Multi-Link Operation for Low-Latency Real-Time Applications

Marc Carrascosa Zamacois

UPF DOCTORAL THESIS / YEAR 2024

Thesis director
Boris Bellalta, Giovanni Geraci, Anders Jonsson
Department of Information and Communication Technologies

Universitat Pompeu Fabra
Barcelona

Abstract

Next generation applications such as Cloud gaming, Virtual Reality and Augmented Reality will have strict network requirements to operate, such as higher throughput and very low latency. As Wi-Fi continues to grow, it needs to adapt to these new technologies and requirements. In this thesis, we focus on the topic of latency reduction over Wi-Fi networks, focusing on one of the main features of the upcoming Wi-Fi 7: Multi-Link Operation (MLO). With MLO, a device can associate to an Access Point (AP) over multiple links at the same time. This device can then transmit over multiple channels simultaneously, multiplying its bandwidth. This feature however can also be used to improve network delay, as using independent backoffs in different links allows the device to adapt to the changing conditions of the links, choosing the least congested one opportunistically. We focus on this aspect of MLO, showing that this can lead to an order of magnitude delay reduction over the worst-case delay in traditional Single-Link Wi-Fi. Our analysis of MLO also uncovers a potential performance anomaly in its implementation which can result in worsened delay performance over current Wi-Fi, and we discuss how channel allocation schemes need to take MLO devices into consideration to avoid collapsing under the excess contention that they create. Finally, to showcase the utility of MLO as a powerful enabler of new technologies, we analyze the traffic of Cloud gaming and Virtual Reality applications, modelling their traffic and studying the use of MLO to deliver their traffic in a timely manner.

Resum

Les aplicacions de nova generació com el Cloud gaming, la realitat virtual i la realitat augmentada tindran estrictes requisits de xarxa per operar, com un major throughput i una latència molt baixa. A mesura que el Wi-Fi continua creixent, necessita adaptar-se a aquestes noves tecnologies i requisits. En aquesta tesi, ens centrem en la reducció de latència per a xarxes Wi-Fi, utilitzant una de les característiques principals del proper Wi-Fi 7: Multi-Link Operation (MLO). Amb MLO, un node es pot associar a un Punt d'Accés (AP) amb multiples links al mateix temps. El node pot aleshores transmetre simultàneament per multiples canals, multiplicant la seva amplada de banda. Aquesta característica pot ser utilitzada també per a millorar el retard de la xarxa, ja que utilitzar backoffs independents en cada link permet adaptar-se a les condicions dels links, escollint el menys congestionat de manera oportunista. Ens centrem en aquest aspecte de MLO, mostrant que aquesta característica pot resultar en una reducció del retard d'una ordre de magnitud respecte el retard en el pitjor cas del Wi-Fi tradicional amb un sol link. Amb el nostre anàlisi de MLO també descobrim una potencial anomalia en el rendiment de MLO que pot resultar en pitjors retards que utilitzant l'actual Wi-Fi d'un sol link. Discutim com els mètodes d'assignació de canals han de considerar si el node és MLO per tal d'evitar el col·lapse de la xarxa a causa de l'excés de contenció que aquests nodes creen. Finalment, amb l'intenció de mostrar el potencial de MLO per facilitar l'ús de noves tecnologies, analitzem el tràfic d'aplicacions cloud gaming i realitat virtual, modelant el seu tràfic i estudiant l'ús de MLO per l'enviament del seu tràfic de manera ràpida i efectiva.

Contents

List of Figures

Glossary

AP Access Point.

BSS Basic Service Set.

MLMR Multi-Link Multiple Radio.

MLO Multi-Link Operation.

MLSR Multi-Link Single Radio.

NSTR Non-Simultaneous Transmit and Receive.

OBSS Overlapping Basic Service Set.

SLO Single-Link Operation.

STA Station.

STR Simultaneous Transmit and Receive.

VR Virtual Reality.

List of publications

1. Carrascosa, M., & Bellalta, B. (2022). Cloud-gaming: Analysis of google stadia traffic. Computer Communications. **Best Paper Award 2023**.

2. Carrascosa, M., Geraci, G., Knightly, E., & Bellalta, B. (2022, May). An experimental study of latency for IEEE 802.11 be multi-link operation. In IEEE ICC.

3. Carrascosa-Zamacois, M., Geraci, G., Knightly, E., & Bellalta, B. (2023). Wi-Fi Multi-Link Operation: An Experimental Study of Latency and Throughput. IEEE/ACM Transactions on Networking.

4. Carrascosa-Zamacois, M., Galati-Giordano, L., Jonsson, A., Geraci, G., & Bellalta, B. (2023, March). Performance and coexistence evaluation of IEEE 802.11 be multi-link operation. In IEEE WCNC.

5. Carrascosa-Zamacois, M., Geraci, G., Galati-Giordano, L., Jonsson, A., & Bellalta, B. (2023, September). Understanding multi-link operation in Wi-Fi 7: Performance, anomalies, and solutions. In IEEE PIMRC.

6. Carrascosa-Zamacois, M., Galati-Giordano, L., Wilhelmi, F., Fontanesi, G., Jonsson, A., Geraci, G., & Bellalta, B. (2024). Performance Evaluation of MLO for XR Streaming: Can Wi-Fi 7 Meet the Expectations?. In IEEE CAMAD.

List of publication collaborations

1. Adame, T., Carrascosa, M., & Bellalta, B. (2019, April). The TMB path loss model for 5 GHz indoor WiFi scenarios: On the empirical relationship between RSSI, MCS, and spatial streams. In IEEE Wireless Days.

2. Adame, T., Carrascosa-Zamacois, M., & Bellalta, B. (2021). Time-sensitive networking in IEEE 802.11 be: On the way to low-latency WiFi 7. Sensors, 21(15), 4954.

3. Wilhelmi, F., Carrascosa, M., Cano, C., Jonsson, A., Ram, V., & Bellalta, B. (2021). Usage of network simulators in machine-learning-assisted 5G/6G networks. IEEE Wireless Communications, 28(1), 160-166.

4. Adame, T., Carrascosa, M., Bellalta, B., Pretel, I., & Etxebarria, I. (2021). Channel load aware AP/Extender selection in Home WiFi networks using IEEE 802.11 k/v. IEEE Access, 9, 30095-30112.

5. Bellalta, B., Carrascosa, M., Galati-Giordano, L., & Geraci, G. (2023). Delay Analysis of IEEE 802.11 be multi-link operation under finite load. IEEE Wireless Communications Letters, 12(4), 595-599.

6. Galati-Giordano, L., Geraci, G., Carrascosa, M., & Bellalta, B. (2024). What will Wi-Fi 8 be? A primer on IEEE 802.11 bn ultra high reliability. IEEE Communications Magazine, 62(8), 126-132.

7. Casasnovas, M., Michaelides, C., Carrascosa-Zamacois, M., & Bellalta, B. (2024). Experimental Evaluation of Interactive Edge/Cloud Virtual Reality Gaming over Wi-Fi using Unity Render Streaming. arXiv preprint arXiv:2402.00540.

8. Shaabanzadeh, S. S., Carrascosa-Zamacois, M., Sánchez-González, J., Michaelides, C., & Bellalta, B. (2024). Virtual reality traffic prioritization for Wi-Fi quality of service improvement using machine learning classification techniques. Journal of Network and Computer Applications, 230, 103939.

Funding Sources

The work done in this thesis has been partially funded by:

- Doctoral Scholarship PRE2019-088690 from Ministerio de Ciencia e Innovación, linked to the Maria de Maeztu Excellence Grant MDM-2015-0502-19-2.
- Machine Learning for Wireless Networking in Highly Dynamic Scenarios (WINDMAL) PGC2018-099959-B-I00 (MCIU/AEI/FEDER,UE).
- Mixed Augmented and Extended Reality Media Pipeline (MAX-R) HEu-CL4-MAX-R-101070072.
- Low-latency Wireless Networking for interactive eXtended Reality applications (Wi-XR) PID2021-123995NB-I00 (MCIU/AEI/FEDER,UE).

Chapter 1

INTRODUCTION

The internet continues to evolve, and new applications reach the market every day. Extended Reality (XR) applications, which include Virtual Reality (VR) and Augmented Reality (AR), are growing in popularity as they unlock novel use cases across many domains, such as healthcare, industry, education and gaming. The market for online gaming alone currently has more than 1.1 billion users, and is projected to go from 26.14 billion dollars in 2023 to 32.45 million in 2027, with VR forecasting a revenue of $2.5 billion [1]. Other online gaming applications surged in popularity during the pandemic, with cloud-gaming applications like GeForce Now providing service to over 20 million users [2]. These new applications are planned for indoor use, and Wi-Fi is expected to become the main technology to support them [3], with most headsets including high-grade Wi-Fi capabilities[1], and services like *Steam Link*[2] allowing to stream games wirelessly from computer to headset. Wi-Fi is the leading wireless technology, delivering more than 80% of all wireless data traffic [4]. There are 18 billion Wi-Fi devices worldwide, and Wi-Fi shipments will increase to four billion annually by 2024, and its global economic

[1] `https://www.meta.com/help/quest/articles/headsets-and-accessories/oculus-link/connect-with-air-link/`

[2] `https://store.steampowered.com/app/353380/Steam_Link/`

value of $3.3 trillion in 2021 is expected to grow to $4.9 trillion by 2025 [5]. As the aforementioned new types of application have strict latency requirements, Wi-Fi will have to consider the delivery of time-sensitive traffic.

Historically, Wi-Fi has struggled to attain delay guarantees due to its use of the Distributed Coordination Function (DCF) model for the MAC layer channel access. It uses Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA), by which network nodes that wish to transmit need to sense the channel to ascertain if it is available. If the channel is sensed busy, the node defers its transmission until the channel is idle again and backoff resumes, otherwise, once backoff reaches zero, the node can transmit and other devices will defer their transmission accordingly. As more nodes appear in the network, the contention and consequently the delay increases, thus keeping a consistent low delay can be difficult. To mitigate this issue, recent amendments added features that can be used to improve delay. New features include Target Wake Time (TWT) in IEEE 802.11ax, which can be used for a more deterministic channel access to avoid contention [6]. A modification called Restricted Target Wake Time (rTWT) was added to IEEE 802.11be [7], allowing Access Points (AP) to reserve resources for latency sensitive traffic. IEEE 802.11bn also includes as an objective a mode of operation capable of reducing latency by 25% in the 95th percentile [8] over 802.11be. One of the main features of IEEE 802.11be, Multi-Link Operation, allows a device to associate to an AP through multiple channels (links) at once, thus running multiple backoffs at the same time and allowing simultaneous transmission of packets. This feature also allows the device to transmit opportunistically through the first available link, thus, if there is congestion in a link, another one can still be used and service can continue without issue, avoiding contention based delay. This potential to reduce network latency is the main focus of this thesis.

In this work, we focus on the need for bounded latency in emergent applications and the tools that Wi-Fi can use to achieve reduced delays, namely MLO. Using real application traces, we model latency-sensitive traffic and simulate MLO features following the standardization efforts,

showing the suitability of MLO as a tool to reduce network latency and enable these new types of communication. The main contributions of this thesis are:

- The study of emerging interactive video streaming applications, their throughput, latency requirements, and their traffic shaping. For cloud-gaming, a model is provided to replicate its traffic for different settings.
- We studied the performance of Multi-Link Operation channel access, considering several implementations and their potential benefits for delay reduction, finding up to an order of magnitude improvement over Single-Link Operation.
- The discovery and definition of the MLO delay anomaly, by which MLO nodes can starve other nodes, leading to worse-than-legacy performance. We show that channel allocation needs to be reworked to consider MLO devices and the extra contention that they bring to the network.
- We explore the suitability of MLO to enable interactive Virtual Reality streaming applications and its low latency requirements, showing the potential of MLO to enable multiple simultaneous users in the same BSS (Basic Service Set) by spreading the same bandwidth over a higher number of narrow channels.

The rest of this document is structured as follows: Chapter 2 details the technologies used and the related literature, Chapter 3 presents the main findings of the thesis: Section 3.1 focuses on cloud-gaming and its traffic analysis, Section 3.2 showcases our results of MLO using the football stadium dataset. Section 3.3 focuses on the study of MLO coexistence. Section 3.4 studies the use of MLO to deliver VR traffic, and finally, Chapter 4 concludes the thesis. For the remainder of this thesis, we refer to each publication by the number specified in the list of publications from page XIII (for example: publication #1).

Chapter 2

ENABLING TECHNOLOGIES

In this chapter we introduce the technologies and environments used and discussed throughout the thesis, such as the testbeds used to study streaming applications, the traffic captured, and the specific Wi-Fi features we studied.

2.1 Testbeds

For both regular gaming and VR gaming, the video quality (resolution and frame rate) depend on the hardware used. Both types of applications require expensive hardware to achieve good performance, and in both cloud gaming and VR streaming, the computational complexity can be offloaded to a remote server, thus reducing the cost of the client used. This was one of the key selling points of Google Stadia, a cloud gaming service in which users could play the latest video games remotely with any device that could access the newest version of their Chrome web browser, while the games were rendered on Google servers. For VR, the Head Mounted Display (HMD) comes with hardware strong enough to support some games, but to achieve the highest quality, it is required to connect to a strong PC to render the video and audio. This is the same offloading idea as cloud-gaming, and HMDs today already come packaged with a Wi-Fi interface to enable it.

Two testbeds were deployed to capture the behavior of the applications considered in this thesis, following the architecture shown in Figure 2.1. For Google Stadia, a fully wired setup was used, with an Ethernet connection between the AP and client. Google servers were delivering the content over the internet to our AP, to which we connected a laptop using Wireshark to capture the traffic received and transmitted. For VR streaming, our own server was used to render the games, and ALVR [9] was used to encode the video and transmit it over Wi-Fi to the VR headset. In this case, as we had physical access to the server, we used Wireshark to capture traffic on both server and client. The modelling of VR traffic then could use the server data to replicate the traffic before any interference or shaping effects from the network. Further details on the testbed and data obtained can be found in papers #1 and #6.

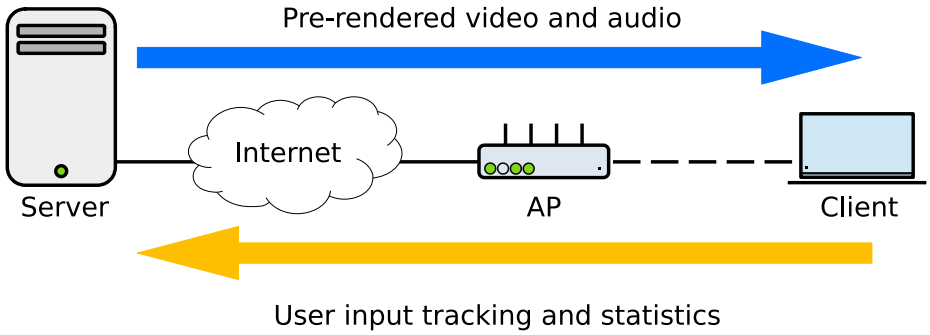


Figure 2.1: Testbed diagram

2.2 Cloud gaming traffic and VR traffic

Cloud gaming has existed for a long time, with one of the first systems being presented in the year 2000 [10]. Other systems like OnLive and Gaikai launched in 2010 and 2011 respectively, and were bought by Sony years later to integrate onto their own system [11]. Cloud gaming saw

a rise in popularity in 2019 with the launch of Google Stadia and Nvidia GeForce Now in early 2020, and while Stadia was shut down in 2023 [12], GeForce Now remains with 20 million users as of 2022 [2]. Other cloud gaming services like Xbox Cloud Gaming [13] and Amazon Luna [14] are still active as well.

These recent platforms have received a lot of attention. The traffic of Stadia, Geforce Now and PSPlus is studied in [15], in which the protocols used by each service are identified, as well as their respective bitrates according to the resolution used, which can go up to 45 Mbps. The performance of Stadia over cellular networks is also analyzed, showing that 3G networks can only deliver 720p streams, while 4G can deliver a better performance over 1080p and even 4K. A testbed is created in [16] to study the performance of a cloud gaming system over Ethernet, WiFi and LTE, identifying the most relevant key performance indicators for assessing quality of experience, and showing the suitability of wireless to enable cloud gaming systems. A performance evaluation of Xbox cloud gaming and GeForce Now can be found in [17], testing both platforms with limited bandwidths and latency, showcasing how each system adapts to the network conditions. Performance is sustainable with up to 100 ms of delay, but video quality and frame rate adjustments can be observed starting at 30 ms. Similarly, the work in [18] analyses the performance of four cloud gaming services under different network conditions, highlighting that each platform has its own application layer adaptation mechanism, and that as a result of being application-level, they all take too much time to react to changes in the network, while lower layer mechanisms could react faster and provide better service. Another testbed is used in [19] to create a dataset containing the traffic of multiple cloud gaming platforms, as well as other streaming applications to then perform traffic classification and identification of cloud gaming traffic so as to deliver it to priority queues. They find that decision trees can allow for up to 98.5% accuracy in identifying cloud gaming traffic even with degraded network conditions.

Virtual Reality (VR) is another emergent application that saw a popularity increase in a similar time period. A crowd-funding campaign was

launched in 2012 for the Oculus Rift, which earned 2.4 million dollars. Facebook then bought the Oculus shortly after, and in the same year Sony announced their first VR headset [20]. Nowadays, there are multiple companies offering VR headsets for gaming, and the market is expected to grow from 12 billion dollars in 2022 to 22 billion dollars in 2025 as the devices get smaller and more practical to use in other sectors such as healthcare, education and manufacturing [21].

Much like cloud gaming, VR streaming requires high throughput and low latency, and whether wireless networks can or cannot handle such traffic has been the focus of recent VR research. In [22], the performance of VR over Wi-Fi is studied using connections at different distances, showing that VR requires stable and high throughput connections to accommodate its high frame rate, as weak Wi-Fi sigals struggle to maintain steady performance. The study in [23] tests the user experience and cybersickness for both wired and wireless VR setups, finding the wireless performance to be similar to using a wired connection so long as there was direct line of sight with the AP. The traffic characteristics of VR are studied in [24] with different video encoders, showing that older encoders can struggle to process the video data in a timely manner, that bitrate can be modified according to game demands, and finally commenting that MAC layer scheduling could be necessary to safely deliver VR traffic. In order to prioritize VR traffic over non-VR traffic, Machine Learning models are used in [25] to classify VR traffic, showcasing the correlation between downlink and uplink as a major defining feature. Simulations are then used to test how this classification can be used to deliver VR traffic over Wi-Fi with delays 4.2 times lower than without prioritizing it.

In this thesis, we study both cloud gaming and VR traffic delay, as well as ways to enable these applications over future networks using MLO.To the best of our knowledge, ours was the first paper to do a deep dive on the protocols used by Google Stadia, as well as testing its performance under bandwidth and delay constraints. Further, we also produced a model capable of replicating its traffic. Our work in VR focuses on its coexistence with new Wi-Fi features like MLO and their capacity to improve worst-case VR delay. Publications $\#1$ and $\#6$ focus on cloud gaming and VR

traffic respectively, and publication #3 uses the model proposed in #1 to test cloud gaming traffic over MLO.

2.3 Wi-Fi 7 and MLO

Maintaining consistent low delay on Wi-Fi networks has become a popular topic of study in recent years. A lot of early discussion on Wi-Fi 7 revolved around real time applications and the need for low, bounded latency [26]. While Wi-Fi 7 has many features, the use of MLO has been at the forefront of this discussion, some initial Wi-Fi 7 studies discussed to separate the control and data plane between links, or to split low and high priority traffic in different links [27], other proposals aimed to transmit the same packet through multiple channels to avoid the delays related to packet loss [28]. As the draft evolved, MLO performance received a lot of attention, especially to support real-time applications. The work in [29] gives an overview on different MLO implementations, showing their respective throughput based on increased external interference. They show that the single-link and non-simultaneous transmit and receive variants can underperform compared to multi-link variants under low loads, but achieve good performance when under heavy interference. In [30], MLO with independent links is analyzed, showing that an order of magnitude reduction can be obtained over the Single-Link 90th percentile delay and that it is able to keep the delay below the 10 ms necessary for low-latency applications. Similarly, in [31] MLO is compared to SLO for an increasing number of users while keeping the same total traffic load, showing that MLO benefits increase with the user count, with up to 8x lower delays than SLO for 10 users. In [32], traffic allocation policies for MLO are considered, such as distributing traffic flows equally among interfaces, or distributing based on the congestion at each link. They find that under external interference, assigning traffic to the single least-loaded link results in better results, as it reduces the overall contention in the network. The work in [33] implements similar traffic allocation policies for MLO to split the number of MPDUs delivered to each link. Their findings show

that unrestricted MLO functions better than any dynamic policy, as the random nature of IEEE 802.11 channel access tends to prevent the simultaneous use of multiple links, thus packets end up spending more time in the queue with the dynamic policies.

While the previously mentioned papers analyse MLO performance, our work is unique in the use of a full 5 GHz dataset to understand the effect of using different channel occupancies over each link. We also discovered and presented the MLO delay anomaly, which has an impact on the delay of crowded MLO scenarios. Publication $\#4$ focused on the coexistence of MLO nodes with both legacy nodes and MLO nodes, and publication $\#5$ centered around our discovery of the MLO anomaly, an issue found when multiple MLO nodes contend for the same channels, which leads to starvation and lower delays than legacy Single-Link networks. Finally, publication $\#6$ studies the use of MLO to enable VR traffic and its stringent latency requirements.

2.4 MLO channel access

Four channel access methods are discussed in this thesis. The first is **Single-Link Operation (SLO)**, in which a single radio interface is used to connect between an AP and a STA. Once there are packets in the buffer, a backoff counter is started at the transmitter, and transmission begins when backoff ends. Figure 2.2 shows the sequential transmission of three packets using SLO, where it can be observed that packets 2 and 3 arrive at the buffer while the link is busy, and have to wait for their transmission.

Next is **Multi-Link Multiple Radio Simultaneous Transmit and Receive (MLMR STR)**. Two (or more) links are used, each with their own backoff counter, and by using orthogonal channels, links can transmit opportunistically, independently of what the other link is doing. Figure 2.3 shows MLMR STR operation, in which packet 2 arrives at the buffer while packet 1 is transmitting, thus starting a backoff counter in link 2. Backoff ends and packets 1 and 2 can transmit at the same time, greatly reducing delay $\#1$ in comparison to Figure 2.2. Once packet 3

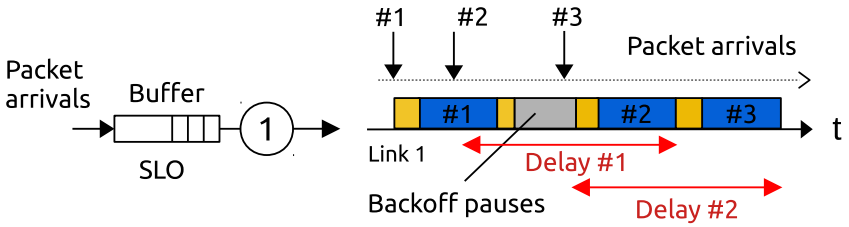


Figure 2.2: Illustration of Single-Link Operation.

arrives, link 1 is busy, so the backoff counter is started in link 2, and once the first link is free, a backoff is started there as well. Link 2 ultimately ends its backoff first, and packet 3 is transmitted with a shortened delay #2. Over the course of the thesis we referred to MLMR STR with interchangeable terms based on context (i.e., standardization process and methods considered in each paper). The terms MLMR STR, MLMR, STR, and MLO are used to all refer to the same MLMR STR mechanism. Publications #2 and #3 use STR, publications #4 and #5 use MLMR, and publication #6 uses MLO.

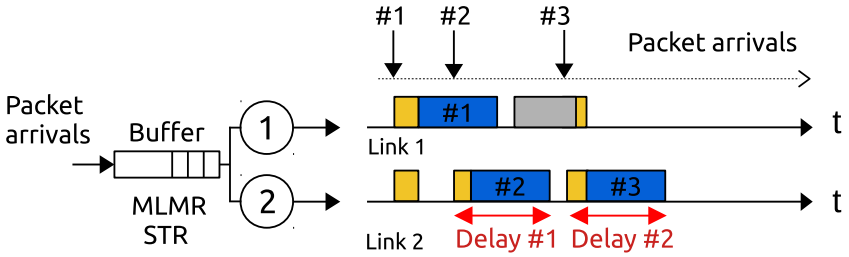


Figure 2.3: Illustration of MLMR STR operation.

Another MLO implementation is **Multi-Link Single Radio (MLSR)**. With MLSR, multiple links are used with independent backoffs, much like with MLMR STR, but the main difference is that only one link is used for transmission. This allows the device to benefit from the opportunistic nature of MLO, selecting whichever link is free, but avoids adding

extra contention on multiple links during transmission, as well as any potential self-interference between the radio interfaces. Figure 2.4 shows the MLSR implementation, in which packet packet 2 can be transmitted through link 2, avoiding the busy period found in link 1, which SLO could not avoid. This reduces delay $\#1$ in comparison to SLO. Similarly, once packet 2 is fully transmitted, backoffs are started in both links, and link 1 wins the contention, transmitting packet 3 and reducing delay $\#2$ over SLO as well.

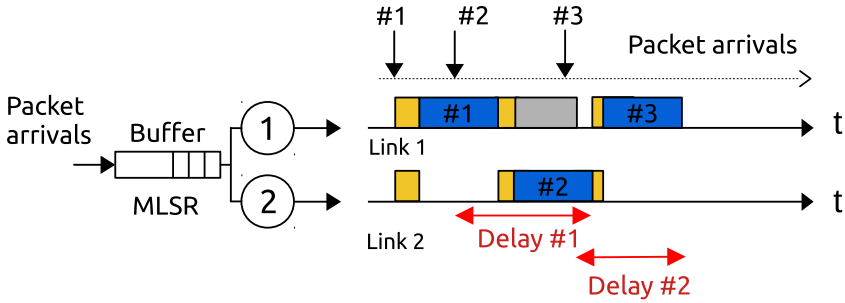


Figure 2.4: Illustration of MLSR operation.

Finally, we consider **Multi-Link Multiple Radio Non-Simultaneous Transmit and Receive (MLMR NSTR)**. This is an earlier implementation of what NSTR would become in the IEEE 802.11be draft, designed to avoid interference between radio interfaces. One link is designated as the primary and all others as secondary. The primary link runs a backoff, and when it ends, if any other link has been idle for the duration of a PIFS, it can also be used for simultaneous transmission, otherwise, transmission starts only over the primary link, and the other link remains idle. Figure 2.5 shows the NSTR operation. For the first transmission, as there is only one packet in the buffer, it is fully transmitted over the first link. Then, once packet 2 arrives, the link is busy and backoff is halted, and by the time the link is free packet 3 has arrived at he buffer. Then, backoff is performed over the first link, and since the second link remains idle, one packet can be transmitted through both links, thus delay $\#1$ remains the same as for SLO, but delay $\#2$ has been reduced significantly.

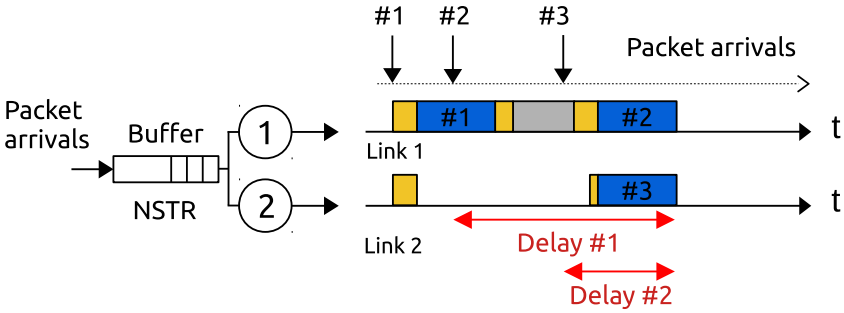


Figure 2.5: Illustration of NSTR operation.

Publication #2 and #3 discuss the use of MLMR STR and NSTR, publications #4 and #5 focus on MLMR STR and MLSR, and publication #6 discusses configurations for MLMR STR.

Chapter 3

ON THE INTERPLAY BETWEEN VR STREAMING AND MLO: MAIN FINDINGS

3.1 Stadia: Understanding cloud-gaming

Cloud-gaming requires high throughput and low latency, which can be a challenge for wired networks, and even more so for wireless ones. In this chapter we focus on the Google Stadia platform, which allowed users to stream games directly from a server to their PC, without requiring high performance hardware. Stadia transmitted high definition video at a frame rate of 60 frames per second (FPS), and it allowed the user to select different profiles based on their network capacity, modifying the resolution of the video received: 720p required 10 Mbps, 1080p required 28 Mbps, and 4K required 35 Mbps or more.

To analyze the traffic of Stadia, we used Wireshark to capture several traces over a wired connection. Our initial analysis considered 10 games, and we selected 3 that were most representative of the collective patterns. The 3 chosen games were of different genres and presentation styles, which had an impact on the throughput perceived. These games were Tomb Raider, Thumper and Spitlings, a third person action game,

an on-rails rhythm game and 2D platformer.

