Is WTSN the missing piece for low latency in general-purpose Wi-Fi?

Milind Kumar Vaddiraju[1], William Sentosa[1], Qinjun Jiang[1], Sarita Adve[1], Dave Cavalcanti[2], Dibakar Das[2], P. Brighten Godfrey[1,3], Javier Ramirez-Perez[4], Deepak Vasisht[1]

[1]University of Illinois Urbana-Champaign, [2]Intel Corporation, [3]Broadcom, [4]Ofinno

Abstract

The high latency and variability of current Wi-Fi networks severely impairs interactive networked applications like extended reality and cloud gaming, and even negatively affects web browsing. Recently, wireless Time-Sensitive Networking (WTSN) has emerged to offer powerful time synchronization and scheduling capabilities that can enable deterministic low latency. However, WTSN relies on precise advance knowledge of packet arrival times and tight integration between applications and a centralized network controller, limiting its scope to niche settings. Resolving WTSN's dependence on knowledge of packet arrival times is key to determining whether it can be a low latency enabler in general-purpose Wi-Fi. Thus, in this work, we ask: *are the stringent assumptions of WTSN necessary to achieve the low latency benefits?* Contrary to prevailing assumptions, we find that it is indeed possible to enable low tail and mean latency *without* prior knowledge of precise packet arrival even in the presence of high throughput background flows. We demonstrate this in simulation using a WTSN-enabled multipath design that partitions the network into two logical paths: one with very low latency and high reliability, and another offering high throughput at the expense of latency and reliability. Further, we describe how our design and WTSN can both complement the powerful OFDMA capabilities of Wi-Fi and present initial results for the same. We conclude by discussing deployability and promising future directions.

CCS Concepts

• **Networks** → Network protocol design; **Link-layer protocols**; **Wireless local area networks**.

Keywords

WTSN, Heterogeneous Virtual Channels, Multipath, OFDMA, XR

1 Introduction

Low latency, especially in the tail, is vital for burgeoning interactive applications like cloud gaming where a mere 0.5% rise in video frame stall rate due to high tail latency can drop user retention by 33% [27]. Applications like Virtual Reality (VR) and Augmented Reality (AR), collectively known as Extended Reality (XR), often adopt edge and cloud-assisted paradigms for lower power and higher compute capabilities. They rely on the network to provide extremely low latency to meet the stringent motion-to-photon latency requirements, e.g. 20ms for VR [5]. Further, this low tail latency must be achieved in the presence of flows that simultaneously increase uplink and downlink utilization. For instance, consider collaborative XR and emerging bidirectional 3D telepresence systems like Project Starline [13], which require 30 to 100 Mbps per device to be immersive with higher fidelity streaming likely requiring more. Highly time-sensitive flows within these applications must therefore successfully coexist with these bandwidth hungry flows over WLANs to ensure a satisfactory experience. This will only prove harder with the proliferation of services like 8K streaming and UHD VR increasing utilization [15] in homes and enterprises.

With its powerful time synchronization and time-aware scheduling capabilities, Wireless Time Sensitive Networking (WTSN) can enable deterministic latencies over Wi-Fi [3, 4]. Given the arrival times of time sensitive packets for, say, a VR application, WTSN reserves the medium in advance to prevent contention-related delays from Wi-Fi's randomized channel access. This has been successful for industrial applications in controlled settings [18, 23]. However, critically, it requires precise prior knowledge of packet arrival times, leading to two practical barriers. (1) It necessitates prohibitively tight integration between applications and a central network controller that collects information about packet priorities, arrival times, flow volumes, etc. to determine a suitable schedule. (2) Most applications do not have precisely predictable packet timing. As we discover (§3), even applications like XR with ostensibly periodic traffic patterns exhibit large jitter with respect to the ≤ 1 ms reservations of WTSN scheduling. This means either the application will often miss its sending slots, thus also missing the corresponding latency guarantee; or slots would have to be very large, leading to inefficient channel utilization. This is even truer for other common applications like web browsing which could benefit from low latency but have unpredictable, aperiodic traffic. Thus, WTSN's capabilities can presently only be efficiently utilized by applications with highly periodic traffic, severely limiting its practicality.

