

# Buffer Management in Wireless Full-Duplex Systems

Nader Bouacida, Ahmad Showail, and Basem Shihada

Computer, Electrical, and Mathematical Sciences and Engineering (CEMSE) Division

King Abdullah University of Science and Technology

Thuwal, Saudi Arabia

Email: {nader.bouacida, ahmad.showail, basem.shihada}@kaust.edu.sa

**Abstract**—Wireless full-duplex radios can simultaneously transmit and receive using the same frequency. In theory, this can double the throughput. In fact, there is only little work addressing aspects other than throughput gains in full-duplex systems. Over-buffering in today’s networks or the so-called “bufferbloat” phenomenon creates excessive end-to-end delays resulting in network performance degradation. Our analysis shows that full-duplex systems may suffer from high latency caused by bloated buffers. In this paper, we address the problem of buffer management in full-duplex networks by using Wireless Queue Management (WQM), which is an active queue management technique for wireless networks. Our solution is based on Relay Full-Duplex MAC (RFD-MAC), an asynchronous media access control protocol designed for relay full-duplexing. We compare the performance of WQM in full-duplex environment to Drop Tail mechanism over various scenarios. Our solution reduces the end-to-end delay by two orders of magnitude while achieving similar throughput in most of the cases.

**Index Terms**—Full-duplex wireless, Bufferbloat, Relay full-duplex MAC, WQM

## I. INTRODUCTION

The long-held assumption that wireless devices can only operate in half-duplex mode is not true after the appearance of the full-duplex wireless systems. Traditionally, a radio is not able to transmit and receive simultaneously using the same channel because the antenna at the receiver side is going to hear its own transmission which is hundreds or thousands of times stronger than the signal coming from other nodes. In fact, full-duplex systems [9], [15] have succeeded in challenging this assumption by using cancellation techniques to cancel self-interference and eliminate the noise created by transmit signal.

Full-duplex systems can theoretically double the throughput. Moreover, these systems have shown great potential to solve important challenges in wireless networks such as hidden terminals, loss of throughput due to congestion, and large end-to-end delays [9]. In fact, the idea of receiving and forwarding simultaneously can reduce the large end-to-end delays in multi-hop networks since a full-duplex node can simultaneously start forwarding a packet to the next hop while receiving it. However, with today’s bloated networks, full-duplex relaying is not sufficient to solve the problem of latency due to bufferbloat phenomenon [12] [20]. For instance, wired networks operate in full-duplex mode and still suffer from unacceptable delays. If we take into consideration the fact

that wireless spectrum is a shared resource between a set of neighboring nodes even in full-duplex mode, the situation will be worse. In order for existing wireless full-duplex designs to have large scale deployments, they should address such challenges.

With declining memory prices and the fallacy that says “more is always better”, industrialists equip the network devices with larger buffering capacities that aim to improve throughput by limiting packet drop. Over-buffering in today’s networks, or the so-called Bufferbloat phenomenon, results in prolonging the time that the packet spends in the queue and thus introducing more latency in the network. While throughput is the dominant performance metric, latency also have a huge impact on user experience not only in real time applications but also in daily used applications such as web browsing which is sensitive to latencies in the order of hundreds of milliseconds. Furthermore, a large buffer may distrust the operation of the TCP congestion control algorithm. More precisely, TCP reduces its transmission window based on packet loss. Thus, with over-sized buffers, it cannot detect the loss and adjust its transfer rate in an effective manner.

Nowadays, wireless devices support rates ranging from 1 Mbps to several Gbps. For example, the peak transmission rate of 802.11ad is 7 Gbit/s [3]. As shown in [20], a statically sized buffer may not be suitable for both high and low transmission rates at the same time. To give an example, a 256 packets buffer needs more than three seconds as a draining time at 1 Mbps which is considered catastrophic for real-time applications. To prove the impact of bufferbloat on network latency, *Gettys & Nichols* [12] recorded a smokeping while moving 20 GB of data to a nearby server. They found that the latency is in the order of seconds.

To address the issue of bufferbloat in wireless full-duplex systems, we propose using the Wireless Queue Management (WQM) mechanism to manage the buffers in the full-duplex nodes. WQM dynamically adjusts the buffer size according to queue draining time and current size. The best of our knowledge, this work is the first attempt to address the buffer management issue in wireless full-duplex systems. We demonstrate through simulation that WQM has succeeded to decrease latency by two orders of magnitudes while achieving better throughput compared to Drop Tail mechanism.

## II. PRELIMINARIES

In this section, we provide some essential background material about the evolution of wireless full-duplex systems as well as bufferbloat battling efforts in the literature.