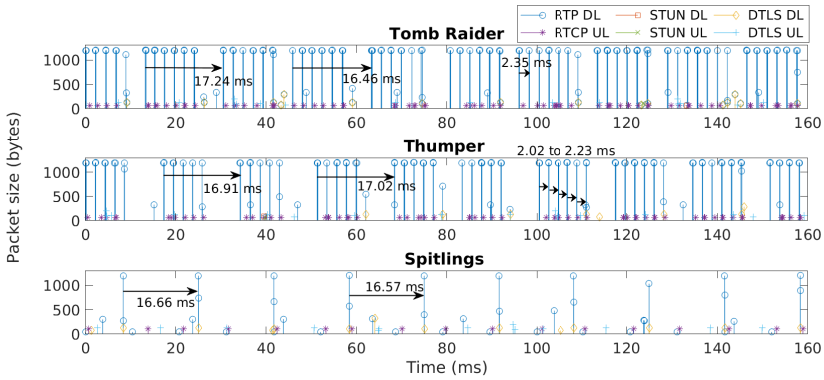
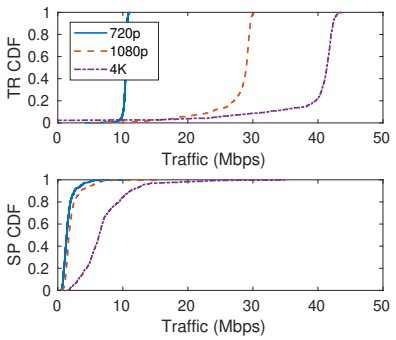


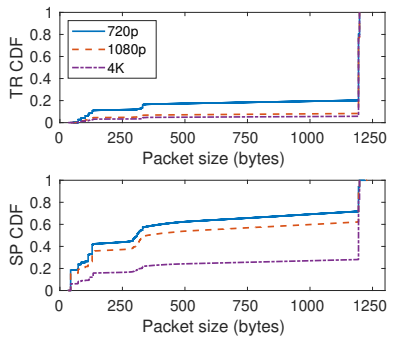
Figure 3.1: Temporal evolution of Stadia traffic for Tomb Raider, Thumber and Spitlings.

Figure 3.1 shows the traffic patterns for each game considered, separated by protocol. Most of the traffic (over 90% in all cases) is concentrated in the downlink (DL), with Real Time Protocol (RTP) packets representing more than 99% of the DL traffic. RTP packets encapsulate the video and audio, and they are transmitted in batches spaced by the frame rate of 60 FPS, which results in an inter-arrival time of $\frac{1}{60} = 16.6$ ms on average. This pattern is consistent on all games. Each batch consists of several smaller groups of packets, which are spaced by ≈ 2 ms. The amount of groups of packets per batch is different across each game, which may be a consequence of large video frames being created at the source and needing to be split into smaller packets. As Tomb Raider is the most graphically complex game, it has the most groups of packets per batch, while Spitlings has the lowest, being a simple 2D game with lower graphical requirements. On the uplink (UL), Real Time Control Protocol (RTCP) packets are found after each group of DL RTP packets. RTCP is designed to send back statistics on packet reception and performance on the client, to help adjust settings at the source. These represent less than 0.5 Mbps over all games. STUN and DTLS packets represent an even lower portion of the throughput, as they take care of stream synchroniza-

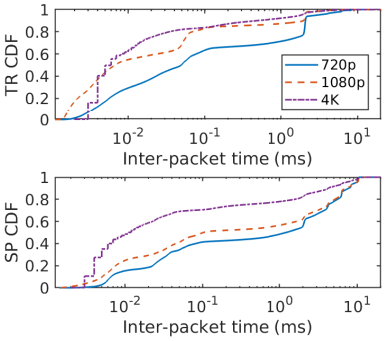
tion and security, we infer that DTLS also encapsulates the user inputs to transmit them back to the server.



(a) Traffic load



(b) Packet size



(c) Inter-arrival time

Figure 3.2: Impact of the resolution on metrics.

Next, we analyzed the evolution of the traffic load, packet size and inter-arrival time as we increased the video resolution. For this specific experiment, each game was captured 3 times (one per resolution), running the same section of 10 minutes. Figure 3.2a shows the CDF of the traffic load required for Tomb Raider and Spitlings. For Tomb Raider, it can be observed that both 720p and 1080p match closely the requirements stated by the app of 10 and 28 Mbps respectively. However, for 4K we observed higher loads than the recommended 35 Mbps for 88% of the time, going

up to 43.74 Mbps. Spitlings on the other hand shows a negligible increase in the traffic load going from 720p to 1080p. Increasing further to 4K shows does increase the traffic load significantly, but it is still well below the requirements stated by the app. The traffic load then appears to be application specific, with more complex games requiring more resources. This has a direct impact on the number of packets per batch, which can be observed in Figure 3.2b, with lower resolutions having less packets over 1000 bytes, and also in Figure 3.2c, where the inter-arrival time is higher when the resolution is lower, i.e., the time between batch starts (or video frames) is always 16.6 ms, but when the traffic load is high, the amount of extra groups of packets reduces the inter-arrival between the last group of packets of a video frame and the start of the next.

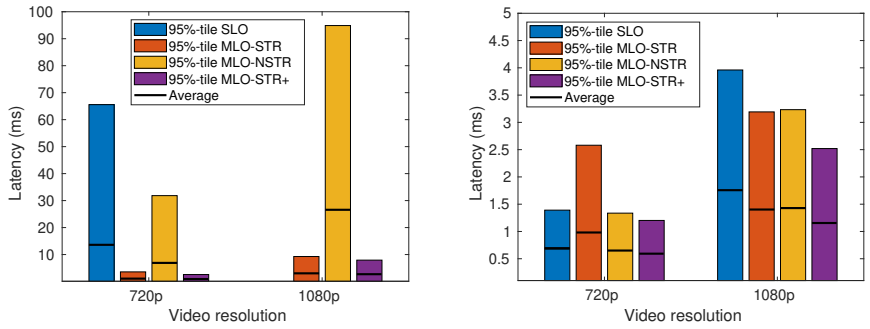
The main challenge of handling cloud-streaming traffic stems from the video streaming pattern, as a lot of packets are transmitted in bursts, thus filling the buffer quickly and increasing the queueing time. As buffering is not an option for these kind of applications, fast delivery of large amounts of data is required to ensure a good end-user experience. Publication #1 contains a detailed and in-depth study of the Stadia traffic, comparing the impact that the chosen videogame has on the patterns, as well as a comparison to other live streaming applications. The impact of different network conditions such as extra delay and change in the bandwidth is studied, and a model to replicate Stadia traffic is also provided.

3.2 MLO latency under real spectrum measurements

In order to study the real-world capabilities of Wi-Fi 7 and MLO under external (OBSS) interference, we used traces from the WACA dataset [34, 35], which contains over-the-air traces of the occupancy (in RSSI) for every channel in the 5 GHz band. The extensive measurement campaign covers a variety of locations, such as an apartment building, a shopping center, and a football stadium. We focused our study on the stadium scenario, as it contains a wide range of channel occupancies. These traces

were used as an input for our own discrete-event simulator based on C++, which replicates Wi-Fi behavior and MLO features (see [36] for more details).

We considered a main MLO-BSS, in which the AP is transmitting in the downlink to a single STA. A Fixed MCS of 256-QAM 5/6 was used, and as the dataset contains the average RSSI perceived over time in $10\mu s$ intervals, we adapted our MLO simulator to round up the IEEE 802.11 backoff slots to this same $10\mu s$. All contention perceived by the main AP comes from the dataset, which contains the OBSS channel occupancy data. Two MLO modes were considered for the main AP: STR and NSTR. For STR, two different options were also considered: STR, in which packets are assigned to an interface prior to backoff, and STR+, where assignment is delayed until the last possible moment (i.e., backoff is finished). Further, we used the Stadia traces from the previous chapter to generate realistic cloud-gaming traffic for 720p and 1080p resolution.



(a) Primary 40% and secondary 40% occupancy

(b) Primary 10% and secondary 70% occupancy

Figure 3.3: Delay vs. video resolution using Google Stadia traffic for (a) symmetric and (b) asymmetric link occupancy.

Figure 3.3a shows the average and the worst-case (95th percentile) delay obtained for both resolutions using channels of equivalent occupancy of 40%. Delay-sensitive applications such as Stadia require a worst-case delay below 10 ms for good performance, and it can be observed that SLO

cannot achieve this result with the lowest resolution of 720p, showing a worst-case delay of over 60 ms. MLO-STR shows an order of magnitude reduction for the 95th percentile compared to SLO, keeping the delay below 10 ms, and MLO-STR+ reduces this delay even further. A resolution of 1080p can only be supported by MLO-STR as well, and while MLO-NSTR offers an improvement over SLO, it underperforms compared to STR.

Figure 3.3b shows the same results but with an asymmetrical configuration of links, with a primary of 10% occupancy and a busy secondary of 70% occupancy. In this case, as SLO uses the primary link, we observe a much lower delay for all cases, with every method achieving lower than 10 ms. However, MLO-STR here struggles to use its multiple links efficiently, and by assigning packets to the overloaded link, it increases SLO delay by 85.5%, performing worse than MLO-NSTR, and showing the need to delay the packet assignment until later, as MLO-STR+ achieves the lowest delay among all options.

To further understand the gains of MLO, we tested the performance of using SLO with comparable bandwidth to two MLO links. We also allowed dynamic channel bonding with preamble puncturing to improve the performance of 40 MHz and 80 MHz channels, using the lowest 20 MHz channel as the primary channel. We used Poisson traffic, normalized based on the maximum SLO capacity at 20 MHz so as to ensure that all methods were in a non-saturating regime. A primary and secondary links of 40% and 70% occupancy were used respectively.

Figure 3.4 shows the latency achieved by SLO and STR+ as we increase the traffic load. In all cases, increasing the bandwidth used leads to better delays, and we can observe that if we use 40 MHz channels for SLO we can obtain better delay than with 20 MHz channels in MLO-STR+. This is a direct result of the dynamic channel bonding selecting whichever primary channel is less occupied. Indeed, for MLO-STR+ with two 20 MHz links, there is no primary channel selection, as they are assigned statically. However, once MLO-STR+ uses wider channels as well (2x40 MHz), it shows higher delay reduction than using a single link of 80 MHz (9.5x lower delays with MLO-STR+ vs 6.1 for SLO).

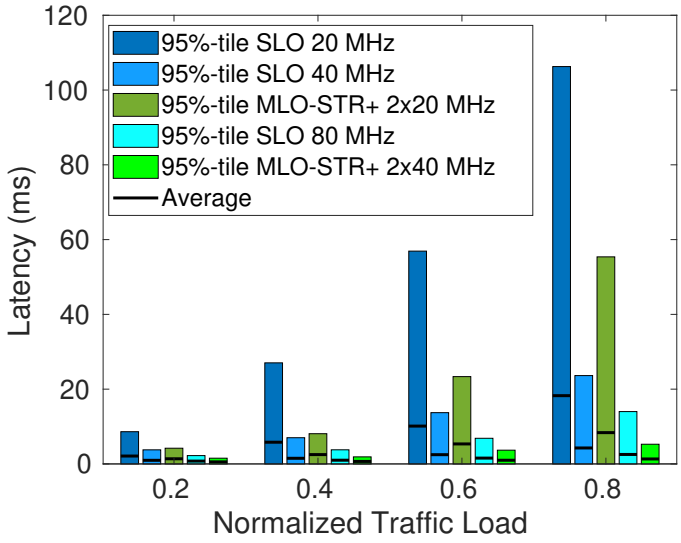


Figure 3.4: Latency for occupancy of $\{40\%, 70\%\}$ with dynamic primary channel selection

MLO-STR+ can be a powerful enabler for next-generation applications, not only increasing the bandwidth used but also reducing the delay by up to an order of magnitude compared to SLO. MLO-STR+ greatly benefits from the ability to use links opportunistically, operating multiple backoffs simultaneously, and transmitting over the earliest available channel. Nevertheless, channel selection is an important consideration in extracting better performance, as a poor selection of links could lead to worse results than SLO with equivalent bandwidth. Dynamic channel allocation and preamble puncturing can help MLO-STR+ reach its full potential. Publications $\#2$ and $\#3$ delve deeper into the STR and NSTR implementation, showing the throughput gains as well as the delay improvement when using links with symmetrical and asymmetrical occupancy.

3.3 MLO contention anomaly and coexistence

To continue analyzing MLO performance as a tool to improve network latency, we extended the MLO simulator to support multiple nodes so that we can generate different scenarios with multiple MLO and SLO users, allowing us to study the interplay between contending devices using similar or different channel access modes.

First, we studied MLO coexistence as the number of links increases. We considered two BSS with one STA each, using an MCS of 256-QAM 3/4 and 2 spatial streams. We used the performance of isolated SLO as a baseline, as two SLO BSS could use orthogonal channels, while MLO BSS would have to share them.

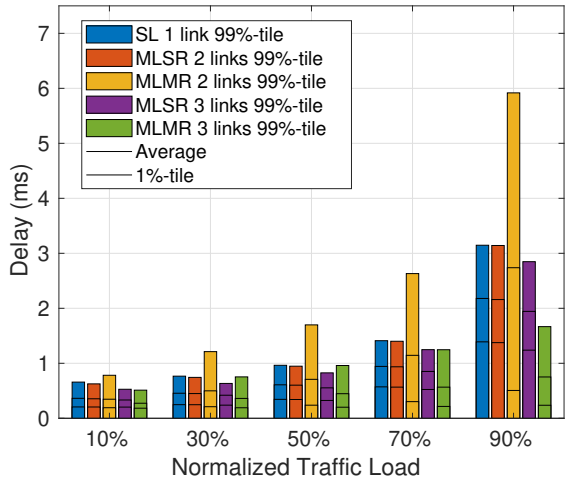


Figure 3.5: Scenario II: Delay performance for each transmission method as traffic load increases. Traffic load values refer to the fraction of the SL full-buffer throughput.

Figure 3.5 shows the best-case delay (1st percentile), average and worst-case delay (99th percentile) for each setup as the traffic load increases. The traffic load is normalized based on the maximum achievable by SLO. MLSR with two links results in a slight delay reduction over

SLO over all traffic loads, and adding a third link further reduces the delay. MLMR however, shows worse 99th percentile delay in all situations for two links, while also having the lowest 1st percentile delay. This effect is exacerbated as the traffic load increases. Adding a third link reduces the delay, but this could simply be a result of the third link significantly increasing the available bandwidth, thus a higher traffic load would likely result in similar results to the two link configuration. We refer to this as the MLO anomaly.

The MLO anomaly is fully described in Figure 3.6. We consider two MLMR APs transmitting to a single STA each. For the first portion without contention, AP 1 receives packet 1, starts its backoff and obtains a transmission opportunity in link 1. As packet 2 has also arrived by this point, packets 1 and 2 are aggregated and transmitted through link 1. Then packet 3 arrives, and since link 1 is busy, AP 1 performs backoff on link 2, transmitting packet 3 at the same time as packets 1 and 2, and reducing its delay compared to SLO transmission. Next we consider the case with contention, where AP 1 starts transmission of packets 4 and 5 over link 1, and then both AP 1 and 2 receive a new packet (packet 6 and 1 respectively). They then both perform backoff on link 2, and due to the random nature of the backoff, AP 1 wins contention, transmitting packets 6 and 7 through link 2, while AP 2 finds all its links busy, forcing it to defer transmission until AP 1 finishes on link 1. At this point, the buffer of AP 2 has multiple packets waiting, and it needs to aggregate more packets per transmission, leading to longer transmission times (i.e., blocking the channel for longer periods of time). The MLO anomaly can be found in this instance, where one MLO device takes all available links at once, starving the other devices. Note that while AP 1 benefited in this instance (thus achieving a really low 1st percentile delay), the anomaly will happen in reverse at some point, and AP 2 will starve AP 1 as well, leading to the higher 99th percentile seen previously.

Next, we analyze the interplay between SLO and MLO nodes. We set up a toy scenario with 3 BSS in a line: the middle AP is using MLO, while the outer APs use SLO, each of them using one of the same channels used by the MLO BSS. We first analyze the throughput capacity of each

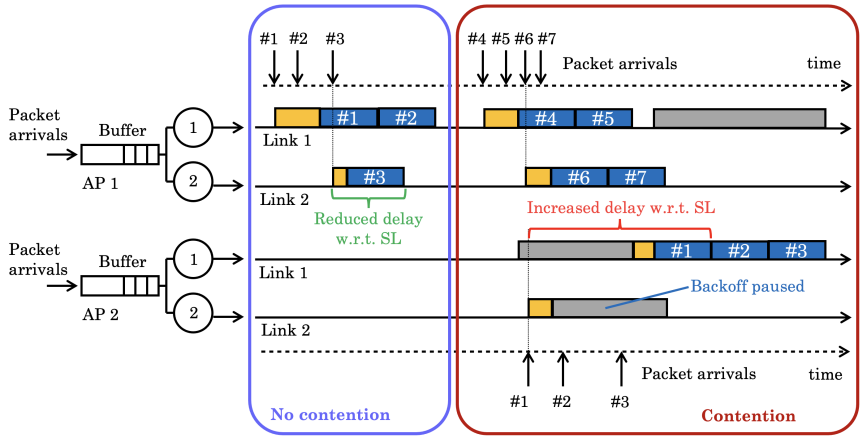


Figure 3.6: Illustration of MLMR operations and packet interactions over two links without (left) and with (right) contention. Grey, orange, and blue slots denote occupied channels, ongoing backoffs, and successful transmissions, respectively. Consecutive blue slots indicate aggregated packets. For illustration purposes, all transmissions are downlink and the corresponding ACKs are omitted. In the example, for AP 1, packet #3 experiences a lower delay than it would under SL operations. For AP 2 instead, packet #1 undergoes a higher delay than it would with SL.

method, which we show in Figure 3.7. MLSR achieves similar throughput to SLO nodes, as it can only fully transmit in one channel, thus its main advantage over SLO is limited to the faster channel access as a consequence of using two backoffs. MLMR however obtains double the SLO throughput, but to do so it takes away transmission opportunities from the other APs, thus limiting their maximum achievable throughput. To analyze the delay, we set the MLO node to have a traffic load of 70%, and then modify the traffic load at the outer APs in different symmetric and asymmetric configurations. Figure 3.8a shows the delay for the MLSR configuration, and Figure 3.8b shows the MLMR configuration. Comparing both, we can observe that MLMR always achieves lower delays than MLSR, and that for the first three configurations, MLMR leads the SLO

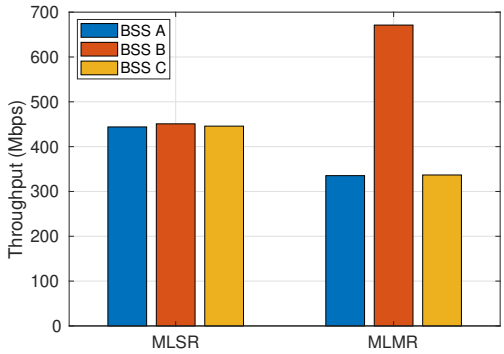
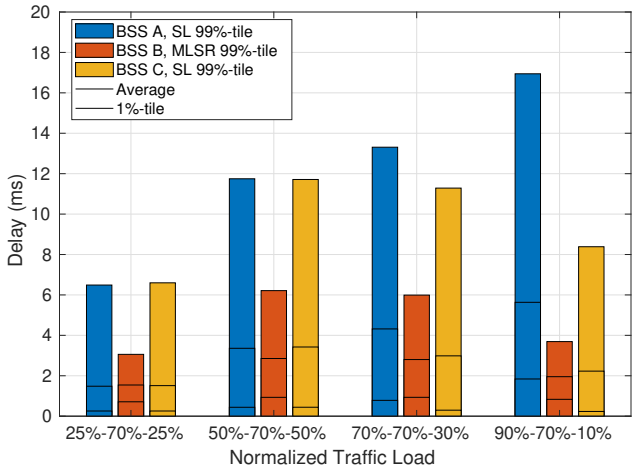


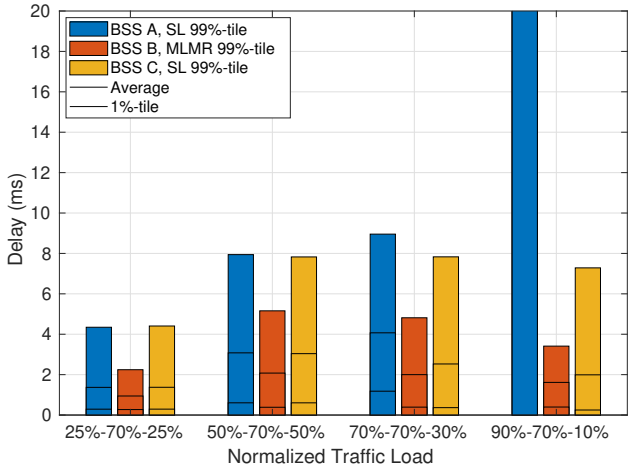
Figure 3.7: Scenario III: individual BSS throughput when BSS B employs MLSR (left) and MLMR (right). BSSs A and C are assume to employ a single link.

BSS to reach lower delays than the MLSR case. Having said that, the last MLMR configuration shows the SLO BSS with a 90% load saturating, as the greedy nature of MLMR stops the SLO BSS to obtain the necessary transmission opportunities to handle all its traffic. In this way, we can surmise that MLSR is the less disruptive technology, and that MLMR can be beneficial to all nodes in a network, but under high traffic conditions it can disrupt other BSS.

By transmitting over multiple links simultaneously, MLO devices can greatly reduce their delay and increase their throughput. However, when paired with other MLO devices, they can starve each other and lead to longer delays than SLO networks using orthogonal channels, thus channel selection needs to account for other MLO devices, and avoid using the same pair of channels for devices in the same coverage area. When paired with SLO devices, MLO can opportunistically use the links to reduce its own delay, as well as the delay of the SLO devices, but it can also starve other networks when the link capacity is limited. Publications $\#4$ and $\#5$ focus on the delay anomaly of MLMR and the MLSR implementation. Publication $\#5$ specifically shows that smart channel assignment with two links can lead to better performance than a costly array of five links.



(a) MLSR delay



(b) MLMR delay

Figure 3.8: Delay in unbalanced scenarios. MLSR has a low impact on nearby BSSs, while MLMR has a severe impact on highly loaded channels, saturating BSS A. Traffic load values refer to the fraction of the SL full-buffer throughput.

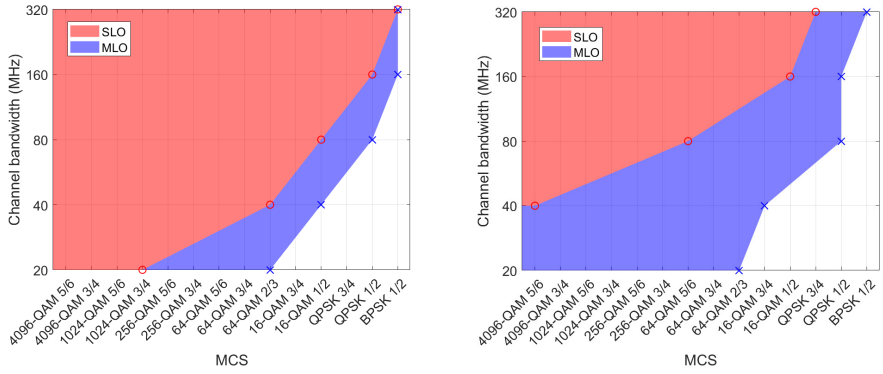
3.4 VR over Wi-Fi

In this chapter, we studied the use of MLO as an enabler for Virtual Reality, a challenging new traffic type with stringent throughput and delay requirements. We reproduced realistic VR traffic based on real traces, and studied the relationship between MCS, channel bandwidth, and how to handle multiple VR users in the same network.

To analyze and replicate VR traffic, we used the traces from [37], which can be found in Zenodo as a dataset [38]. The downlink of VR shows similar behaviors to the traces shown in Figure 3.1 of Chapter 3.1, with batches of numerous packets with an inter-arrival time defined by the frame rate. In this case, both the traffic load and frame rate are higher than Stadia, with 100 Mbps and 90 FPS respectively. The behavior in the uplink is also dictated by the frame rate, with 4 small packets being sent in each interval of one video frame. These packets in the uplink represent only 1 Mbps, but they contain information on the inputs and perspective of the user, and thus it is important to ensure their timely delivery. We consider the delay thresholds set by the Wi-Fi Alliance for VR gaming [39], which stipulate different thresholds for the downlink and uplink. For the downlink: up to 5 ms for the 75th percentile delay, 10 ms for the 95th percentile and 50 ms for the 99.9th percentile. For the uplink: up to 2 ms for the 90th percentile, and 10 ms for the 99.9th percentile. Note that as previously mentioned, the uplink thresholds are stricter than the downlink.

We began by measuring the minimum MCS and channel bandwidth required to be able to obtain a good VR performance according to the aforementioned requirements. We show the necessary combination for both SLO and MLO in Figure 3.9. As expected, the uplink requirements (Figure 3.9b) are harder to meet than the downlink ones (Figure 3.9a). Focusing on the uplink, we can determine that SLO cannot deliver VR using a 20 MHz channel no matter what MCS is used. It could be achieved with 40 MHz, but only with the highest MCS available in Wi-Fi 7 (4096-QAM). Indeed, even with the widest possible channel, SLO requires at least QPSK 3/4, meaning that VR devices need to be close to the AP

to find a good performance (i.e., not the entire coverage area of the AP is suitable for VR). MLO has an easier time meeting the requirements, even delivering good performance at the lowest MCS as long as 320 MHz channels are used. Of course, MLO uses two links instead of one, but even if we compare equivalent bandwidths (SLO at 40 MHz with MLO at 20 MHz), MLO only requires 64-QAM, while SLO needs 4096-QAM. Thus, the MLO benefits go beyond the extra bandwidth, and are a direct consequence of having multiple simultaneous backoffs and transmissions.



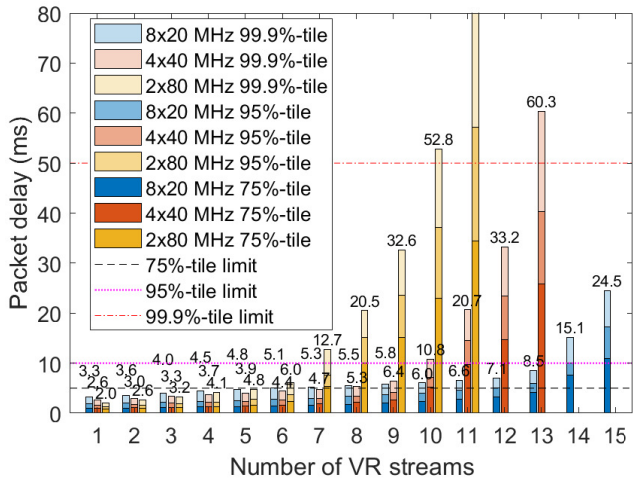
(a) Downlink (75%-tile of 5 ms) (b) Uplink (90%-tile of 2 ms)

Figure 3.9: Minimum MCS to accomplish Wi-Fi Alliance thresholds for different channel widths.

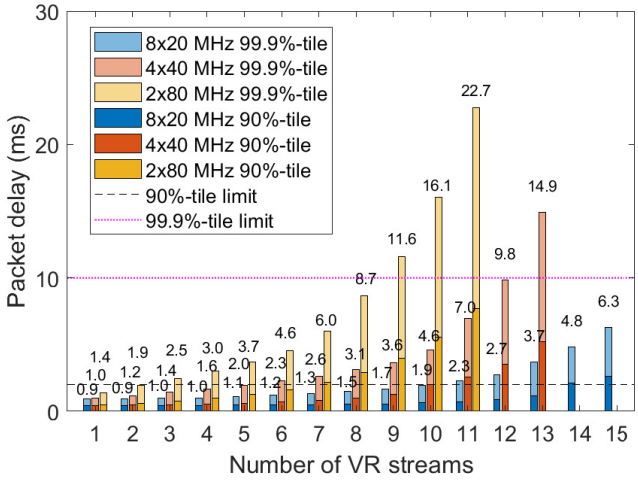
Next, we studied the relationship between bandwidth and number of MLO links, and how to properly configure an MLO device to serve as many VR users as possible. We considered two links of 80 Mhz, four links of 40 MHz, and eight links of 20 MHz. Figure 3.10 shows the packet delay as the number of VR streams increase, with the Wi-Fi Alliance thresholds marked in horizontal lines. Starting with the downlink in Figure 3.10a, two links allows us to serve up to 6 users before crossing the 75th threshold, four links brings the user count up to 9, and eight links allows for good VR performance with up to 13 users. If we aim to minimize the delay however, we observe that for up to 3 users, two

links leads to the minimum delay possible, four links is the optimal option for 4 to 8 users, and after that eight links leads to the lowest delays. This is a result of the way that VR traffic is shaped. As packets arrive in bursts, having an excess of links leads to idle links that do not transmit anything, and in those circumstances, having a wider channel reduces the transmission time. When there are multiple users it is better to have a lot of links instead, thus delivering all the packets simultaneously and avoiding queueing delays. For the uplink in Figure 3.10b, we find the same tendency when it comes to number of users and links required, but in all cases it appears to be beneficial to have more links to minimize the delay. Indeed, the uplink are short transmissions that arrive regularly, thus having extra bandwidth does not benefit the transmission time, as no aggregation is present in the uplink. Reducing the queueing time and the access time then becomes more important than for the downlink.