We argue that the key challenge in leveraging WTSN for general-purpose low latency is to *relax* its strict assumption about application traffic patterns. We ask: *Can applications still obtain the low latency benefits of WTSN without explicit prior knowledge of its packet arrivals?* A straightforward solution is to use WTSN to

schedule devices to transmit in a round robin (RR) fashion. While this improves performance over Wi-Fi's standard Carrier Sense Multiple Access (CSMA), it does not adequately improve tail latency. However, we observe that for most interactive applications (e.g. VR), only a *fraction* of traffic (e.g. pose information) is time sensitive and the remaining traffic just requires high bandwidth. This prompts a different approach: we propose partitioning the network into two logical "paths" (by which we mean two different types of service provided in parallel on a single wireless network) using WTSN tools: A Low Latency Path (LLP) offers low latency and high reliability with near zero packet drops, while a High Bandwidth Path (HBP) offers high throughput albeit with no latency guarantee. To design these paths, we exploit the following insights: (a) CSMA offers very low latencies at low utilization, and (b) RR efficiently supports high load. Therefore, we use WTSN to provision alternating periods of transmission for LLP and HBP traffic. The relatively small fraction of traffic that is time sensitive and uses the LLP receives quick access to the medium via CSMA while the large volumes of HBP traffic are transmitted without contention via RR.

We build a Wi-Fi simulator to evaluate such a scheme. We choose to design our own simulator as existing simulators like ns-3 do not currently support features such as Restricted Target Wake Time [7] that would enable us to implement the division of the network into multiple paths as described above. Our preliminary simulations show such a hybrid schedule improves significantly over CSMA, RR and its variants, and even Orthogonal Frequency-Division Multiple Access (OFDMA) schedules. Our design provides < 5 ms tail latency with high reliability for time sensitive traffic via the LLP while also serving the throughput needs of applications via the HBP, for up to 39% utilization of the network under typical wireless channel conditions. Finally, we show how latencies can be improved further with an adaptive scheme using just the long-term average load of each device, which is easily available information. Note that, here, by low tail latency we mean that the delivery time of packets (from the time of arrival from the application at the sender) is low (e.g. ≤ 5 ms) with a high probability (e.g ≥ 99%). Further, we expect this to be achieved without packets being dropped at some point in the network if they exceed the, say, 5 ms latency value in an attempt to ensure all delivered packets have low latency. Thus, we aim for a performance goal in between best-effort service (with no guarantees whatsoever) and perfectly deterministic latency (with guaranteed upper bound on the latency), offering generally consistent low latency with occasional outliers, which can benefit real-time applications [16, 27] (and others like web browsing [21]).

Previous work has referred to hybrid schedules like ours as *Heterogeneous Virtual Channels* (HVCs) and showed its utility for improving application performance [21, 25] in the context of 5G using network slices offering high bandwidth (called Enhanced Mobile Broadband or eMBB) and ultra-reliable low latency (called URLLC). However, that work took as a starting assumption that HVCs exist; we effectively explore how to build them. In any case, techniques from cellular networks cannot directly be leveraged in Wi-Fi networks due to fundamental differences in architecture, signaling and capabilities (§6). Our work presents an alternative for achieving HVCs in Wi-Fi networks with WTSN.

This work is a starting point for multiple interesting directions, including: the combining our HVC scheduling with OFDMA, which has strikingly similar problems to WTSN (special frames for time synchronization to avoid contention, and need for information about workload to apportion frequency resources); designing better HVCs by leveraging Multi-Link Operation (MLO); and alternate approaches to avoid WTSN's tight coordination with applications. Overall, we hope to prompt a discussion in the community about how WTSN and Wi-Fi's new capabilities can address the pressing need for *general-purpose* low latency wireless.

2 Background

CSMA/CA. Wi-Fi employs the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol, requiring devices to listen first, ensure the medium is idle, select a random duration to wait and transmit only if the medium is sensed idle at the end of that duration. This needs no central control and at low utilization, gives devices nearly immediate access to the medium as no one else is likely transmitting. At higher utilization, performance worsens as packets wait for extremely long periods before transmission due to the randomness of access [2, 9], which degrades tail latency severely [16, 24]. This will continue to be an issue even with the introduction of the 6 GHz band as utilization increases due to immersive applications (§1).

OFDMA. Wi-Fi 6 introduces OFDMA which allows users to be scheduled in the frequency domain in addition to the time domain. This is of particular interest in the uplink where multiple devices can transmit to the access point (AP) simultaneously by sharing the channel's bandwidth. To achieve this, the AP *triggers* multiple devices using a special trigger frame (TF) that informs them about the frequency resources, modulation schemes, etc. to use and allows them to transmit together. As the AP coordinates medium access, UEs no longer have to contend via CSMA and with appropriate scheduling, packets may experience lower latency. In addition to this scheduled access, OFDMA is also capable of uplink OFDMA-based Random Access (UORA) wherein the AP can designate some frequency resources as unscheduled and allow devices to contend for them using a random backoff based mechanism. While intended for association requests and control signalling, UORA presents several new opportunity for low latency network design.