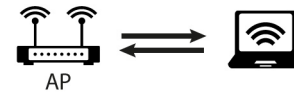
### A. Wireless full-duplex

Nowadays, wireless devices generally operate in half-duplex mode which means they can either transmit or receive on a single channel, but not do both at the same time. Recent research efforts have proved that wireless full-duplexing, illustrated in Fig. 1, is feasible with the use of interference cancellation techniques. *Choi et al.* [9] have succeeded to achieve a single channel full-duplex wireless communication by implementing a full-duplex IEEE 802.11.4 testbed using two USRPv1 equipped with two 2.4 GHz radio daughterboards (RFX 2400). In their design, they introduced a novel self-interference cancellation scheme called “Antenna Cancellation”. The insight behind antenna cancellation is that transmissions from two antennas are added destructively where a separate receive antenna is placed. For the signals to be added destructively and cancel each other for a specific wavelength  $\lambda$ , the transmitter should be placed at distances  $d$  and  $d + \frac{\lambda}{2}$  away from the receive antenna. For this implementation, antenna cancellation provided about 30 dB of self-interference cancellation. Combined with analog cancellation based on QHx220 noise canceller chip and digital cancellation, it managed to cancel about 60 dB of interference. Overall, single channel full-duplexing gives a gain of 84% in throughput without a significant loss in network reliability. However, this design suffers from many limitations. First of all, the prototype uses 7 inches of spacing between antennas which raises the question if such prototype can fit into today’s small wireless cards. In addition, it supports neither wide bandwidths like the 20 MHz 802.11 Wi-Fi signals nor high transmit powers. Moreover, antenna cancellation is very sensitive to antennas placement mismatch.

To overcome these limitations, *Jain et al.* [15] proposed a full-duplex radio design based on a balanced/unbalanced (Balun) transformer. The mechanism known as “balun cancellation” exploits signal inversion using a balun circuit in an adaptive manner to match the self-interference signal. This design, unlike the antenna cancellation based design, eliminates the bandwidth constraint and supports high transmit powers. Balun cancellation combined with digital cancellation can achieve up to 73 dB of cancellation for a 10 MHz OFDM signal. While it solves many problems related to full-duplex, this cancellation is very sensitive to delays and needs very sophisticated electronic components. Further, TX and RX antennas are separated by 20 cm which represents an engineering limitation especially for mobile devices like tablets and mobile phones.

In 2012, a group of researchers from Rice University [10] have implemented a practical 20 MHz IEEE 802.11 multi-antenna full-duplex system using WARP boards. Their design achieves almost the intended doubling of throughput. In addition, *Hong et al.* [13] introduced a transparent spectrum slicing scheme called “Picasso” which allows simultaneous

### Bidirectional full-duplexing



### Relay full-duplexing



Fig. 1: Bidirectional full-duplexing and relay full-duplexing

transmission and reception on separate and arbitrary spectrum fragments using a single antenna. Picasso solves the problem of leaking of interference into adjacent spectrum especially in Wi-Fi OFDM signals.

Later, other implementations of full-duplex systems appeared such as the implementation of full-duplex 802.11ac radio using a single antenna for both transmit and receive [7]. This achievement was made possible thanks to the use of an analog cancellation board and a circulator. With more sophisticated digital and analog cancellation techniques, the prototype reduces self-interference by 110 dB and achieves a median throughput gain of  $1.87\times$ .

The research about full-duplex didn’t stop at this point. *Bharadia & Katti* [6] have demonstrated that full duplex can be combined with MIMO systems putting an end to comparisons between the performance of MIMO half-duplex and full-duplex systems. Conceiving a MIMO full-duplex design wasn’t a simple task because a single antenna will not only suffer from the ordinary self-interference but also from a very strong cross-talk coming from neighboring antennas in the TX chain. The replication of SISO full-duplex is considered as ineffective due to several complexity and scalability limitations. To reduce complexity, the design exploits the fact that neighboring MIMO antennas share a similar radio environment. The solution was based on creating a cascaded filter structure composed of  $M$  cancellation circuits for  $M$ -antennas MIMO system that leaves a negligible 1 dB of the overall self-interference. Moreover, digital cancellation was used to cancel any residual self-interference. The experimental setup of  $3 \times 3$  Wi-Fi MIMO ensures a 95% gain in throughput compared to half-duplex.

### B. Bufferbloat

*Van Jacobson* drew attention to what so called “persistently full buffer” in 1989, leading to the development of RED (Random Early Detection) algorithm [11]. RED represents one of the early AQM (Active Queue Management) techniques which attempt to prevent large queue buildup at the bottleneck by implementing a proactive packet drop when the queue size reaches certain threshold. Many variations of RED were proposed after that but none of them succeeded to gain traction because they tend to be hard to configure and also due to their

slow response to fast changes in the environment.

Recently, a new AQM technique called CoDel (Controlled Delay) [17] has been proposed. Unlike other AQM techniques, CoDel monitors how long each packet stays in the queue. For a given interval, the algorithm finds the lowest queuing delay experienced by all packets. If this lowest packet sojourn time exceeds a predefined target, the packet is dropped at the queue egress and the interval is going to be shortened. Alternatively, if the lowest packet sojourn time for that interval is still in the acceptable range, the packet is forwarded and the interval is reset to 100 milliseconds (initial default value). This algorithm was essentially designed to detect bad queues, which are defined as the queues that last longer than one RTT (Round Trip Time) resulting in a constantly high buffering latency. CoDel is a self-configurable algorithm and has shown good performance over traditional AQM solutions [14].

Another AQM technique has also been proposed recently, called PIE (Proportional Integral controller Enhanced) [18], which combines the benefits of both RED and CoDel. Similar to RED, PIE randomly drops a packet when experiencing congestion. However, congestion detection in PIE is based on the queuing delay instead of the queue length. In fact, latency moving trends helps PIE to determine the congestion level in the network.

In a systemic evaluation of the bufferbloat effect, *Cardozo et al.* [8] suggested that bufferbloat might not be a significant problem in practice. Considering the microscopic view of the buffer architecture of typical network devices, they drew attention to the impact of varying the buffer size of various buffers in the network transmit stack. They found out that the occurrence of the bufferbloat phenomenon is not common. This is in agreement with what *Allman* found in his empirical evaluation of bufferbloat [5]. In his study, he concluded that although bufferbloat could happen, it does not happen that often.