VR traffic creates self-contention between the downlink and uplink, and with SLO, the downlink contention stops the uplink from delivering its packets in a timely manner. By using multiple links, MLO alleviates this contention on the uplink and more easily delivers a good VR performance. It also requires lower MCS than SLO, thus giving a wider margin for user placement. MLO can split the same bandwidth over more or less links, and using more links with a shorter bandwidth can allow more VR users in a network than wider links. However, a trade-off can be achieved in which the downlink delay can be minimized based on the number of users present by using wider links. Publication $\#6$ focuses on streaming VR over Wi-Fi using MLO, showcasing the real traffic traces used, our modelling of the same traffic, and the added performance that MLO offers over SLO.



(a) Downlink



(b) Uplink

Figure 3.10: Packet delay for different configurations of links and bandwidth.

Chapter 4

CONCLUSIONS

In this thesis, we studied the use of new Wi-Fi features to improve the network latency in order to support emergent streaming applications that require both a high throughput and low delay. In order to do so, we first did a comprehensive study of a cloud gaming application with real traces, identifying its main traffic patterns and how they are affected by the transmitted media, video resolution, and frame rate. We then monitored the IEEE 802.11be standardization efforts, identifying Multi-Link Operation as a key feature with the potential to both increase network throughput and reduce latency. We implemented the main MLO modes in our own simulation tool, and using real traces with different occupancy levels, tested MLO performance when the links are similar or dissimilar in occupancy, showing how MLO can opportunistically select the least busy link to avoid contention, thus improving overall performance and offering up to an order of magnitude reduction in the worst-case delay. We continued this study by examining the effect of having multiple MLO nodes in a network, discovering the delay anomaly which can worsen worst-case delay over traditional SLO. Following that, we studied how to avoid the anomaly by ensuring that MLO nodes used different channels for at least some of their links, thus avoiding contention as much as possible. Finally, we analyzed another streaming application with VR, finding similar traffic patterns to those of cloud gaming. We compared SLO and

MLO performance, showing that MLO can enable VR with a lower MCS (worse connection) and that with double the bandwidth of SLO, MLO can allow more than double the VR users.

In our study of MLO, we have found its ability to avoid congestion to be the most relevant to the needs of current Wi-Fi networks. Indeed, increasing the available throughput is useful, but its capacity for reducing network delay is unmatched by other features. There are multiple varieties of MLO, but we have found STR to be the most versatile. Having independent links allows not only the opportunistic use of different links to avoid congestion, but they can also be used to undertake different tasks, such as having a dedicated link for control packets, or low-latency traffic. Through our work however, we have noticed that enabling all MLO links to transmit every kind of traffic outperforms most complex schemes that assign different tasks to each link, as it reduces the contention perceived by them, reducing the queueing times and overall delay. Still, with future devices possibly having up to four links, a combination of schemes seems possible, for instance reserving one link for best-effort traffic, thus allowing all other links to quickly dispatch sensitive traffic opportunistically.

In the future, MLO links could be used independently to enable more advantages over SLO, separating traffic types over different links, or modifying the packet aggregation scheme to take into consideration the multiple links available, avoiding long transmissions over one link, when they could be split over two or more, reducing the link contention. Further, future IEEE 802.11 amendments will have to consider MLO as a core aspect of their architecture, thus it can be used in conjunction with other techniques to improve performance even more. One of the key features of IEEE 802.11bn is going to be multi-AP coordination, enabling the use of coordinated OFDMA and spatial reuse to reduce overall contention. Distributed MLO will also enable a network to associate users using MLO to links from different APs, thus avoiding roaming delays. Coordinated beamforming can also be used in conjunction with MLO to facilitate more transmission opportunities and lower MLO delay.

Bibliography

[1] TechReport. Online Gaming Statistics: Trends and Analysis of the Industry. https://techreport.com/statistics/software-web/gaming-statistics/, 2024. Last accessed 17/07/2024.

[2] PC MAG. Nvidia's GeForce Now Game-Streaming Service Tops 20 Million Users. https://uk.pcmag.com/pc-games/142289, 2022. Last accessed 17/07/2024.

[3] Lorenzo Galati Giordano, Giovanni Geraci, Marc Carrascosa, and Boris Bellalta. What will Wi-Fi 8 be? A primer on IEEE 802.11bn ultra high reliability. *arXiv preprint arXiv:2303.10442*, 2023.

[4] Wi-Fi Alliance. Wi-Fi 6E Insights. https://www.wi-fi.org/system/files/Wi-Fi_Alliance_Wi-Fi_6E_Insights_Newsletter_202311_0.pdf, 2023. Last accessed 17/07/2024.

[5] Wi-Fi Alliance. 6 GHz Wi-Fi: Connecting to the future. https://www.wi-fi.org/system/files/6_GHz_Wi-Fi_Connecting_to_the_future_202210.pdf, 2022. Last accessed 17/07/2024.

[6] Maddalena Nurchis and Boris Bellalta. Target wake time: Scheduled access in IEEE 802.11 ax WLANs. *IEEE Wireless Communications*, 26(2):142–150, 2019.

[7] Xiaoqian Liu, Yuhan Dong, Yiqing Li, Yousi Lin, Xun Yang, and Ming Gan. IEEE 802.11 be Wi-Fi 7: Feature Summary and Performance Evaluation. *arXiv preprint arXiv:2309.15951*, 2023.

[8] IEEE 802.11 TGbn. IEEE 802.11bn PAR. https://mentor.ieee.org/802.11/dcn/23/11-23-0480-03-0uhr-uhr-proposed-par.pdf, 2022. Last accessed 17/07/2024.

[9] zarik5. Air Light VR. https://github.com/alvr-org/ALVR, 2024. Accessed 28 July 2024.

[10] Verdict. Cloud Gaming Timeline. https://www.verdict.co.uk/cloud-gaming-timeline, 2021. Last accessed 24/07/2024.

[11] How SONY Bought, And Squandered, The Future Of Gaming. https://www.theverge.com/2019/12/5/20993828, 2020. Accessed 12 June 2020.

[12] Google. Stadia Announcement FAQ. https://support.google.com/stadia/answer/12790109, 2023. Accessed 24 July 2024.

[13] Xbox Cloud Gaming Official website. https://www.xbox.com/es-es/play, 2024. Accessed 24 July 2024.

[14] Amazon. Amazon Luna official website. https://luna.amazon.com/, 2024. Accessed 24 July 2024.

[15] Andrea Di Domenico, Gianluca Perna, Martino Trevisan, Luca Vassio, and Danilo Giordano. A Network Analysis on Cloud Gaming: Stadia, GeForce Now and PSNow. *Network*, 1(3):247–260, 2021.

[16] Oswaldo Sebastian Peñaherrera-Pulla, Carlos Baena, Sergio Fortes, Eduardo Baena, and Raquel Barco. Measuring key quality indicators in cloud gaming: Framework and assessment over wireless networks. *Sensors*, 21(4):1387, 2021.

[17] Minzhao Lyu, Yifan Wang, and Vijay Sivaraman. Do Cloud Games Adapt to Client Settings and Network Conditions? In *Proc. IFIP Networking*, 2024.

[18] Xavier Marchal, Philippe Graff, Joël Roman Ky, Thibault Cholez, Stéphane Tuffin, Bertrand Mathieu, and Olivier Festor. An analysis of cloud gaming platforms behaviour under synthetic network constraints and real cellular networks conditions. *Journal of Network and Systems Management*, 31(2):39, 2023.

[19] Philippe Graff, Xavier Marchal, Thibault Cholez, Bertrand Mathieu, and Olivier Festor. Efficient identification of cloud gaming traffic at the edge. In *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, pages 1–10. IEEE, 2023.

[20] Virtual Reality Society. History of Virtual Reality. `https://www.vrs.org.uk/virtual-reality/history.html`, 2019. Accessed 28 July 2024.

[21] Thomas Alsop and Statista. Virtual Reality (VR) statistics and facts. `https://www.statista.com/topics/2532/virtual-reality-vr/#topicOverview`, 2024. Accessed 28 July 2024.

[22] Matthijs Jansen, Jesse Donkervliet, Animesh Trivedi, and Alexandru Iosup. Can My WiFi Handle the Metaverse? A Performance Evaluation Of Meta's Flagship Virtual Reality Hardware. In *Companion of the 2023 ACM/SPEC International Conference on Performance Engineering*, pages 297–303, 2023.

[23] Guilherme Gonçalves, Pedro Monteiro, Miguel Melo, José Vasconcelos-Raposo, and Maximino Bessa. A comparative study between wired and wireless virtual reality setups. *IEEE access*, 8:29249–29258, 2020.

[24] Seyedmohammad Salehi, Abdullah Alnajim, Xiaoqing Zhu, Malcolm Smith, Chien-Chung Shen, and Leonard Cimini. Traffic characteristics of virtual reality over edge-enabled wi-fi networks. *arXiv preprint arXiv:2011.09035*, 2020.

[25] Seyedeh Soheila Shaabanzadeh, Marc Carrascosa-Zamacois, Juan Sánchez-González, Costas Michaelides, and Boris Bellalta. Virtual reality traffic prioritization for Wi-Fi quality of service improvement using machine learning classification techniques. *Journal of Network and Computer Applications*, 230:103939, 2024.

[26] Adrian Garcia-Rodriguez, David López-Pérez, Lorenzo Galati-Giordano, and Giovanni Geraci. IEEE 802.11 be: Wi-Fi 7 strikes back. *IEEE Communications Magazine*, 59(4):102–108, 2021.

[27] Cailian Deng, Xuming Fang, Xiao Han, Xianbin Wang, Li Yan, Rong He, Yan Long, and Yuchen Guo. IEEE 802.11 be Wi-Fi 7: New challenges and opportunities. *IEEE Communications Surveys & Tutorials*, 22(4):2136–2166, 2020.

[28] Evgeny Khorov, Ilya Levitsky, and Ian F Akyildiz. Current status and directions of IEEE 802.11 be, the future Wi-Fi 7. *IEEE access*, 8:88664–88688, 2020.

[29] Cheng Chen, Xiaogang Chen, Dibakar Das, Dmitry Akhmetov, and Carlos Cordeiro. Overview and performance evaluation of Wi-Fi 7. *IEEE Communications Standards Magazine*, 6(2):12–18, 2022.

[30] Gaurang Naik, Dennis Ogbe, and Jung-Min Jerry Park. Can Wi-Fi 7 support real-time applications? On the impact of multi link aggregation on latency. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[31] Ana Jeknić and Enis Kočan. Multi-Link Operation for Performance Improvement in Wi-Fi 7 Networks. In *2024 28th International Conference on Information Technology (IT)*, pages 1–4. IEEE, 2024.

[32] Alvaro López-Raventós and Boris Bellalta. IEEE 802.11 be multi-link operation: When the best could be to use only a single interface. In *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 1–7. IEEE, 2021.

[33] Molham Alsakati, Charlie Pettersson, Sebastian Max, Vishnu Narayanan Moothedath, and James Gross. Performance of 802.11 be Wi-Fi 7 with multi-link operation on AR applications. In *2023 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2023.

[34] Sergio Barrachina-Muñoz, Boris Bellalta, and Edward Knightly. Wi-Fi All-Channel Analyzer. In *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*, pages 72–79, 2020.

[35] Sergio Barrachina-Muñoz, Boris Bellalta, and Edward W Knightly. Wi-fi channel bonding: An all-channel system and experimental study from urban hotspots to a sold-out stadium. *IEEE/ACM Transactions on Networking*, 29(5):2101–2114, 2021.

[36] Marc Carrascosa-Zamacois, Giovanni Geraci, Edward Knightly, and Boris Bellalta. Wi-Fi Multi-Link Operation: An Experimental Study of Latency and Throughput. *IEEE/ACM Transactions on Networking*, 2023.

[37] Costas Michaelides, Miguel Casasnovas, Daniele Marchitelli, and Boris Bellalta. Is Wi-Fi 6 Ready for Virtual Reality Mayhem? Aa Case Study Using One AP and Three HMDs. *Authorea Preprints*, 2023.

[38] Costas Michaelides, Miguel Casanovas, Danielle Marchitelli, and Boris Bellalta. VR Gaming Dataset - Multi-users tests. `https://doi.org/10.5281/zenodo.8169785`, 2023.

[39] Wi-Fi Alliance. Wi-Fi delivers immersive VR gaming. `https://www.wi-fi.org/system/files/VR_Gaming_`

Highlights_20231215_0.pdf, 2023. Last accessed 17/06/2024.

Appendix A

PUBLICATIONS

Cloud-gaming: Analysis of Google Stadia traffic

Marc Carrascosa-Zamacois*, Boris Bellalta

Wireless Networking Research Group, Universitat Pompeu Fabra
Carrer de Roc Boronat 138, 08018 Barcelona, Spain

Abstract

Interactive, real-time, and high-quality cloud video games pose a serious challenge to the Internet due to simultaneous high-throughput and low round trip delay requirements. In this paper, we investigate the traffic characteristics of Stadia, the cloud-gaming solution from Google, which is likely to become one of the dominant players in the gaming sector. To do that, we design several experiments, and perform an extensive traffic measurement campaign to obtain all required data. Our first goal is to gather a deep understanding of Stadia traffic characteristics by identifying the different protocols involved for both signalling and video/audio contents, the traffic generation patterns, and the packet size and inter-packet time probability distributions. Then, our second goal is to understand how different Stadia games and configurations, such as the video codec and the video resolution selected, impact on the characteristics of the generated traffic. We also evaluate the ability of Stadia to adapt to different link capacity conditions, including cases where the capacity drops suddenly, as well as sudden increases in the network latency. Our results and findings, besides illustrating the characteristics of Stadia traffic, are also valuable for planning and dimensioning future networks, as well as for designing new resource management strategies. Finally, we compare Stadia traffic to other video streaming applications, showcasing the main differences between them, and introduce a traffic model using our captures. We show that this model can be used in simulations to further investigate the network performance in presence of Stadia traffic.

Keywords: Cloud-gaming, Google Stadia, Traffic Measurements and Analysis

*Corresponding author

1. Introduction

Video streaming is more popular than ever. It represents a share of 60.6% of all internet downlink traffic, far above the second and third places: web browsing, which takes 13.1%, and gaming with 8.0% [1]. Video on demand services like Netflix, Amazon Prime Video and Disney+ report 183 million, 150 million and 50 million subscribers each [2], Youtube has 2 billions of logged-in users each month [3], and Twitch reports an average of over 1.9 million concurrent users [4].

Cloud gaming is the next step in streaming video content on demand, with several companies already offering subscription services that allow users to play video games remotely. In cloud gaming, the games are run in a remote server, and then streamed directly to the users, thus removing the need for dedicated gaming computers or consoles. While this has many advantages, it challenges the network infrastructure to support both high-throughput and low-latency, as otherwise the service will simply not run.

Some of the first companies to attempt this model were OnLive and Gaikai, which were later bought by Sony, who entered the market with PlayStation Now [5]. Microsoft has its own service called XCloud, offering to play their games on Android phones and tablets [6]. Nvidia offers GeForce Now, allowing users to remotely play the games that they have bought previously in online stores on Windows, Mac, and Android devices [7]. Lastly, Google has also entered the market with Stadia, a service that runs on Google Chrome or on Chromecast. Stadia has its own shop to purchase games and a subscription service offering free games each month. A key characteristic of Stadia is that it supports a 4K video resolution only for its PRO subscribers.

To deliver this service, Google uses Web Real Time Communication (WebRTC), an open standard by the World Wide Web Consortium (W3C), and the Internet Engineering Task Force (IETF), allowing peer-to-peer voice, video and data communication through browsers

with a JavaScript API. WebRTC has a strong presence in videocalling and messaging. It is used by applications such as Google Hangouts or Amazon Chime, and it has the third spot in the global messaging market share after Skype and Whatsapp [1].

In this paper, we offer a comprehensible overview of how Stadia works, and the main characteristics of the traffic that Stadia generates. To do that, we perform a measurement campaign, obtaining a large dataset covering all aspects of interest, which also serves as a snapshot of Stadia behavior as a home user would perceive it near its launch[1]. Our main goal is to characterize the traffic generated by Stadia, how it changes for different games, video codecs and video resolutions, and how Stadia reacts to different network conditions. To the best of our knowledge, this is the first work providing a detailed analysis of Stadia traffic, and likely, the first work analyzing the traffic generated by a cloud-gaming solution in such a detail.

The main findings of this paper are:

1. Different games have different traffic characteristics such as the packet size, inter-packet times, and load. However, we have found the traffic generation process follows a common temporal pattern, which opens the door to develop general but parameterizable Stadia traffic models. We also found that some of these patterns are present in other WebRTC applications.
2. The use of different video codecs (VP9 and H.264) and different video resolutions does not change the traffic generation process. Although we expected that the recent VP9 codec to significantly outperform H.264 in terms of traffic load, our results show they perform similarly, with H.264 even resulting in less generated traffic in some cases.
3. Stadia works properly for different link capacities. We show that Stadia strives to keep the 1080p resolution at 60 frames

[1]The dataset with all the collected traffic traces is available in open access here: `https://github.com/wn-upf/Stadia_cloud_gaming_dataset_2020`

per second (fps) even if the available bandwidth is far below its own pre-defined requirements, and only switches to a lower resolution of 720p as the last resort. In all cases, Round Trip Time (RTT) values consistently remain below 25 ms, with average values between 10 and 15 ms, even when the experiments and measurements have been done at different times in a temporal span of several months.

4. Stadia attempts to recover from a sudden drop in the available link capacity almost immediately, entering a transient phase in which Stadia aims to find a new configuration to compensate the lack of network resources. This transient phase however, can last over 200 seconds. During this time, although the user is able to continue playing, the quality of experience is heavily affected, with constant resolution changes, inconsistent framerate and even audio stuttering.
5. Stadia traffic adapts to sudden increases in network latency by increasing uplink traffic (i.e., client to server feedback), and decreasing the downlink video traffic. In contrast to the drops in bandwidth, the user experience is less affected when the latency increases, as no packets are dropped by the receiver and framerate is kept consistent for the whole capture.
6. Traffic patterns found on Stadia are similar to those of other WebRTC applications, especially for video packets. For non-video packets, all the WebRTC applications tested show application specific patterns.

Further, a traffic model is presented, which uses the patterns found in our captures such as the time between frames, packet size and number of packets that arrive in batches to accurately replicate Stadia traffic for a specific game (Tomb Raider) using different video resolutions. The model is built in a way it can be easily implemented as a traffic generator, either in a simulator or in the real world, which makes it suitable to be used in the performance evaluation of networking systems. Moreover, its parameters can be easily updated to cover a

broad range of traffic generation patterns even if they do not belong to any particular game.

The rest of the paper is organized as follows: Section 2 provides an overview of the related work. Section 3 introduces Google Stadia, WebRTC and all protocols involved in their operation. The details of the experimental setup and definition of the datasets is presented in Section 4. A study of the bandwidth required to play multiple Google Stadia games can be found in Section 5. An analysis of the main characteristics of Google Stadia traffic can be found in Section 6. Section 7 studies the traffic evolution as the game changes states, and the effects of the video encoding and resolution can be found in Section 8. Performance under different available bandwidths is presented in Section 9, and the effects of such bandwidth changes on latency is investigated in Section 10. Section 11 analyses how Stadia adapts when latency increases suddenly. Section 12 compares Stadia traffic patterns to those of other WebRTC applications and analyzes their similarities, and a model based on the traces is presented in Section 13. Finally, Section 14 closes the paper.

2. Related Work

Cloud gaming has received some attention, especially in regards to finding methods of compensating latency issues. The authors in [8] present a crowdsourced study of 194 participants in which gameplay adaptation techniques are used to compensate up to 200 ms of delay. Some of these modifications include increasing the size of targets or reduce their movement speed, and the results show an improvement of the user QoE. If taken too far however, these latency adaptations reduce the perceived challenge of the game, resulting in an unsatisfying experience for users. An overview of the main issues with cloud gaming can be found in [9], where the performance of OnLive is tested. OnLive is one of the first cloud gaming services, highlighting the cloud overhead of 100 ms as a major challenge for player interaction with certain games. Authors in [10] emulate OnLive's

system, streaming content from a PlayStation 3 and applying packet loss and delays to then perform a subjective study on the quality of the service perceived by users. As it could be expected, they observe that reduced delays are very important for fast paced games, while slower games suffer more from packet loss and degraded image quality than from latency issues. Lastly, the relationship between framerate and bitrate is studied in [11], presenting a QoE prediction model using both variables, and commenting that the graphical complexity of different games is a challenge in generalizing such a model. Their results also show that for low bitrates, reducing the framerate leads to a better experience.

WebRTC performance on videoconferencing has been studied at length. The authors in [12] test the congestion control capabilities of WebRTC. Performing tests with different latencies, it is shown that performance is maintained while latency is below 200 ms. They also conclude that in presence of TCP cross-traffic, WebRTC video streams can heavily reduce their datarate to avoid an increase in latency at the cost of a lower video quality. The work in [13] does another in-depth analysis of the adaptability of WebRTC to different network conditions. WebRTC performance is evaluated in both wired and wireless networks, showing that the bursty losses and packet retransmissions from wireless connections have a severe impact on the video quality. WebRTC performance in presence of TCP downloads is studied in [14], where multiple queue management methods are applied to avoid WebRTC performance degradation.

The quality of WebRTC services is assessed by defining two groups of metrics in [15]: Quality of Service (QoS) ones such as latency, jitter, and throughput; and Quality of Experience (QoE) ones, which focus on the user satisfaction with the service. The evaluation of the QoE includes the use of both subjective methods, such as collecting feedback from users, and objective methods such as the Peak-to-Noise Ratio (PSNR) and the Structure Similarity (SSMI) index to calculate image degradation after compression and transmission. The work in

[16] studies the use of Chrome's WebRTC internals as a source of QoS statistics, showing that the reported values for throughput and packet loss correlate well with user perceived issues in the connection.

This paper aims to characterize the performance of Google Stadia, studying its traffic generation patterns under several different configurations, and analyzing its mechanisms for traffic adaptation. Similar studies of audio and video applications can be found in the literature. The work in [17] studies how Skype changes its frame size, inter-packet gap and bandwidth according to network limitations, codec used and packet loss, noting on how packet size increases with them, retransmitting past blocks to compensate. The authors in [18] used their own hardware to create a cloud gaming setup and study 18 different games, separated by genre, and find how said genre affects bandwidth, packet rate and video characteristics. In [19], a study on cloud gaming platforms OnLive and StreamMyGame performed controlled experiments modifying the network delay, packet loss rate and bandwidth at the router to test and compare the performance of each service.

To be the best of our knowledge, this was the first paper focusing on the analysis of Stadia traffic when it was uploaded to arxiv in 2020 [20]. Since then, it has served to other authors [21–25] as the starting point for their works on Stadia and other cloud gaming platforms. These papers provide complementary results to the analysis presented here by considering different network conditions, as well as particular network technologies such as Wi-Fi and mobile networks.

3. How Stadia Works: WebRTC

In this Section we introduce how Stadia works. We first detail the user-server interaction, to then introduce the different protocols and mechanisms involved, such as the Google Congestion Control algorithm. Finally, we overview how "negative latency" could have been implemented in Stadia, since to the best of our knowledge, there is no information available on this aspect.

3.1. User-server interaction

On a computer, Google Stadia can be played through Google Chrome. Once users reach `http://stadia.google.com`, they can either access the shop to acquire games (either by buying them or by just acquiring the ones provided for free with a PRO account), or go to their main page to choose one of the already available games to play. This part of Stadia is just regular web browsing until the user chooses a game, at which point the browser starts a WebRTC video session, switches to a full screen mode, and the user can start playing.

Once the video session begins, the server transmits both video and audio, while the user transmits their inputs (coming from a gamepad or their mouse and keyboard). This way, both the video stream and the input stream have different traffic loads for each game: an action game will require constant inputs from the player, while a puzzle game will have a slower and more methodical playstyle. Inside a game, since there are different states (menus, playing the game, idle, pause screen, etc.), traffic loads are also variable, although it is expected users will stay in the "play" state for most of the time.

The stream of the game is customized according to the user's needs, and there are two parameters that have a direct impact in the quality of a stream: resolution and video codec. Stadia offers three different resolutions: 1280x720 (i.e., 720p), 1920x1080 (i.e., 1080p) and 3840x2160 (i.e., 2160p or 4K). The resolution can change mid stream automatically, according to the network state, but it can also be restricted by the users. Since higher resolution will require higher downlink traffic, Stadia allows users to restrict it to "limited" (720p), "balanced" (up to 1080p) and "Optimal" (up to 4K). Note that Stadia only restricts the maximum resolution (i.e., a balanced configuration may still use 720p if network conditions are unfavorable).

The video encoding is selected automatically at the beginning of the session, and kept unchanged until it finishes. Stadia uses two video coding formats:

- **H.264:** It is the most supported video format nowadays, with 92% of video developers using it in 2018, followed by its successor H.265 with 42% [26]. While H.265 promises half the bitrate of H.264 at the same visual quality [27][28], H.264 has been supported in phones, tablets and browsers for years. Thus, for Stadia and other content providers, it serves as a fallback to ensure that no user will have issues decoding their media.
- **VP9:** Developed by Google in 2013 as the successor of VP8. It is royalty-free, as opposed to H.264 and H.265, and has a comparable performance to H.265 [29][30]. It is already used by Youtube, and Google reports that VP8 and VP9 make up 90% of WebRTC communications through Google Chrome [31].

Audio encoding is done through **Opus** [32], an open audio codec released in 2012 and standardized by the IETF. Designed with voice over IP and live distributed music performances in mind, it can scale audio from 6 kbps to 510 kbps. It is used widely by applications such as WhatsApp[2] and Jitsi[3].

3.2. WebRTC

Google Stadia uses WebRTC to provide its services, which uses the following protocols:

1. **ICE, STUN and TURN:** Interactive Connectivity Establishment (ICE) [33] is a protocol designed to facilitate peer-to-peer capabilities in UDP media sessions via Network Address Translator (NAT). It uses Session Traversal Utilities for NAT (STUN) and Traversal Using Relay NAT (TURN). STUN is used to query a server about the public IP of the user. Once both users know their respective IP and port, they can then share this information with the other user to establish a direct connection. TURN is used when a direct connection is not

[2]WhatsApp `https://www.whatsapp.com/`
[3]Jitsi `https://meet.jit.si/`

possible (because of a symmetric NAT). In such occasions, a TURN server is used as the intermediary between the users.
In the signaling process, the Session Description Protocol (SDP) is used to negotiate the parameters for the session (audio and video codecs, IP and ports for the RTCP feedback, etc). These are exchanged along with the ICE candidates, which inform both peers of their connectivity options (IP and port, direct connection or through a TURN server, transport protocol used, etc). UDP is the preferred protocol for most live streaming applications, but Transmission Control Protocol (TCP) is also supported by ICE.

2. **DTLS:** Datagram Transport Layer Security (DTLS) [34] [35] is used to provide security in datagram based communications. It was designed as an implementation of TLS that did not require a reliable transport channel (i.e., it was designed for data exchanges that do not use TCP), as a consequence of the increased use of User Datagram Protocol (UDP) for live streaming applications, which prioritize timely reception over reliability. DTLS has become the standard for these kind of live streaming applications, and it is used in WebRTC to secure the RTP communication.
3. **RTP and RTCP:** Real-Time Protocol (RTP) and Real-Time Control Protocol (RTCP) [36] [37] are used to transmit media over the secured channel. RTP is used by the sender of media (Stadia), while RTCP is used by the receiving client to provide feedback on the reception quality. RTP packets are usually sent through UDP, and contain a sequence number and a timestamp. The endpoints implement a jitter buffer, which is used to reorder packets, as well as to eliminate packet duplicates, or compensate for different reception timing.
   RTCP provides metrics that quantify the quality of the stream received. Some of these metrics are packet loss, jitter, latency, and the highest sequence number recieved in an RTP packet. RTCP packets are timed dynamically, and are recommended to

account for only 5% of RTP/RTCP traffic.

Once the ICE connection and DTLS encryption are in place, the connection consists mostly of RTP and RTCP packets for the video stream, and the application data (which we assume includes user inputs) sent through DTLS packets and STUN binding requests being sent periodically to ensure that the peer to peer connection can be maintained. RTP packets encrypted with DTLS maintain their structure, and do not use DTLS headers, which is why we can identify DTLS and RTP packets separately.

3.3. Congestion Control

Google Congestion Control [38] has two main elements: a delay-based controller at the client side, and a loss-based controller at the server side. The delay-based controller uses the delays between the video frame transmission at the sender and its arrival at the receiver to estimate the state of buffers along the path (i.e., it compares the time it took to transmit a full video frame at the sender, with the time it took for the sender to receive all packets that form that frame). Using this information, as well as the receiving bitrate in the last 500 ms, it calculates the required bitrate A_r, and forwards it to the sender. Notification messages from the client to the server are sent every second unless there is a significant change (i.e., a difference higher than 3%) in the estimated bitrate with respect to the previous one, in which case they are forwarded immediately. The loss-based controller works at the server side. It estimates the bitrate A_s based on the fraction of packets lost provided by RTCP messages. Its operation is simple: If the packet loss is below 0.02, the A_s is increased. On the contrary, if it is above 0.1, the bitrate A_s is decreased. In case packet losses are in between 0.02 and 0.1, A_s remains the same. The sender then uses the minimum of the two bitrates, i.e., $\min(A_r, A_s)$ to choose the current bitrate at which packets are transmitted.

3.4. Negative Latency

The mechanisms used by Stadia to manage network delays have been named *negative latency* by Google. There is much speculation on what the term means, but the details have not been made available to the public. According to Google Stadia's designers: "*We created what we call a negative latency. It allows us to create a buffer for the game to be able to react with the user, and that accommodates the latency that you have on the network*" [39]. Their hardware infrastructure has also been mentioned [40], citing that 7500 Stadia edge nodes have been deployed at partnered ISPs to reduce the physical distance from server to user.