WTSN adapts Ethernet's Time Sensitive Networking (TSN) [6] to Wi-Fi. Next, we describe its primary capabilities and the associated network management model. We refer to access points (APs) and User Equipments (UEs) collectively as stations (STAs).

Time synchronization: TSN-based time synchronization over Wi-Fi uses the IEEE 802.1 AS [1] standard implemented over 802.11 to achieve synchronization of the order of hundreds of nanoseconds (compared to 10s of microseconds or more for Wi-Fi's Timing Synchronization Function (TSF)). Further, TSF only synchronizes the AP and UEs belonging to a single Basic Service Set (BSS), whereas TSN allows synchronization both within a BSS and across APs.

Time-aware scheduling: In Ethernet networks, 802.1Qbv scheduling uses time synchronization and gates queues at the egress port of the Ethernet switch to regulate which one transmits, thus creating a schedule. In the wireless domain, this gating of queues is done over the 802.11 MAC layer. With precise time synchronization and gating of queues, a schedule dividing time into periods (or slots) when distinct STAs can transmit can be created and enforced.
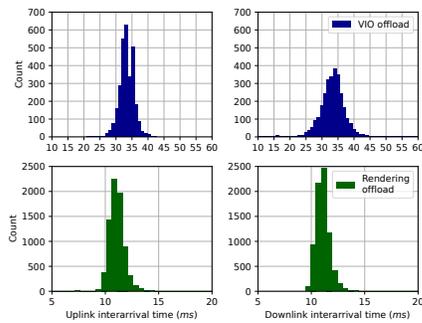
Figure 1: The uplink and downlink data interarrival histogram for VIO and rendering offload.

Network management: WTSN has been envisioned for use in managed networks like controlled industrial settings and enterprises with a central controller acting as the global scheduler. This scheduler collects flow- and packet-level information from the UEs, computes a suitable schedule based on QoS requirements and shares it with all devices. Applications must follow this schedule precisely to obtain deterministic latency. This needs tight integration between the applications on UEs and the controller. Removing this by making the schedule independent of precise packet arrival information is our main goal. Questions of coordination between multiple APs, wired routers and multiple controllers are left to future work.

3 Are common applications predictable enough for WTSN?

Next, we show that the traffic of many applications is quite unpredictable, making it hard to design WTSN's schedules.

Extended reality (XR) traffic. We examined XR traffic as it is expected to benefit greatly from WTSN due to its periodicity. We collected traffic from the open-source XR system ILLIXR [8], which supports offloading its (1) Visual Inertial Odometry (VIO, used for head tracking [10]) and (2) application rendering components. Once offloaded, the network is critical. We ran both the client and server processes in a single machine and logged packet timing.

The VIO traffic consists of a client sending two compressed stereo camera images (each at 2 Mbps), plus the acceleration and angular velocity of the user's head from the inertial measurement unit (IMU), to the server. The data delivery follows the camera frequency at 30 frames per second (FPS). Upon receiving the data, the server runs the VIO algorithm, computes a 6-degree-of-freedom (6DOF) pose of the user's head, and sends the pose back to the client. The rendering traffic consists of the client sending poses to the server and the server sending the compressed rendered frames to the client, with bitrate of 240 Mbps and a frame rate of 90 FPS.

Fig. 1 shows the interarrival times of the uplink and downlink data for VIO offloading and rendering offloading. Although XR traffic is periodic, there is still variability in the interarrival times. For VIO offloads, both uplink and downlink shows a mean interarrival time of 33.3 ms, with standard deviation 2.15 ms and 4.79 ms for uplink and downlink respectively. This variability also depends on the component being offloaded. In the case we analyzed, VIO offloading (10.3% Coefficient of Variation or CV) has somewhat more variability than offloading rendering (8.2% CV).

The variability can be addressed by either extending the allocated time slot or increasing the frequency of slot openings. However,

this can hurt bandwidth. Another approach is to apply Real Time Computing mechanisms to make application behavior more deterministic and synchronized with the network's schedule. This is a valid approach that may be an interesting research direction; however, it limits the practicality of the approach for general-purpose (non-RTC) applications. In our work, we choose to focus on decoupling the schedule from any expectation of packet arrivals.

Web browsing traffic does not require the kind of deterministic latency enabled by WTSN for industrial applications. However, previous work [21] has shown how ultra-reliable low-latency can significantly improve page load times and user experience. Thus, we wish to explore what benefits WTSN could bring to applications like web browsing. Web traffic represents a general-purpose workload that is aperiodic; we verified this numerically by collecting packet traces of loading the 3 most popular pages according to [22] (the landing pages of Google, YouTube, and Facebook). We grouped packets into bursts, and found the time between bursts varies roughly uniformly between 1 ms - 25 ms, with outliers up to $\approx$ 1 sec (standard deviation 3.85 and 3.62 times the mean for uplink and downlink respectively). Thus, if we hope to help such applications, we cannot depend on alignment with the specific times they choose to send packets.