### III. APPROACH

Typically, wireless devices have both receive and transmit buffers. However, receive buffers usually do not get bloated since today wireless devices possess good processing capabilities. More precisely, the wireless device processes incoming packets and forward those packets to the transmit queue in case this node isn't their final destination. That's why, we should focus on managing the transmit buffer because it may cause long queuing delays. We believe that we can reuse the same approach as half-duplex adaptive buffer sizing in the full-duplex domain with minor changes. Moreover, adaptive buffer sizing can also be used to improve the energy efficiency of radio design.

Our primary goal in this paper is to study the interaction between AQM based techniques for buffer management and wireless full-duplex systems. We already know the performance of many buffer management techniques on wireless half-duplex systems even the majority of them are designed for wired networks. Nevertheless, there is a lack in the literature about the suitable method to manage the buffer of a full-duplex

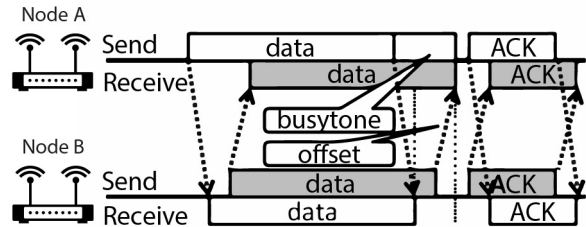


Fig. 2: RFD-MAC time sequence

node or what is the impact of such techniques on networks metrics like latency and throughput.

To be able to analyze the performance of buffer management methods in the literature in wireless full-duplex systems, we modify the buffer module in our wireless full-duplex testbed in order to stimulate the wanted adaptive buffer management behavior. The AQM techniques previously discussed in Sec. II-B are not initially designed for wireless networks. They may not be capable to support various challenges related to the wireless medium like changing transmission rates, frame aggregation, link scheduling *etc.* Hence, we don't have a clear idea about their performance in multi-hop wireless networks where the impact of bufferbloat is more serious.

*Showail et al.* [21] recently proposed Wireless Queue Management (WQM), a queue management scheme for wireless networks. WQM is an adaptive buffer sizing algorithm that estimates periodically the buffer draining time using current transmission rate, backlog queue and the number of neighboring nodes. In fact, WQM increases or decreases the buffer size depending on buffer draining time which raises an alarm when exceeding a predefined value. In comparison to CoDel and PIE, WQM demonstrates a good performance in an ordinary IEEE 802.11n half-duplex testbed.

The novelty of WQM lies in the fact that it is designed to address the unique challenges of buffer sizing in wireless networks [22]. In reality, WQM is the first queue management technique that is designed to be aggregation-aware, that is to say it accounts for frame aggregation when selecting the optimal buffer size in wireless networks. Second, the link rate in wireless networks changes dynamically in response to variations in network conditions. The capacity of the network controls the BDP (Bandwidth-Delay Product) which reflects the amount of buffering needed in the network. To address this issue, WQM is the only scheme in the literature that is tightly coupled with the rate control algorithm to quickly respond to changes in the environment. Last but not least, WQM accounts for the number of active users when selecting the optimal buffer size. This number affects the packet service rate due to MAC random scheduling in wireless networks. All those advantages motivated us to implement WQM on top of our wireless full-duplex implementation with minor changes as detailed in the next section.

#### IV. IMPLEMENTATION

We based our work on a simulation of full-duplex communication in a multi-hop environment using the discrete-event network simulator NS-3 [1]. This simulation deploys the Relay Full-Duplex MAC protocol [23] by extending the Wi-Fi module [2] of NS-3.20. Relay Full-Duplex MAC (RFD-MAC) [23] is a media access control protocol which is designed for relay full-duplexing. Multi-hop networks can handle bidirectional full-duplexing as well as relay full-duplexing. The difference between these two schemes is illustrated in Fig. 1. Bidirectional full-duplexing means that the wireless node is able to send and receive to/from the same node at any given point of time. In Relay full-duplex scenario, node 2 can simultaneously receive a frame from node 1 and forward a frame to node 3 at the same time. To maximize full-duplex capability, RFD-MAC should properly choose a secondary transmission node. For example, node 2 has two candidates for a secondary transmission node: nodes 1 and 3. If the MAC protocol selects a node that does not have a frame, then relay full-duplexing does not occur. Also, RFD-MAC take into account the possible collision between a primary transmission and a secondary transmission. Collision occurs at receiver node when a primary transmission node is placed within the transmission range of a secondary transmission node and vice versa. Every node builds its own surrounding node table and exploits this table to choose a secondary transmission node in such a way that avoids a collision between primary and secondary transmission. The algorithm of selecting a secondary node is based on a priority set.

Whenever a node completes the transmission of the frame before receiving is done, RFD-MAC uses a busytone until the reception is complete. Then, the primary and secondary transmission nodes exchange ACK frames to finalize the full-duplex transmission session as shown in Fig. 2. This scheme does not completely eliminate collision because receiver is susceptible to collisions until it finishes receiving the packet header.