In [41], several aspects are considered to reduce latency in cloud gaming. Their approach consists of several aspects: a) Future state prediction with a Markov chain on inputs that they deem predictable, combined with error compensation for mispredictions (graphical rendering to quickly hide the misprediction and correct course); and b) Rollback: when a new input appears that contradicts a prediction, in-between frames are dropped to avoid propagating the error, this serves as a state recovery system, syncing the user and server when they drift. Basically, the system predicts future inputs from the user several frames before the input is taken. Then, these predictions (i.e., the video frame associated with each prediction) are transmitted to the client giving the illusion of instantaneous reaction. There are two types of inputs: "predictive" and "speculative". For "predictive" inputs (user's movement) a Markov Chain is used to reduce the number of speculative frames to transmit. For impulsive inputs they render multiple possibilities and send them all, only showing the correct one at the user side. Supersampling is also used (i.e., sampling the inputs of the user at a higher rate than the screen framerate), thus being able to receive multiple inputs per frame instead of limiting their inputs to one per frame. This reduces the sampling noise, and so improves the prediction accuracy.

The authors in [8] adjust the gameplay to compensate for latency

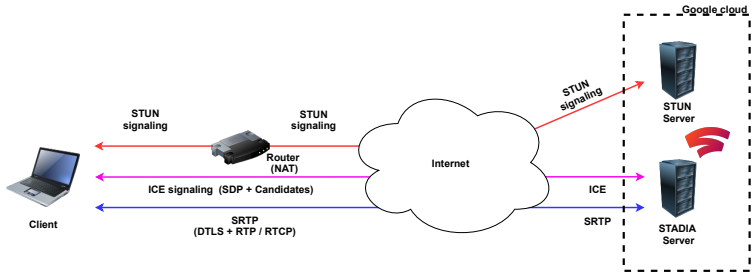


Figure 1: Google's Stadia: Main components and data streams.

issues on cloud gaming. They modify the size of the targets, their hitbox (i.e., the target is visually the same size, but the game registers hits on a larger area), their speed and quantity. These changes result in a better experience for most users. Authors in [42] do the same, modifying a game of tank combat, and showing that latency severely affects those actions that require speed and precision.

Overall the consensus seems to be that "negative latency" is a combination of closeness to the datacenter, predictive inputs, server side running at a higher frame rate for faster reactions, and parallel sending of speculative frames (i.e., many possible actions at once).

4. Experiments and Measurements

This section introduces the considered testbed, the designed experiments, and the obtained datasets.

4.1. Experimental Setup

With the aim to identify the most relevant characteristics of Stadia traffic, we have designed a set of experiments to provide answers to the following five questions:

1. How does Stadia generate the traffic?
2. Is the traffic constant regardless of the different game states?
3. How does the selection of the video codec and resolution affect the generated traffic?

(a) Rise of the Tomb Raider (b) Thumper (c) Spitlings

Figure 2: Screenshots of the games considered in this paper.

4. Is Stadia able to adapt to different link/network capacities?
5. How does Stadia react to a sudden network capacity change?

For each experiment, we performed extensive traffic measurements while playing Stadia. All experiments were done in an apartment building of Sarria-St.Gervasi neighborhood, in the city of Barcelona, Catalonia, Spain.

We consider a network deployment as the one depicted in Figure 1. The network consists of a laptop connected to a Wi-Fi-AP acting as an Ethernet switch.[4] The AP is a TP-Link Archer C7 v.5.0 running OpenWRT 19.07.2 [5]. The client is a Dell Latitude 5580 running Ubuntu 16.04 (kernel 4.15.0-96) [6].

All tests are performed using the Chrome browser (version 81.0.4044.92), with Wireshark (version 3.2.2)[7] running in the background to capture all incoming and outgoing traffic. Wireshark captures are processed via tshark to extract a .txt file that is later interpreted via MATLAB 2019a [8]. Filters include all RTP, RTCP, DTLS and STUN frames, and we extract the arrival timestamps, inter-packet time

[4]To avoid any influence of the Wi-Fi channel on the measured traffic characteristics, we opted to do all tests using a CAT5 Ethernet cable.

[5]OpenWRT: `https://openwrt.org/`

[6]Captures in 4K are run in Windows 10 due to Ubuntu drivers are not able to reach 60 fps.

[7]Wireshark: `https://www.wireshark.org/`

[8]Matlab: `https://mathworks.com/products/matlab.html`

and packet size. Once the capture is finished, WebRTC performance metrics (number of frames decoded, codecs used, resolution, round trip time, and jitter buffer) are extracted from Chrome via `chrome://webrtc-internals`.

Apart from Section 5 in which we compare the bandwidth requirements of multiple Stadia games, all other experiments are based on the following three games: Rise of the Tomb Raider: 20th anniversary edition (TR), Thumper (TH) and Spitlings (SP). Tomb Raider is a third person action adventure game with an open world and a lot of freedom on player inputs, as well high definition graphics that require high throughput. Thumper is an on rails rhythm game, with more limited player inputs and predictable gameplay, contrasting the freedom on TR while still requiring high throughput. Spitlings is a 2D platformer, and the game that required the lowest throughput out of all the games we tested. An illustrative screenshot of each game can be found in Figure 2.

Although we perform the measurements at the client side, we assume the network between server and client is of high capacity, and therefore its effects on shaping the traffic are negligible, and do not significantly alter Stadia traffic characteristics. This assumption is later discussed in Section 6, where, from the collected traffic traces, we conjecture it is accurate.

4.2. Datasets

After the experiments and measurement campaign, we have generated nine datasets[9]. Each dataset can contain multiple traffic traces depending on the number of experiments carried out in each category. Each traffic trace is a text file that includes two or more of the following variables as columns:

- Y_1: Packet arrival time in Epoch format.

[9]The datasets are publicly available at Github: https://github.com/wn-upf/Stadia_cloud_gaming_dataset_2020

Dataset	Name	Variables	Characteristics
D1	Temporal patterns	Y_1, Y_2, Y_3	TR, TH, SP; R: 1080p; C: VP9; T: 30 sec; DL & UL; RTP, DTLS, STUN
D2	Traffic characteristics	Y_1, Y_2, Y_3	TR, TH, SP; R: 1080p, C: VP9, T: 600 sec; DL & UL, RTP, DTLS, STUN
D3	Game states	Y_1, Y_2, Y_3	TR, SP; R: 1080p; C: VP9, T: 540 sec; DL & UL
D4	Codecs	Y_1, Y_2, Y_3	TR, SP; R: 1080p; C: VP9 & H.264, T: 600 sec, DL
D5	Resolutions	Y_1, Y_2, Y_3	TR, SP; R: 720p, 1080p, 4K; C: VP9, T: 600 sec, DL
D6	Different bandwidths	Y_1, Y_2, Y_3	TR, SP; R: 720p, 1080p; C: VP9, T: 60 sec, DL
D7	Sudden bandwidth changes	Y_4, Y_5, Y_6, Y_7	TR, SP; R: 720p, 1080p; C: VP9, T: 500 sec, DL
D8	Latency	Y_7, Y_8	TR, SP; R: 720p, 1080p; C: VP9, L: 60-600 sec, DL
D9	Latency changes	Y_5, Y_6, Y_9, Y_{10}	TR: 1080p; C: VP9, L: 180 sec, DL & UL

Table 1: Variables included in each file and dataset. R: resolution, C: video codec, T: duration, DL: downlink traffic, and UL: uplink traffic. In case the trace includes only packets from a specific protocol, it is also indicated.

- Y_2: Arrival time relative to previous packet in seconds.
- Y_3: Length of UDP payload in bytes.
- Y_4: Video frame height in pixels.
- Y_5: Video frames per second.
- Y_6: Round Trip Time in seconds.
- Y_7: Packets lost per second.
- Y_8: Jitter Buffer Delay per second.
- Y_9: Uplink RTCP data in bits per second.
- Y_{10}: Downlink RTP data in bits per second.

Variables Y_1, Y_2 and Y_3 are obtained via Wireshark, and correspond to the filters *frame.time_epoch*, *frame.time_delta_displayed* and *udp.length*. The rest of the variables are extracted via webRTC internals: receiver frame height and jitter buffer delay are taken from *RTCMediaStreamTrack*, frames decoded per second, packets lost and downlink RTP data from *RTCInboundRTPVideoStream*, uplink RTCP data from *RTCDataChannel* and round trip time from *RTCICECandidatePair*.

Table 1 specifies which of the previous variables are used in each of the datasets, as well as information of each dataset, such as the games, the duration of the measurement, the video codec used, and the resolution, among other aspects.

5. Games Overview

The following Sections, including this one, focus on the analysis of Stadia traffic. The main goal is to identify the main characteristics of Stadia traffic, and determine how it changes for different games, video codecs and video resolutions. Moreover, we will also focus on how Stadia adapts its traffic to different and changing network conditions.

In this section we overview 10 games available through the Stadia pro subscription (their name and genre can be found in Table 2). We perform captures at a video resolution of 1080p for all games, selecting only 60 seconds in which the game is being played for the analysis. Here, we compare the throughput of these 10 games for both uplink and downlink.

Figure 3a shows the boxplot for downlink UDP traffic in Mbps for each of the games tested. Each game requires a different amount of traffic, and some games have a much higher variance than others. It can be observed that most of the games require a heavy downlink load, as 80% of them have the 25th percentile over 10 Mbps and their median over 20 Mbps. The highest variance can be found on the racing game Grid, as it shows values ranging from 9.83 Mbps to 41.6 Mbps (standard deviation of 8.6), while the 2D platformer Spitlings

Game	Genre
Tomb Raider	Open world action-adventure
Thumper	On-rails ryhtm game
Spitlings	Side-scrolling platformer
Gylt	Action-adventure
Grid	Racing
SuperHot	First person shooter
Samurai Shodown	1 on 1 fighting
Serious Sam 3	First person shooter
Farming Simulator 2019	Simulation
Panzer Dragoon	On-rails third person shooter

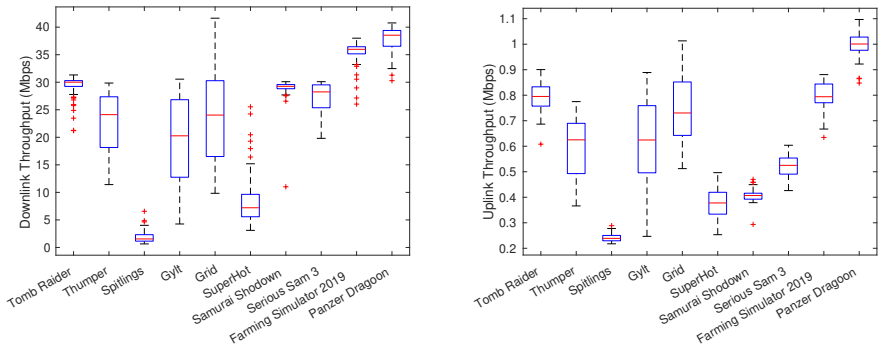
Table 2: Games considered and their respective genre.

shows the lowest, ranging only from 0.645 Mbps to 6.56 Mbps (standard deviation of 1.2). Figure 3b shows the uplink UDP traffic in Mbps. While the boxplots show similar shapes to the downlink ones, the traffic values are much lower, as all games stay below 1.1 Mbps.

Finding: Every Stadia game has different bandwidth requirements, with average traffic going from 1.95 Mbps to 37.7 Mbps depending on the game. In general, most games seem to demand a high amount of traffic, but there are some exceptions such as Spitlings. Uplink traffic is in average 37.63 times lower than downlink traffic, ranging from 0.21 Mbps to 1.1 Mbps depending on the game.

6. Stadia traffic

In this Section, we first determine the traffic share between RTP, DTLS and STUN streams, showing that, as expected, in terms of network utilization, only RTP traffic is relevant. We also evidence the existence of temporal patterns in the generated traffic, as well as the existing correlation between downlink and uplink streams. We finally delve in the packet size, inter-packet time and traffic load characteristics. Starting with this Section we will focus on the three



(a) Downlink traffic

(b) Uplink traffic

Figure 3: Boxplot of the throughput for different games.

main games mentioned in Section 4: Tomb Raider, Thumper and Spitlings.

6.1. RTP, DTLS and STUN streams

We are interested in quantifying the fraction of traffic that belongs to RTP/RTCP, DTLS and STUN streams. We use dataset D1 (Table 1 in Section 4.2). It includes three different games: Tomb Raider, Thumper and Spitlings. While Tomb Raider and Thumper are 3D games, Spitlings is a 2D game. Therefore, we expect Tomb Raider and Thumper will require higher network resources than Spitlings. Note that D1 includes only traces of the games in the "play" state.

Table 3 shows the average packet size, average inter-packet time, and average traffic load of RTP, RTCP, STUN and DTLS streams for the three considered games. It can be seen that most of the traffic corresponds to RTP as it carries the game video and audio contents from the server to the client. Note that while more than 90% of all traffic corresponds to the downlink (91.64% for Spitlings, and 98.13% and 98.14% for Tomb Raider and Thumper), the uplink can take up to 30.81% of all transmitted packets, showing that Stadia requires a consistent stream of short periodic reports to function, even if their

Parameter	Avg. Packet size (bytes)	Avg. inter packet time (ms)	Load (Mbps)
Downlink			
TR RTP	1118.01	0.34	25.60
TH RTP	1154.64	0.49	18.33
SP RTP	677.21	2.81	1.87
TR STUN	81.39	265.23	0.0024
TH STUN	81.50	263.31	0.0024
SP STUN	81.50	264.36	0.0024
TR DTLS	118.59	7.44	0.12
TH DTLS	132.44	10.52	0.097
SP DTLS	137.38	11.31	0.094
Uplink			
TR RTCP	65.99	1.44	0.35
TH RTCP	65.99	1.98	0.26
SP RTCP	113.76	9.84	0.090
TR STUN	79.37	265.13	0.0024
TH STUN	79.25	261.04	0.0023
SP STUN	79.10	264.35	0.0023
TR DTLS	123.17	7.10	0.13
TH DTLS	114.66	9.96	0.089
SP DTLS	119.60	10.62	0.087

Table 3: Traffic characteristics for RTP/RTCP, DTLS and STUN streams. TR: Tomb Raider, TH: Thumper, SP: Spitlings.

overall network utilization seems negligible.

Finding: RTP/RTCP traffic is the only stream of Stadia traffic in which the packet size, inter-packet times, and traffic load depend on the game played. DTLS and STUN traffic are game-independent traffic streams.

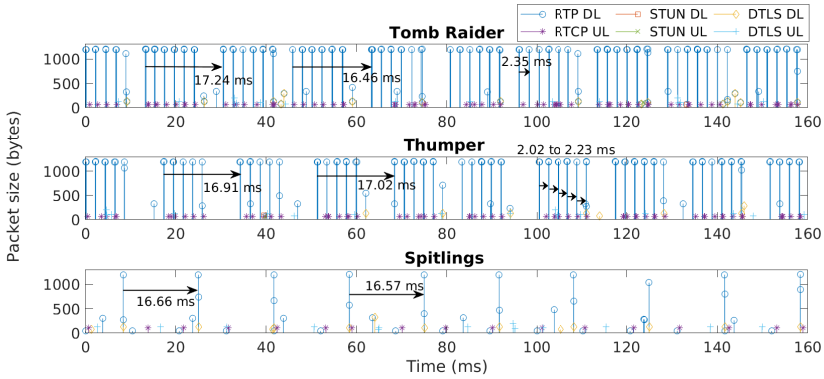


Figure 4: Temporal evolution of Stadia traffic for Tomb Raider, Thumber and Spitlings.

6.2. Temporal patterns

We aim to visualize how Stadia traffic looks like over time. We use dataset D2. This time we focus on a small snapshot of the dataset to visualize the temporal traffic evolution.

Figure 4 shows the traffic evolution of Tomb Raider, Thumper and Spitlings during 160 ms. First, in all three games, and focusing only on the downlink RTP packets, we can observe a clear pattern that repeats every ≈16.67 ms, and that corresponds to the video frame rate of Stadia (i.e., 1/60 fps)[10]. Second, between two consecutive frames, we can observe several groups of packets separated by 2 ms. The number of groups and the number of packets in each group depend on the game. For Tomb Raider, there are 6 groups of 7 to 9 packets each, while for Spitlings, there is only 1 group of 1-2 packets each. Thumper has 6 groups of 5 to 7 packets each. The existence of those groups may be due to the generation of large video frames at the source, which need to be spread among multiple packets. However, since values change for different games, we conjecture this is already implemented at the source. The smaller RTP packets of around 360

[10]The same frame-based pattern can be observed in all other games tested.

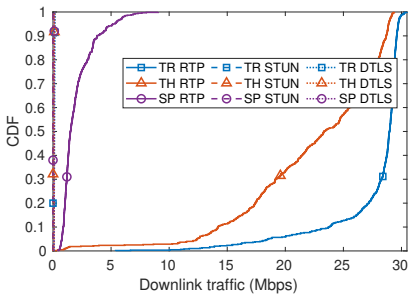
bytes that appear with a periodicity of 20 ms, represent the audio stream, which has a bitrate ≈120 kbps in all three games. We observe the same patterns through all the traffic captures beyond the 160 ms shown in Figure 4, and this consistency allows us to surmise that they are not affected by the transport network. Lastly, the existing correlation between downlink RTP and uplink RTCP streams is clearly observable.

Finding: The temporal evolution of RTP/RTCP traffic follows a well-defined pattern. Inside each frame period, RTP packets are sent in groups. The number of groups and the size of each group in number of packets depend on the game.

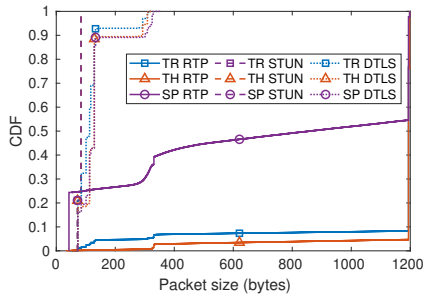
6.3. Traffic load, packet sizes and inter-packet times

We have seen that the temporal structure of Stadia traffic follows a clear periodic pattern. Here, we aim to further validate previous results by showing the probability distribution of the packet size, the inter-packet delay, and traffic load. We use the complete dataset D2 that covers 10 minutes of gameplay for each game.

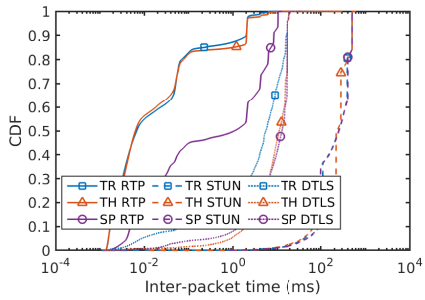
Figure 5a shows the ecdf of the traffic load for each game and protocol, where we can observe that both STUN and DTLS traffic represent a small fraction of the total traffic generated by Stadia, with a maximum of 3.2 Kbps and 159.9 Kbps, respectively. As we have seen before, RTP represents most of the traffic, with Tomb Raider and Thumper generating, respectively, 28.97 and 23.32 Mbps at the 50th percentile. Spitlings generates much lower traffic loads, with only 1.5 Mbps at the same percentile. Figure 5b shows the ecdf of the packet size. For Tomb Raider and Thumper, more than 90% of the RTP packets are equal or larger than 1194 bytes, while for Spitlings these larger packets only represent the 45.42%. For STUN and DTLS, we observe they transmit very small packets. The average packet size of STUN is 81.47 bytes for Tomb Raider, 81.48 bytes for Thumper and 81.47 bytes for Spitlings. For DTLS, their average packet size is 114.47, 133.36 and 129.24 bytes respectively. Finally,



(a) Downlink traffic



(b) Packet size



(c) Inter-packet time

Figure 5: cdfs for each RTP, DTLS and STUN in the downlink, for TR, TH and SP.

Figure 5c shows the inter-packet time for each game and protocol. As expected, inter-packet times are inversely proportional to the traffic load, with Tomb Raider and Thumper showing that, respectively, the 87.33% and 85.07% of their RTP packets have an inter-packet time below 1 ms. Spitlings has higher inter-packet times in general, with only 49.72% of them below 1 ms.

We can also observe that Spitlings seems to have consistent low traffic, with occasional peaks, while Tomb Raider is the opposite. It generates a high traffic load in general, with occasional dips.

Finding: In this section, by comparing the different probability distributions, we further confirm that DTLS and STUN traffic are almost

identical regardless of the game. With respect to the RTP/RTCP traffic, similarly, we also confirm that the video traffic generation process is common in all three games, just adapting for each game the number of groups of packets per frame, and the number of packets inside each group. Although this paper does not focus on traffic modelling, these results open the door to develop a general but parameterizable traffic model for Stadia traffic.

7. Inside a game

In the previous section we showed that the traffic generated by Stadia in the "play" state depends on the game. However, in addition to the "play" state, a game has other states, such as the "menu", "idle", and "pause". Here, we investigate how is the traffic generated in each state of a game.

7.1. Different game states, different traffic loads

In this section we check how the traffic load generated by Stadia changes based on the different states (or types of screens) that we can find in a game. We also study how user input changes the load perceived.

We use dataset D3, that includes traces from Tomb Raider and Spitlings. We omit Thumper, as we have seen before that it behaves similar to Tomb Raider. We play both games at 1080p using the VP9 codec, and change the game state every 120 seconds. We consider a state to be a "screen" that shows unique characteristics in both gameplay and video needs. For example, the main menu is text based, with an animated background, and the player just selects items from a list. This is different than the game itself, in which there is a changing environment, with many active elements on the screen, and multiple possible actions for the player. We have identified the following states:

1. **Main Menu:** A screen with several text items (e.g., new game, continue, options, etc.), and a dynamic background (i.e., an of-

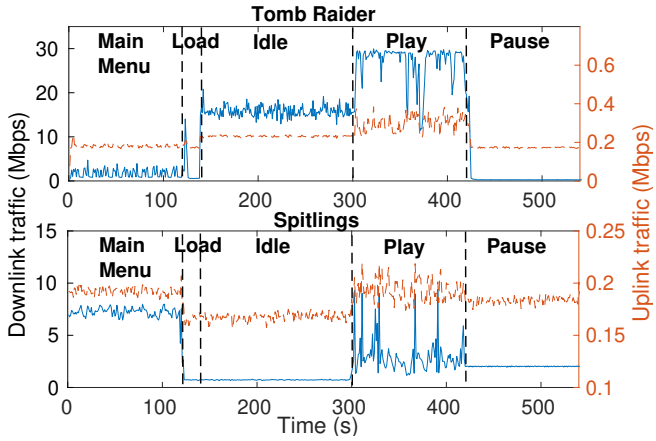


Figure 6: Traffic evolution over 4 states.

fice with lightning in the windows for Tomb Raider, and several characters moving around for Spitlings).

2. **Loading screen:** A black screen with some text that appears while a level is loading.
3. **Idle:** Inside of the game state, but with the player not performing any actions.
4. **Play:** The player interacts with the environment by taking actions.
5. **Pause:** The pause menu is another text menu, overlayed on top of the frozen playing screen.

Figure 6 shows the traffic evolution for the different states of Tomb Raider and Spitlings, confirming that the traffic load depends on the current state. Note that the "play" state is the only one where the player is pressing buttons frequently. If we compare the "idle" and 'play' states, we can observe that there is a significant difference when the player interacts with the environment in terms of the traffic load. This is a result of the player's actions changing the information presented on screen, i.e., when "idle" or in a 'menu', most of the elements on screen are static, and require no extra data. However,

during playtime, new data needs to be sent to the user constantly. The lowest traffic load is found during the loading screen, as it is only a black background with a couple of lines of text. We can also observe that the traffic required per state is not similar across the games. Tomb Raider has a much higher throughput for the "Idle" state than for the "Menu", while the opposite is true for Spitlings. We also find that the variance in each state is not related to the average throughput. For example, in the 'main menu' of Tomb Raider we have an average traffic of 1.73 Mbps and a standard deviation of 0.96 Mbps, while in the "idle" state, which has a much higher average throughput of 15.79 Mbps, the standard deviation is only of 1.24 Mbps.

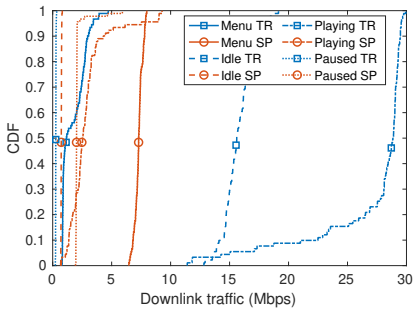
Finding: Each game state has different traffic characteristics. Depending on the game, it could be the case that the "play" state may not be the one with the highest traffic load, as it could be initially expected. While we expected some variance between states, some games can have more than 20 Mbps of difference between them, which could have quite an impact on a network's performance.

7.2. Variability of the traffic load

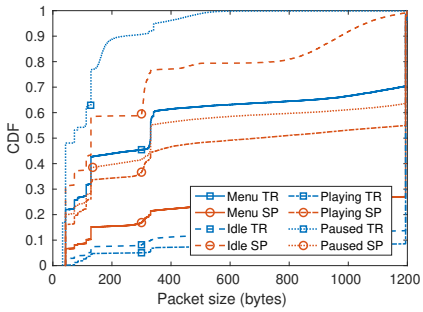
This section explores the traffic variability in each of the different game states. A higher variability should be expected in the "play" state than in the other states due to the expected interaction with the user.

We use the same dataset as in the previous section (D3). For each game state of Tomb Raider and Spitlings we calculate the empirical cumulative density function using 90 seconds of data (we use the "middle" section of each state, as to avoid interference from other states, such as the initial spikes in traffic when changing between states). We consider only the downlink RTP traffic, which includes the video and audio contents.

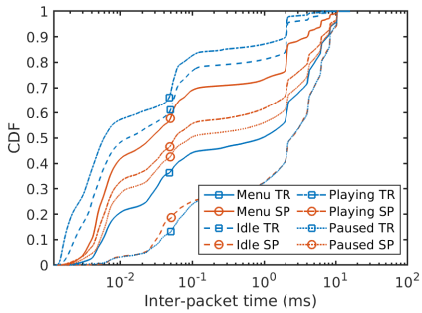
Figure 7a shows the ecdf for each state, where we observe that the "play" and "idle" states of Tomb Raider result in the highest traffic



(a) Traffic load



(b) Packet size

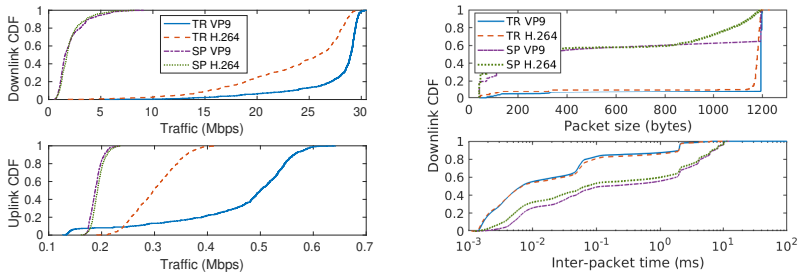


(c) Inter-packet time

Figure 7: cdfs for different games, and different game states inside a game.

load by far, reaching almost 30 Mbps and 20 Mbps, respectively. For Spitlings, as mentioned in previous section, the highest load appears in the main menu, which stays below 8 Mbps. All remaining states have relatively low loads, mostly staying below 5 Mbps.

Most states show little variability: the "pause" state has a standard deviation of 0.02 Mbps for Tomb Raider and 0.49 Mbps for Spitlings; the main menu has 0.98 Mbps and 0.34 Mbps respectively. The "idle" state has a deviation of 1.26 Mbps for Tomb Raider and 0.026 Mbps for Spitlings. The highest deviation appears in the "play" state for both games, with 4.30 Mbps for Tomb Raider, and 1.72 Mbps for Spitlings, showing that the player actions have a clear impact on the generated traffic.



(a) Traffic characteristics for H.264 and VP9 video codecs

(b) Packet size and inter-packet CDF for downlink traffic

Figure 8: Video codec characteristics.

Finding: The results confirm that states with high user interactions show the highest variance in their traffic, showing that the player actions have an impact on the generated traffic.

8. Video Codecs and Resolution

This section investigates the effect of the video codec and the video resolution on the traffic generated by Stadia.

8.1. Codecs

Google Stadia supports two different types of video encoding: VP9 and H.264. This section aims to identify if the use of different video codecs affects the generated Stadia traffic. The same parts of Tomb Raider and Spitlings ("play" state) are played twice, one for each codec. Dataset D4 is used in this section.

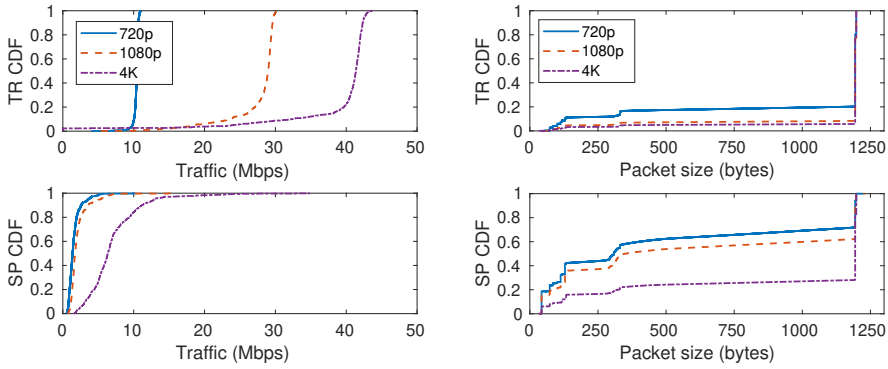
Figure 8a shows the ecdf of the traffic load for both codecs. While Spitlings shows almost no differences between encodings, playing Tomb Raider using the H.264 codec represents less traffic. In average, the traffic load required for H.264 is 23.63 Mbps for Tomb Raider and 2.07 Mbps for Spitlings, and for VP9 it is 27.56 Mbps and 2.10 Mbps respectively.

