Packet Mass Transit: Improving Frame Aggregation in 60 GHz Networks

Hany Assasa
IMDEA Networks Institute and
Universidad Carlos III, Madrid, Spain
hany.assasa@imdea.org

Adrian Loch
IMDEA Networks Institute
Madrid, Spain
adrian.loch@imdea.org

Joerg Widmer
IMDEA Networks Institute
Madrid, Spain
joerg.widmer@imdea.org

***Abstract*—The impact of frame aggregation on wireless network performance increases dramatically with higher data rates. The key problem is that the transmission time of packets decreases while the medium access, preamble and packet header overhead remain the same. Recent 802.11 standards address this issue using frame aggregation, i.e., grouping multiple data frames in a single transmission to reduce the overhead. This already provides substantial efficiency gains in networks operating in the 2.4 GHz and 5 GHz bands, and for future 60 GHz networks such as 802.11ad, gains are even more pronounced due to the order-of-magnitude higher data rates. In 802.11ad, frame aggregation becomes crucial to achieve the multi-gbps data rates that are possible in theory, since medium access overhead can be 20x larger than the time required to transmit a single packet. While frame aggregation is essential, it very much depends on the traffic patterns present in the wireless network, and a node may not always have enough packets in the transmit queue to achieve a sufficiently large aggregated frame size. In this paper, we investigate in which case nodes should wait to construct a larger aggregated packet before starting the channel access procedure. We present a simple waiting policy for the uplink case that either waits for a minimum number of packets or for a maximum amount of time, whichever comes first. For the downlink case, we utilize a maximum weight scheduling policy with a maximum waiting time. Our results show that both policies significantly improve medium utilization, thus increasing throughput and reducing end-to-end delay.**

I. Introduction

Introducing artificial delay can improve the performance of wireless networks. This somewhat counter-intuitive idea becomes particularly relevant for wireless systems that achieve multi-gbps data rates, such as 802.11ad [1]. The underlying reason is frame aggregation, which plays a fundamental role in recent 802.11 standards. Its impact is amplified in the case of IEEE 802.11ad due to the large bandwidth available in the 60 GHz band—802.11ad channels are 2.16 GHz wide—and the resulting very high data rates. Any time spent for medium access control (MAC) backoff, inter-frame spacing, or retransmissions is highly detrimental to performance. To give an intuition, transmitting a single packet of size 1500 bytes at a moderate 802.11ad rate requires around $3\,\mu s$. In contrast, the MAC overhead for channel access is $60\,\mu s$, that is, 20× larger. Thus, transmitting large frames that include as many data packets as possible is crucial; 802.11ad must use frame aggregation. Moreover, this has such a large impact on 802.11ad performance that it is crucial to exploit as many aggregation opportunities as possible. Figure 1 gives an example. The first timeline shows the arrival of packets at the transmit queue of a wireless 802.11ad station. Packets p_1 and p_2 arrive while the medium is busy, and p_3 shortly after it becomes free. Case I in Figure 1 shows the behavior of existing 802.11 devices, that is, aggregate all the data which is available when the medium is free and transmit. However, this behavior misses the opportunity of aggregating p_3 and incurs significant overhead to transmit it later. In contrast, in this paper we design a mechanism that introduces an artificial delay t_{wait} before transmitting. In Case II, this allows the station to aggregate all packets in a single transmission, and thus significantly reduce medium utilization. Also the average delay is reduced. While p_1 and p_2 are slightly delayed in Case II of Figure 1, p_3 arrives at the receiver much earlier than in Case I, and the average delay is smaller.

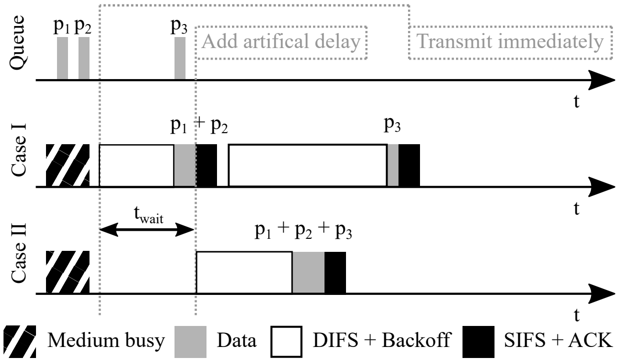

Fig. 1. Aggregation opportunity example.

Waiting for packets is particularly beneficial for bursty packet arrivals such as in Figure 1, which is often the case for typical Internet traffic. In infrastructure-based networks, this is highly relevant for the uplink, where waiting time may be used by the access point (AP) or other stations (STAs) which might have accumulated more packets. Moreover, introducing a waiting time improves performance in case of high contention since nodes access the medium less often, the more packets they aggregate. As a result, fewer collisions occur. Introducing a waiting time for the AP is more problematic since traffic is usually asymmetric—forcing the AP to idle may lead to undesirable queue build-up and delay.

