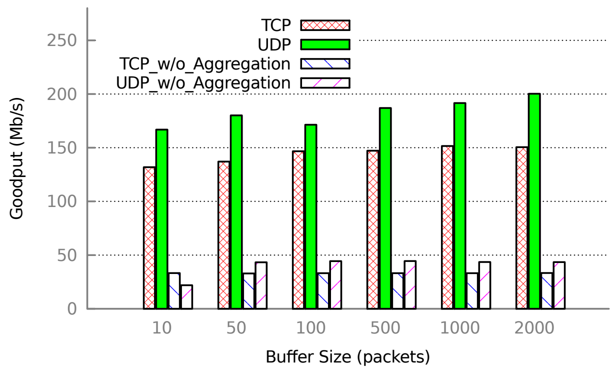
2.5 Multi-hop challenges

Multi-hop wireless networks further exacerbate the challenges described above. Due to the shared nature of wireless spectrum, a flow not only competes for transmission opportunities with other flows (*inter-flow contention*), but also contends with its own packet transmissions along the hops to the destination (*intra-flow contention*). This adds to the link scheduling and variable packet inter-service time challenges described above. Further, the abstraction of a 'bottleneck' in a shared wireless medium translates to a set of nodes in a part of the network that experiences high channel contention. A flow may traverse multiple hops in this part of the network, and hence the bottleneck spans multiple nodes. It is unclear how to size buffers in this distributed environment. Moreover, a multi-hop node may also relay traffic for other nodes in the network, and so it needs additional measures to provide isolation and fairness between flows.

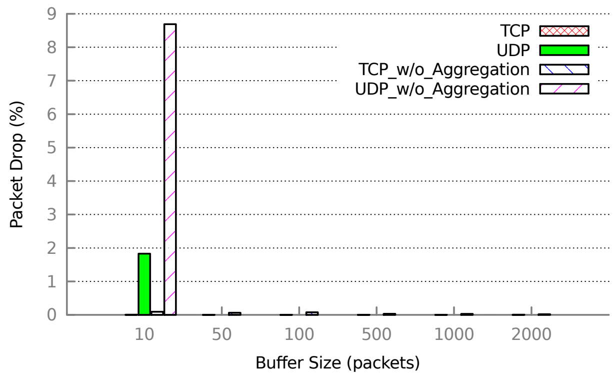
To illustrate the effects of mulithop topologies on network dynamics, we performed a large file transfer between two hosts in our testbed while varying the number of intermediate hops from one to four. We observe from Fig. 5 that the maximum RTT increases by 3× (from 2.69 to 7.95 seconds) when the hop count changes from one to two while the network goodput decreases by half (from 4.87 to only 2.41 Mb/s). Similar behavior, *i.e.* longer delays and lower goodput, is experienced when we increase the hop count between the sender and the receiver. These persistently full buffers also affect the network fairness characteristics. To study this, we repeated these experiments with a bidirectional file



(a) Latency analysis



(b) Goodput analysis



(c) Drop analysis

Figure 4: Delay, goodput and packet drop comparison of both TCP and UPD flows with and without A-MPDU frame aggregation

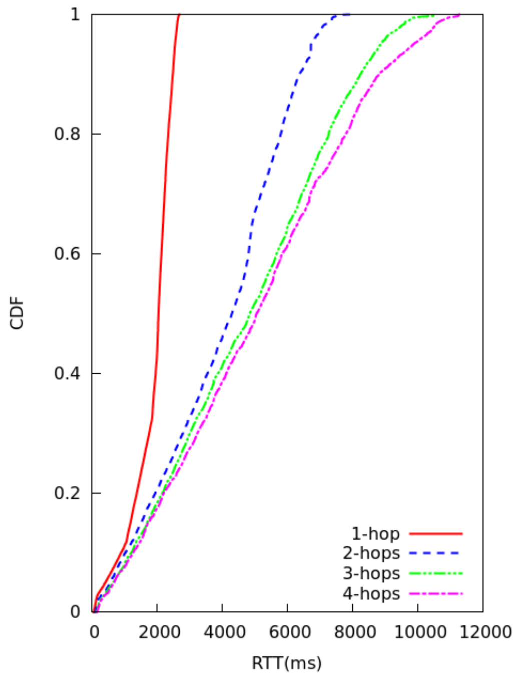


Figure 5: End-to-end delay CDF of a large file transfer over topologies with increasing number of hops