4 Traffic-agnostic WTSN

Clearly, real world applications do not have precise traffic periodicity and may even have fully random traffic. Thus, we evaluate in simulation how WTSN does when packet arrival is unpredictable in order to generalize it for any application. We describe our simulator next and present the various schemes – RR, LDRR, OFDMA – and their performance in §4.2, and in §4.3, we design HVCs.

4.1 Simulation details

Wi-Fi simulator. We built a simulator[1] for the CSMA/CA behaviour of stations utilizing the Distributed Coordination Function of WiFi. The simulator performs discrete-event simulations with events including random backoff, data transmission, SIFS, DIFS and acknowledgement. If no STA requires access to the medium, time advances in increments of 10 μs. In our simulations, STAs first wait for a short period (DIFS) and sense if another device is transmitting. If one is, then the remaining STAs defer transmission. If not and a STA has packets to send, it waits for an additional randomly selected backoff period before transmitting. Each transmission consists of sending data, a short wait and then an acknowledgment. STAs can aggregate multiple IP packets into one transmission with the same MAC/PHY headers to reduce overhead. All aggregated packets have the same delivery time. If multiple STAs choose the same backoff, then a collision occurs, packets from all STAs are dropped and their contention window is doubled. We enable TSN schedules by determining periods, called slots, in which only specified STAs transmit. If multiple STAs are allocated to a slot, they will still compete using CSMA/CA. When a slot is assigned to a single STA, it will inevitably win and successfully transmit data The slot's length limits the maximum number of packets a STA can aggregate at once (equivalently throughput) as transmissions do not spill over into the next slot. In uplink OFDMA simulations,

---

[1]https://bit.ly/42oHY14

only the AP contends for the medium and wins. It then sends a trigger frame, waits (SIFS), receives uplink data from UEs, waits (SIFS) and finally sends a multi-user block acknowledgement. UEs can aggregate packets and all packets have the same delivery time as the UE that takes the longest to transmit owing to how OFDMA operates in the physical layer. We ignore RTS and CTS. In essence, our simulator primarily captures link layer (and some physical layer) behaviour. The layers above are represented by a trace that indicates packet arrival times-these can be randomly generated (as described below) or obtained from an application.

Experiment configurations: For all results we assume 9 STAs: 1 AP and 8 UEs. DIFS is 34 μs. Each STA uses 80 MHz when transmitting, has 2 spatial streams and may aggregate up to 150 packets, each of size 964 B, into one transmission. All STAs operate at an MCS of 7 transmitting data using 64 QAM and 5/6 coding rate. We assume an SNR of 24 dB, for IEEE channel model D and thus a packet error rate (PER) of 0.1. Our OFDMA experiments have the same setup except that all the UEs are triggered every time in uplink with each getting an 8 MHz share of the bandwidth. This is a valid division of the 80 MHz channel from before.

Traffic: Packets follow a Poisson arrival process with the rate λ for each STA. Poisson arrival essentially represents no knowledge of packet arrival times. The behaviour of STAs following a schedule for 10s is simulated multiple times for varying λ. During every transmission, a STA transmits using packet aggregation with earlier packets being selected for service first. We measure the latency of a packet as the difference between its arrival and delivery times.

4.2 Transmission Schemes and Results

Fig. 2a presents the results of simulating CSMA, round robin (RR) and load dependent round-robin (LDRR) and the HVC schemes (§4.3). Fig. 2b presents the results of simulating CSMA, OFDMA under specific conditions and an adaptive HVC scheme from §4.3.

CSMA vs. RR: From Fig. 2a, CSMA has lower latency at low loads but degrades very rapidly. In round robin (RR), all STAs take turns trasmitting in a round robin fashion. WTSN enables this by (a) specifying a schedule for various STAs in the network on top of CSMA and (b) synchronizing STAs precisely so that they do not trespass into each other's slots. RR outperforms CSMA for moderate to high loads. We emphasize that this gain occurs with a predetermined schedule that was designed without knowing the application packet arrival times. For RR, the point of inflection in the latency curve indicates the throughput achievable. There is a clear trade-off between throughput and latency: smaller slot sizes (400 μs), have lower latency but lower throughput and longer ones (2000 μs) have higher latency but also higher throughput (about 3×). We exploit this to design an adaptive RR scheme.