While the underlying idea is simple, the above scheme raises a number of vital questions. When should a station wait? And, most importantly, for how long should it wait? Without clear guidelines on how to answer both questions, a waiting scheme may reduce performance rather than improve it.

In this paper, we design uplink and downlink scheduling policies that take aggregation opportunities into account. Our design is local, that is, it incurs no communication overhead among nodes. It requires a STA to either wait for a certain minimum number of packets, or for a maximum amount of time, whichever comes first. This is in contrast to most existing queuing work (c.f. Section II), and our work is the first to use such a policy for MAC-level frame aggregation. For the downlink case, it is beneficial to transmit to the STA with the highest number of packets in the AP queue, i.e., the one that allows for the highest level of aggregation, rather than introducing a waiting time. To this end, we use a maximum weight scheduling policy together with a waiting time limit to ensure fairness among stations with different traffic patterns. Throughput improves for both uplink and downlink since the medium is used more efficiently. We make the following contributions:

- We design and implement downlink and uplink scheduling policies that are optimized for aggregation.
- We study the benefits of maximum weight scheduling and introducing artificial delays for a range of the above thresholds and traffic patterns.
- We show experimentally that wrong parameterizations of our waiting policy are not harmful in most cases.

The paper structure is as follows. In Section II we survey related work on aggregation. Section III presents the waiting policy. Section IV describes our evaluation scenario, and Section V presents the results from our simulation study based on ns-3. In Section VI we discuss the implications of these results for 802.11ad. Finally, Section VII concludes the paper.

II. Background and Related Work

We first give some background on 60 GHz networks and then survey existing work on frame aggregation.

A. Background on 60 GHz networks

Millimeter-wave communication is particularly interesting for wireless networking because a large amount of bandwidth is available for unlicensed use in the 60 GHz band. However, 60 GHz networks face a number of challenges such as high attenuation [2], [3]. While transmitters use directional antennas to overcome attenuation, recent work shows that consumer-grade phased antenna arrays for 60 GHz systems have many side lobes [4]. Hence, interference among transmitters is significant in spite of directional communication. Further, the MAC layer must operate efficiently to ensure multi-gbps rates. Since channel bandwidth is typically in the order of a couple of GHz, any inter-frame spacing and control data transmitted at a low modulation and coding scheme (MCS) significantly reduce MAC efficiency. Standards such as 802.11ad [1] and WiGig [5] address this issue through frame aggregation. That is, a transmitter transmits multiple packets back-to-back such that they share the channel access overhead.

Existing simulation studies of 60 GHz networks show their feasibility [6], [7], and evaluate their efficiency regarding different medium access techniques such as carrier sense multiple access with collision avoidance (CSMA/CA) and polling [8]. Experimental evaluation of 60 GHz networks is often difficult since hardware is not available yet. Practical work in this area typically infers network performance from individual 60 GHz links [9], [10] rather than an entire network. Existing commercial hardware does not allow us to modify the medium access control and physical layers, which is crucial to evaluate the impact of frame aggregation. Hence, we must base our evaluation in Section V on simulations.

B. Frame aggregation

Frame aggregation is a key feature to reduce the aforementioned MAC inefficiency. Starting with the 802.11n standard, wireless networks may use two types of aggregation, namely, A-MSDU and A-MPDU [11]. A-MSDU aggregates few frames has a single frame check sequence (FCS). That is, if one packet is lost during transmission, the entire aggregated frame needs to be retransmitted. In contrast, A-MPDU can aggregate more frames since it has individual FCSs for each packet, but at the same time this incurs higher overhead. Existing studies show that aggregation can achieve up to 95% channel utilization [12], and that hybrid of A-MSDU and A-MPDU performs best [13]. Related work discusses improvements to these 802.11 aggregation schemes, such as including additional headers to allow A-MSDU to retransmit individual packets [14], and compressing the per-packet subheaders in an A-MSDU frame [15]. Moreover, other approaches allow aggregating packets addressed to different stations. This is feasible if stations use the same physical layer rate [16], or if the transmitter transmits the packets for each station on disjoint 802.11 subcarriers [17]. In [18], the authors propose a joint spatial multiplexing and packet aggregation scheme for MU-MIMO 802.11ac. However, they neither take into account the two types of aggregation supported by the standard nor consider the uplink case which is relevant for aggregation. Finally, the authors in [19] use fuzzy control to determine the optimum aggregation buffer delay before accessing the channel. Unlike our work where we consider two variables for the waiting policy, namely the maximum waiting time and maximum number of packets, the authors only evaluate the impact of buffer delay on the end-to-end latency.

Frame aggregation can be modeled as a batch service queue since the transmitter provides service to all packets in an a frame together. Early models for such queues consider that the service time is independent of the batch size [20]. However, this does not hold in our case since the transmission time increases with the number of aggregated packets. More sophisticated models [21], [22] take this into account and study additional features such as limiting the maximum/minimum batch size N [23], and allowing for server vacations [24], which model waiting times. The design that we sketch in

Section I considers both a minimum batch size and a maximum waiting time. Hence, it is a batch queue with N-policy and interrupted vacation [25]. We apply this queuing policy to wireless networking. Earlier work on 802.11 networks only considers aggregation based on a non-interruptible waiting time [26]. Further, [26] does not allow the transmitter to aggregate packets while waiting for the channel to become available. In contrast, our design allows for this, which is a more realistic assumption.