Flow #	1 hop	2 hops	3 hops
Flow 1	18.53 Mb/s	16.98 Mb/s	9.85 Mb/s
Flow 2	18.45 Mb/s	0.10 Mb/s	0.77 Mb/s

Table 1: Per flow goodput for a bidirectional file transfer over a multihop wireless network

transfer. Tab.1 lists per flow goodput over various hops. We observe severe unfairness between the two flows, with the flow starting first starving out the flow starting later in the experiment. This is because the flow starting first quickly saturates the buffers at the intermediate hosts, resulting in dropped packets and timeouts for the flow starting later.

2.6 Implementation challenges

Implementing buffer sizing mechanisms on modern operating systems represents another challenge as buffers exist on multiple layers in the software stack. It is unclear as to which of these buffers should be tuned. For example, the Linux network stack uses *txqueue* to buffer packets between the kernel network subsystem and the device driver. *txqueue* may be scheduled using a variety of queueing disciplines. In addition to *txqueue*, packets may also queue at the device driver ring buffers (also called Tx/Rx descriptors). These buffers are used to hide the latency of the interrupt processing overhead. One of the main issues with device driver ring buffers is the fact that it is sized by the number of descriptors which vary in size. As a result, the actual time to empty the buffer cannot be estimated precisely.

3. BUFFER SIZING SOLUTIONS

We categorize the current research on buffer sizing methods for wireless networks based on network architecture to single-hop solutions and multi-hop solutions. These are discussed below.

3.1 Single-Hop Network Solutions

Li *et al.* [7] have studied adaptive tuning of IEEE 802.11 Access Point (AP) buffers. They proposed three algorithms: emulating BDP (eBDP), Adaptive Limit Tuning (ALT), and A* algorithm.

eBDP extends the classical BDP rule to AP buffers. Because the link rates in a wireless network may change dynamically, the eBDP algorithm adaptively sets the buffer size limit based on measurements of current mean service time for packet transmission, T_{serv}. T_{serv} is the time difference between the packet getting to the head of the queue and its successful transmission. The goal is to limit T_{serv} to some predefined maximum T_{max}. The AP buffer Q_{eBDP} is decreased when T_{serv} increases, and vice versa. This algorithm is formalized in the following equation:

$$Q_{eBDP} = min(T_{max}/T_{serv} + c, Q^{eBDP}_{max}) \qquad (1)$$

where Q^{eBDP}_{max} is the maximum allowable buffer size and c is a constant added to accommodate short-term packet bursts.

Although simple in concept, eBDP has a fundamental limitation. Packet service time is a good indication of channel contention, but does not capture queueing delays. ALT feedback algorithm improves on eBDP as follows: it monitors buffer occupancy and modifies the size accordingly. However, ALT suffers from low convergence rate, *e.g.*, it takes three minutes to converge to a small buffer value when the number of competing upload flows increase from 0 to 10.

A* is a hybrid approach that combines the two methods mentioned above. This algorithm calculates two queue sizes: (1) Q_{eBDP}, by monitoring the mean service time of packet transmissions, and (2) Q_{ALT}, by monitoring the buffer occupancy percentage. It then simply chooses the minimum of these two values. The ALT part of the A* can be used to further tune the buffer size. One of the main limitations of the A* algorithm is that it only works on AP buffers; it is unclear if similar scheme can also be implemented on client devices to manage queueing delays for uplink flows.