We choose to implement our proposed method using NS-3 because it was built to improve the realism of the transmit stack of real devices [19]. In the Wi-Fi module of NS-3, the Wi-Fi Net Devices internal queues have different architecture than the traditional ones used elsewhere. In fact, to implement any sort of AQM method in NS-3, one must modify the source files of NS-3 Wi-Fi module. The insight beyond our work is to handle the transmit packet queue via modifying the “*Enqueue*” function belonging to “*WifiMacQueue*” class.

WQM defines an upper limit on buffer size to avoid bufferbloat. In fact, WQM estimates the maximum needed buffer size in the network  $B_{max}$  using the well known buffer sizing rule which is the bandwidth delay product that can be calculated as per the following equation:

$$B_{max} = \lambda \cdot RTT \quad (1)$$

where  $\lambda$  is the maximum possible transmission rate. WQM also selects an optimal buffer size to start with based on several

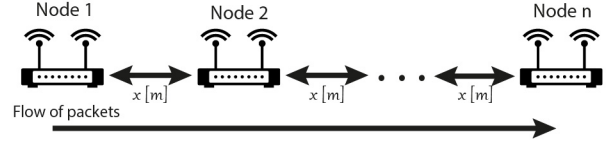


Fig. 3: Topology of single flow scenario

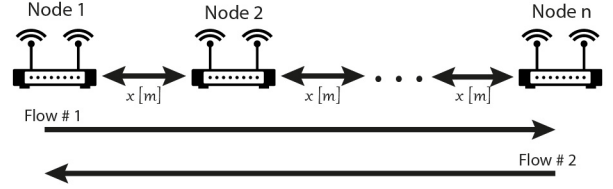


Fig. 4: Topology of bidirectional flows scenario

network parameters. The initial queue size can be calculated as:

$$B_{initial} = R \cdot RTT \quad (2)$$

where  $R$  is the current transmission rate.  $RTT$  is the sum of the TCP segment transmission time and acknowledgement transmission time which can be calculated based on the IEEE 802.11a standard [4] as:

$$RTT = T_{d-DATA} + T_{d-ACK} \quad (3)$$

After doing the math, we found out that the maximum needed buffer size in our network should be set to 24 packets and the initial buffer size should be 2 packets. The buffer size is not allowed to be less than one packet ( $B_{min} = 1$ ). We drew the attention of the reader that the default buffer size for Drop Tail in NS-3 is defined as 400 packets.

We would like to note that our current implementation is slightly different from the original WQM implementation. First, in our implementation we use an Ad-Hoc Wi-Fi network based on IEEE 802.11a standard [4] in which the transmission range can vary between 6 Mb/s and 54 Mb/s. All devices are configured with Adaptive Auto Rate Fallback (AARF) rate control algorithm [16] instead of Minstrel. AARF attempts to increase the transmission rate after a predefined number of successful transmissions at the current rate. To ensure the stability of the channel, AARF increases its success threshold before trying to use a higher rate by remembering the number of failed probes. In case of two consecutive packet losses, it lowers the transmission rate one step and resets the success threshold to 10. In its original version, WQM is synchronized with Minstrel which is tuned every 100 ms. This is not the case for our current setup since AARF is based on packet transfer status. Thus, we choose simply to tie running the algorithm with every incoming packet to the transmit queue. Second, the simulation of full-duplex communication [1] is not QoS-enabled and hence it doesn't support frame aggregation. So, we modify the WQM algorithm to deal with disabled frame aggregation.

We test our implementation over two scenarios: single flow scenario as illustrated in Fig. 3 and bidirectional flows scenario as illustrated in Fig. 4. This topology illustrated represents a typical relay full-duplex Ad-Hoc network. The distance between nodes in the network is fixed to 90 m. We repeat every experiment multiple times while varying the number of nodes in the network and the sender transmission rate. The simulation specifications are summarized in Table I.

Prameter	Value
Wi-Fi Standard	IEEE 802.11a
Radio Band	5 GHz
Packet size	1500 Bytes
Default m_queue size	400 packets
Amount of data transferred	100 MB
Distance between nodes	90 m
Routing protocol	AODV
Rate control algorithm	AARF

TABLE I: Simulation parameters summary

## V. PERFORMANCE EVALUATION

In this section, we present our performance evaluation of WQM in the full-duplex simulation in comparison to Drop Tail.

### A. Single flow scenario

As shown in Fig. 3, when the simulation starts, node 1 sends a flow of packets towards node n, which is the last node in the network. This setup tries to mimic the transfer of a large file between nodes in the network. Every packet carries 1500 bytes worth of data. The simulation ends when the node n receives 100 MB worth of bytes or when 1000 seconds is elapsed from the start of simulation, whichever occurs first. We start collecting various network metrics after the receiver node receives the first 100 packets. In all our experiments, we vary the source rate gradually from 100 packets/s to 1000 packets/s. We repeat each experiment multiple times while varying the number of nodes in the network from 3 to 5. To increase the reliability of our results, we run every experiment at least twice and report the average of the results. End-to-end delay is shown in Fig. 5, goodput results are shown in Fig. 6 and the collision rate is shown in Fig. 7. We would like to note that we use the logarithmic scale for the y-axis in the latency figures to be able to show the difference between the two schemes.