Adjusting the *maximum* batch size can be beneficial, too. For instance, the optimal length of A-MSDU depends on the packet error rate (PER) since these frames have a single FCS [27]. Further, 802.11ac only performs channel equalization at the beginning of A-MPDU/A-MSDU frames. Hence, frames should not be longer than the coherence time of the medium [28]. While these schemes deal with the maximum aggregation size, we focus on how to achieve that size given bursty traffic. Hence, these approaches are orthogonal to our work.

III. Scheduling Policy

For our design, we consider an infrastructure-based 60 GHz network with one AP and N STAs. We first present the details of the aggregation-aware uplink and downlink scheduling policies we introduced in Section I and then outline a method to estimate suitable policy parameters.

A. Uplink case

The key idea is to allow STAs to wait for a limited amount of time in order to receive more packets of the current burst and thus increase aggregation. A basic waiting policy would be to wait for a fixed amount of time whenever a STA is ready to transmit. While simple, this approach may wait unnecessarily. For instance, if the STA had to wait for the medium to become available, chances are that its queue already holds a sufficient number of packets. Since the STA aggregates all of these packets, the per packet overhead for that medium access may already be acceptable. In this case, waiting for a fixed amount of time may allow to aggregate more packets but the additional benefit is limited, since the per packet overhead decreases as $1/k$, where k is the number of packets in the queue of the STA. In contrast, if k is small when the medium becomes available, any additional packet that the STA receives during the waiting time significantly reduces the per packet overhead. That is, the number of packets k in the queue of a STA is critical to decide whether to wait for more packets. Hence, we design our waiting policy based on two thresholds, namely:

1) The **packet threshold** P_s is the minimum number of packets that a STA must have, to transmit *before* the elapsed waiting time reaches the time threshold.
2) The **time threshold** T_s is the maximum waiting time duration that a STA must not exceed even if it has *fewer* packets than the packet threshold requires.

Hence, if $k \geq P_s$ the STA transmits. It also transmits if it waited for a time of T_s even if $k < P_s$. This prevents an uncontrolled increase of the artificial delay that we introduce through our waiting policy.

The transmit queue of the STA contains packets addressed to the AP and the STA must check the above thresholds whenever a new packet arrives at the queue. Specifically, when the *first* packet arrives to an empty transmit queue, the STA sets a timer to expire after a time of T_s. Whenever a packet arrives at the queue which results in $k = P_s$, the STA initiates medium access according to 802.11 (i.e., start the backoff procedure in case of CSMA/CA), and the timer is cancelled. Once the STA is granted access to the medium, it aggregates *all* packets that arrived up to this point in time (including packets that may have arrived during backoff), if the maximum possible frame length of the device and the standard allow this. Otherwise, the maximum allowed number of packets are aggregated and sent, and the timer is reset to expire after $T_s - (t - t_a)$, where t is the current time and t_a is the arrival time of the oldest packet in the queue.

To avoid excessive out-of-order packet delivery, the waiting policy does not apply to retransmissions. These are handled separately from normal data packets and medium access follows the usual 802.11 procedure. Note that also for an immediate retransmission, any other packets that are in the queue at that time are aggregated with it.

STAs locally decide when and for how long to wait. Thus, our waiting policy does not incur any control overhead—any benefit that results from waiting is at *zero cost* (other than the delay itself). The specific values of P_s and T_s have a significant impact on the performance of our policy. Hence, we study them in detail in Section V.

B. Downlink case

The downlink case is different from the uplink case since the transmit queue of the AP usually contains packets for multiple STAs, and forcing the AP to wait can be detrimental to performance since downlink traffic usually exceeds uplink traffic. Instead, the AP uses a maximum weight scheduling policy to exploit aggregation opportunities. When the AP gains a transmission opportunity, it transmits to the STA with highest number of packets in the queue. This way, AP adds some *implicit* waiting time to the packets for other STAs in its queue, providing the opportunity for more packets to these STAs to arrive.

To prevent starvation for STAs with low traffic, also the AP uses a maximum waiting time T_{ap}. Whenever the AP gains access to the channel, it only transmits to the STA with highest number of packets if $t - t_a < T_{ap}$, where t_a is again the arrival time of the oldest packet in the queue. Otherwise, it transmits to the STA that is the destination of the head-of-queue (i.e., oldest) packet.

C. Setting the Parameter

Parameters P_s, T_s and T_{ap} determine the performance of our waiting policy. The AP's maximum waiting time threshold T_{ap} essentially limits unfairness among downlink flows. It is primarily of importance in case low rate flows with strict delay constraints compete with high rate flows. In this case, T_{ap} should be set to the desired maximum wireless delay.