To dig a bit more on the differences between VP9 and H.264, we plot the downlink ecdf of the packet size and inter-packet time in Figure 8b. The average packet size using VP9 in the downlink is 1116.12 bytes and 530.04 bytes for Tomb Raider and Spitlings, respectively (in the uplink, the average is 75.71 bytes and 116.73 bytes). When the H.264 codec is used, the average packet size of downlink packets is 1064.10 bytes and 480.34 bytes for Tomb Raider and Spitlings, with 123.83 and 118.24 bytes for the uplink. In general, VP9 attempts to use the same packet size whenever possible, which we can observe in the VP9 downlink of Tomb Raider, where 68.79% of the packets are 1194 bytes long. Differently, for H.264, the most common packet size is 1183 bytes long, with 3.58% of all packets being that size. Regarding inter-packet times, it can be observed that even if there are differences in the traffic load and packet size distributions, the two codecs have almost the same inter-packet time ecdf, which means that the traffic generation process is independent on the codec used.

Finding: In our experiments, the use of H.264 codec has resulted in lower traffic loads than using VP9 for Tomb Raider. This is an unexpected result since VP9 is on paper a more advanced codec, and H.264 is supposed to be kept just for compatibility across all devices. Also, we can observe that the traffic generation process is exactly the same for both codecs, only slightly changing the packet size distribution, which in turn, results in different traffic loads. Although fully subjective, we did not experience any difference in terms of quality between the two video codecs.

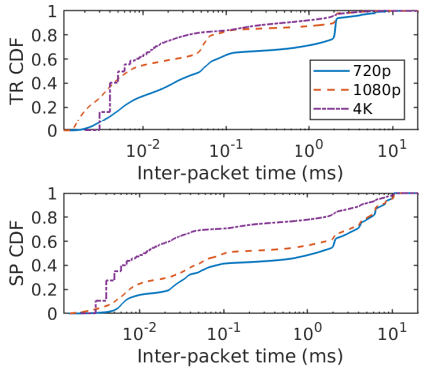
8.2. Resolution

Stadia offers 3 different resolutions: 1280x720 (720p), 1920x1080 (1080p) and 3840x2160 (4K). Stadia recommends a minimum connection of 10 Mbps, 28 Mbps, and 35 Mbps to enjoy their service at 720p, 1080p, and 4K, respectively [43]. This section explores the relationship between the resolution and the traffic load. We also aim to validate if those capacity recommendations are accurate in practice.



(a) Traffic load

(b) Packet size



(c) Inter-arrival time

Figure 9: Impact of the resolution on metrics.

This section uses dataset D5, in which we play the same section of each game 3 times, one for each resolution. We use VP9 in all cases, and play the games for 600 seconds.

Figure 9a shows the ecdf of the Stadia traffic in the downlink for Tomb Raider and Spitlings. For Tomb Raider, we can observe that the ecdf for the 720p and 1080p resolutions closely follows the recommended link capacity. However, for the 4K resolution, the traffic load is higher than 35 Mbps for 88% of the time, reaching up to

43.74 Mbps. For Spitlings, increasing the resolution from 720p to 1080p does not represent a large increase of the traffic generated, while for 4K, the increase is significant. For instance, considering the 95th percentile, the traffic load from 720p to 1080p increases in a 47.63% (opposed to the 177.00% for Tomb Raider), while from 1080p to 4K increases in a 144.15% (only 43.09% for Tomb Raider).

We have seen that using different resolutions results in a change on the traffic generated by Stadia. Different traffic loads are obtained by changing both the packet size distribution (Figure 9b) and the inter-packet time distribution (Figure 9c). First, regarding the packet size distribution, we can observe that reducing the resolution results in transmitting less packets over 1000 bytes. The number of packet groups inside each video frame period, as well as the number of packets in each group is also reduced. Second, as it could be expected, transmitting less packets affects also the inter-packet time distribution. However, we can observe that in spite of those differences, the shape of the ecdf is similar for all three resolutions, meaning that the traffic generation process follows the same general pattern regardless the resolution employed.

Finding: Supporting a resolution of 4K (i.e., up to 43 Mbps) requires a network capacity almost 4 times higher than for a resolution of 720p (which can go up to 11 Mbps). Moreover, the suggested link capacity values for each resolution are surpassed in all three cases. Changing the resolution affects the number of video packets generated, although the general packet generation process is unaffected. Subjectively, the use of higher resolutions is clearly observed in the quality of the image.

9. How Stadia adapts to the available bandwidth

This section studies the adaptability of Stadia to changing network conditions. We limit the available bandwidth at the client, and observe how performance is affected. We consider two cases: a) the

game starts with the bandwidth limit already in place, and therefore, Stadia knows before starting the game the effective link capacity, and b) the available link capacity suddenly changes in the middle of the game, enabling us to observe how Stadia reacts.

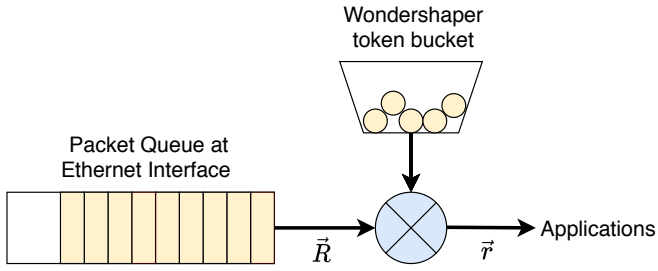


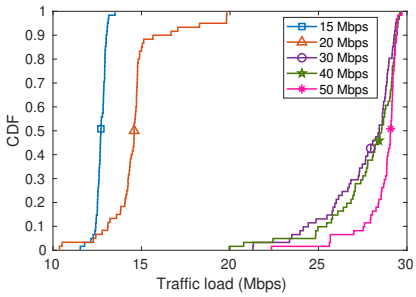
Figure 10: Wondershaper operation: a token bucket allows new packets to get processed at a specified rate.

9.1. Different initial available bandwidths

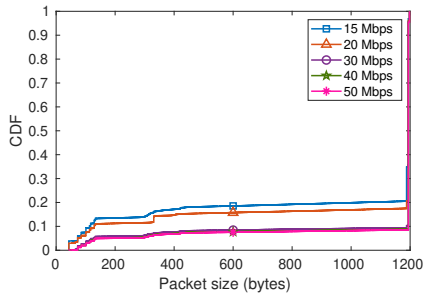
We start by applying a limit to the available bandwidth at the receiver before the game starts. Using different bandwidth limits will allow us to understand how Stadia deals with different link capacities.

To limit the available bandwidth, we use Wondershaper[11], a tool that uses Ubuntu's traffic control capabilities to limit incoming traffic. Wondershaper is installed on the receiving laptop, where it creates a virtual interface that receives incoming traffic and sends it to the physical interface following our specification. We use limits $\vec{r}$ of 5, 10, 15, 20, 30, 40 and 50 Mbps. For each limit, we play the same section of Tomb Raider for 60 seconds at 1080p with VP9. Figure 10 shows the operation of Wondershaper, and how it uses a token bucket to limit the network interface bandwidth $\vec{R}$ to the newly imposed $\vec{r}$. In this section we use dataset D6.

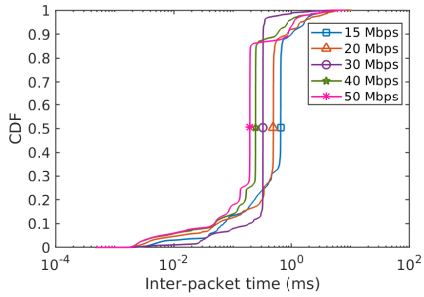
[11]Wondershaper: `https://github.com/magnific0/wondershaper`



(a) Downlink traffic CDF



(b) Packet size



(c) Inter-packet time

Figure 11: Different link bandwidths.

Figure 11a shows the ecdf of the traffic load for each bandwidth limit. We can observe that Wondershaper guarantees that the traffic streams will use a bandwidth lower than the limit in average. For example, for bandwidths limits of 15 Mbps and 20 Mbps, Stadia generates a mean traffic load of 12.71 Mbps and 14.64 Mbps, respectively. For 30, 40, and 50 Mbps, the traffic generated by Stadia is almost the same in all the three cases. The mean traffic load values are 27.54 Mbps, 27.80 Mbps and 28.66 Mbps for each bandwidth limit, respectively. In addition, we can also observe that the highest traffic peak is never higher than 30 Mbps, even when bandwidth limits of 40 and 50 Mbps are in use.

Figure 11b shows that the packet size distribution is almost identical in all cases regardless the imposed bandwidth limit. For 15 and 20

Limit	Avg. packet size (bytes)	STDEV	Min	Max	3 most common
15 Mbps	995.61	405.42	43	1198	1194 (57.45%), 1188 (13.38%), 43 (3.66%)
20 Mbps	1026.61	379.14	43	1198	1194 (53.10%), 1198 (2.75%), 1188 (2.15%)
30 Mbps	1105.50	285.92	43	1198	1194 (78.78%), 1198 (4.14%), 73 (1.22%)
40 Mbps	1113.25	272.86	43	1198	1194 (80.34%), 1198 (4.22%), 73 (0.96%)
50 Mbps	1115.21	269.84	43	1198	1194 (83.78%), 1198 (4.40%), 73 (1.01%)

Table 4: Packet characteristics.

Mbps, there is a higher amount of small packets, but overall, the results are very similar to the cases of 30, 40 and 50 Mbps. Table 4 shows the average packet size and standard deviation for each case. We can observe how the average packet size increases along with the bandwidth limit, but the standard deviation decreases, as most of the packets are of the same size. We also show the most common packet sizes, where we can see the clear preference for packets of 1194 bytes no matter the bandwidth limit used.

Similarly to the packet size, the distribution of inter-packet times (Figure 11c) shows an identical tendency in all cases, with the highest bandwidth limits leading to slightly lower inter-packet times. Stadia already sustained 1080p with the 20 Mbps limit, only switching to 720p for a link capacity of 15 Mbps. This leads us to conjecture that Stadia can further stress the video encoding to reduce the traffic load, changing the resolution only as a last resort. We also used limits of 5 Mbps and 10 Mbps, but these low bandwidth values lead Stadia to stop the game shortly before starting, showing a message informing that the network was unsuited for the service.

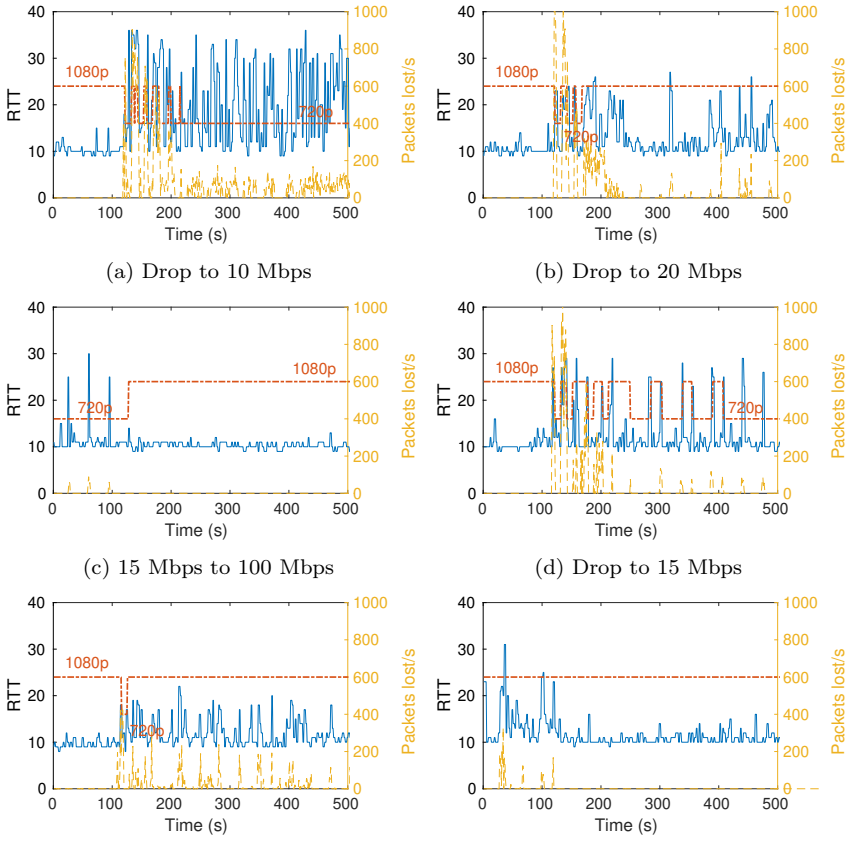
Finding: Stadia adapts to the available bandwidth seamlessly when the limitation is set before the game starts. We have found that Stadia strives to keep the 1080p resolution at 60 fps even if the available bandwidth is far below its own pre-defined requirements, and only switches to a lower resolution of 720p as the last resort. In this regard, subjectively speaking, the more compressed 1080p streams were still preferable than the 720p streams, justifying Stadia's behavior.

9.2. Sudden changes on the available bandwidth
Here, we test the ability of Stadia to adapt to a sudden change in the available bandwidth. Limiting the bandwidth during the gameplay, we aim to investigate how Stadia responds to congestion in real time.

This section uses dataset D7. We start Wondershaper in the background after 120 seconds of playing without any bandwidth limit (i.e., using the default link capacity of 100 Mbps). We first consider bandwidth drops to 10 Mbps, 15 Mbps, 20 Mbps and 30 Mbps. After that, we do the opposite, we start Stadia under a bandwidth limit (as in the previous section), to remove it after 120 seconds. To showcase the impact of these changes, we use the metrics obtained from `chrome://webrtc-internals/`, such as the RTT (calculated as the time between the transmission of STUN packets and the arrival of the response), video packet losses and the rate of successful delivered video frames to the application. The use of Chrome's statistics to quantify QoS was studied in [16].

Figure 12 shows the RTT, video packets lost and video resolution of the stream over time for each of the bandwidth limits. We can observe that once the bandwidth decreases, Stadia changes resolution repeatedly for a while. Once the client reports that packets are being lost, Stadia drops to the lowest resolution. Then, after a short time, Stadia attempts to use a new encoding configuration with a higher resolution. If the new configuration still results in dropping packets, Stadia repeats the process again until a viable configuration is found.

A change in the available bandwidth leads to transitory periods of variable duration during which Stadia attempts to recover by finding a viable configuration. The bandwidth drop to 10 Mbps results in Stadia changing resolutions 12 times during 95 seconds before remaining at 720p until the end. It is worth to mention here that this game session was completed successfully, while in the previous section, starting with the limit of 10 Mbps already in place resulted in Stadia deciding to close the game. The bandwidth drop to 15 Mbps in Figure 12d leads to the longest transitory period, spanning



(a) Drop to 10 Mbps

(b) Drop to 20 Mbps

(c) 15 Mbps to 100 Mbps

(d) Drop to 15 Mbps

(e) Drop to 30 Mbps

(f) 20 to 100 Mbps

Figure 12: Round Trip Time (continuous line), video packets lost (dashed line) and resolution (dash-dotted line).

287 seconds before Stadia stops changing resolutions and remains at 720p. Dropping the available bandwidth to 20 Mbps results in a comparatively quick reconfiguration of 47 seconds, and dropping it to 30 Mbps results in a single switch to 720p for 11 seconds before returning to 1080p.

When we start Stadia with a bandwidth limit and then remove it, the time to find a new viable configuration is much faster, as it could be expected. For the case where the initial limit was 15 Mbps, Stadia just jumps to 1080p. In the case that the initial bandwidth limit was 20 Mbps, Stadia never changed the resolution, as we started at 1080p already, but the generated traffic load does increase, going from 17.75 Mbps before removing the limit, to 24.71 Mbps afterwards, further confirming the existence of different coding settings for the same resolution.

The RTT increases rapidly when the bandwidth limits are enforced, going from an average of 10 ms for most captures to 19.52 ms for the bandwidth drop to 10 Mbps. The variance in the RTT is higher for the lower bandwidth limits, as we can observe peaks of more than 35 ms in Figure 12a, while we only reach 22 ms for 30 Mbps in Figure 12e. We can also observe an increase in the packets lost when the bandwidth limits are enforced, reaching as far as 1017 packets lost in a single second in Figure 12b. We can observe that the packets lost decrease during the transitory period in Figures 12a and 12d, stabilizing afterwards. For the two cases in which the bandwidth limit is removed, we observe that the RTT stabilizes soon at 10 ms, and Stadia stops dropping packets, showing the opposite tendency to the previous cases.

Figure 13 shows the video frames per second decoded by the browser over time for the bandwidth drops to 15 and 20 Mbps. Correct playback should lead to a stable 60 fps. However, we can observe that during the transitory period the framerate varies strongly, dropping as low as 10 fps for 15 Mbps, and 5 fps for 20 Mbps. Combined with the changes in the resolution (see Figure 9a for the impact of resolu-

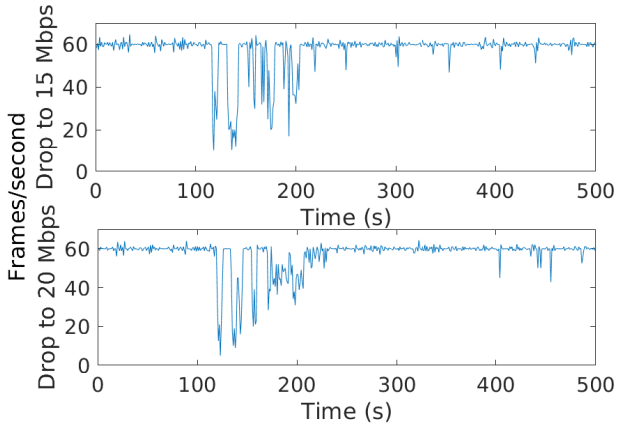


Figure 13: Framerate over time.

tion on traffic), the reduction on the framerate results in a noticeable loss of quality for the user. While the resolution has an effect on the quality of the image, the impact of the framerate is on the smoothness of the video reproduction. Frame drops such as these found in Figure 13 result in severe delays between the time the player presses a button and the time the corresponding action appears in the screen, as well as in parts of the video being skipped, resulting in characters "teleporting" to another place due to the missing frames. Finally, let us mention that in the previous case, when Stadia started with a bandwidth limit already in place, the framerate remains stable for the entire capture, so the user notices the low bandwidth availability only in the image resolution.

Finding: Stadia attempts to recover from a drop in the available link capacity almost immediately, entering a transient phase in which Stadia aims to find a new configuration to compensate the lack of network resources. This transient phase however, can last over 200 seconds. During this time, although the user is able to continue playing, the quality of experience is heavily affected, with constant resolution changes, inconsistent framerate and even audio stuttering.

In the case that the available bandwidth increases, Stadia also reacts immediately, easily finding a new viable configuration.

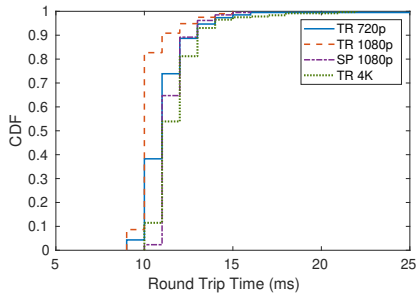
10. Latency related to bandwidth changes

In most of previous experiments we have intentionally avoided any reference to the end-to-end latency, so we can focus on it in this and the following sections. In this one, we analyse the latency of our previous experiments, i.e., latency under normal conditions and then with sudden drops in link capacity. We use the RTT and jitter buffer delay metrics from WebRTC internals (`chrome://webrtc-internals`). The RTT is computed as the total time elapsed between the most recent STUN request and its response, and it is reported every second. It shows how long it takes for an action to obtain a response. The jitter buffer delay represents the amount of time RTP packets are further buffered at the client side to guarantee a smooth data delivery to the user (i.e., fixing packet order, since UDP does not offer such control by itself). Note that the jitter buffer delay is adaptive and so it may change with time.

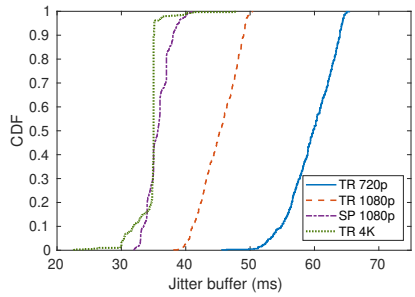
High latency can have a negative impact on player enjoyment. After pressing a button, players expect that the effects of the corresponding action will appear instantaneously on the screen. If it takes too long, it can make the game unenjoyable, and in some cases, fully disrupt the gameplay. Here, we compare the RTT and jitter buffer delay metrics for different games and resolutions.

Dataset D8 presents the WebRTC internals that correspond to the same traffic captures as in Sections 8.2 and 9.1.

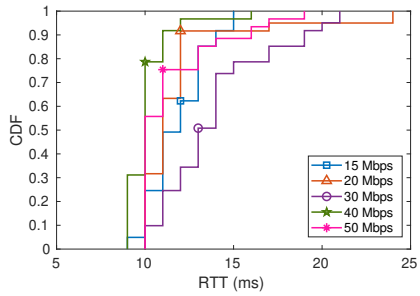
Figure 14a shows the ecdf of the RTT for both Tomb Raider and Spitlings during the "play" state at 1080p, as well as the RTT for Tomb Raider at 720p and 4K. All of them show similar RTT values, averaging between 10.28 ms and 12.30 ms. In all cases, the 95 % percentile of the RTT values is lower than the duration of a single video frame (16.67 ms), meaning that in these tests Stadia had the



(a) RTT for each game



(b) Jitter buffer delay



(c) RTT when bandwidth is limited

Figure 14: Round Trip Time and jitter buffer delay for different configurations.

opportunity to interact with the player's actions without any perceptible delay. As expected, since the available link capacity was large enough, the game and resolution have no impact on the perceived latency. Only when the traffic load is close to the network capacity, such as in the previous section, the RTT is affected.

Figure 14b shows the empirical ecdf of the jitter buffer delay for Tomb Raider and Spitlings. For a resolution of 1080p, Spitlings shows a lower jitter buffer than Tomb Raider, with an average of 35.76 ms and 45.35 ms, respectively, which corresponds to 2-3 video frames at a framerate of 60 fps. Considering Tomb Raider and different resolutions, we observe that the jitter buffer decreases as the resolution increases. For Tomb Raider the jitter buffer averages 58.42 ms, 45.34 ms and 35.35 ms for 720p, 1080p and 4K, respectively. This is an

interesting result that shows the buffer jitter delay is directly related to the inter-packet arrival time (i.e., lower inter-packet arrival times also result in lower jiter buffer values, and the opposite is also true).

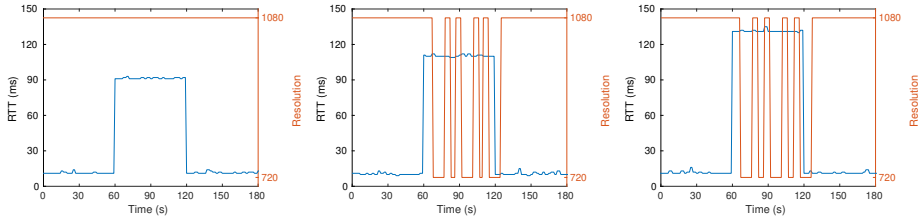
Figure 14c shows the RTT for the different initial bandwidth limits considered in Section 9.1. We can observe that the RTT is quite similar to the ones on Figure 14a, showing that our bandwidth limitations do not have an impact on the RTT, and only affect the throughput of the network.

Finding: In all the considered cases for different games, resolutions, and available bandwidth limits, RTT values have consistently been below 25 ms, with average values between 10 and 15 ms. This is especially relevant since all the experiments and measurements have been done in a temporal span of several months from March to July 2020, hence, meaning that in absence of network congestion, RTTs are extremely stable and clearly below 60 ms (i.e., the value at which users start noticing the latency issues [41]). In terms of the jitter buffer, we have found that it is higher for lower resolutions, as it directly depends on the amount of traffic received at the client.

11. How Stadia reacts to latency changes

In this section we directly modify the latency in our network to investigate how Stadia traffic adapts to changes in the delay between packets. We use the Ubuntu Traffic Control (tc) package to change the latency of both incoming and outgoing packets in our PC, and much like before, we use the WebRTC internals metrics from Chrome to assess how latency changes impact the stream performance.

We start playing the game without any added latency. Then, after 60 seconds, we add the same extra latency value to both incoming and outgoing packets from the client. We add 40 ms, 50 ms and 60 ms to each link (i.e., a perceived addition of 80 ms, 100 ms and 120 ms to the Round Trip Time). Then, after 60 seconds of high latency, it is removed for another 60 seconds.



(a) RTT increase of 80 ms (b) RTT increase of 100 ms (c) RTT increase of 120 ms

Figure 15: Round Trip Time and video resolution over time.

Figure 15 shows the RTT of Stadia traffic over time for Tomb Raider, as well as the video resolution. It can be observed that for an extra RTT latency of 80 ms the video resolution does not change. However, when we add 100 and 120 ms, the video resolution starts to go up and down, showing that Stadia reacts to the perceived congestion by trying different video configurations.

Google Congestion Control works based on the bitrate, packet loss and delay of transmissions as explained in Section 3.3. Increasing the latency does not cause packet losses, and so changes in the configuration come purely from the delay-based controller. Once the latency goes back down, after 120 s, the system returns to the initial configuration quickly. In all cases, Stadia shows a message warning the user that their network is unstable after around 40 seconds have passed since the increase of latency. When the latency added is higher than 80 ms, if it is kept for more than 60 seconds, the system would sometimes shut down on its own, giving the user another message explaining the situation. Gameplay continued uninterrupted for the duration of the experiment, with additional lag being noticeable, but not unmanageable. Framerate was consistent around 60 fps through the entire time, with few video frames being dropped and only occasional stuttering. As mentioned before, Chrome reported no packets lost either. In terms of player experience, the changes in bandwidth were a much bigger issue than the latency ones, as long as Stadia does not terminate the session by itself.

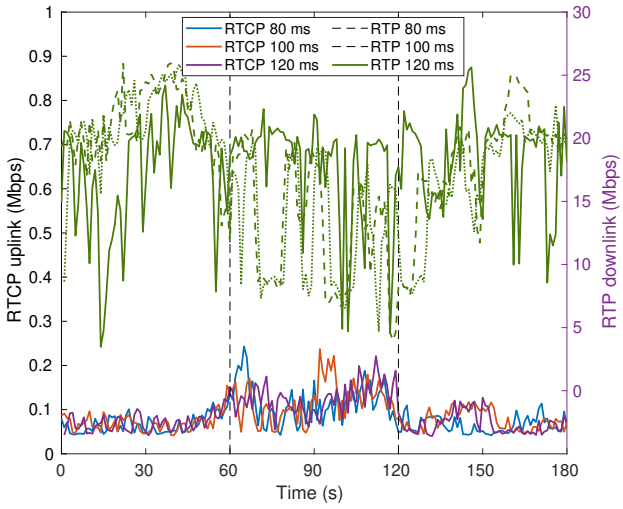


Figure 16: Impact of latency changes on the throughput.

Figure 16 shows the impact of the latency on the data exchanged with the Stadia server. An increase in uplink data can be observed during the period going from 60 seconds to 120 seconds (i.e., when the latency is increased), which then is reduced when latency returns to normal. We can also observe a decrease in downlink RTP video for all cases on the high latency period, including the case of 80 ms in which resolution did not change. In the 80 ms case, it seems that the changes to the configuration are less aggressive, confirming that much like with bandwidth, changes in Stadia configuration are not just based on a binary overloaded/underloaded approach and also continuous adjustments to the video codec are applied based on the gathered feedback.

Finding: Stadia reacts to latency changes quickly, increasing the client-side reports and adapting the video configuration (including video resolution) to reduce downlink traffic. Warnings are given to the client if the RTT is considered too high. After a few minutes of high latency (over 70 ms RTT) the connection is considered unsus-

tainable, and the game is stopped.

12. Is Stadia traffic similar to other WebRTC applications?

Stadia traffic is comprised mostly of video, and so we want to study which parts of the traffic are unique to Stadia and which are just part of a standard video application. As Stadia uses WebRTC, we will also compare its traffic to some video conferencing applications that use it, such as Google meet.

For this section we perform captures with other video services. We use the same setup we used for Stadia, but with Google meet[12] and Jitsi[13] and a remote caller. Both of these applications are based on WebRTC, so we expect their traffic to have some similarities to that of Stadia. For both applications, both users use the highest video settings available, which means that the streams are 720p. The video codec used by Google Meet is VP9, just like Stadia. Jitsi however used VP8 in our captures.