LDRR In RR, the slot lengths are fixed despite what λ is. We now present a scheme, which we call load dependent round-robin (LDRR), where λ is assumed known beforehand (in practice it can easily be estimated by observing traffic for a short period of time). The slot length is set in advance by making it long enough to allow the transmission of the average number of packets that would arrive while a STA is waiting its turn in a RR sequence for each value of λ. As Fig. 2a shows, LDRR outperforms the static RR scheme for a large range of arrival rates except at very high throughput where
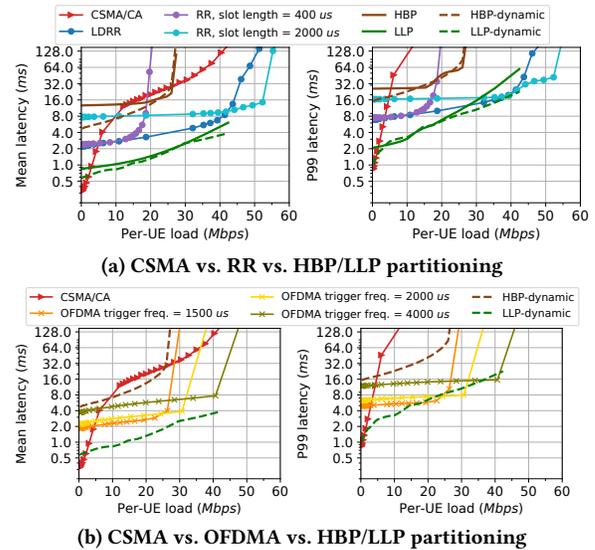
(a) CSMA vs. RR vs. HBP/LLP partitioning

(b) CSMA vs. OFDMA vs. HBP/LLP partitioning

Figure 2: The mean and p99 latency of multiple scheduling schemes under various per-UE loads.

its selection of a shorter slot deprives it of the benefits of retries that the 2000 μs scheme has. This could be addressed with further refinement of LDRR which we leave to future work. Although we have considered all STAs to have the same λ in our examples, when different STAs have different λs, slots can be sized proportionally to serve each user's traffic while keeping the latency low. Note that this scheme will fall short when λ changes rapidly due to the overhead of repeatedly communicating a new schedule to all STAs.

OFDMA In uplink OFDMA, when the AP chooses to trigger and who it triggers greatly affect UEs' latency. To do this well, the AP needs to know channel conditions, UE queue lengths, packet priorities, etc. It also relies heavily on a vendor's implementation. We discuss these practical concerns in greater detail in §5 and describe how WTSN and HVCs can be used to mitigate their overheads. Here, however, we assume that the AP already knows that all UEs always have traffic and ignore the overheads of collecting queue information, etc. Therefore, it chooses to trigger all UEs with some trigger frequency. In our schedule, the AP triggers UEs in the uplink 8 times before performing a downlink transmission using the full bandwidth the ninth time. The fraction of airtime dedicated to uplink and downlink are the same as in the RR case. The interval between successive triggers limits how many packets can be aggregated in one transmission. From Fig. 2b, just like in the RR case, throughput increases with the trigger interval and there is a trade-off with latency. When used this way, OFDMA outperforms CSMA due to more efficient medium utilization.

While RR and LDRR do reduce latency, they cannot cater to applications requiring extremely low tail latency, even if only for a small fraction of their traffic. While OFDMA does perform well, it is with very specific assumptions. Consequently, we explore another way to leverage WTSN to meet the needs of interactive applications.

4.3 HVCs using WTSN

Leveraging HVCs Given the tradeoff between throughput and latency, one approach is to provide two types of service in parallel – one high throughput (without latency or reliability guarantees), and

one low latency (while sacrificing throughput significantly). This pattern is referred to in [25] as Heterogeneous Virtual Channels (HVCs); an example is 5G's eMBB and URLLC slices. Multipath solutions such as DChannel [21] have shown how paths such as eMBB and URLLC can be used simultaneously to improve web browsing Page Load Time by 16-40%. Solutions like DChannel and the transport layer solution of [25] boost application performance by speeding up a small portion of the traffic via the low latency path while utilizing the high throughput path for the remaining traffic. For example, the low latency path may carry ACKs, small packets, control messages, XR pose information, or in [25] the base layer of a scalable multi-layer video codec. Both works envision using the URLLC and eMBB slices offered by 5G as HVCs. Previous work [27] has also shown how two paths, of which one has lower latency but can only be sparingly utilized, can be used in parallel to improve the performance of cloud gaming applications.

HVCs in Wi-Fi with WTSN From Fig. 2, it is evident that there is a trade-off between throughput and latency. Further, CSMA offers the lowest latency at low utilization while RR performs best at high load. We exploit this and utilize WTSN to create two virtual paths in the Wi-Fi network: one with low latency and high reliability properties but low throughput and the other with high throughput that sacrifices latency and reliaiblity.