To summarize, eBDP deals with changes in service rate while ALT monitors the queue occupancy in order to avoid long queueing delays. The two schemes may complement each other, and form the basis of A*. In reality, several challenges are not yet addressed. For example, neither of these three methods were evaluated using 802.11n/ac hardware. Hence, it is unclear if the small buffer sizing approach recommended in these methods will scale with frame aggregation, where sufficient buffers may be required to assemble the large aggregates supported by the standards. Indeed, some results suggest that these algorithms yield sub-par performance for some practical 802.11g/n networks [13]. Furthermore, extending these schemes for multi-hop networks may not be straightforward due to the following reasons. First, eBDP accounts only for inter-flow contention, while intra-flow contention is also common over multi-hops. Second, the three schemes select the buffer size independently. However, as the bottleneck in multi-hop networks spans multiple nodes, some coordination between nodes may be needed to find the optimal buffer size.

3.2 Multi-Hop Network Solutions

There is limited work addressing the challenges associated with buffer sizing of multi-hop wireless networks. Shihada and Jamshaid address buffer sizing in the context of static Wireless Mesh Networks (WMN) [10]. Since the bottleneck in a wireless network is the radio spectrum shared between multiple nodes, the authors proposed a distributed buffer sizing approach. The interfering nodes are identified using collision domains. Collision domain for a link l is a set of links that interfere with l. For a multi-hop flow, the end-to-end rate is bottlenecked by a collision domain that experiences full channel utilization. This is the bottleneck collision domain.

The authors consider the buffer sizing problem in two parts. First, they determine the cumulative buffer required for saturating the bottleneck collision domain. This is the BDP of the network which factors in various overheads associated with frame transmissions in a wireless network. Second, this cumulative buffer is distributed among the nodes that constitute the bottleneck. Various distribution criterion can be used: the authors propose a cost function where buffers are assigned in a way such that packets are more likely to be dropped closer to the source nodes than to the destination. The authors show that using this approach results in small buffer sizes for each node. This minimizes queueing delays, while allowing a node to achieve close to full link utilization.

This approach was evaluated under two scenarios: (1) large file transfer with TCP, and (2) simultaneous TCP and UDP flows. In both cases, this scheme reduces the end-to-end delay to acceptable values when compared to the default buffer size while incurring slight drop in network goodput.

This work has several limitations. First, accurate detection of collision domains using a low overhead mechanisms is a challenge. Second, the analysis of cumulative buffer sizing is optimized for a single TCP flow. It is unclear if similar small sized buffers would work well with multiple flows. Moreover, these small buffers result in suboptimal performance when using A-MPDU aggregation with 802.11n radios. Our testbed measurements show a goodput drop of up to 20% compared to results with 1000 packet buffers. We found that these small buffers prevent a node from transmitting large A-MPDUs. In our measurements, average A-MPDU length dropped from 13.5 frames to 7.5 frames per aggregate with small buffers, resulting in goodput drop.

To overcome these limitations, Showail *et al.* recently proposed WQM [12], an-aggregation aware queue management scheme that sizes buffers for Wi-Fi based networks. WQM relies on passive channel measurements to determine the buffer size at each node. It operates in two phases: in the first phase, the buffer size is initialized based on a variant of the BDP rule-of-thumb while accounting for frame aggregation. In the second phase, the queue drain times are monitored and the buffer sizes adjusted accordingly. The queue drain time reflects the transmission rate and the contention from other active nodes sharing the radio spectrum. One important feature in WQM is enforcing a lower bound on the buffer size so it is not allowed to be less than the average A-MPDU length for any node. This allows a node to transmit multiple packets back-to-back in a single channel access which quickly deflates the buffer and reduces the queueing delays. WQM was evaluated using multiple topologies over both single-flow and multi-flow scenarios. Results show that WQM reduced the end-to-end delay by upto 8× compared to Linux default buffers.