To compare Stadia with the conferencing apps, we use a 720p capture of Tomb Raider so that all captures have the same resolution. In terms of framerate, Stadia is the only one that uses 60 fps, while Google meet had an average of 24 fps, and Jitsi used 15 fps [14]. Figure 17 shows the downlink traffic patterns of the three WebRTC applications we use over a period of 500 ms. For Stadia we can find the same patterns we described in Section 6, mainly that RTP packets arrive in batches, separated by around 16.67 ms, which is the time between two video frames at 60 fps. This is the main part of the traffic, with STUN and DTLS packets happening less frequently, and with small sizes. For Jitsi we find that the patterns are very similar,

[12] `https://meet.google.com/`

[13] `https://meet.jit.si/`

[14] Jitsi can go up to 30 fps depending on settings, but in our HD captures it settled around 15-20 fps.

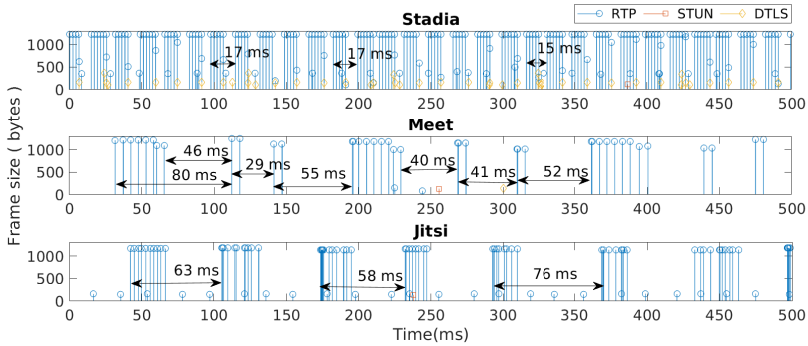


Figure 17: Temporal patterns of downlink traffic for WebRTC applications.

with RTP packets arriving in batches, but with much larger intervals. A framerate of 15 fps would result in video frames lasting an average of 66.7 ms, which is consistent with the timing we observe. For Google meet however the RTP patterns are a bit different, where we can observe a large batch of 6 groups of packets, with two extra packets at the end that are smaller in size, followed by two small batches of 2 groups of packets. The timing for an average of 24 fps would be 41.7 ms, and we cannot identify when video frames start as easily as with the other two applications. The time between the bigger batches and the smaller ones is quite large at 80 ms, but the time between the last packet of the batch and the first of the next one is 46 ms, which fits much better with the framerate. The timing between the smaller batches also seems to fit closer to the average of 41.7 ms. Overall, while there are some differences between the traffic shape of meet and the other apps, the underlying patterns are quite similar, with large RTP packets sent in batches with intervals related to the video framerate.

Table 5 summarizes other traffic parameters such as average packet size, inter packet time and traffic load. Average packet size is consistent across the three applications, as RTP traffic has the largest packets, while STUN and DTLS have much smaller ones. As a conse-

Parameter	Avg. Packet size (bytes)	Avg. inter packet time (ms)	Load (Mbps)
TR RTP	1159.9	0.88	10.54
Meet RTP	1061.3	5.28	1.63
Jitsi RTP	912.27	4.40	1.66
TR STUN	115.38	263.05	0.0035
Meet STUN	134.00	2502.9	0.00042
Jitsi STUN	118	1242.08	0.00078
TR DTLS	148.82	6.38	0.19
Meet DTLS	135	17479.01	0.000070
jitsi DTLS	N/A	N/A	N/A

Table 5: Traffic characteristics for RTP/RTCP, DTLS and STUN streams of WebRTC applications.

quence of the lower framerates, we can observe that the average inter packet time is much larger for Meet and Jitsi, both 5 times higher than that of Stadia. In terms of RTP load we can clearly observe that Meet and Jitsi are incredibly close, both at 1.6 Mbps, while Stadia requires a much larger load of 10.54 Mbps.

Both STUN and DTLS protocols show clear differences in their traffic patterns. For STUN, the inter packet times for Meet and Jitsi are over 9 and 4 times larger than Stadia respectively. Both conferencing apps use very infrequent packets for STUN, and this is also true for the DTLS traffic. For Stadia, we find DTLS packets every 6.38 ms, while for meet we find them every 17.4 seconds. This is the biggest difference, in that Stadia has frequent application data transmitted using DTLS, while Meet sends packets very infrequently, and for Jitsi, we only have the initial handshake of the call setting up DTLS encryption. After that, no application data is sent using DTLS. This application data for Stadia and Meet may be reports containing statistics, or information pertaining to the users involved. Which would explain its absence in Jitsi, as it does not require reg-

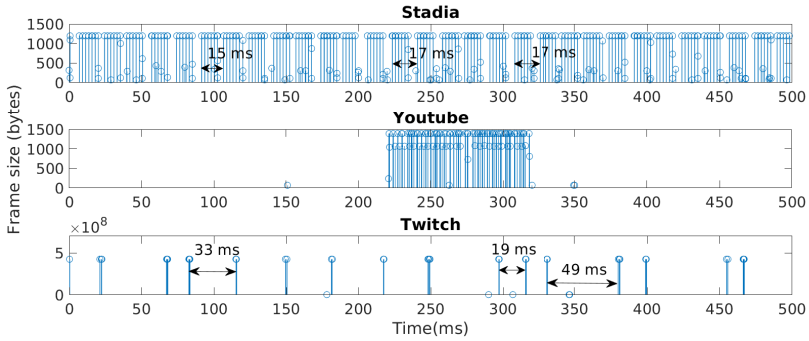


Figure 18: Temporal patterns of downlink traffic for high definition live video applications.

istering as a user to be used.

For the three applications, RTP is the main type of traffic, and it is clear that the RTP patterns are tied to the video that is being transmitted. In this regard, the main difference seems to be in terms of volume, as the load for Stadia traffic is much higher than that of the conferencing apps. The higher framerate of Stadia seems to be a main reason for the higher traffic, but this is not so clear, as the decrease in video frames (60% and 75% for Meet and Jitsi) is not met with an equivalent decrease in traffic load (85% for both). Meet and Jitsi also use the same traffic load despite the fact that they operated with very different framerates, which could be a consequence of their use of different video codecs (VP9 for Meet and VP8 for Jitsi).

To further test other video applications, we check two live streaming websites that offer live video at 1080p and 60 fps: Youtube and Twitch. We perform two captures of live streams that we can compare to our 1080p captures of Tomb Raider and check their similarities. However, neither of these two services use WebRTC. Youtube uses UDP over QUIC, and Twitch uses TCP. Figure 18 shows the traffic patterns of all three applications, where we can observe that they operate very differently. Youtube for instance sends packets in

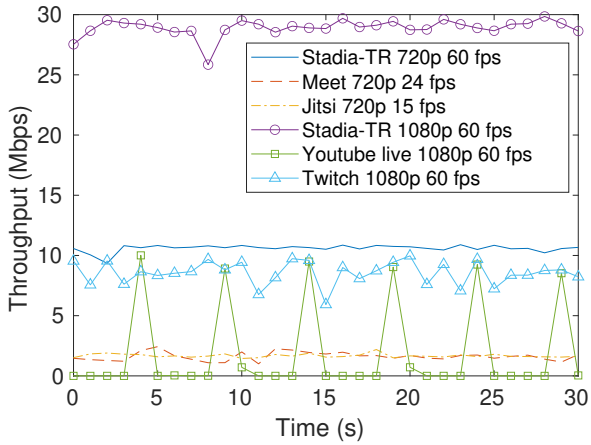


Figure 19: Throughput over 30 seconds for live streaming video applications used.

batches every 5 seconds, which clearly shows that even though the streams are live, they do some buffering at the transmitter (adding a small latency), and in the end they operate as any other Youtube video. For Twitch the same seems to be true, but packet batches appear much more frequently than for Youtube and they have very varied timing between batches. While live streams usually have chat interaction between content creator and viewers, this interaction does not need such strict responses as Stadia, and so both applications can use a margin of time to buffer the stream and solve typical UDP issues, such as missing packets or arrivals out of order.

Finally, Figure 19 shows the video throughput (RTP, UDP and TCP) over 30 seconds for all the captures in this section. Here we can observe that both Twitch and Youtube, even when using 60 fps, have a much lower load than Stadia. Both live streaming applications use 10 Mbps at 1080p, close to that of a 720p Stadia stream, but less than half of a 1080p stream. Here we can also observe clearly the unique patterns of Youtube traffic, with arrivals being spaced 5 seconds, while all other services tend to use a constant stream of data.

Finding: The traffic patterns found on Stadia streams are similar to those of other WebRTC applications. For RTP traffic the patterns are mainly a factor of video resolution and framerate, but DTLS and STUN traffic patterns seem to be application specific, as all three applications use them in different ways. The main difference in video traffic between Stadia and other WebRTC applications is the magnitude, as Stadia needs close to ten times more RTP traffic than other applications.

Other live streaming applications such as Youtube and Twitch use different protocols and traffic patterns, even when using the same resolution and framerate, which further cements that Stadia RTP video traffic is heavily rooted in WebRTC operation, while DTLS and STUN traffic patterns are unique to Stadia.

13. Modelling Tomb Raider's traffic

In this section we present a traffic model for a single Stadia game, Tomb Raider, when the VP9 codec is used, and for the three available video resolutions using the VP9 video codec: 720p, 1080p, and 2160p.

Although there are many similarities between different Stadia games (and WebRTC apps in general, as we have seen in the previous section), the particularities of each game make it difficult to generalize, and therefore we opted to focus on a single game only, leaving for future work such a task.

The model presented in this section has been developed by analyzing the traces from datasets D1 and D5. It takes into account the traffic patterns observed in Section 6. It has been designed to be both accurate and simple, so it can be easily used in the performance evaluation of communication networks.

The traces in dataset D1 and D5 show that regardless of the employed video resolution, TR traffic follows a clear temporal pattern that closely matches the framerate of 60 fps. In all three video resolutions

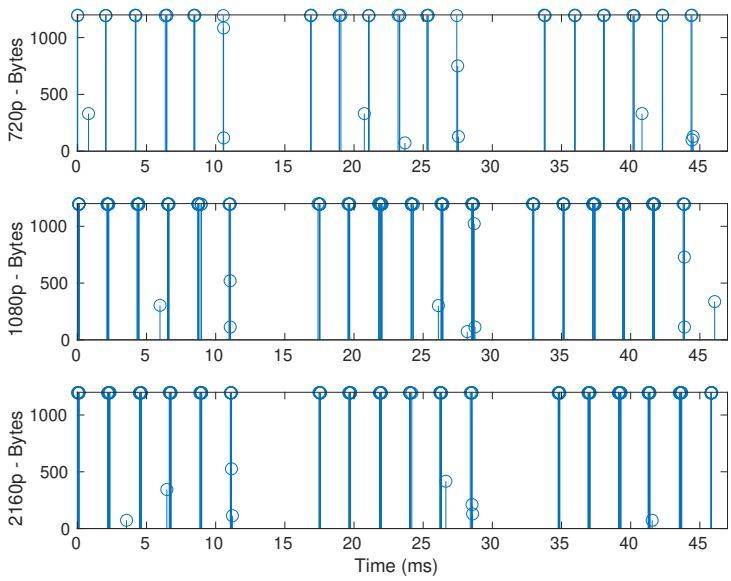


Figure 20: TR traffic patterns for different video resolutions. The y-axis indicates the size of the packets in bytes. It can be observed how the video frame timing is generally preserved, as well as the number of groups per frame. The main difference is the number of packets in each group, i.e., the batch size.

(720p, 1080p, 2160p), between two video frames, we find six groups of packets in general, with an average separation of 2 ms between two groups of packets. However, the number of packets in each group depends on the video resolution. Moreover, we can also observe long (>1100 Bytes) and short packets, which respectively represent video (RTP) and non-video traffic (audio, STUN, and DTLS). All these aspects are shown in Figure 20, where a 50 ms temporal snapshot of the Tomb Raider traffic is depicted for the three available video resolutions.

The observed video traffic temporal patterns can be then represented as shown in Figure 21, and characterized using only six parameters: the time between two frames (T_f), the packet size (L_v), the number

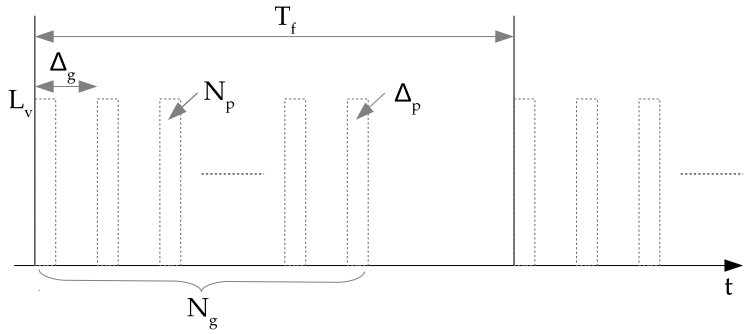


Figure 21: Representation of the temporal pattern followed by the video traffic in Stadia.

of groups of packets per frame (N_g), the time between two groups of packets (Δ_g), the number of packets in each group (N_{p}), and the time between two packets inside a group (Δ_p).

Regarding the non-video traffic, it can be observed in the traces that the load of non-video traffic is independent of the video resolution in use, an equal to 0.5 Mbps in all three cases. Similarly, the average packet size remains between 250 and 300 Bytes in all cases too.

Taking those observations into account, the resulting TR traffic model is parameterized as follows. It consists of two independent streams: video and non-video traffic, that are independently modelled:

1. The common parameters for **video traffic** in all three resolutions are: Frame duration $T_f = 1/60$ seconds, Video packet size $L_v = 1194$ Bytes, Number of groups of packets $N_g = 6$ per frame, time between groups of packets $\Delta_g = \mathcal{U}(1.5, 2.5)$ ms. The time between two consecutive packets within the same group of packets is set to a constant value of $\Delta_p = 0.02$ ms. Finally, depending on the video resolution, the number of packets per group (N_{pg}) is: $N_{\text{p}} = 3$ for 720p, $N_{\text{p}} = 8$ for 1080p, and $N_{\text{p}} = 12$ for 2160p.
2. Since the characteristics of the **non-video traffic** are inde-

pendent of the video resolution, we model it as a single traffic stream of 0.5 Mbps where packet arrivals follow a Poisson process. Also, packets sizes are exponentially distributed with mean $E[L_{\text{nv}}] = 275$ Bytes.

For all three video resolutions the number of packets per frame is set in a way that matches the observed traffic load in the traces (see Section 8.2 and Figure 9). For example, for the 1080 resolution, we have 48 video packets per frame, which results in a load of $60 \cdot (6 \cdot 8 \cdot (1194 \cdot 8)) = 27.5$ Mbps.

With the aim to illustrate how the presented model can be used to obtain further insights in terms of the network response in different scenarios, we extend the simulator used in [44] to include the TR traffic model, and consider the following two examples:

1. **Sharing a buffer with background traffic**: We investigate the capacity of a best-effort link with respect to the number of supported Stadia streams in presence of background traffic. The best-effort link may perfectly represent the link between the Ethernet switch and the final user as considered in the measurement campaign. The transmission rate of the link is set to $R = 100$ Mbps. Background traffic is generated using a Poisson source: packet sizes are exponentially distributed with an average of 12000 bits. The duration of each simulation is 100 s.
   Figure 22 shows the delay (average and 99th percentile) of TR traffic with respect to the load of the background traffic when the TR traffic is generated using the presented model (model), and when it is generated directly from the traces (trace). Results are consistent since a higher background traffic load and a higher resolution, i.e., higher TR traffic load, results in higher delays. Moreover, the results obtained using the model and the ones obtained using the traces are very similar, confirming that the presented model is accurate.

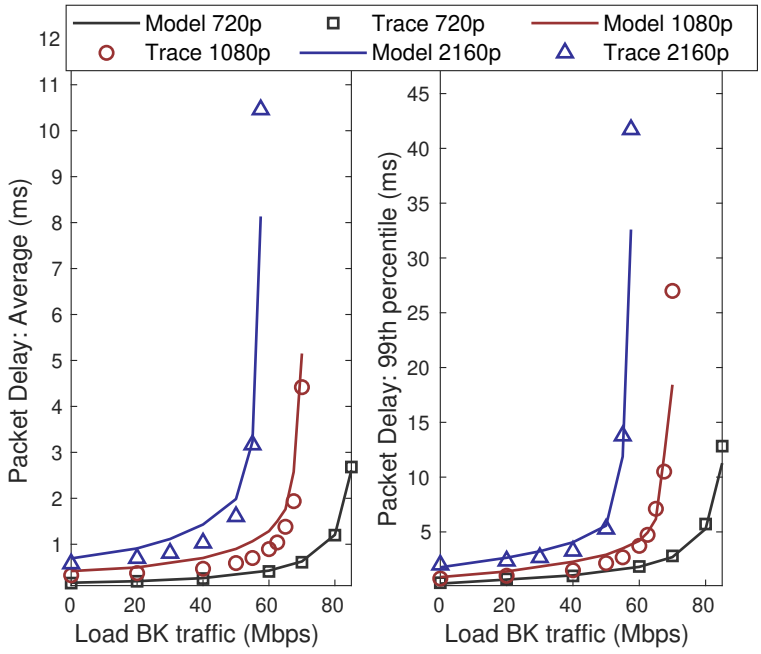


Figure 22: Average and 99th percentile delay for Tomb Raider against background traffic load. The three available video resolutions in Stadia are considered.

2. **Scaling the number of players**: We now consider the same 100 Mbps link. However, instead of sharing the link between Stadia and background traffic, we investigate how the latency increases when several Stadia players share the same link. To perform such an experiment, we execute as many instances of the traffic model as players. The duration of the simulation is 100 s, and each instance is initiated at a random instant of time during the first second of the simulation.

   Figure 23 shows the average and 99th percentile delay when the number of Stadia players increases, and so it does the number of traffic flows, for the three different supported video resolutions. We can observe that we can guarantee a packet delay below 3

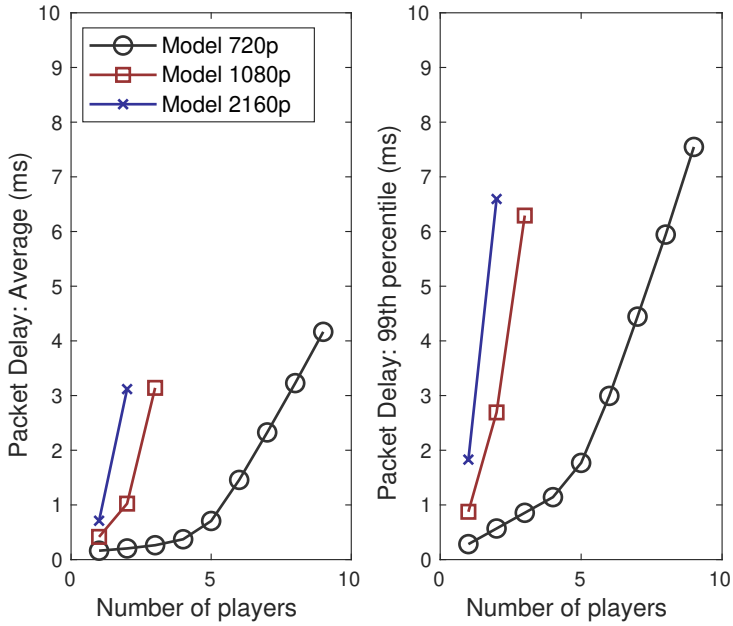


Figure 23: Increasing the number of Stadia streams

ms, for example, in the 99% of the cases for a single 4K player, two 1080p players, and six 720p players. Adding more users beyond those values, although supported in terms of throughput, would result in higher delays and a likely degradation of the user experience.

To conclude this section, we would like to point out that the value of the model parameters can be easily adjusted to represent other traffic generation patterns, even if they are not extracted from real traces, and so it enables to investigate how a certain network reacts to different traffic generation patterns.

14. Conclusions

In this paper we have investigated the characteristics of the Stadia traffic. We first designed a set of experiments that implied playing some specific games under pre-defined Stadia configurations while we captured the traffic over an Ethernet network. Then, we used the collected traffic measurements to learn about the characteristics of Stadia traffic, covering from how Stadia generates the traffic at the packet level in both downlink and uplink, to how it adapts to sudden changes in the network capacity and latency.

This paper aims to serve as a reference for future research in the area of real-time and interactive networking. In the future, it would be interesting to test Stadia performance on a Wi-Fi network, where a variety of factors can have an impact on performance, such as signal strength, number of users in the network, or the presence of other networks, as well as the particular Wi-Fi technology. Specifically, we would like to investigate how Wi-Fi is able to support low-latency in unlicensed bands [45]. Other Stadia dynamics require also to be analyzed, specially in terms of its response to network latency and background traffic. Our model can also be extended, particularizing it to represent other available games.

15. Acknowledgements

This work was supported by grants WINDMAL PGC2018-099959-B-I00 (MCIU/AEI/FEDER,UE), and SGR017-1188 (AGAUR).

References

[1] Sandvine. Global internet phenomena report. Technical report, 2019. https://www.sandvine.com/hubfs/Sandvine_Redesign_2019/Downloads/Internet%20Phenomena/Internet%20Phenomena%20Report%20Q32019%2020190910.pdf, Accessed on: 09 June 2020.

[2] Ariel Shapiro. Netflix Adds 15.8 Million Subscribers In First Quarter. https://www.forbes.com/sites/arielshapiro/2020/04/21/netflix-stock-up-5-after-hours-reports-158-million-additional-subscribers/#5e0c10dd3d18, 2020. Accessed 09 June 2020.

[3] Youtube statistics for press. https://www.youtube.com/intl/en-GB/about/press/, 2020. Accessed 09 June 2020.

[4] Twitch statistics and charts. https://twitchtracker.com/statistics, 2020. Accessed 09 June 2020.

[5] How SONY Bought, And Squandered, The Future Of Gaming. https://www.theverge.com/2019/12/5/20993828/sony-playstation-now-cloud-gaming-gaikai-onlive-google-stadia-25th-anniversary, 2020. Accessed 12 June 2020.

[6] Xcloud offical website. https://www.xbox.com/en-US/xbox-game-streaming/project-xcloud, 2020. Accessed 12 June 2020.

[7] Nvidia GeForce Now official website. https://www.nvidia.com/en-us/geforce-now/, 2020. Accessed 12 June 2020.

[8] Saeed Shafiee Sabet, Steven Schmidt, Saman Zadtootaghaj, Babak Naderi, Carsten Griwodz, and Sebastian Möller. A latency compensation technique based on game characteristics to mitigate the influence of delay on cloud gaming quality of experience. In *Proceedings of the 11th ACM Multimedia Systems Conference*, MMSys '20, page 15–25, New York, NY, USA, 2020. Association for Computing Machinery.

[9] Ryan Shea, Jiangchuan Liu, Edith C-H Ngai, and Yong Cui. Cloud gaming: architecture and performance. *IEEE network*, 27(4):16–21, 2013.

[10] M. Jarschel, D. Schlosser, S. Scheuring, and T. Hoßfeld. An evaluation of qoe in cloud gaming based on subjective tests. In *2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 330–335, 2011.

[11] S. Zadtootaghaj, S. Schmidt, and S. Möller. Modeling gaming qoe: Towards the impact of frame rate and bit rate on cloud gaming. In *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6, 2018.

[12] Varun Singh, Albert Abello Lozano, and Jorg Ott. Performance analysis of receive-side real-time congestion control for webrtc. In *2013 20th International Packet Video Workshop*, pages 1–8. IEEE, 2013.

[13] Bart Jansen, Timothy Goodwin, Varun Gupta, Fernando Kuipers, and Gil Zussman. Performance evaluation of webrtc-based video conferencing. *SIGMETRICS Perform. Eval. Rev.*, 45(3):56–68, March 2018.

[14] Ewa Janczukowicz, Arnaud Braud, Stéphane Tuffin, Ahmed Bouabdallah, and Jean-Marie Bonnin. Evaluation of network solutions for improving webrtc quality. In *2016 24th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–5. IEEE, 2016.

[15] Boni García, Micael Gallego, Francisco Gortázar, and Antonia Bertolino. Understanding and estimating quality of experience in webrtc applications. *Computing*, 101(11):1585–1607, 2019.

[16] Doreid Ammar, Katrien De Moor, Min Xie, Markus Fiedler, and Poul Heegaard. Video qoe killer and performance statistics in webrtc-based video communication. In *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, pages 429–436. IEEE, 2016.

[17] Dario Bonfiglio, Marco Mellia, Michela Meo, and Dario Rossi. Detailed analysis of skype traffic. *IEEE Transactions on Multimedia*, 11(1):117–127, 2008.

[18] Mirko Suznjevic, Justus Beyer, Lea Skorin-Kapov, Sebastian Moller, and Nikola Sorsa. Towards understanding the relationship between game type and network traffic for cloud gaming. In *2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2014.

[19] Kuan-Ta Chen, Yu-Chun Chang, Hwai-Jung Hsu, De-Yu Chen, Chun-Ying Huang, and Cheng-Hsin Hsu. On the quality of service of cloud gaming systems. *IEEE Transactions on Multimedia*, 16(2):480–495, 2013.

[20] Marc Carrascosa and Boris Bellalta. Cloud-gaming:Analysis of Google Stadia traffic, 2020.

[21] Andrea Di Domenico, Gianluca Perna, Martino Trevisan, Luca Vassio, and Danilo Giordano. A Network Analysis on Cloud Gaming: Stadia, GeForce Now and PSNow. *Network*, 1(3):247–260, 2021.

[22] Xiaokun Xu and Mark Claypool. A First Look at the Network Turbulence for Google Stadia Cloud-based Game Streaming. In *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 1–5, 2021.

[23] Philippe Graff, Xavier Marchal, Thibault Cholez, Stéphane Tuffin, Bertrand Mathieu, and Olivier Festor. An Analysis of Cloud Gaming Platforms Behavior under Different Network Constraints. In *2021 17th International Conference on Network and Service Management (CNSM)*, pages 551–557, 2021.

[24] Hassan Iqbal, Ayesha Khalid, and Muhammad Shahzad. Dissecting Cloud Gaming Performance with DECAF. *Proc. ACM Meas. Anal. Comput. Syst.*, 5(3), dec 2021.

[25] Franck Aumont, Frédérique Humbert, Christoph Neumann, Charles Salmon-Legagneur, and Charline Taibi. Dissecting Cloud Game Streaming Platforms Regarding the Impacts of Video Encoding and Networking Constraints on QoE. In *Proceedings of the Workshop on Game Systems (GameSys '21)*, GameSys '21, page 13–19, New York, NY, USA, 2021. Association for Computing Machinery.

[26] Bitmovin. Video Developer Report. `https://go.bitmovin.com/hubfs/Bitmovin-Video-Developer-Report-2018.pdf`, 2018. Accessed 18 June 2020.

[27] MultiCoreWare Inc. HEVC/H.265 Explained. `http://x265.org/hevc-h265/`, 2020. Accessed 18 June 2020.

[28] ITU-T. Joint Collaborative Team on Video Coding. `https://www.itu.int/en/ITU-T/studygroups/2017-2020/16/Pages/video/jctvc.aspx`, 2020. Accessed 18 June 2020.

[29] J. Bienik, M. Uhrina, M. Kuba, and M. Vaculik. Performance of H.264, H.265, VP8 and VP9 Compression Standards for High Resolutions. In *2016 19th International Conference on Network-Based Information Systems (NBiS)*, pages 246–252, 2016.

[30] T. Uhl, J. H. Klink, K. Nowicki, and C. Hoppe. Comparison Study of H.264/AVC, H.265/HEVC and VP9-Coded Video Streams for the Service IPTV. In *2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–6, 2018.

[31] Google. Chromium Blog: Celebrating 10 years of WebM and WebRTC. `https://blog.chromium.org/2020/05/celebrating-`

10-years-of-webm-and-webrtc.html, 2020. Accessed 18 June 2020.

[32] Opus. Opus audio codec . https://opus-codec.org/, 2020. Accessed 06 August 2020.

[33] Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal. https://datatracker.ietf.org/doc/html/rfc8445, 2018. Accessed 29 August 2021.

[34] Datagram Transport Layer Security Version 1.2. https://datatracker.ietf.org/doc/html/rfc6347, 2018. Accessed 29 August 2021.

[35] Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP). https://datatracker.ietf.org/doc/html/rfc5764, 2010. Accessed 29 August 2021.

[36] RTP: A Transport Protocol for Real-Time Applications. https://datatracker.ietf.org/doc/html/rfc3550, 2003. Accessed 29 August 2021.

[37] Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF). https://datatracker.ietf.org/doc/html/rfc4585, 2006. Accessed 29 August 2021.

[38] Gaetano Carlucci, Luca De Cicco, Stefan Holmer, and Saverio Mascolo. Analysis and design of the google congestion control for web real-time communication (webrtc). In *Proceedings of the 7th International Conference on Multimedia Systems*, MMSys '16, New York, NY, USA, 2016. Association for Computing Machinery.

[39] TechRepublic. Google Stadia's biggest challenge with streaming and meeting gamers' expectations . https://www.techrepublic.com/article/google-stadias-biggest-challenge-with-steaming-and-meeting-gamers-expectations/, 2020. Accessed 27 July 2020.

[40] IEEE spectrum Jeremy Hsu. How YouTube Paved the Way for Google's Stadia Cloud Gaming Service. https://spectrum.ieee.org/tech-talk/telecom/internet/how-the-youtube-era-made-cloud-gaming-possible, 2020. Accessed 06 August 2020.

[41] Kyungmin Lee, David Chu, Eduardo Cuervo, Johannes Kopf, Yury Degtyarev, Sergey Grizan, Alec Wolman, and Jason Flinn. Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '15, page 151–165, New York, NY, USA, 2015. Association for Computing Machinery.