As expected, WQM outperforms Drop Tail in terms of latency reduction. When the network is bloated, WQM reduces the latency by an average of  $109\times$  for the three nodes case,  $208\times$  for the four nodes case and  $49\times$  for the five nodes case. In comparison to Drop Tail, WQM manages to reduce up to 99% of the encountered latency. For example, in the four nodes case and using a transmission rate of 800 packets/s, WQM reduces the end-to-end delay from 4094.51 ms to only 18.91 ms, which represents two orders of magnitude reduction. In fact, the default buffering scheme leads to latency in the order of seconds in all scenarios. When the source rate reaches

more than 300 packets/s, the end-to-end latency reaches 1 s for the three nodes case, around 4 s for the four nodes case and approximately 1.5 s for the five nodes case. On the contrary, WQM maintains a delay less than 10.27 ms for the three nodes topology, less than 20.69 ms for the four nodes topology and less than 29.32 ms for the five nodes topology. It is obvious from these figures that selecting the optimal buffer size results in significant queuing delay reduction.

As shown in Fig. 6, the goodput results of WQM suffer from less variation compared to Drop Tail. In the case of three nodes topology, WQM drops the goodput by an average of 9% compared to the default buffering scheme in NS-3. However, when the number of hops is increased, WQM improves the network goodput. On average, it increases the goodput by 7.7% and 25% for the four and five nodes scenario respectively when the network is congested. Furthermore, We attribute this to the ability of WQM to reduce the collision rate between the primary and secondary transmissions when the number of hops is increased as illustrated in Fig. 7. In fact, as the source rate increases, the buffers in the case of the default scheme fill up quickly leading to extra latency and higher collision rates. The bufferbloat point is located between 300 packets/s and 400 packets/s which corresponds to the default static buffer size in our implementation (400 packets). The situation is worse when there are more nodes in the network. This is a clear proof that static buffers are not suitable for full-duplex wireless networks.

### B. Bidirectional flows scenario

In this section, we evaluate the performance of WQM in the presence of bidirectional flows in the network. We run two 50 MB flows in opposite directions between the edge nodes in the network as illustrated in Fig. 4. Similar to the single flow scenario, we vary the number of nodes from three up to five and vary the source rate gradually from 100 packets/s to 1000 packets/s. The end-to-end latency, goodput and collision rate between primary and secondary transmission are shown in Fig. 8, 9 and 10 respectively.

Similar to the single flow scenario, when the source rate is equal or higher than 300 packets/s, WQM reduces network latency by an average of  $126\times$  for the three nodes case,  $181\times$  for the four nodes case and  $40\times$  for the five nodes case. For instance, when the network consists of three nodes and both the sources operate at 600 packets/s, WQM manages to drop the end-to-end latency from 1646.15 ms in Drop Tail case to only 12.61 ms which represents two orders of magnitudes delay reduction. This achievement comes at cost of only 10.75% of goodput reduction. It could be noticed that when the sources are operating at high transmit rates, WQM suffers from 50% drop in goodput in the three nodes case. We attribute this to the high collision rate in this case as shown in Fig. 10.

As we increase the number of nodes in the network, we notice that WQM outperforms Drop Tail in terms of network goodput. In the five nodes case, the average increase in goodput is about 24 %. We would like to note that in the four nodes topology, Drop Tail have slightly better goodput than



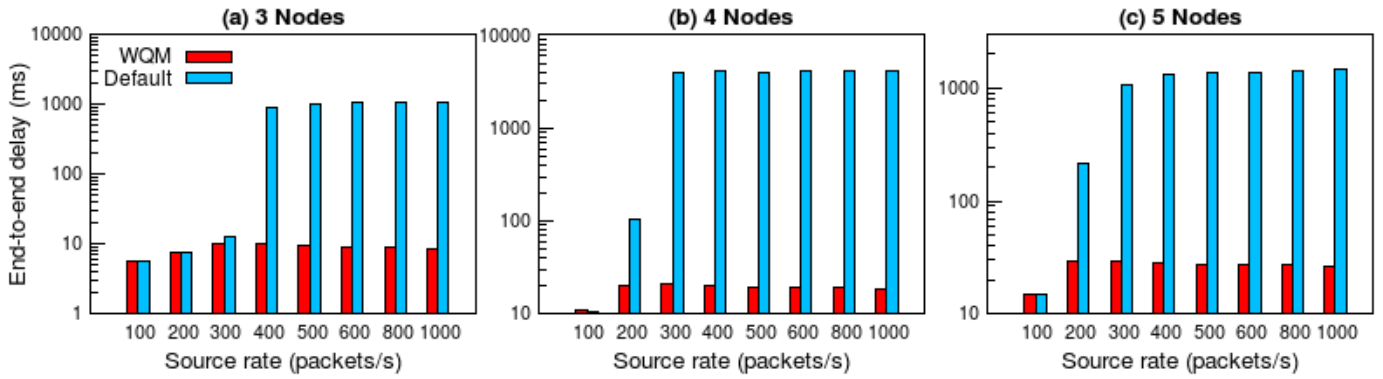


Fig. 5: End-to-end latency while varying source rate for the single flow scenario