Section V shows that the optimal value of P_s and T_s depends on the traffic pattern and the number of nodes in the network. Designing in detail a mechanism to estimate the number of nodes in the network a priori is out of our scope, since we focus on analyzing the waiting policy itself. However, we provide a simple *a posteriori* method, and evaluate in Section V whether it is suitable for our waiting policy.

Essentially, a node can just follow an adaptive trial-and-error approach for P_s and T_s. That is, while transmitting data, it tries P_s and T_s values and observes their impact on performance. If performance improves, i.e., an increase in P_s and T_s did allow to include more packets in a frame, the node continues increasing the parameters. Otherwise, it returns to a previous value known to provide gains. This requires no coordination among the nodes of the network, since each node can probe P_s and T_s values independently. Nodes try new values in two cases—first, periodically to determine if a different parameterization is more beneficial, and second, whenever performance decreases without having changed the parameters. This allows nodes to adapt dynamically to changes in the network. Additionally, if network conditions are stable, nodes can deduce throughput and delay trends after probing some P_s and T_s combinations. As a basic approach, nodes can attempt to fit curves on the values they observe, and use the curves to estimate performance for new parameter combinations. For obvious reasons, P_s should never be set higher than the maximum aggregation level allowed by the standard.

In Section V we analyze whether the behavior of our waiting policy is suitable for such a trial-and-error approach. Moreover, we show that the above fitting approach performs well, too, yielding good sum of squared errors (SSE) values when comparing our estimation to the actual performance.

IV. Scenario

In the following, we describe the network scenario and the traffic pattern that we consider in our evaluation.

A. Network

We consider an indoor 60 GHz network with one AP and N STAs following the 802.11ad standard. All STAs are located in one room, and have a line-of-sight link to the AP. Even though the STAs may use directional beamforming patterns to transmit data to the AP, recent work (c.f. Section II-A) shows that consumer-grade phased antenna arrays for 60 GHz devices exhibit significant side lobes. Hence, we consider that all STAs interfere but also overhear each other if they transmit simultaneously. This reduces the number of collisions due to deafness, and is thus a worst case scenario for our waiting policy, which particularly benefits from collisions in the uplink. That is, in a scenario with deafness the performance of our waiting policy would be even better than our results in Section V. Further, we use CSMA/CA at the MAC layer, which is likely to become the main protocol for medium access in 802.11ad hardware. At the time of writing, existing consumer-grade 60 GHz devices only implement CSMA/CA at the MAC layer [4]. We consider an error-free channel where all packet drops are due to collisions only. In addition, both AP and STAs use the same transmission rate for communication.

B. On-Off Markov Model Traffic Pattern

The statistical characteristics of packet arrivals have a strong impact on aggregation opportunities. The burstier the traffic, the higher are the benefits of the aggregation policy described in this paper. For the evaluation, we use a basic On-Off Markov Model (OOMM) traffic generator to simulate bursty application layer traffic.

The OOMM models the traffic as a two-state Markov model. In the "on" state, the traffic generator sends packets with a fixed length at a constant data rate, whereas in the "off" state it does not transmit at all. The traffic generator performs random experiments and based on the outcome switches back and forth between both states. The duration of the "on" and "off" periods follows truncated exponential distributions. Figure 2 shows an OOMM traffic output example. The model allows us to adjust the burstiness of the traffic by tuning the probabilities of switching between states (and thus the distributions of state duration) as well as the packet generation rate. Table I gives an overview on the statistical parameters of OOMM.

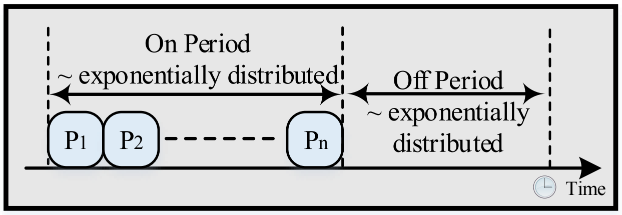

Fig. 2. On-Off Markov Model example.

TABLE I
On-Off Markov Model Traffic Pattern Parameters

Parameter	Distribution	Value
"On" duration	Truncated Exponential	Constant Data Rate = 1 gbps
		Mean = 600 μs, Bound = 6 ms
"Off" duration	Truncated Exponential	Mean = 2 ms, Bound = 20 ms

V. Evaluation

Practical evaluation of wireless networking in the 60 GHz band is challenging due to the lack of both consumer-grade and experimental hardware. In particular, software-defined radio hardware is not available at the time of writing. This prevents us from evaluating our waiting policy in practice, since it interacts with the MAC layer and the Physical Layer (PHY). Hence, we must resort to simulations to assess our scheme in 802.11ad networks. Specifically, we use the 802.11ad model provided by [29]. The model provides an accurate level of implementation for 802.11ad frame structure and channel access periods in ns-3.

We classify our results by scenario, that is, uplink/downlink using User Datagram Protocol (UDP) as transport protocol.