[42] Mark Claypool and Kajal Claypool. Latency can kill: Precision and deadline in online games. In *Proceedings of the First Annual ACM SIGMM Conference on Multimedia Systems*, MMSys '10, page 215–222, New York, NY, USA, 2010. Association for Computing Machinery.

[43] Google. Bandwidth, data usage, and stream quality . https://support.google.com/stadia/answer/9607891?hl=en, 2020. Accessed 19 June 2020.

[44] Boris Bellalta. On the low-latency region of best-effort links for delay-sensitive streaming traffic. *IEEE Communications Letters*, 25(3):970–974, 2020.

[45] Toni Adame, Marc Carrascosa-Zamacois, and Boris Bellalta. Time-sensitive networking in IEEE 802.11 be: On the way to low-latency Wi-Fi 7. *Sensors*, 21(15):4954, 2021.

Performance and Coexistence Evaluation of IEEE 802.11be Multi-link Operation

Marc Carrascosa-Zamacois*, Lorenzo Galati-Giordano
Anders Jonsson, Giovanni Geraci, and Boris Bellalta

*Universitat Pompeu Fabra, Barcelona
♭ Nokia Bell Labs, Stuttgart

Abstract

Wi-Fi 7 is already in the making, and Multi-Link Operation (MLO) is one of the main features proposed in its correspondent IEEE 802.11be amendment. MLO will allow devices to coordinate multiple radio interfaces to access separate channels through a single association, aiming for improved throughput, network delay, and overall spectrum reuse efficiency. In this work, we study three reference scenarios to evaluate the performance of the two main MLO implementations —Multi-Link Multi-Radio (MLMR) and Multi-Link Single-Radio (MLSR)—, the interplay between multiple nodes employing them, and their coexistence with legacy Single-Link devices. Importantly, our results reveal that the potential of MLMR is mainly unleashed in isolated deployments or under unloaded network conditions. Instead, in medium- to high-load scenarios, MLSR may prove more effective in reducing the latency while guaranteeing fairness with contending Single-Link nodes.

M. Carrascosa and B. Bellalta were supported in part by grants Wi-XR PID2021-123995NB-I00 and WINDMAL PGC2018-099959-B-I00 (MCIU/AEI/FEDER,UE), PRE2019-088690 (MCIU/FPI). G. Geraci was in part supported by MINECO's Project RTI2018-101040 and by a "Ramón y Cajal" Fellowship from the Spanish State Research Agency.

I. Introduction

Wi-Fi is more popular than ever. There will be 628 million Wi-Fi hotspots by 2023, four times up from 2018, 11% of which adopting Wi-Fi 6 and 6E [1], [2]. Meanwhile, a new generation of Wi-Fi—IEEE 802.11be, or Wi-Fi 7—is in the making, with technical discussions underway to determine the specific implementation of several disruptive new features [3]–[9]. The new capabilities of IEEE 802.11be will include 320 MHz bandwidth channels, 16 spatial streams, hybrid automatic repeat request (HARQ), and multi-band/channel aggregation and operation [10].

This last feature, commonly known as Multi-Link Operation (MLO), refers to the joint use of multiple radio interfaces on a single device. Owing to its promised augmented throughput and reduced delay, MLO is arguably the new feature drawing the most attention from industry and academia alike [11]–[17]. However, the performance of specific MLO implementations, the interplay between multiple devices implementing MLO, and the coexistence of MLO with legacy channel access schemes are all crucial issues that remain largely unexplored.

In this paper, we bridge the above gap and investigate the performance of two MLO implementations as well as their coexistence with other legacy devices. In particular, we conduct extensive experiments comparing three channel access mechanisms: i) traditional single-link (SL) operations, where a device avails of a single radio interface; ii) multi-link single radio (MLSR), where multiple radio interfaces are available but only one at a time can be opportunistically used; and iii) multi-link multi radio (MLMR), where the multiple available radio interfaces can be used concurrently. Our study unfolds as follows:

- We begin by considering an isolated Basic Service Set (BSS) setting devoid of channel contention. In this case, MLSR—only accessing one interface at time—can merely reduce the backoff time, only yielding anecdotal delay gains over SL. MLMR does curb the worst delays by five-fold when availing of a second interface, though adding a third interface provides diminishing returns.
- We then consider two MLO BSSs contending for channel access. Contending MLSR BSSs retain the same delay as contention-free

SL BSSs, as they opportunistically react to the evolving channel occupancy. However, MLMR BSSs may surprisingly incur higher delays than those of SL and MLSR, since they sometimes starve one another.
- We conclude by assessing the coexistence between a MLO BSS and two independent legacy SL BSSs. A MLMR BSS boosts its throughput at the expense of a nearly equivalent reduction for the two coexisting SL BSSs. Nonetheless, MLMR also allows its SL neighbors to achieve lower delays in all cases except when these are highly loaded.

II. A Primer on Multi-link Operation

MLO is being introduced in IEEE 802.11be to enable Wi-Fi devices to exchange data in a flexible manner over one or multiple wireless interfaces.[1] Compared to legacy SL devices, where multiple radios are operated through different and separate transmitter-to-receiver associations as if they were part of different BSS, MLO devices can benefit of using all available radios through a single association. Indeed, MLO devices can dynamically select one of the available interfaces, or even all at the same time, thus achieving opportunistic channel access or spectrum aggregation, respectively, and thus potentially higher transmission rates and lower delays.

In this paper, we consider two MLO mechanisms that are likely to be found in the upcoming Wi-Fi 7 certified products (based on IEEE 802.11be). These two mechanisms, along with the legacy SL approach, are introduced in the sequel.

- **Single-Link (SL)**: Default channel access, following the Distributed Coordination Function (DCF) and running over a single radio interface. At the transmitter side, when a packet is available in the transmission buffer, a backoff instance is initiated and the packet is transmitted once the backoff expires.
- **Multi-link Single Radio (MLSR)**: To support opportunistic spectrum access at a reduced cost, Wi-Fi devices can be equipped

[1]The terms link, channel, radio, and interface are used interchangeably.

with a single fully functional 802.11be radio plus several other low-capability radios able only to decode IEEE 802.11 control packets (e.g., Wi-Fi preambles). At the transmitter side, once a packet is available in the buffer, an independent backoff instance is initiated on each wireless interface, with the data packets then being allocated to only one of such interfaces according to a specific strategy, e.g., to the one whose backoff expires first. No other transmission is initiated until the one ongoing on the selected interface is completed. Once the transmitter determines the interface to use for the ongoing transmission, it informs the receiver, which in turn switches its fully functional 802.11be radio to the selected interface, receives the data packet, and responds with the corresponding ACK.

- **Multi-Link Multiple Radio (MLMR)**: For a device implementing this approach, all multiple radio interfaces are 802.11be compliant and they are able to operate concurrently, thus performing multiple simultaneous transmissions. At the transmitter side, once a packet is available in the transmission buffer, a backoff instance is initiated on all inactive wireless interfaces, allocating the data packets progressively to the interfaces as their backoffs expire. At the receiver side, packets are then received on all links used by the transmitter.

Complexity: Implementing the above three mechanisms (namely SL, MLSR, and MLMR) entails an increasing level of complexity. Indeed, SL employs a single 802.11 radio; MLSR requires an 802.11be radio as well as $S-1$ 'dummy' radios—S being the number of interfaces—for channel sensing; MLMR requires S full-blown 802.11be radios. We will show that the gains (or lack thereof) arising from an increased complexity may heavily depend on the specific scenario.

Example: Fig. 1 exemplifies the operation of SL, MLSR, and MLMR. All the available interfaces (circles) share a single buffer and packets can thus be scheduled to either available interface. The figure illustrates the following:

- For SL, as packets arrive, the backoff starts and they are sent through the only available interface.
- For MLMR, Packet 1 arrives while Channel 1 is busy and Channel 2 is idle, thus it is transmitted through the latter. During this trans-

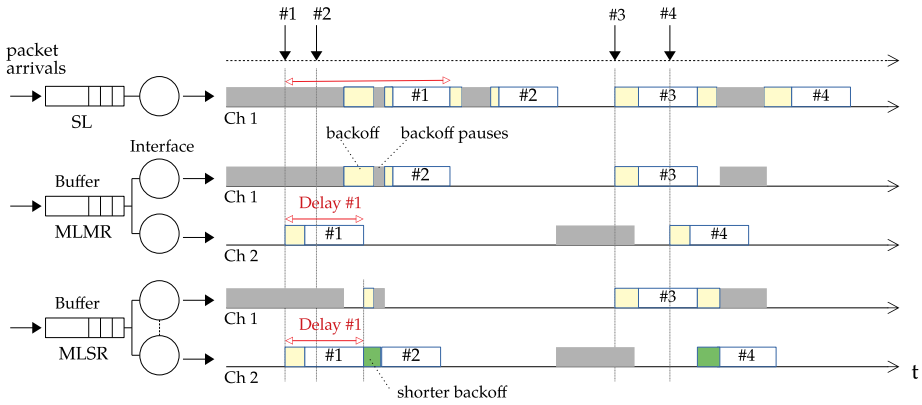


Fig. 1: Illustration of SL, MLMR, and MLSR operations. The transmission representation (white box) includes control and acknowledgment frames.

mission, another backoff instance begins on Channel 1 for Packet 2, resulting in transmitting both packets with shorter delays than in SL.

- For MLSR, Packet 1 is also transmitted through Channel 2, and then the backoff is restarted on both channels. Packet 2 is sent through Channel 2, whose backoff expires first, and the backoff on Channel 1 is cancelled.
- As for Packets 3 and 4, MLMR transmits them simultaneously as soon as they arrive, with the transmission for Packet 4 starting during the transmission of Packet 3.
- For MLSR instead, Packet 3 is transmitted first on Channel 1. As Packet 4 arrives, it must wait for the ongoing transmission to be completed. Channel 2 then becomes available first and is used for Packet 4. Note that MLSR transmits Packet 4 more slowly than MLMR, but faster than SL.

III. Evaluation Methodology

To carefully evaluate the performance of MLO as well as its coexistence with legacy SL, we consider the three representative scenarios

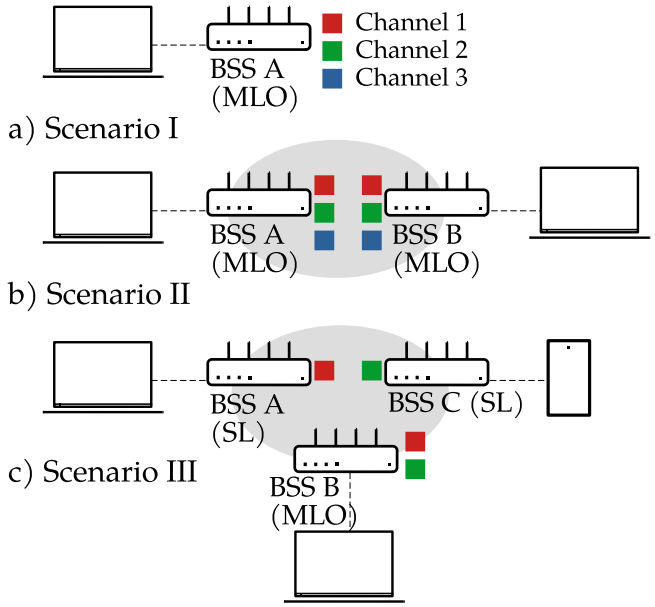


Fig. 2: Deployment scenarios considered in this paper: (a) a single MLO BSS, (b) two contending MLO BSSs, and (c) one MLO BSS coexisting with two legacy SL BSSs.

depicted in Fig. 2. In particular:

- *Scenario I* is used to asses the performance gains experienced by a single and isolated BSS (BSS A) as it gets upgraded from SL to MLO.
- *Scenario II* models the mutual interaction between two MLO BSSs (BSSs A and B) to study its effect on both.
- *Scenario III* features a single MLO BSS (BSS B) and two independent SL BSSs (BSSs A and C), serving the all-important purpose of evaluating the coexistence of 802.11be MLO with legacy SL devices.

All three scenarios share the following features: i) All BSSs are within each other's coverage area and therefore neither hidden terminal issues or deployment asymmetries arise; ii) Only downlink traffic is considered, i.e., from the AP to an associated single station; iii) Traffic

arrival follows a Poisson process and all arriving packets have a constant size of $L = 12000$ bits; $iv)$ APs have a transmission buffer size of 1000 packets; $v)$ A fixed modulation and coding scheme is employed, based on 256-QAM with rate 3/4 and 2 spatial streams[2]; $vi)$ A-MPDU packet aggregation is enabled for up to 64 packets, and the instantaneous number of aggregated MPDUs is chosen at the start of each transmission; $vii)$ The transmission duration depends on the A-MPDU size, thus ranging from 0.25 ms to 3.4 ms; $viii)$ The Request-to-Send/Clear-to-Send (RTS/CTS) mechanism is used to reserve the channel and, in the case of MLSR, to indicate which link will be used in the upcoming data transmission. The main system model parameters are summarized in Table I. In order to isolate the gains of MLO and its effect of legacy devices, the main system parameters are intentionally chosen according to 802.11be's predecessor and current standard, 802.11ax [18].

The MLO BSSs considered in the different scenarios are equipped with up to 3 radio interfaces, each one operating on a different 80 MHz radio channel. The channel mapping strategy implemented by each BSS is depicted in Fig. 2, where the colored boxes denote the corresponding links/channels in use. In particular:

- In Scenario I, BSS A uses Channels 1, 2, and 3.
- In Scenario II, both MLO BSSs use Channels 1, 2, and 3 simultaneously, thus modeling contention.
- In Scenario III, the two SL BSSs A and C employ orthogonal channels: Channel 1 and Channel 2, respectively. The MLO BSS B, employing both channels, thus contends with BSS A to access Channel 1 and with BSS C to access Channel 2.

In all three scenarios, MLO BSSs operate according to either the MLSR or MLMR strategies presented in Section II. These new MLO features are implemented atop a Wi-Fi state machine originally developed to study channel bonding and spatial reuse under SL, thus bringing to the next level our previous work that only focused on IEEE 802.11ax

[2]We set the transmission rate according to a link distance of 7 m, a transmit power of 20 dBm, and the 802.11 TGax path loss model for residential scenarios [18], resulting in a path loss of 72.51 dB and a received power of -58.51 dBm. No other channel impairments are considered, in order to isolate the effects of the three channel access schemes under consideration.

TABLE I: Wi-Fi state machine parameters for our studies.

PHY	
Channel width	80 MHz
Modulation	256 QAM 3/4
Transmission power	20 dBm
Legacy (HE single-user) preamble	20 μs (52 μs)
OFDM (legacy) symbol duration	16 μs (4 μs)
Number of spatial streams	2
MAC	
Short (DCF) InterFrame Space	16 μs (34 μs)
Service field	32 bits
MAC header	272 bits
Tail (delimiter) bits	6 bits (32 bits)
ACK (block ACK) bits	112 bits (256 bits)
RTS (CTS)	160 bits (112 bits)
Frame size	12000 bits
A-MPDU size	1–1024 packets
Backoff (Best effort Access Category)	CW_{min}=15
AP buffer size	4096 packets

networks [19], [20]. We carry out long-run simulations of 100 s, gathering traces with more than 150000 entries to guarantee an accurate characterization of both throughput and delay. [3]

IV. Performance and Coexistence of MLO

In this section, we consider the three scenarios described in Fig. 2 and evaluate the performance of a MLO BSS as well as its coexistence with other MLO and legacy BSSs.[4]

[3]While not reported for space constraints, the accuracy of our simulator was thoroughly validated using an extended version of the Markovian analytical models in [19], showing an excellent match.

[4]The values of throughput and delay ensue from the specific system model assumed. While the absolute values may differ from the performance limits of actual Wi-Fi 7 networks, their qualitative trends help understand the interplay and relative performance of the different approaches.

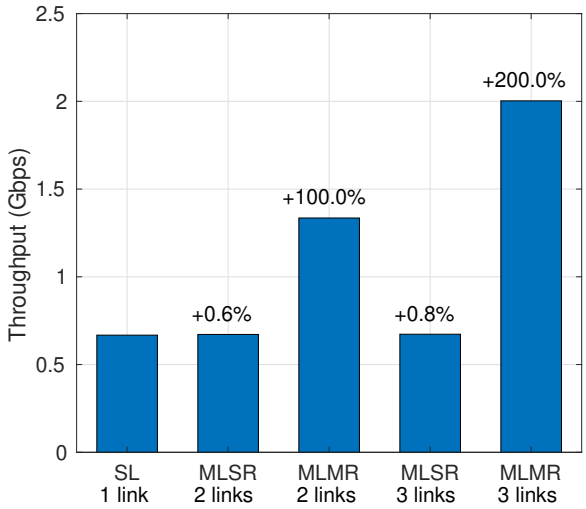


Fig. 3: Scenario I: Throughput of a single BSS using Single-Link (SL) and Multi-Link (MLSR or MLMR) modes.

A. Scenario I: Single MLO BSS

We begin by evaluating performance gains provided by MLO in an isolated BSS setting, i.e., devoid of channel contention, as described in Scenario I. To this end, we compare the throughput and delay experienced by a MLO BSS to the one of a legacy SL BSS. We assume the latter to operate on a single radio interface, e.g., on Channel 1, and the former to jointly operate two or three radio interfaces, each on a different channel.

Fig. 3 shows the throughput achieved by each transmission method as a function of the number of radio interfaces. For MLSR, the throughput is almost identical to that of SL regardless of the number of available interfaces. Instead, the MLMR throughput increases linearly with the number of interfaces, i.e., two- and three-fold with two and three links, respectively. Indeed, the latter is due to a lack of channel contention, allowing MLMR to transmit proportionally more data as the number of interfaces grows.

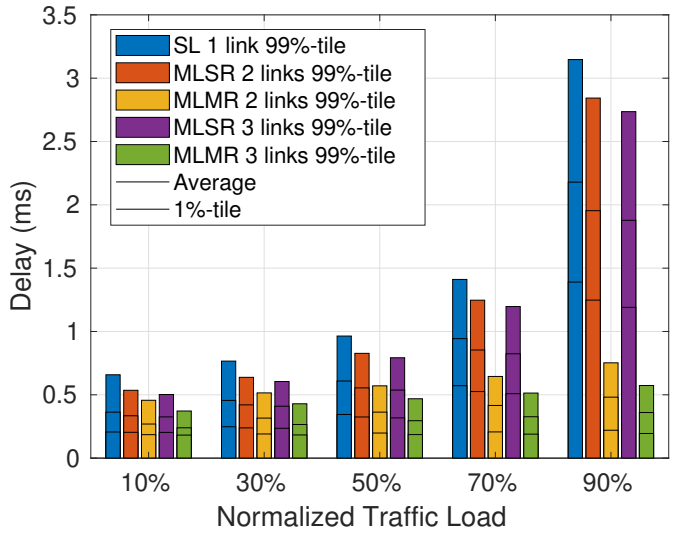


Fig. 4: Scenario I: Delay of a single BSS using Single-Link (SL) and Multi-Link (MLSR or MLMR) modes.

Fig. 4 presents the associated average, 99%-tile, and 1%-tile delay, the last two serving as a proxy for worst- and best-case performance, respectively. The traffic load indicated is normalized to the SL throughput, i.e., 670 Mbps as per Fig. 3. For instance, 10% and 30% correspond to loads of 67 and 201 Mbps, respectively.

As MLSR can only transmit through one radio interface at a time, its slightly reduced delay over SL is simply due to running simultaneous backoff counters and accessing the interface whose counter expires first. In the absence of channel contention, merely reducing the backoff time yields anecdotal delay reductions, as this is negligible with respect to the transmission time.

Unlike MLSR, MLMR does significantly decrease the delay since transmitting over multiple radio interfaces allows data packets to be received faster. MLMR is particularly effective at high loads, where availing of a second interface curbs the 99%-tile delay at least by a factor of four. Note that, for the traffic loads considered, adding a third

interface provides diminishing returns in delay reduction.

B. Scenario II: Two Contending MLO BSSs

We now study the coexistence of two MLO BSSs contending for channel access, as described in Scenario II. The MLO BSSs are equipped with two or three radio interfaces each and can be operated either in MLSR or MLMR mode. Fig. 5 shows the delay vs. traffic load under this setup as compared to the one experienced by two SL BSSs operating on orthogonal channels (and thus not contending for access).[5]

On the one hand, contending MLSR BSSs—with either two or three interfaces each—retain the same delay as contention-free SL BSSs. The opportunistic use of a single radio allows MLSR to react to the evolving contention levels over different channels. Additionally, since MLSR with two interfaces always leaves at least one channel idle to the contending BSS, it results in a fairer share of the spectrum. In the case of MLSR with three interfaces, an extra backoff instance can be allocated to each MLO BSS, further reducing the channel access delay.

On the other hand—and despite its higher complexity—MLMR with two interfaces somewhat surprisingly incurs higher 99%-tile delays than those of SL and MLSR, even at loads as low as 30%. Since MLMR BSSs can transmit through multiple interfaces at once, they can sometimes starve one another. As a result of this greedy policy, the best-case delays (e.g., 1%-tile) are reduced but the worst-case ones (e.g., 99%-tile) are increased. A workaround to this shortcoming is to add an extra interface to each MLO BSS. Although the channel used by the extra interface is to be shared between the two MLO BSSs, its presence significantly increases the likelihood of finding an idle interface. Indeed, the 99%-tile delay with a third interface drops to around half that of SL.

C. Scenario III: Contending MLO and SL BSSs

We conclude by assessing the coexistence between a MLO BSS and two independent legacy SL BSSs, as in Scenario III. Specifically, we

[5]While not shown for brevity, SL, MLSR, and MLMR all attain the same throughput as they handle the same incoming traffic load which is all successfully delivered.

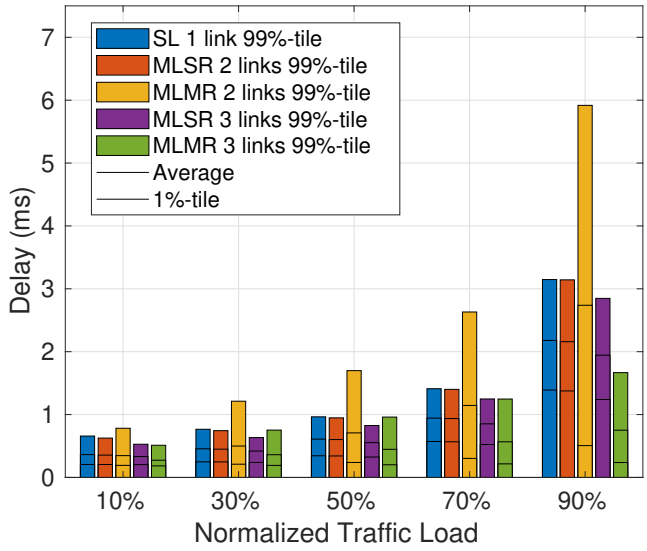


Fig. 5: Scenario II: Delay performance for each transmission method as traffic load increases. Traffic load values refer to the fraction of the SL full-buffer throughput in Section IV-A.

assume two radio interfaces and a constant traffic load for the MLO BSS (BSS B), and consider two different traffic loads for the SL BSSs (BSS A and BSS C), namely symmetric and asymmetric. Our aim is to shed light on how MLSR/MLMR affect the performance of neighboring legacy BSSs and how MLO BSSs handle symmetric and asymmetric activity in their operating channels. Fig. 6 shows the full-buffer throughput achieved by all three BSSs when BSS B employs either MLSR or MLMR. When operating in MLSR mode, BSS B achieves almost identical throughput as SL BSSs A and C. However, when employing MLMR, BSS B boosts its own throughput at the expense of a nearly equivalent reduction for the two coexisting SL BSSs A and C. Fig. 7a and Fig. 7b show the delay for each BSS when BSS B employs MLSR and MLMR, respectively. We consider several combinations for the traffic load fed to each BSS, indicated again as a fraction of the full-

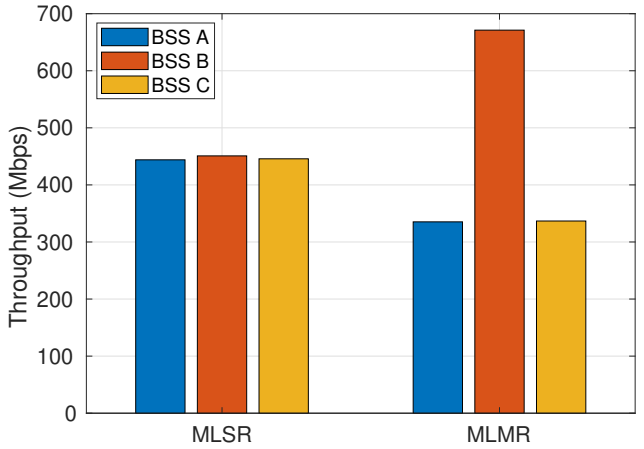
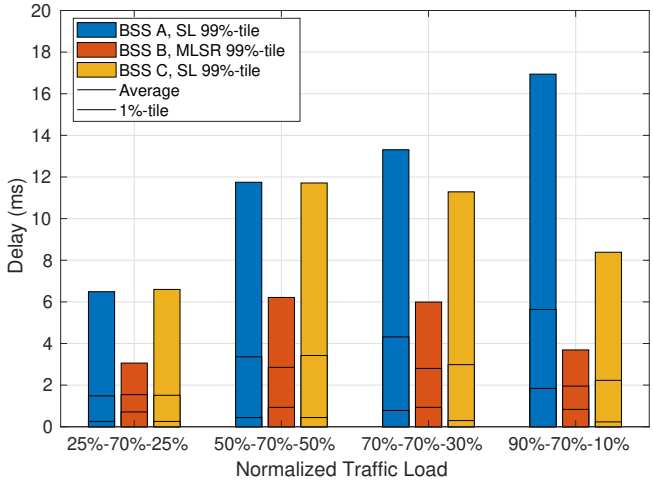


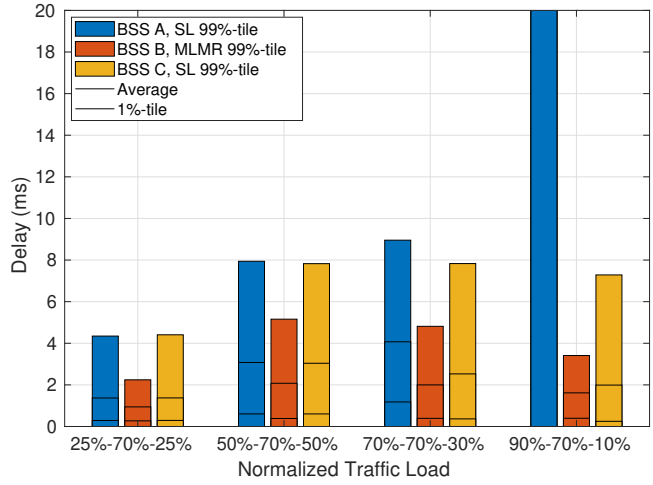
Fig. 6: Scenario III: individual BSS throughput when BSS B employs MLSR (left) and MLMR (right). BSSs A and C are assume to employ a single link.

buffer throughput achieved by a SL BSS in Scenario I (670 Mbps as per Fig. 3). Specifically, the load on BSS B is set to 70% while the loads on BSSs A and C are varied to model symmetric scenarios (with BSSs A and C experiencing the same load) and asymmetric scenarios (with BSS C increasingly less loaded than BSS A).

Both Fig. 7a and Fig. 7b show that MLSR/MLMR can opportunistically leverage the availability of an emptier channel. For instance, by comparing case {50%–70%–50%} to case {90%–70%–10%} with same aggregated contending traffic from BSSs A + C, we note that the latter results in a lower delay for MLSR/MLMR. In all traffic configurations considered, employing MLMR is only slightly more beneficial than MLSR for BSS B. As for the coexistence between the multi-link BSS B and the SL BSSs A and C, MLMR allows its SL neighbors to achieve lower delays than MLSR does in most cases. However when a SL BSS is highly loaded (i.e., 90%), MLMR can occasionally cause it to starve and thus experience higher delays.



(a) MLSR delay



(b) MLMR delay

Fig. 7: Delay in unbalanced scenarios. MLSR has a low impact on nearby BSSs, while MLMR has a severe impact on highly loaded channels, saturating BSS A. Traffic load values refer to the fraction of the SL full-buffer throughput obtained in Section IV-A.

V. Conclusion

This work is devoted to the understanding of the performance of two specific MLO implementations, the interplay between multiple devices implementing them, and their coexistence with legacy single-link channel access schemes. Through our extensive study—which compared i) traditional single-link operations, ii) multi-link single radio (MLSR), and iii) multi-link multi radio (MLMR)—, we were able to draw the following key insights:

- In an isolated BSS setting devoid of channel contention, MLMR with two interfaces can reduce the worst delays by a factor of five, whereas adding a third interface provides immaterial extra gains.
- Two contending MLSR BSSs experience same delay as SL does in a contention-free setup. Surprisingly, and despite its increased complexity, MLMR BSSs may instead incur higher delays by occasionally preventing one another from timely accessing the channel.
- When surrounded by legacy SL BSSs, a MLMR BSS boosts its own throughput at the expense of its SL neighbors', but also allows them to achieve lower delays for low-to-medium traffic loads.

The present work is, to the best of our knowledge, the first providing a well-grounded performance comparison of the two most relevant implementations of MLO and addressing the critical aspect of backward-compatibility with legacy SL devices. Extensions are underway from different standpoints:

Non-Poisson traffic: By considering non-Poisson traffic with batch arrivals—a key feature, being MLO capable of transmitting multiple packets in the same batch at once.