As before, consider 9 STAs transmitting at rate λ. Assume that each STA enqueues 5% of its traffic for which it would like extremely low tail latency in a queue called Low Latency Path (LLP) and the remaining 95% in a queue called High Bandwidth Path (HBP). What this 5% is can be determined using a heuristic like in [21] or specified by the application [20, 25, 26]. With this information, we design a schedule consisting of 1.5 ms slots. In every odd numbered slot, starting from the first, all 9 STAs compete using CSMA but only with the LLP packets, thus ensuring low utilization. Unlike in our RR schedules, the LLP packets are transmitted at MCS 2 i.e using QPSK (versus MCS 7's 64 QAM) and a coding rate of 3/4 in order to make transmissions impervious to multipath fading and dropping PER to 0 for the channel conditions we consider. In the even numbered slots, we use an RR schedule with the HBP traffic from STAs.

Fig. 2a shows the LLP traffic has tail latency lower than the LDRR scheme until about 27 Mbps (per STA), and less than 5ms up to 15 Mbps! In effect, this creates a low latency HVC with high reliability within the Wi-Fi network which operates with no prior knowledge of packet arrivals. However, low LLP latency comes at the price of throughput. HBP packets have a permissible latency only until about 25 Mbps – about half the throughput of RR.

Dynamic HVCs HBP experiences elevated latencies even at low loads (Fig. 2a). This is as we choose a static slot length irrespective of λ. To redress this, we adjust the slot lengths depending on λ as we did with LDRR, except accounting for the facts that (a) there are twice as many slots between successive HBP transmissions for a UE due to there being two paths and (b) the LLP uses a much lower MCS for reliability and thus needs correspondingly longer slots. The dynamic-HBP has much lower latency than HBP. Interestingly, dynamic-LLP also offers lower latency than LLP at higher λ. This is due to the selection of longer slots for transmission which increase the likelihood of aggregating more packets at once, thereby decreasing tail latency. Dynamic-LLP also offers lower latency than OFDMA-based schemes until a per-UE load of about 18 Mbps. Thus,

we conclude that WTSN-based HVCs offer a valuable design point in Wi-Fi. In §5, we discuss the trade-offs of HVCs and how they can be coupled with OFDMA for even lower latency.

5 Discussion and open questions

Trade-offs of HVCs. Compared to RR, the HBP clearly sacrifices throughput to enable the LLP's low latency and reliability guarantees. Despite this, we believe that this is a promising design for several reasons. First, our parameters for slot lengths were chosen to be illustrative. They can be tuned depending on channel conditions which might allow an MCS higher than our very conservative MCS 2 for LLP, or even using finer grained knowledge of application needs, if available. Further, this scheme offers one operating point-the network can simply switch to any of the other schemes when it detects a performance deterioration. Second, we assume a modest setting of MCS 7 for the HBP to account for typical conditions. However, in settings like enterprise VR, devices can operate close to the AP with Line of Sight and utilize the higher rates (up to 2.4 Gbps) that newer Wi-Fi generations offer. Then, some throughput can be easily be given up to attain low, deterministic latency.

OFDMA: practical concerns. Our OFDMA simulations used optimistic assumptions about traffic knowledge and trigger behaviour which may not apply in practice, e.g., [17] showed commercial OFDMA-enabled APs will not perform uplink OFDMA until obscure vendor-specific conditions are met, resulting in increased latencies. Even if APs trigger OFDMA, collecting frequent queue information from UEs for scheduling incurs significant overhead as Wi-Fi lacks a dedicated control channel. Finally, throughput-heavy background flows may cause OFDMA transmissions to lengthen, thus delaying latency-sensitive packets sharing frequency resources with them. We discuss how these issues might be addressed next.

OFDMA, WTSN and HVCs. Schemes such as what we describe in §4.2 can be implemented with Restricted Target Wake Time (R-TWT), a Wi-Fi feature that is part of WTSN. It can help OFDMA as it allows partitioning time into slots and specifying AP triggering. With Wi-Fi's native time synchronization TSF (§2), these slots cannot be made too small as a significant portion has to be reserved as a guard interval. However, with WTSN, these slots (equivalently, intervals between successive triggers) can be made smaller, leading to lower latency. In fact, Fig. 2b already illustrates how smaller slot lengths can decrease latency and create new operating points. Further, WTSN's time-aware scheduling can be used to shape traffic at queues in order increase the efficiency of OFDMA transmissions.