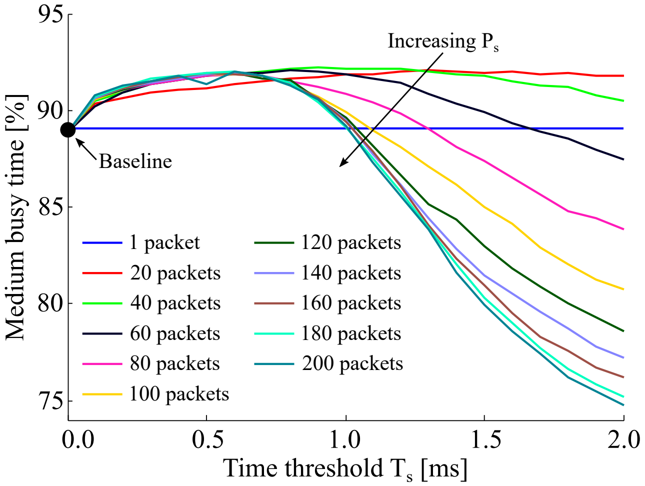

Fig. 3. UDP uplink scenario: medium usage

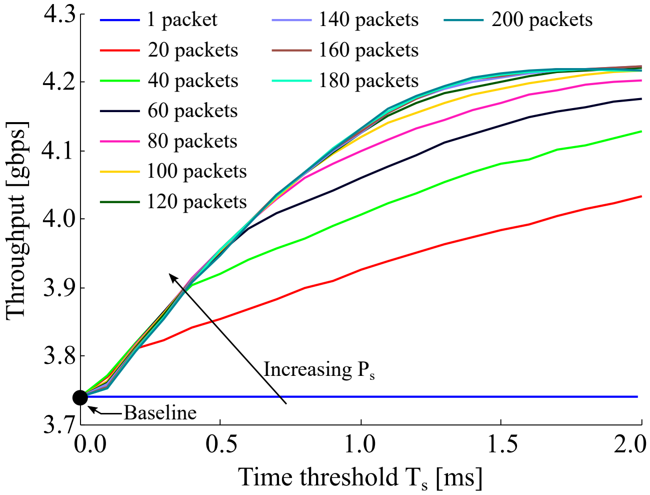

Fig. 4. UDP uplink scenario: total throughput.

Table I summarizes further parameter values of our traffic patterns. For our results, we consider the following metrics:

- **Medium busy time**, which we compute as the ratio of the total time spent transmitting—including preamble, header, and payload of all frames—-to the total simulation time.
- **Total throughput** received at the MAC Service Access Point (SAP) of the access point (uplink case) or the sum of this throughput received at all STAs (downlink case).
- **Packet Delivery Delay**, which we measure from the moment the application generates packet until it is received successfully by the the intended receiver.

We note that the results of each experiment is the average of 20 to 60 seconds of simulation time. Moreover, all our results in this section show the performance gain of our waiting policy *on top* of regular frame aggregation gains.

A. Uplink Scenario

We start with the uplink scenario in which multiple STAs have some data to upload to the AP. For each experiment, we study a range of P_s and T_s values. In most cases, we set $P_s \in [1..200]$ in steps of 20 packets, and $T_s \in [0..2]$ ms in steps of 100 μs. Further, we consider $N \in [5, 10, 15]$ STAs. Figure 3 depicts the medium usage for different packet thresholds P_s and time thresholds T_s for the case of 15 STAs. The black dot at $P_s = 1$ and $T_s = 0$ is the baseline, since with these parameters a STA does not introduce any artificial delay and transmits as soon as it has at least one packet. In Figure 3, the medium busy time initially rises for all packet thresholds until reaching a certain time threshold, beyond which it decreases significantly. The underlying reason is that each STA is able to aggregate more packets, thus reducing MAC overhead and specifically time spent for backoff. Figure 4 confirms the corresponding throughput increase. The medium usage decreases again beyond the aforementioned time threshold occurs as soon as we aggregate enough to deliver all packets. Beyond this threshold, further aggregation just increases the channel idle time. Again, this matches Figure 4, which shows that the throughput increase stabilizes after that time threshold.

Further, Figure 4 shows that the higher the packet threshold, the higher the throughput gain that we can achieve for a certain time threshold. The reason is that for low values of P_s we often wait less than T_s since we receive enough packets to satisfy the packet threshold before reaching the time threshold. Hence, we aggregate less. Conversely, if we set T_s to a large value, we wait longer on average and thus aggregate more.