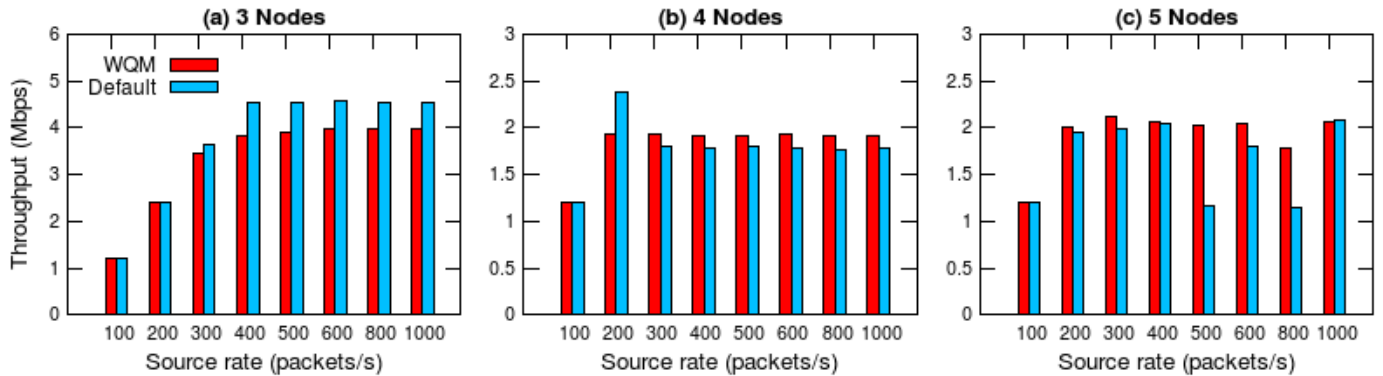


Fig. 6: Goodput while varying source rate for the single flow scenario

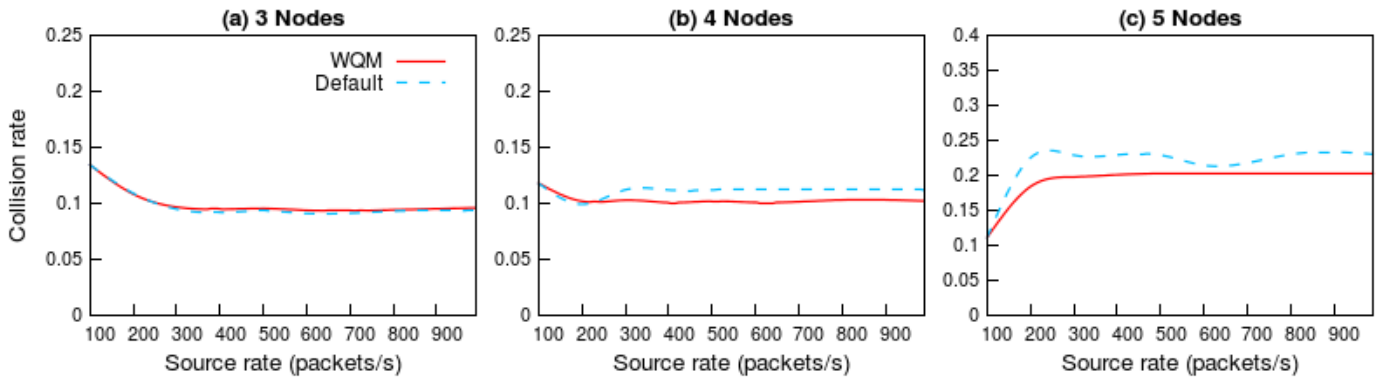


Fig. 7: Collision rate while varying source rate for the single flow scenario

WQM in several cases even though the latter achieves lower collision rate as shown in Fig. 10. We investigate this issue and find that WQM have inferior full-duplex ratio in this case as illustrated in Fig. 11. This reduction varies between 2.87 % and 8.81 % in comparison to Drop Tail and may limit the ability of WQM to utilize the drop in collision rate. Despite this limitation, WQM manages to enhance goodput from 2.48 Mbps to 5.92 Mbps when both sources send 800 packets/s.

The collision rate for both WQM and Drop Tail are shown in Fig. 10. For the three nodes case, WQM have higher collision rate than Drop Tail by 1.15% on average. This fact combined

with limited buffer size may explain the significant drops in goodput mentioned before. However, with larger topologies, our implementation achieves about 1.46% and 7.63% average decrease in collision rate respectively for four and five nodes topology. The ability of WQM to increase the goodput with larger networks compared to Drop Tail is undoubtedly a great achievement.

Overall, WQM keeps the end-to-end latency below 15.57 ms for the three nodes topology, below 20.81 ms for the four nodes topology and below 34.84 ms for the five nodes topology which is an acceptable delay for real time applica-

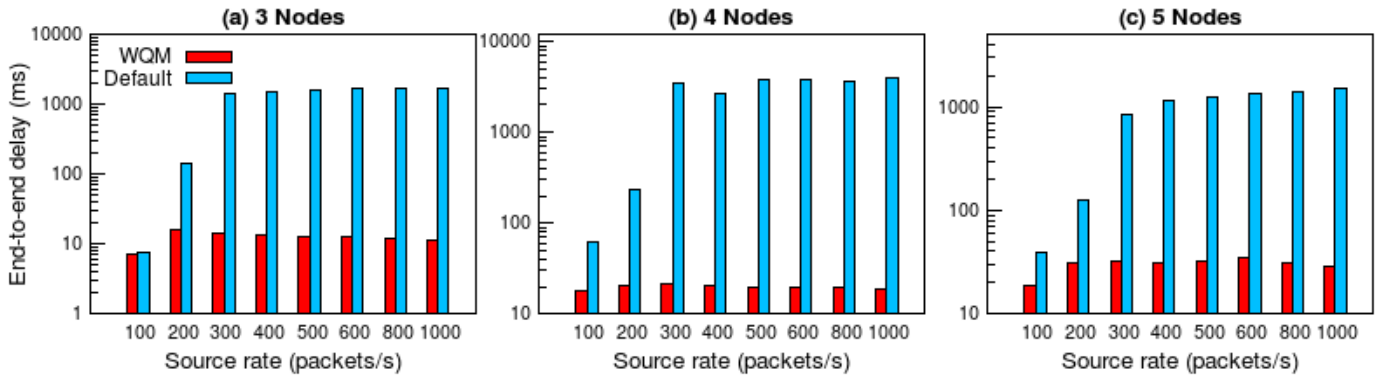


Fig. 8: End-to-end latency while varying source rate for the bidirectional flows scenario