Short packet transfer. A problem that both OFDMA and queuing-based schedules reveal is that it is very inefficient to transfer just a few bytes of information (e.g., queue lengths) quickly, either because of the medium access protocols or the overhead of headers and frame design. This is because networks are optimized for long packets [11] and throughput. Exploring *how a few bytes of information can be transferred quickly* over a Wi-Fi network could help create a control channel for queue information and accelerate small application messages, transport ACKs, etc. This exploration is especially timely given similar discussions in the cellular domain.

Learning-based approaches. A number of traffic workloads and a number of network configurations are possible. A learning algorithm can be used to characterize possible workloads into templates, map a current or predicted workload to a template and

determine how to tune parameters in order to optimize a metric (say, latency). Previous work on self-driving radios [12] is in a similar vein, although in the context of PHY in cellular networks.

6 Related work

Many recent efforts to support interactive applications over Wi-Fi operate at the higher layers of the network stack. For instance, they might attempt to enable low latency by accelerating congestion control (Zhuge [16]) or supplementing WiFi with a cellular path (AUGUR [27]). However, they are still vulnerable to Wi-Fi's latency and could greatly benefit from our solution. Theoretical work, such as airtime slicing [19] continues to optimize for throughput, only focuses on the downlink and leverages dropping packets for bounded delay, contrary to our requirement of ultra-reliability. Other works on scheduling [14] assume perfect knowledge of queues at UEs without considering the associated overheads that we have described in §5. Finally, while our proposal bears similarity to network slicing based solutions in 5G such as eMBB and URLLC, techniques from the cellular domain are inapplicable in Wi-Fi. This is because Wi-Fi (a) lacks dedicated control channels that collect channel and queue information and, (b) simply does not support techniques like puncturing already scheduled transmission which cellular networks can accomplish at fine time and frequency granularity.

7 Conclusion

We have explored some temptingly useful designs that WTSN enables, the practical challenges of utilizing techniques like OFDMA and the multiple research directions therein. With this paper, we hope to draw the community's attention towards the exciting possibilities for achieving low latency in general-purpose Wi-Fi.

Acknowledgments

References

[1] IEEE 802.1. 2024. Time-Sensitive Networking (TSN) Task Group. https://1.ieee802.org/tsn/ [Last accessed on 27 June 2024].

[2] Luciano Bononi, Marco Conti, and Enrico Gregori. 2000. Design and performance evaluation of an asymptotically optimal backoff algorithm for IEEE 802.11 wireless LANs. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*. IEEE, 10–pp.

[3] Dave Cavalcanti, Carlos Cordeiro, Malcolm Smith, and Alon Regev. 2022. WiFi TSN: Enabling Deterministic Wireless Connectivity over 802.11. *IEEE Communications Standards Magazine* 6, 4 (2022), 22–29.

[4] Dave Cavalcanti, Javier Perez-Ramirez, Mohammad Mamunur Rashid, Juan Fang, Mikhail Galeev, and Kevin B Stanton. 2019. Extending accurate time distribution and timeliness capabilities over the air to enable future wireless industrial automation systems. *Proc. IEEE* 107, 6 (2019), 1132–1152.

[5] Eduardo Cuervo. 2017. Beyond reality: Head-mounted displays for mobile systems researchers. *GetMobile: Mobile Computing and Communications* 21, 2 (2017), 9–15.

[6] Janos Farkas, Lucia Lo Bello, and Craig Gunther. 2018. Time-sensitive networking standards. *IEEE Communications Standards Magazine* 2, 2 (2018), 20–21.

[7] Tom Henderson. 2024. Wi-Fi module: Recent changes and future work. "https://www.nsnam.org/tutorials/consortium24/wns3-2024-wi-fi-update-final.pdf" [Last accessed on 13 January 2025].

[8] Muhammad Huzaifa, Rishi Desai, Samuel Grayson, Xutao Jiang, Ying Jing, Jae Lee, Fang Lu, Yihan Pang, Joseph Ravichandran, Finn Sinclair, et al. 2021. ILLIXR: Enabling end-to-end extended reality research. In *2021 IEEE International*

[9] Xiaoyu Ji, Yuan He, Jiliang Wang, Kaishun Wu, Ke Yi, and Yunhao Liu. 2013. Voice over the dins: improving wireless channel utilization with collision tolerance. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*. IEEE, 1–10.