Routing protocols play a significant role in the performance of multi-hop wireless networks. Both [10] and [12] use static routing. In contrast, adaptive load-aware routing protocols that route around the congested parts of the network may yield better performance. Multi-path routing protocols can also be used, where a data stream is distributed as multiple data flows that can take link-disjoint or even node-disjoint paths. However, the performance gains of these protocols may be limited by the network topology. For example, in infrastructure WMNs where the bulk of traffic is routed either towards or away from a single gateway router providing Internet connectivity, load-aware or multi-path routing has limited leeway. Routing protocols can also be used to address channel contention issues such as hidden terminal or exposed terminal problems. To address some of these issues, Dousse [3] proposes a hole routing scheme. The main idea behind this scheme is to reduce the queue size on relay nodes to only one packet to mitigate the problem of low goodput in multi-hop networks. Hence, every node has either a packet or a hole. This routing scheme helps solving bandwidth allocation problem. Similarly, Xue and Ekici [15] use adaptive routing, among other techniques, to increase energy efficiency in multi-hop networks. Finally, Draves *et al.* [4] tackle the problem of routing multi-radio devices. They come up with a routing protocol that takes into consideration loss rate and channel bandwidth to be able to choose a high throughput path.

4. AQM BASED SOLUTIONS

Active Queue Management (AQM) techniques address the problem of persistently full buffers from an aspect other than direct buffer sizing, and as such, are complementary to these efforts. They attempt to prevent large queue buildup at intermediary network hosts through proactive, probabilistic packet drop. However, these algorithms failed to gain traction because of the complexity of setting the configuration parameter knobs effectively. Recently, a no-knobs AQM technique called CoDel [8] has been proposed. Unlike traditional AQM techniques, CoDel does not monitor queue size or queue occupancy directly. Instead, it keeps track of the packet sojourn time through the queue. Once the queueing delay exceeds a predefined value for a fixed amount of time, the algorithm goes into the dropping phase. Packet dropping will stop only if the queuing delay goes below the predefined value again or if the queue has less than one MTU worth of bytes.

Another no-knobs AQM variant, called PIE [9], has also been proposed recently. PIE determines the level of network congestion based on latency moving trends. Upon packet arrival, the packet may be dropped according to a dropping probability that is determined by the dequeue rate and the length of the queue.

Neither CoDel nor PIE has been specifically designed for wireless networks. Hence, it is unclear how they can be effectively used in multi-hop wireless networks where the bottleneck spans multiple distributed nodes. Furthermore, these schemes may not be capable to support fast mobility in wireless devices *e.g.*, vehicular speed mobility. Finally, CoDel allows the buffer to be as small as one frame which will restrict aggregate formation resulting in lower utilization.

5. FUTURE DIRECTIONS

We believe that fixing bufferbloat at the wireless edge requires work along multiple lines, creating complementary solutions that, taken together, may address the myriad challenges described in this paper.

5.1 Frame aggregation schedulers

IEEE 802.11 standard specifications have left the design of A-MPDU aggregation schedulers open to vendor implementation, creating the space for well-designed schedulers that can balance the various performance tradeoffs in a wireless network. Our prior work shows that efficient design of A-MPDU aggregation schedulers can boost goodput while simultaneously reduce end-to-end delays [11]. This can be attributed to two factors: (1) Each A-MPDU includes a single PHY preamble and header, significantly reducing this overhead as these headers are usually transmitted at base rate for backward compatibility with 802.11 a/b/g nodes. (2) A single channel access can transmit as many as 64 subframes, and in response receive a single block ACK. However, even with aggregation enabled, RTT values can still exceed approximately 100 ms over a single wireless hop. We anticipate that the performance will deteriorate further in noisy radio environment as well as in multi-hop networks. These delays may potentially further exacerbate with the emerging 802.11ac standard which supports aggregates as large as 1 MBytes.

5.2 Wireless compatible queue management

Buffer sizing and AQM algorithms may be considered as complementary solutions that can be used in conjunction. As such, the combined effect of the two schemes needs to be studied through both analyses and experimentation. Traditional AQM algorithms may fail in a wireless environment where the queue size may not always be the best indicator of network congestion. Newer algorithms, such as CoDel, address this challenge by using the packet sojourn time to interpret congestion. Thus, the optimal queue backlog is a function of the buffer drain time, and this varies in response to changing channel conditions. Analyzing and adapting the behavior of AQM algorithms with dynamic buffer sizing under this environment needs to be studied in more details.