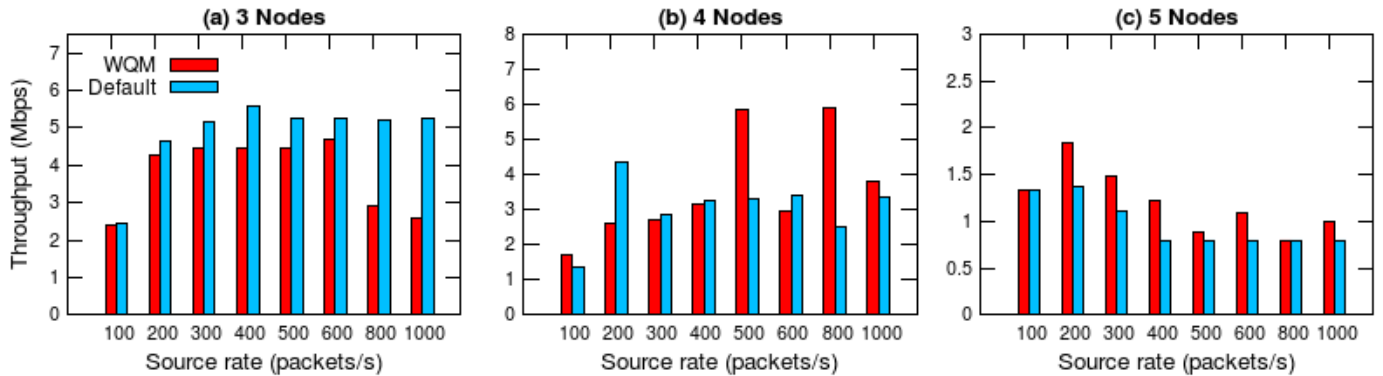


Fig. 9: Goodput while varying source rate for the bidirectional flows scenario

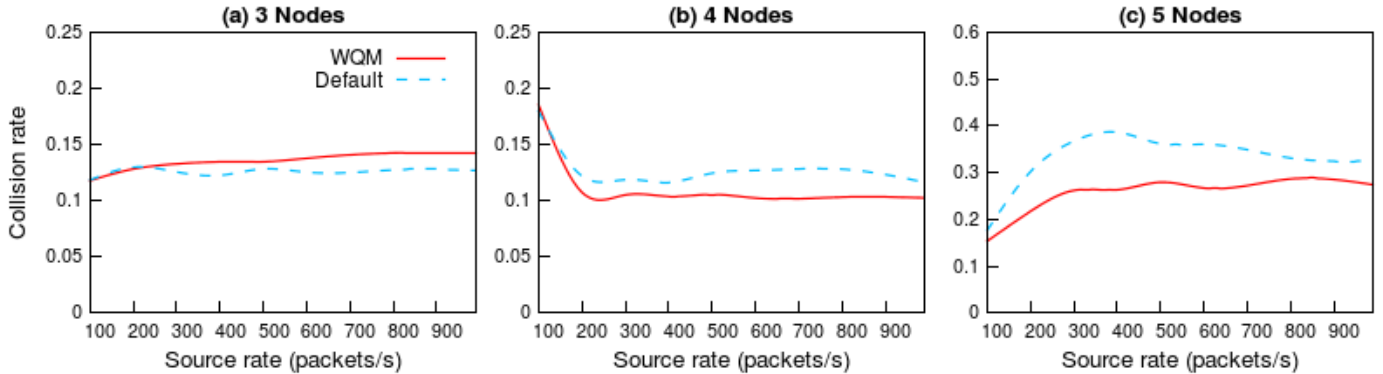


Fig. 10: Collision rate while varying source rate for the bidirectional flows scenario

tions such as VoIP, online gaming, video streaming, *etc.* As mentioned earlier, WQM manages the buffers in an effective manner and prevents the large buffers buildup at the bottleneck links.

## VI. CONCLUSIONS AND OPEN ISSUES

Wireless networking is expected to see a huge shift in the next few years. Mobile devices such as mobile phones and tablets are replacing the traditional desktops and becoming principal computing devices. As a result, improving the performance of wireless systems such as latency and throughput seems crucial. Recently, wireless full-duplexing imposes itself

as a reality turning on very promising area of research. Now, we should look for more sophisticated full-duplex systems.

This paper tackles the problem of buffer management in full-duplex systems and analyses the gains in terms of latency of implementing AQM on top of such systems. We prove through simulation over many scenarios that WQM, an adaptive buffer sizing algorithm, can decrease latency in relay full-duplex networks by two orders of magnitude. We believe that this work opens a new research direction by evaluating the interaction between buffer management and full-duplex design.