Figure 5 shows the delay for 5 and 10 STAs. The delay includes propagation delay and channel access delay. For both cases, the delay decreases until reaching a minimum. Beyond that, it increases quasi-linearly with the time threshold but changes its slope at a certain point. This slope change occurs for the time threshold beyond which, *on average*, the policy hits the P_s threshold before it reaches the T_s threshold. This is the reason why the slope change takes place at lower T_s for lower P_s. We observe similar slope changes in Figure 4—the throughput increases less beyond that point since we wait for fewer packets. Note that Figure 4 shows the case for 15 nodes, while Figure 5 depicts the results for 5 and 10 nodes. While we do not show the throughput figure for 10 nodes due to space constraints, we observe similar effects as those for the case of 15 nodes. These slope changes show the importance of the packet threshold. While P_s limits the throughput increase, it also limits the delay increase, allowing a node to use a larger T_s. For traffic with a highly irregular burst spacing, we expect P_s to have a large impact since it prevents waiting if a node has enough packets.

Further, Figure 5 shows that the delay improvement is much larger for 10 STAs than for 5 STAs. This difference is due to the fact that, the more STAs contend for access to the channel, the higher is the probability of collisions. Such packet losses result in very high MAC overhead. Introducing an artificial delay allows nodes to aggregate more and thus access the channel less frequently, hence reducing the probability of collisions. This has a more significant impact for 10 STAs (and

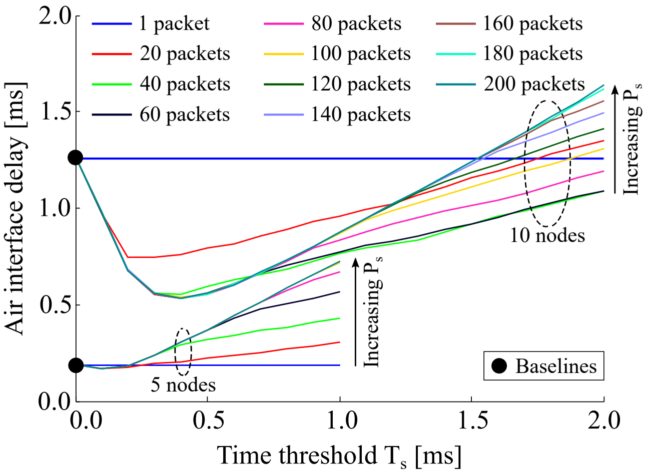

Fig. 5. UDP uplink scenario: air interface delay.

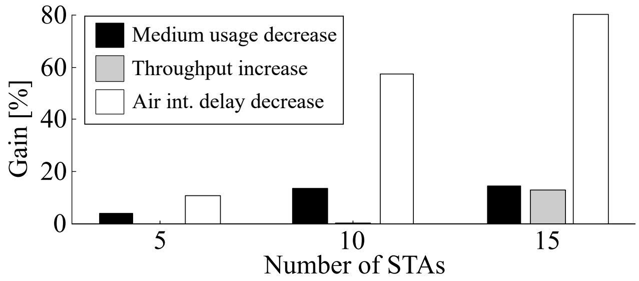

Fig. 6. UDP uplink results.

even more so for 15 STAs), since with 5 STAs the collision probability is low. Thus, waiting only provides a slight delay improvement in the latter case.

Figure 6 shows an overview of the gains for different number of STAs. For clarity, we only show the results for the P_s and T_s values that maximize the gain in each scenario and for each metric. As expected, the higher the number of STAs the higher the medium usage, delay, and throughput gains that we obtain. We observe that our waiting policy is particularly beneficial in terms of air interface delay, achieving up to 80% improvement. This highlights the relevance of artificial delay in contention scenarios.

B. Downlink Scenario

The UDP downlink is fundamentally different from the uplink case, since only the AP accesses the channel and thus no collisions occur. Figure 7 depicts the throughput and delay gains in this scenario where all STAs generate the same amount of traffic. In this case, our baseline to compare against is a FIFO scheduler with aggregation, which sends packets in the order of arrival. Figure 7 shows that the maximum weight scheduling policy—and the implicit delay for STAs that did not yet accumulate as many packets—allows us to achieve up to 247 mbps net throughput gain. In this case, the gain is exclusively due to the burstiness of the traffic pattern, since this scenario does not suffer from collisions.

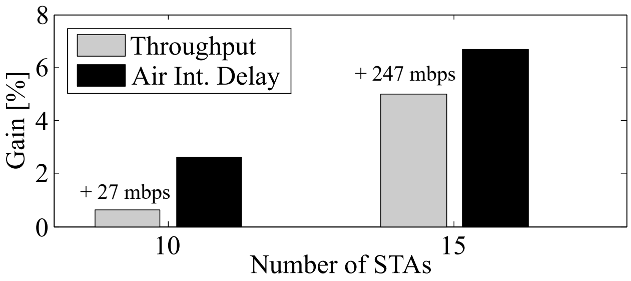

Fig. 7. UDP downlink results.

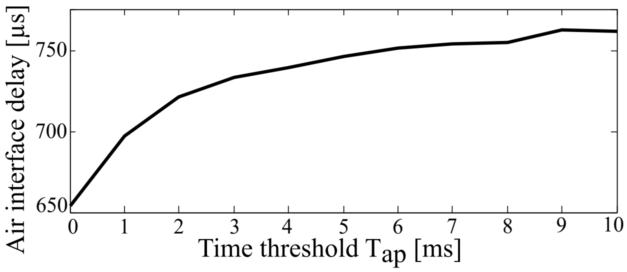

Fig. 8. UDP downlink scenario: air interface delay.