5.3 Virtual queueing

The AP or BS transmits data to multiple client devices, each experiencing different channel conditions. As a result, the buffer size suitable for one client device may deteriorate the performance of another. One solution is to implement a per station virtual queue to segregate the traffic for different nodes. We believe that some variant of Fair Queuing is necessary to isolate the impact of one wireless device from the other. This can also help improve fairness between flows with different congestion window sizes.

5.4 Fine-tuning TCP

End-to-end solutions may be easier to deploy in controlled networks, such as cellular networks. This is particularly beneficial when the operator cannot access/configure bottleneck router buffers along the traffic route. The TCP stack on client devices can be modified through updates pushed out to smartphones locked by the operator. It is more beneficial to implement these changes at the client side (than at the BS) as the client has more accurate information about the last-hop wireless link.

6. CONCLUSION

Wireless networks usually have smaller BDP than wired networks. Hence, they need smaller buffers. On the other hand, extremely small buffers may limit the network overall goodput. In this paper, we identified the challenges of optimally sizing buffers in various types of wireless networks.We showed that optimally sizing buffers is not only important for real-time traffic, but also for TCP flows sharing the bottleneck buffer as well. We classified wireless buffer sizing schemes into two categories based on network topology: Single-hop and Multi-hop solutions. As shown in this survey, it is very difficult to have a single optimal buffer that suites all types of wireless networks. The new advancements in wireless technology, such as 802.11n/ac frame aggregation, make choosing the optimal buffer size even more challenging.

7. 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, 2004.

[2] A. Dhamdhere and C. Dovrolis. Open issues in router buffer sizing. *SIGCOMM Comput. Commun. Rev.*, 36(1):87–92, Jan. 2006.

[3] O. Dousse. Revising buffering in CSMA/CA wireless multihop networks. In *Proc. of the IEEE SECON '07*, June 2007.

[4] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, MobiCom '04, pages 114–128, 2004.

[5] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla. The impact of multihop wireless channel on TCP throughput and loss. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1744–1753 vol.3, March 2003.

[6] H. Jiang, Y. Wang, K. Lee, and I. Rhee. Tackling bufferbloat in 3G/4G networks. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, IMC '12, pages 329–342, 2012.

[7] T. Li, D. Leith, and D. Malone. Buffer sizing for 802.11-based networks. *IEEE/ACM Transactions on Networking*, 19(1):156 –169, Feb. 2011.

[8] K. Nichols and V. Jacobson. Controlling queue delay. *Queue*, 10(5):20:20–20:34, May 2012.

[9] R. Pan, P. Natarajan, C. Piglione, M. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg. Pie: A lightweight control scheme to address the bufferbloat problem. In *High Performance Switching and Routing (HPSR), 2013 IEEE 14th International Conference on*, 2013.

[10] B. Shihada and K. Jamshaid. Buffer sizing for multi-hop networks, Jan. 28 2014. US Patent 8,638,686.

[11] A. Showail, K. Jamshaid, and B. Shihada. An empirical evaluation of bufferbloat in IEEE 802.11n

wireless networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2014.

[12] A. Showail, K. Jamshaid, and B. Shihada. WQM: An aggregation-aware queue management scheme for IEEE 802.11n based networks. In *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*, CSWS '14, pages 15–20, 2014.

[13] D. Taht. What I think is wrong with eBDP in debloat-testing. `https://lists.bufferbloat.net/pipermail/bloat-devel/2011-November/000280.html`.

[14] C. Villamizar and C. Song. High performance TCP in ANSNET. *SIGCOMM Comput. Commun. Rev.*, 24(5):45–60, Oct. 1994.

[15] D. Xue and E. Ekici. Optimal power allocation in multi-hop wireless networks with finite buffers. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1–5, June 2011.