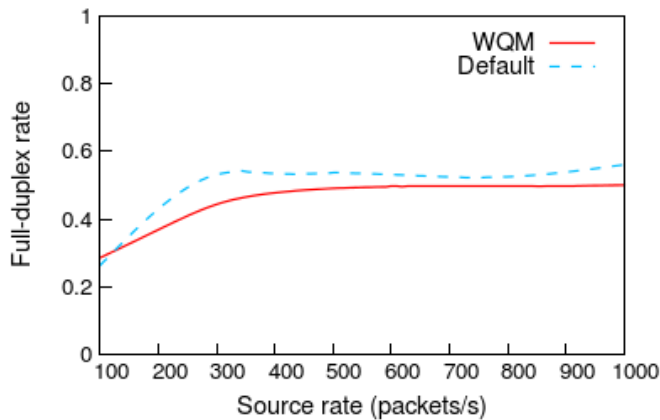


Fig. 11: Full-duplex Ratio for the four nodes scheme

In the future, we are going to consider other methods to evaluate WQM using real testbed such as WARP boards and also test the performance using the most recent Wi-Fi standard. Further, we aim to come up with a novel buffering scheme for full-duplex devices that takes into consideration internal queues as well as ring buffers. Finally, we would like to investigate the effect of buffer management on the energy efficiency of full-duplex systems.

#### REFERENCES

- [1] Implement wireless full-duplex communication in the ns-3 network simulator. <https://github.com/yusuke-sugiyama/ns-3-fdwifi>, accessed April, 2015.
- [2] Wi-Fi Model Library - ns-3. <https://www.nsnam.org/docs/models/html/wifi.html>, accessed April, 2015.
- [3] IEEE standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements–part 11: Wireless lan medium access control (MAC) and physical layer (PHY) specifications amendment 3: Enhancements for very high throughput in the 60 ghz band. *IEEE Std 802.11ad-2012 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012 and IEEE Std 802.11aa-2012)*, pages 1–628, Dec 2012.
- [4] IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: Wireless lan medium access control (MAC) and physical layer (PHY) specifications. *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pages 1–2793, March 2012.
- [5] M. Allman. Comments on bufferbloat. *SIGCOMM Comput. Commun. Rev.*, 43(1), Jan. 2012.
- [6] D. Bharadia and S. Katti. Full duplex MIMO radios. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation, NSDI'14*, pages 359–372, 2014.
- [7] D. Bharadia, E. McMillin, and S. Katti. Full duplex radios. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, SIGCOMM '13*, pages 375–386, 2013.
- [8] T. Cardozo, A. da Silva, A. Vieira, and A. Ziviani. Bufferbloat systematic analysis. In *Telecommunications Symposium (ITS), 2014 International*, pages 1–5, Aug 2014.
- [9] J. I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti. Achieving single channel, full duplex wireless communication. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking, MobiCom '10*, pages 1–12, 2010.
- [10] M. Duarte, A. Sabharwal, V. Aggarwal, R. Jana, K. Ramakrishnan, C. Rice, and N. Shankaranarayanan. Design and characterization of a full-duplex multi-antenna system for WiFi networks. *Vehicular Technology, IEEE Transactions on*, 63(3):1160–1177, March 2014.
- [11] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.*, 1(4):397–413, Aug. 1993.
- [12] J. Gettys and K. Nichols. Bufferbloat: Dark buffers in the internet. *Queue*, 9(11):40:40–40:54, Nov. 2011.
- [13] S. S. Hong, J. Mehlman, and S. Katti. Picasso: Flexible RF and spectrum slicing. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '12*, pages 37–48, 2012.
- [14] T. Hiland-Jrgensen. Battling bufferbloat: An experimental comparison of four approaches to queue management in linux. *Project Report*, Dec. 2012.
- [15] M. Jain, J. I. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha. Practical, real-time, full duplex wireless. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking, MobiCom '11*, pages 301–312, 2011.
- [16] M. Lacage, M. H. Manshaei, and T. Turletti. IEEE 802.11 rate adaptation: A practical approach. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM '04*, pages 126–134, 2004.
- [17] K. Nichols and V. Jacobson. Controlling Queue Delay. *Queue*, 10(5):20:20–20:34, May 2012.
- [18] R. Pan, P. Natarajan, C. Piglion, M. Prabhu, V. Subramanian, F. Baker, and B. VerSteg. PIE: A lightweight control scheme to address the bufferbloat problem. In *High Performance Switching and Routing (HPSR), 2013 IEEE 14th International Conference on*, pages 148–155, July 2013.
- [19] G. F. Riley and T. R. Henderson. The ns-3 network simulator. In K. Wehrle, M. Gnes, and J. Gross, editors, *Modeling and Tools for Network Simulation*, pages 15–34. Springer, 2010.
- [20] A. Showail, K. Jamshaid, and B. Shihada. An empirical evaluation of bufferbloat in IEEE 802.11n wireless networks. In *Wireless Communications and Networking Conference (WCNC), 2014 IEEE*, pages 3088–3093, April 2014.
- [21] 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.
- [22] A. Showail, K. Jamshaid, and B. Shihada. Buffer sizing in wireless networks: Challenges, solutions, and opportunities. *IEEE Communications Magazine*, Accepted, 2014.
- [23] K. Tamaki, H. Ari Raptino, Y. Sugiyama, M. Bandai, S. Saruwatari, and T. Watanabe. Full duplex media access control for wireless multi-hop networks. In *Vehicular Technology Conference (VTC Spring), 2013 IEEE 77th*, pages 1–5, June 2013.