Finally, Figure 8 shows the average end-to-end delay with respect to the time threshold T_{ap} for stations with different traffic patterns. We note that the higher the value of T_{ap}, i.e., the longer the packets may reside in the queue, the more the average end-to-end delay increases. This increases short-term unfairness as packets are delayed until flows which allow for more aggregation are served. Reducing the time threshold in turn reduces this unfairness and ensures that a STA with low traffic intensity are served in a timely manner.

1) Parameter Estimation: Finally, we discuss whether the parameter estimation method that we suggest in Section III-C is suitable for our waiting policy for the uplink case. Our results show that small increases in P_s and T_s do not cause abrupt changes in performance. That is, following a simple trial-and-error approach is unlikely to produce significant performance penalties even in case of wrong P_s/T_s parameterizations. Finally, we also evaluate the curve fitting approach sketched in Section III-C for the case of delay performance. We obtain an SSE value of 0.0282 ms, which suggests that this method is also well suited to estimate P_s and T_s. The corresponding graphs are omitted due to space constraints.

VI. Discussion

Our results in Section V show that although it may be counter-intuitive, introducing artificial delay in wireless networks may significantly increase performance. Most importantly, we provide insights into when and how long a STA should wait, as well as the fundamental tradeoffs of such artificial delay.

Performance. Waiting exhibits a significantly different behavior in the uplink compared to the downlink due to contention (c.f. Section V). However, this effect is not necessarily limited to the uplink, since a network with multiple APs in the same interference domain would have to deal with potentially high high contention. Such a scenario is particularly relevant for 60 GHz networks since such networks may require multiple APs per room to ensure coverage. Further, consumer-grade 60 GHz devices are likely to exacerbate contention due to significant side lobes [4].

Parameterization. In most cases, using our waiting policy without prior knowledge of network conditions is safe. Conservative time and packet thresholds typically provide gains, as discussed in Section V-B1. Based on our results, we provide some rough recommendations on how to set P_s and T_s, as an alternative to estimation methods such as the one we suggest in Section III-C. First, a node should evaluate whether (a) it observes frequent collisions and (b) its transmission queue becomes empty periodically. This provides a basic notion on the network conditions. If neither (a) or (b) occur, the node need not use our waiting policy, since the baseline aggregation policy provides the same gains. However, such a permanently backlogged case is unusual. Second, a node should set P_s and T_s more conservatively the less contention it observes, and the longer the intervals are at which its queue becomes empty.

Cost. A key feature of our waiting policy is that it causes *zero communication overhead*. In other words, a node may simply *test it* to estimate the possible gains with only a minimal risk, i.e., there is little to loose in terms of potential delay, but significant performance improvements to gain.

VII. Conclusion

We design an aggregation-aware MAC scheduling policy for 802.11ad wireless networks which introduces a maximum weight scheduling policy for the downlink and an artificial delay for the uplink. In case of bursty traffic, this enables nodes to aggregate more packets of the current burst at the MAC layer. This is particularly beneficial for 802.11ad since its channel access overhead per transmission is extremely large. Our policy allows STAs to wait for either a minimum amount of packets to aggregate or a maximum amount of time, whichever comes first. We implement and evaluate this policy in ns-3 for uplink and downlink scenarios. Our policy is beneficial in two ways. First, since more aggregation reduces the number of channel accesses, it reduces collisions in case of high contention. Second, for bursty traffic, it avoids that a small number of packets at the end of a burst require a costly individual medium access. In our experiments, we achieve up to 480 mbps throughput increase and 80% channel access delay reduction.

VIII. Acknowledgments

This work is partially supported by the European Research Council grant ERC CoG 617721, the Ramon y Cajal grant from the Spanish Ministry of Economy and Competitiveness RYC-2012-10788, and the Madrid Regional Government through the TIGRE5-CM program (S2013/ICE-2919).

References

[1] IEEE 802.11 Working Group, “IEEE 802.11ad, Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band,” 2012.

[2] P. Smulders, “Exploiting the 60 GHz Band for Local Wireless Multimedia Access: Prospects and Future Directions,” *IEEE Communications Magazine*, vol. 40, no. 1, 2002.

[3] R. C. Daniels and R. W. Heath, “60 GHz Wireless Communications: Emerging Requirements and Design Recommendations,” *IEEE Vehicular Technology Magazine*, vol. 2, no. 3, 2007.

[4] T. Nitsche, G. Bielsa, I. Tejado, A. Loch, and J. Widmer, “Boon and Bane of 60 GHz Networks: Practical Insights into Beamforming, Interference, and Frame Level Operation ,” in *Proc. CoNEXT’15*, 2015.

[5] C. J. Hansen, “WiGiG: Multi-gigabit wireless communications in the 60 GHz band,” *IEEE Wireless Communications*, vol. 18, no. 6, 2011.