[10] Qinjun Jiang, Muhammad Huzaifa, William Sentosa, Jeffrey Zhang, Steven Gao, Yihan Pang, Brighten Godfrey, and Sarita Adve. 2023. Offloading Visual-Inertial Odometry for Low Power Extended Reality. In *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 793–794. doi:10.1109/VRW58643.2023.00239

[11] Xiaolin Jiang et al. 2019. Low-Latency Networking: Where Latency Lurks and How to Tame It. *Proc. IEEE* 107, 2 (2019), 280–306. doi:10.1109/JPROC.2018.2863960

[12] Samuel Joseph, Rakesh Misra, and Sachin Katti. 2019. Towards Self-Driving Radios: Physical-Layer Control using Deep Reinforcement Learning. In *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications* (Santa Cruz, CA, USA) *(HotMobile '19)*. Association for Computing Machinery, New York, NY, USA, 69–74. doi:10.1145/3301293.3302374

[13] Jason Lawrence, Dan B Goldman, Supreeth Achar, Gregory Major Blascovich, Joseph G. Desloge, Tommy Fortes, Eric M. Gomez, Sascha Häberling, Hugues Hoppe, Andy Huibers, Claude Knaus, Brian Kuschak, Ricardo Martin-Brualla, Harris Nover, Andrew Ian Russell, Steven M. Seitz, and Kevin Tong. 2021. Project Starline: A high-fidelity telepresence system. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 40(6) (2021).

[14] Bin Li, Atilla Eryilmaz, and R Srikant. 2017. Emulating round-robin in wireless networks. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 1–10.

[15] Tong Li, Kai Zheng, Ke Xu, Rahul Arvind Jadhav, Tao Xiong, Keith Winstein, and Kun Tan. 2020. Tack: Improving wireless transport performance by taming acknowledgments. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*. 15–30.

[16] Zili Meng, Yaning Guo, Chen Sun, Bo Wang, Justine Sherry, Hongqiang Harry Liu, and Mingwei Xu. 2022. Achieving consistent low latency for wireless real-time communications with the shortest control loop. In *Proceedings of the ACM SIGCOMM 2022 Conference*. 193–206.

[17] Karl Montgomery, Mohamed Kashef Hany, and Richard Candell. 2024. Latency-Sensitive Networked Control Using 802.11ax OFDMA Triggering. In *2024 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. 1088–1095. doi:10.1109/AIM55361.2024.10637136

[18] Javier Perez-Ramirez, Oscar Seijo, and Iñaki Val. 2022. Time-Critical IoT Applications Enabled by Wi-Fi 6 and Beyond. *IEEE Internet of Things Magazine* 5, 3 (2022), 44–49.

[19] Matías Richart, Javier Baliosian, Joan Serrat, Juan-Luis Gorricho, and Ramón Agüero. 2020. Slicing with guaranteed quality of service in wifi networks. *IEEE Transactions on Network and Service Management* 17, 3 (2020), 1822–1837.

[20] Philipp S Schmidt, Theresa Enghardt, Ramin Khalili, and Anja Feldmann. 2013. Socket intents: Leveraging application awareness for multi-access connectivity. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. 295–300.

[21] William Sentosa, Balakrishnan Chandrasekaran, P Brighten Godfrey, Haitham Hassanieh, and Bruce Maggs. 2023. DChannel: Accelerating Mobile Applications With Parallel High-bandwidth and Low-latency Channels. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*.

[22] Similarweb. 2017. Top Websites Ranking. "https://www.similarweb.com/top-websites/" [Last accessed on 24 June 2024].

[23] Susruth Sudhakaran, Vincent Mageshkumar, Amit Baxi, and Dave Cavalcanti. 2021. Enabling QoS for collaborative robotics applications with wireless TSN. In *2021 IEEE international conference on communications workshops (ICC Workshops)*. IEEE, 1–6.

[24] Kaixin Sui, Mengyu Zhou, Dapeng Liu, Minghua Ma, Dan Pei, Youjian Zhao, Zimu Li, and Thomas Moscibroda. 2016. Characterizing and improving wifi latency in large-scale operational networks. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. 347–360.

[25] Talal Touseef, William Sentosa, Milind Kumar Vaddiraju, Debopam Bhattacherjee, Balakrishnan Chandrasekaran, Brighten Godfrey, and Shubham Tiwari. 2023. Boosting Application Performance using Heterogeneous Virtual Channels: Challenges and Opportunities. In *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks*. 139–146.

[26] Zhilong Zheng, Yunfei Ma, Yanmei Liu, Furong Yang, Zhenyu Li, Yuanbo Zhang, Jiuhai Zhang, Wei Shi, Wentao Chen, Ding Li, et al. 2021. Xlink: Qoe-driven multi-path quic transport in large-scale video services. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*. 418–432.

[27] Yuhan Zhou, Tingfeng Wang, Liying Wang, Nian Wen, Rui Han, Jing Wang, Chenglei Wu, Jiafeng Chen, Longwei Jiang, Shibo Wang, et al. 2024. {AUGUR}: Practical Mobile Multipath Transport Service for Low Tail Latency in {Real-Time} Streaming. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. 1901–1916.