[6] M. Park, C. Cordeiro, E. Perahia, and L. L. Yang, “Millimeter-wave Multi-Gigabit WLAN: Challenges and Feasibility,” in *Proc. of IEEE PIMRC’08*, 2008.

[7] C. Cordeiro, D. Akhmetov, and M. Park, “IEEE 802.11Ad: Introduction and Performance Evaluation of the First Multi-gbps Wifi Technology,” in *Proc. of ACM mmCom’10*, 2010.

[8] C. Cordeiro, “Evaluation of Medium Access Technologies for Next Generation Millimeter-Wave WLAN and WPAN,” in *Proc. of IEEE ICC’09*, 2009.

[9] X. Tie, K. Ramachandran, and R. Mahindra, “On 60 GHz Wireless Link Performance in Indoor Environments,” in *Proc. of PAM 2012*, 2012.

[10] S. Sur, V. Venkateswarana, X. Zhang, and P. Ramanathan, “60 GHz Indoor Networking through Flexible Beams: A Link-Level Profiling,” in *Proc. of ACM SIGMETRICS’15*, 2015.

[11] D. Skordoulis, Q. Ni, H.-H. Chen, A. P. Stephens, C. Liu, and A. Jamalipour, “IEEE 802.11n MAC frame aggregation mechanisms for next-generation high-throughput WLANs,” *IEEE Wireless Communications*, vol. 15, no. 1, 2008.

[12] B. Ginzburg and A. Kesselman, “Performance Analysis of A-MPDU and A-MSDU Aggregation in IEEE 802.11n,” in *IEEE Sarnoff Symposium’07*, 2007.

[13] E. H. Ong, J. Kneckt, O. Alanen, Z. Chang, T. Huovinen, and T. Nihtilä, “IEEE 802.11ac: Enhancements for Very High Throughput WLANs,” in *Proc. of IEEE PIMRC’11*, 2011.

[14] “A Reliable A-MSDU Frame Aggregation Scheme in 802.11n Wireless Networks,” *Procedia Computer Science*, vol. 21, 2013.

[15] “An Enhanced A-MSDU Frame Aggregation Scheme for 802.11n Wireless Networks,” *Wireless Personal Communications*, vol. 66, no. 4, 2012.

[16] A. Majeed and N. B. Abu-Ghazaleh, “Packet Aggregation in Multi-Rate Wireless LANs,” in *Proc. of SECON’12*, 2012.

[17] J. Gross, O. Puñal, and M. Emmelmann, “Multi-user ofdma frame aggregation for future wireless local area networking,” in *IFIP Networking’09*, ser. Lecture Notes in Computer Science, 2009, vol. 5550.

[18] B. Bellalta, J. Barcelo, D. Staehle, A. Vinel, and M. Oliver, “On the Performance of Packet Aggregation in IEEE 802.11ac MU-MIMO WLANs,” *IEEE Communications Letters*, 2012.

[19] P. Dely, A. Kassler, N. Bayer, H. J. Einsiedler, and D. Sivchenko, “FUZPAG: A fuzzy-controlled packet aggregation scheme for wireless mesh networks,” in *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*, 2010.

[20] N. T. J. Bailey, “On Queueing Processes with Bulk Service,” *Journal of the Royal Statistical Society. Series B*, vol. 16, no. 1, 1954.

[21] J. H. Hong and K. Sohraby, “On Modeling, Analysis, and Optimization of Packet Aggregation Systems,” *IEEE Transactions on Communications*, vol. 58, no. 2.

[22] M. F. Neuts, “The Busy Period of a Queue with Batch Service,” *Operations Research*, vol. 13, no. 5, 1965.

[23] ——, “A General Class of Bulk Queues with Poisson Input,” *The Annals of Mathematical Statistics*, vol. 38, no. 3, 1967.

[24] M. M. S. Hyo-Seong Lee, “Control Policies for the M$^{\text{X}}$/G/1 Queueing System,” *Management Science*, vol. 35, no. 6, 1989.

[25] V. Makiš, “Optimal Control of a Batch Service Queueing System with Bounded Waiting Time,” *Kybernetika*, vol. 21, no. 4, 1985.

[26] A. Razi, A. Abediy, and A. Ephremides, “Delay Minimization with Channel-Adaptive Packetization Policy for Random Data Traffic,” in *Proc. of CISS’14*, 2014.

[27] Y. Lin and V. W. Wong, “Frame Aggregation and Optimal Frame Size Adaptation for IEEE 802.11n WLANs,” in *Proc. of IEEE GLOBECOM’06*, 2006.

[28] S. Byeon, K. Yoon, O. Lee, S. Choi, W. Cho, and S. Oh, “MoFA: Mobility-aware Frame Aggregation in Wi-Fi,” in *Proc. CoNEXT*, 2014.

[29] H. Assasa and J. Widmer, “Implementation and Evaluation of a WLAN IEEE 802.11ad Model in ns-3,” in *WNS3*, 2016.