Reproducibility: By capturing the behavior of various MLO modes analytically [21], thus allowing a more generalized comparison and wide reproducibility of the results.

Interplay: By studying the performance gains of MLO when paired with other new features being introduced in IEEE 802.11be and beyond, e.g., advanced AP coordination [5].

References

[1] Cisco Annual Internet Report (2018–2023) White Paper. March 2020. Accessed on 05/08/2022.

[2] Wi-Fi alliance Wi-Fi 6E insights. April 2022. Accessed on 05/08/2022.

[3] Mao Yang and Bo Li. Survey and perspective on extremely high throughput (EHT) WLAN—IEEE 802.11 be. *Mobile Networks and Applications*, 25(5):1765–1780, 2020.

[4] David López-Pérez, Adrian Garcia-Rodriguez, Lorenzo Galati-Giordano, Mika Kasslin, and Klaus Doppler. IEEE 802.11be Extremely High Throughput: The next generation of Wi-Fi technology beyond 802.11ax. *IEEE Commun. Mag.*, 57(9):113–119, 2019.

[5] Adrian Garcia-Rodriguez, David López-Pérez, Lorenzo Galati-Giordano, and Giovanni Geraci. IEEE 802.11be: Wi-Fi 7 strikes back. *IEEE Commun. Mag.*, 59(4):102–108, 2021.

[6] Evgeny Khorov, Ilya Levitsky, and Ian F Akyildiz. Current status and directions of IEEE 802.11be, the future Wi-Fi 7. *IEEE Access*, 8:88664–88688, 2020.

[7] Cailian Deng, Xuming Fang, Xiao Han, Xianbin Wang, Li Yan, Rong He, Yan Long, and Yuchen Guo. IEEE 802.11be Wi-Fi 7: New challenges and opportunities. *IEEE Commun. Surveys & Tuts.*, 22(4):2136–2166, 2020.

[8] Toni Adame, Marc Carrascosa-Zamacois, and Boris Bellalta. Time-sensitive networking in IEEE 802.11be: On the way to low-latency WiFi 7. *Sensors*, 21(15):4954, 2021.

[9] Nikolay Korolev, Ilya Levitsky, Ivan Startsev, Boris Bellalta, and Evgeny Khorov. Study of Multi-Link Channel Access Without Simultaneous Transmit and Receive in IEEE 802.11be Networks. *IEEE Access*, 10:126339–126351, 2022.

[10] IEEE P802.11be/D1.5 - Draft 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 8: Enhancements for extremely high throughput (EHT), March 2022.

[11] Mao Yang, Bo Li, Zhongjiang Yan, and Yuan Yan. AP coordination and full-duplex enabled multi-band operation for the next generation WLAN: IEEE 802.11be (EHT). In *Proc. WCSP*, pages 1–7, 2019.

[12] Taewon Song and Taeyoon Kim. Performance analysis of synchronous multi-radio multi-link MAC protocols in IEEE 802.11be extremely high throughput WLANs. *Applied Sciences*, 11(1):317, 2020.

[13] Álvaro López-Raventós and Boris Bellalta. Multi-link operation in IEEE 802.11be WLANs. *IEEE Wireless Commun.*, pages 1–12, 2022.

[14] Álvaro López-Raventós and Boris Bellalta. Dynamic traffic allocation in IEEE 802.11be multi-link WLANs. *IEEE Wireless Commun. Letters*, 11(7):1404–1408, 2022.

[15] Gaurang Naik, Dennis Ogbe, and Jung-Min Jerry Park. Can Wi-Fi 7 support real-time applications? On the impact of multi link aggregation on latency. In *Proc. IEEE ICC*, pages 1–6, 2021.

[16] Guillermo Lacalle, Iñaki Val, Oscar Seijo, Mikel Mendicute, Dave Cavalcanti, and Javier Perez-Ramirez. Analysis of latency and reliability improvement with multi-link operation over 802.11. In *Proc. IEEE INDIN*, pages 1–7, 2021.

[17] Marc Carrascosa, Giovanni Geraci, Edward Knightly, and Boris Bellalta. An experimental study of latency for IEEE 802.11be multi-link operation. In *Proc. IEEE ICC*, pages 1–6, 2022.

[18] IEEE 802.11 TGax. TGax Simulation Scenarios. https://mentor.ieee.org/802.11/dcn/14/11-14-0980-14-00ax-simulationscenarios.docx. accessed on 05/08/2022.

[19] Boris Bellalta, Alessandro Checco, Alessandro Zocca, and Jaume Barcelo. On the inter-

actions between multiple overlapping WLANs using channel bonding. *IEEE Transactions on Vehicular Technology*, 65(2):796–812, 2015.

[20] Francesc Wilhelmi, Sergio Barrachina-Muñoz, Cristina Cano, Ioannis Selinis, and Boris Bellalta. Spatial reuse in IEEE 802.11ax WLANs. *Computer Communications*, 170:65–83, 2021.

[21] Boris Bellalta, Marc Carrascosa, Lorenzo Galati-Giordano, and Giovanni Geraci. Delay Analysis of IEEE 802.11be Multi-link Operation under Finite Load. *IEEE Wireless Communication Letters*, 2022.

Wi-Fi Multi-Link Operation: An Experimental Study of Latency and Throughput

Marc Carrascosa-Zamacois, Giovanni Geraci, *Senior Member, IEEE,* Edward Knightly, *Fellow, IEEE,* and Boris Bellalta, *Senior Member, IEEE*

Abstract

In this article, we investigate the real-world capability of the multi-link operation (MLO) framework—one of the key MAC-layer features included in the IEEE 802.11be amendment—by using a large dataset containing 5 GHz spectrum occupancy measurements on multiple channels. Our results show that when both available links are often busy, as is the case in ultra-dense and crowded scenarios, MLO attains the highest throughput gains over single-link operation (SLO) since it is able to leverage multiple intermittent transmission opportunities. As for latency, if the two links exhibit statistically the same level of occupancy, MLO can outperform SLO by one order of magnitude. In contrast, in asymmetrically occupied links, MLO can sometimes be detrimental and even increase latency. We study this somewhat unexpected phenomenon, and find its origins to be packets suboptimally mapped to either link before carrying out the backoff, with the latter likely to be interrupted on the busier link. We cross validate our study with real-time traffic generated by a cloud gaming application and quantify MLO's benefits for latency-sensitive applications.

M. Carrascosa-Zamacois, G. Geraci, and B. Bellalta are with University Pompeu Fabra, 08018 Barcelona, Spain ({marc.carrascosa, giovanni.geraci, boris.bellalta}@upf.edu).

E. Knightly is with Rice University, Houston TX 77005, USA (knightly@rice.edu).

M. Carrascosa-Zamacois and B. Bellalta were supported by WINDMAL PGC2018-099959-B-I00 and WI-XR PID2021-123995NBI00 (MCIU/AEI/FEDER,UE).

G. Geraci was supported by RTI2018-101040-A-I00, PID2021-123999OB-I00, and a "Ramón y Cajal" Fellowship from the Spanish Research Agency.

E. Knightly was supported by Cisco, Intel, the US National Science Foundation (grant numbers 1955075, 1923782, 1824529, and 2148132), and the Army Research Laboratory (grant W911NF-19-2-0269)

Part of the materials presented in this article have been presented at IEEE ICC 2022 [1].

I. Introduction

Achieving consistent low delay in Wi-Fi networks is a challenge that has attracted growing interest, motivated by new applications with stringent latency constraints, such as gaming, augmented and virtual reality, industrial automation, and remote healthcare—some requiring response times as low as 1 ms [2]. Moreover, Wi-Fi access links have the potential to be the bottleneck in terms of network delay, accounting for more than 60% of the Round Trip Time in connections to domestic servers [3]. Indeed, operating in license-exempt bands brings about the need to coexist with other wireless networks, along with the inherent uncertainty as to how many transmission opportunities will be available, and when. One way to mitigate such uncertainty is by employing multiple radio interfaces for packet transmission. At the time of writing, this approach—termed multi-link operation (MLO)—is one of the main features being proposed and developed for IEEE 802.11be [4], [5], the new amendment that is foreseen to be certified as Wi-Fi 7 [6]–[9].

A. Motivation

Through MLO, IEEE 802.11be will target efficient operations in all the available bands, i.e., 2.4, 5, and 6 GHz, for load balancing, multi-band aggregation, and simultaneous downlink/uplink transmission [10]. In 802.11be, a multi-link device is defined as one with multiple affiliated access points (APs) or stations (STAs), and a single MAC service access point to the above logical link control layer [6]. Multi-link devices could thus transmit and receive packets at the same time, separate the control and data planes, or transmit delay-sensitive traffic through multiple links to ensure its timely reception [2], [11]. Lastly, while MLO is fully transparent to the upper TCP/IP protocols, they will benefit from the faster and more reliable data communication it enables [4].

As consensus has not yet been reached on the specific implementation details of MLO, recent works have compared the performance of different variants [12], [13], studied the feasibility of simultaneous transmission and reception [14], and undertaken the optimization of traffic and resource allocation in MLO [15]–[17]. Besides throughput augmentation, latency reduction has been identified as one of the main

endeavors of MLO, with its delay performance being the object of several recent studies [16]–[21]. These works—and others—have shown that, in many cases, MLO is capable of enabling new applications whose requirements cannot reliably be supported by conventional single-link operation (SLO).

Notwithstanding the insights provided by these works, the literature currently lacks experimental evidence on what performance gains MLO can attain over SLO, in what practical scenarios, and under which channel access methods. This gap prompts us to study, for the first time, the performance of MLO by using spectrum occupancy measurements and real application traces.

B. Contribution and Summary of Results

In this paper, we utilize our over-the-air measurements of spectrum occupancy for the entire 5 GHz band [22], [23][1] and investigate the throughput and latency performance of MLO when operating on two links. Atop these traces, which include scenarios with high AP density and crowded environments and span multiple hours, we develop an emulation tool that fuses a Wi-Fi MLO state machine with the high-resolution spectrum measurements. We feed our MLO state machine with Poisson traffic first, and then validate selected experiments with real-time traffic generated by a cloud gaming application. Besides legacy Wi-Fi SLO, we study two MLO channel access modes defined as follows: *(i)* MLO with Simultaneous Transmit and Receive (MLO-STR), in which both interfaces are available and work independently, and *(ii)* MLO with Non-Simultaneous Transmit and Receive (MLO-NSTR), where both interfaces are available but access to the secondary link is conditioned on the primary also being unoccupied. While our results confirm the potential latency gains of MLO seen in previous works, the use of real spectrum measurements as well as real traffic traces offers new and otherwise inaccessible insights on MLO.

Our main findings can be summarized as follows:

[1]Freely available in the open source WACA dataset: https://github.com/sergiobarra/WACA_WiFiAnalyzer.

- We show that an MLO AP with two radio interfaces achieves throughput higher than the maximum SLO throughput in 53% and 28.5% of the cases by using MLO-STR and MLO-NSTR, respectively. When both links are almost always busy, which may correspond to ultra-dense and crowded scenarios, MLO-STR achieves the highest throughput gains as it is able to leverage the intermittent transmission opportunities over multiple links.
- We find that when primary and secondary links have statistically symmetrical occupancy, MLO-STR yields order-of-magnitude 95th percentile latency benefits over SLO, even in the challenging regime of increasing occupancies and traffic. This is because MLO-STR can utilize either available link, and reduce packet waiting time even when it cannot simultaneously utilize both links.
- In contrast, we surprisingly discover that when using two links with asymmetrical occupancy, MLO-STR can sometimes underperform SLO by up to 112% in terms of 95th percentile latency. This is owed to packets being suboptimally assigned to an interface before carrying out the backoff, with the latter likely to be interrupted on the busier link. This phenomenon is exacerbated when the asymmetry in channel occupancy increases.
- To overcome the aforementioned phenomenon, we define a third MLO channel access approach, denoted MLO-STR+, that employs parallel backoff instances for each interface and allocates packets to the interface whose backoff expires first. While MLO-STR+ is a minor variation on MLO-STR, we study it to better understand the design space and ultimate capabilities of MLO.
- We further validate our results by using, in addition to real-world channel occupancy measurements, real-time traffic generated by a cloud gaming application. This final set of experiments confirm our previous findings, and further demonstrate MLO's capabilities to enable latency-sensitive applications whose traffic load cannot otherwise be delivered in a timely manner through SLO.
- Using channel bonding, we show that splitting the channel bandwidth (80 MHz) between two independent MLO links (2x40 MHz) leads to lower delays than using the entire bandwidth for a single

link. Further, we show that by using wider links, we have another degree of freedom in the primary channel used, and correct selection can lead to an 89.5% delay reduction for MLO-STR+.

The rest of the document is organized as follows. Section II details the experimental setup, including the dataset and methodologies used. Section III studies the achievable MLO throughput based on the links occupancy. Section IV analyses the MLO delay in symmetrically and asymmetrically occupied links. Section V introduces MLO-STR+ as a way to improve the observed drawbacks of MLO-STR operation. In Section VI, real traffic traces are used to validate previous results obtained using Poisson traffic. Section VII studies the use of channel bonding coupled with MLO. Section VIII includes the related work, and finally, Section IX concludes the paper.

II. MLO Protocol and Experimental Setup

We consider a scenario consisting of multiple Basic Service Sets (BSS) and STAs accessing multiple channels in the 5 GHz band. We use a dataset containing real spectrum information from a crowded stadium. Since the dataset contains the aggregate signal strength received at the endpoint, we cannot identify individual devices and we focus on a single AP and STA pair equipped with two MLO-capable interfaces each, denoted the MLO-BSS. We do not add extra model/simulation-driven sources of contention so as to make all our findings only dependent on the real traces.

The MLO interfaces operate in the 5 GHz band on 20 MHz-wide channels, respectively on channel 36 (low 5 GHz) and channel 100 (high 5 GHz), which we denote as *primary* and *secondary*, respectively. On both channels, the MLO-BSS observes the environment activity, i.e., the transmissions generated by the Orthogonal Basic Service Sets (OBSS). The MLO-BSS and OBSS under consideration are illustrated in Figure 1 in blue and red, respectively. The environment activity from the OBSS is characterized by the WACA dataset (Section II-B). For the MLO-BSS, we only consider downlink traffic, i.e., from the AP to the STA. We initially assume packet arrivals to follow a Poisson process, and transmitted packets to have a constant size of $L = 12000$ bits. In

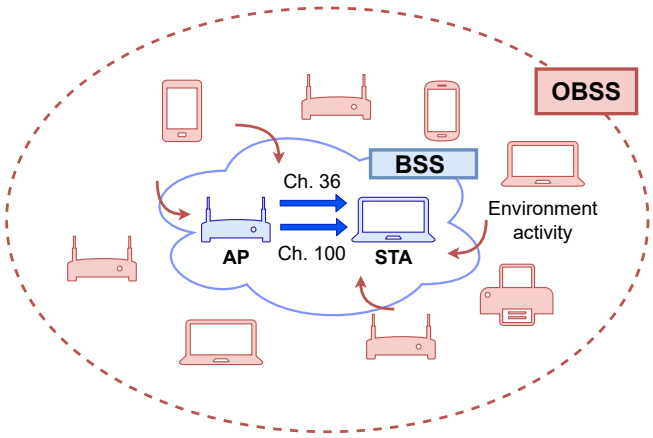


Fig. 1: Scenario considered. The WACA dataset is used to characterize the environment activity (red) observed by the target BSS (blue) on channels 36 and 100 in the 5 GHz band.

Section VI, we consider real-time data traffic instead. Since we do not have client-AP SNR traces, any MCS variation would need to be purely model driven. We therefore employ a fixed MCS (256-QAM, 5/6) in all links.

Next, we detail the channel access schemes considered, the measurement-based channel occupancy model, and the performance evaluation methodology.

A. *Multi-link Channel Access Policies*

We study three channel access policies for the MLO-BSS, namely:

- Conventional single-link operation (SLO), where only the primary interface is available.
- Multi-link operation with Simultaneous Transmit and Receive (MLO-STR), where both interfaces are available and work independently.
- Multi-link operation with Non-Simultaneous Transmit and Receive (MLO-NSTR), where both interfaces are available but access to the secondary is subjected to the state of the primary link.

In particular, the two MLO channel access schemes operate as follows:

1) MLO with Simultaneous Transmit and Receive (MLO-STR): The two radio interfaces operate independently and asynchronously. The first packet waiting for transmission in the buffer is allocated to the first radio interface that becomes available. If both radio interfaces are available, the packet is randomly allocated to either. Once a packet is allocated to an interface, it starts the channel access procedure by initializing a backoff instance.

2) MLO with Non-Simultaneous Transmit and Receive (MLO-NSTR): One radio interface always acts as primary, and the other always as secondary. When there are packets waiting for transmission, the primary interface undergoes contention to access the channel. Once the backoff counter reaches zero, packets are sent through the two interfaces if the secondary one has been idle for at least a PIFS interval before the backoff expiration. Otherwise, only a single packet is transmitted through the primary link.

Figure 2 exemplifies SLO, MLO-STR, and MLO-NSTR operation. SLO follows default Wi-Fi access, where packets are sequentially transmitted over a single link, with packet 1 being the first to be transmitted in the timeline before starting backoff for packet 2, and so on. In the case of MLO-STR, arriving packets are allocated to whichever interface becomes available first. This results in a significant delay reduction for packets #1, #2, and #4. In the case of MLO-NSTR, the secondary link's dependence on the primary sometimes prevents using the two radio interfaces efficiently. As a result, and unlike MLO-STR, the delay for packets #1 and #4 cannot be reduced with respect to SLO.

In order to evaluate the above MLO schemes, we extended the IEEE 802.11 state machine originally developed in [23] by adding functionalities to accurately reproduce the temporal system dynamics under finite traffic loads (i.e., non-full buffer conditions). In order to isolate the combined effect of the access scheme and channel occupancy, we assume a fixed modulation and coding scheme on both interfaces - a 256-QAM with coding rate 5/6 and 2 spatial streams- yielding a transmission time of 0.172 ms (DATA+SIFS+ACK). Since our FCB-WACA dataset (described next) contains measurements taken with a periodicity of 10 μs, we have rounded the duration of IEEE 802.11

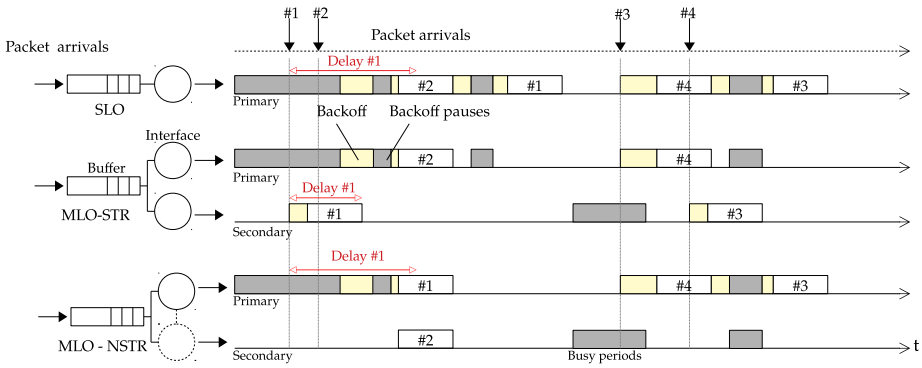


Fig. 2: Illustration of SLO, MLO-STR, and MLO-NSTR operations. Grey, yellow, and white bars respectively indicate occupied channels, random backoffs, and packet transmissions. Packet transmissions include both the data part and the corresponding ACK, as well as DIFS and SIFS.

timings to integer multiples of 10 μs, setting the duration of a backoff empty slot, SIFS, and DIFS to 10 μs, 10 μs, and 30 μs, respectively. Such small approximation implies no loss of generality, as discussed in [23]. The value of the $\mathrm{CW}_{\mathrm{min}}$ used in all cases is 15.

B. Measurement-based Channel Occupancy Dataset

To investigate the performance of the MLO-BSS in a real-world setting—i.e., while considering OBSS activity—we employ the *WACA dataset*, containing over-the-air measurements of the 5 GHz band occupancy that we have recently collected and made publicly available [22], [23]. This dataset was obtained by conducting extensive measurement campaigns on different days and in multiple locations, including a sold-out football stadium (F. C. Barcelona's Camp Nou). In this paper, we focus only on the football stadium measurements since they range from completely idle to fully occupied channels. We will refer to such subset of measurements as the *FCB-WACA dataset*.

The FCB-WACA dataset spans 5 hours and contains 2000 samples of the Received Signal Strength Indicator (RSSI) for each of the 24 20-MHz

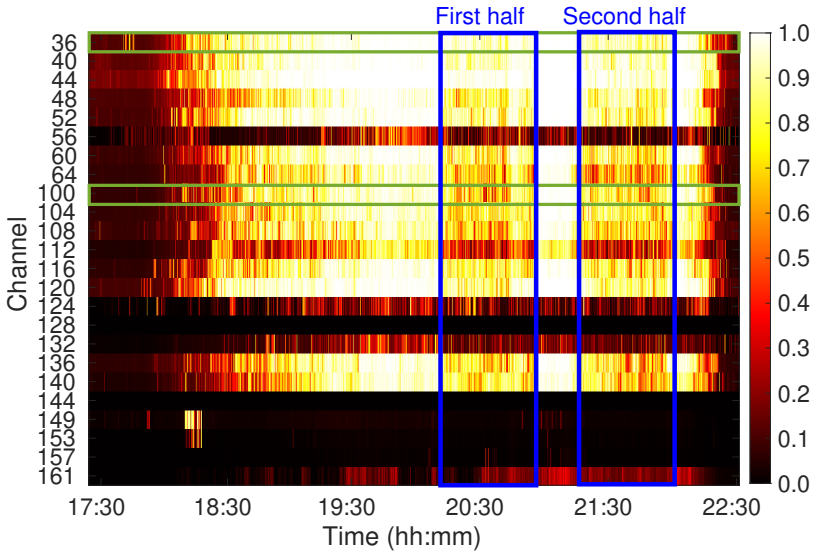


Fig. 3: Average channel occupancy in the FCB-WACA dataset. Channels 36 and 100, used in our experiments, are highlighted horizontally. Time intervals corresponding to the first- and second-half of the football game are highlighted vertically.

channels in the 5 GHz band.[2] Each sample lasts one second and consists of 1000 consecutive 10 μs measurements containing the aggregate signal strength of all nodes in the area. In Figure 3, the spectrum occupancy in the FCB-WACA dataset is displayed as the average number of busy slots in each one-second sample, with a slot considered busy if its RSSI is above -83.5 dBm. We note from Figure 3 that the channel occupancy varies across the measurement campaign, exhibiting: (i) predominantly empty channels prior to the football game, (ii) increasing occupancy up until the game starts and during half-time recess, (iii) lower occupancy during the first- and second-half, and (iv) a rapidly decreasing occupancy from the end of the game onwards.

[2]The F. C. Barcelona Camp Nou's network only supports 20 MHz channels without channel bonding.

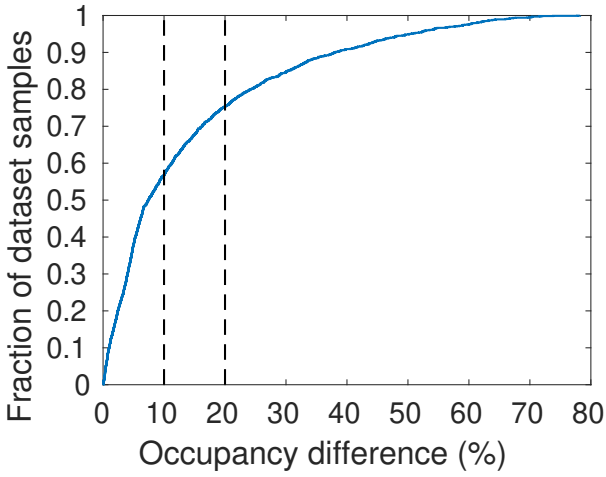


Fig. 4: Distribution of the difference in occupancy between channels 36 and 100.

While the temporal evolution of channels 36 and 100 appears similar from Figure 3 at the macroscopic level, the same does not necessarily hold when observing concurrent one-second samples from the two channels and comparing their average occupancy. To quantify the occupancy disparity, Figure 4 shows the distribution of the absolute value of the difference in occupancy between the two channels. Although such difference is lower than 10% (resp. 20%) in 75% (resp. 57%) of the samples, there is also a non-negligible number of cases with high occupancy disparity. The latter prompts us to evaluate the performance of MLO channel access schemes both under symmetric and asymmetric channel occupancy, as discussed in the remainder of the paper.

In what follows, we employ the FCB-WACA dataset to investigate how different combinations of primary and secondary channel occupancies affect the MLO-BSS performance. In particular, we assume that the MLO-BSS perceives the same spectrum activity as the one captured in the FCB-WACA dataset, and it contends for channel access accordingly. To address the interaction between the simulated node and the traces, we implement the same "hinder" interaction model as in [23]. Using the

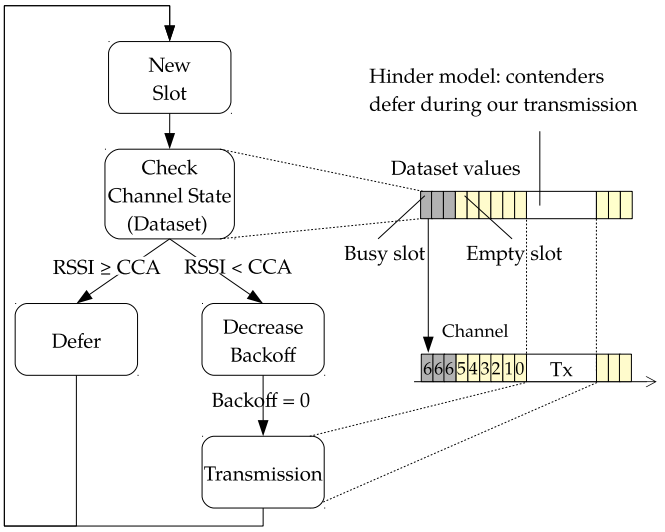


Fig. 5: Wi-Fi state machine (per interface) using WACA dataset to determine channel state.

hinder model, once the MLO-BSS starts a transmission, we assume that the OBSS devices are able to sense the on-going transmission and defer accordingly, allowing the MLO-BSS transmission to finish successfully, thus avoiding collisions. As it is shown in [23], the hinder interaction model keeps the same implicit channel access fairness as in CSMA.

C. Trace-based Simulation Methodology

Figure 5 illustrates how the WACA dataset is used by the Wi-Fi state machine in each of its interfaces. As previously stated, we accurately follow the WACA dataset occupancy measurements to determine the MLO-BSS channel access dynamics.

In order to study the effect of channel occupancy on latency, we treat both channels as independent and partition the available traces in our dataset into different average channel occupancy regimes: $\{10\%, 20\%, \ldots, 90\%\}$, as illustrated in Figure 6.

We perform 20 experiments for each combination of channel occupancy and traffic load, with each experiment considering a pair of one-

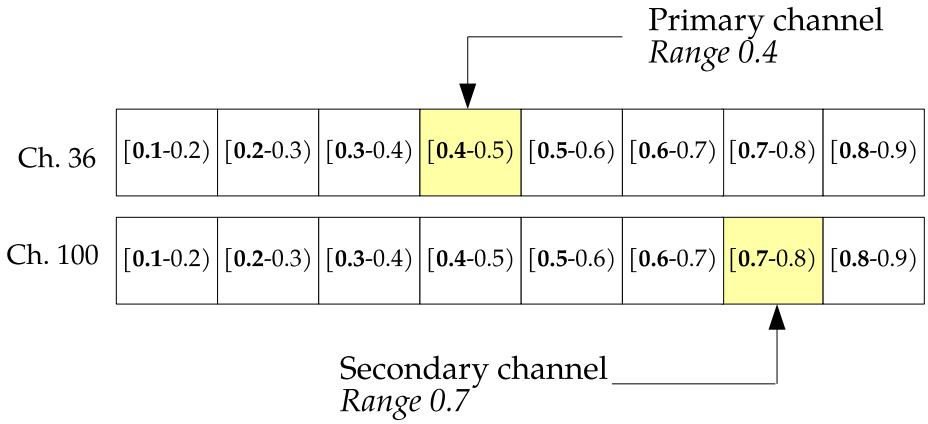


Fig. 6: Partitioning the available traces in the dataset into different average channel occupancy regimes.

second spectrum samples. Samples from channel 36 are assigned to the primary MLO link, and channel 100 to the secondary link.

Each experiment is carried out as follows: *(i)* We select the occupancy regime of interest for the primary and secondary links, e.g., 10% and 40%, respectively. *(ii)* We combine uniformly at random one spectrum sample each for the primary and secondary links from the dataset. *(iii)* For each spectrum sample pair and given a particular traffic load of interest, we compute the packet arrival times at the AP. *(iv)* We execute the Wi-Fi state machine for SLO, MLO-STR, and MLO-NSTR access policies. The same packet arrival times are considered in all cases to allow a direct comparison. *(v)* We store the individual delay experienced by each packet in each experiment.

We then combine the per-packet results (i.e., the individual packet delays) from all runs to obtain the average and the 95th percentile delay. We guarantee that all results are obtained under stability conditions (i.e., the AP does not become backlogged) and thus we discard any experiment where less than 95% of all the transmitted packets are received.

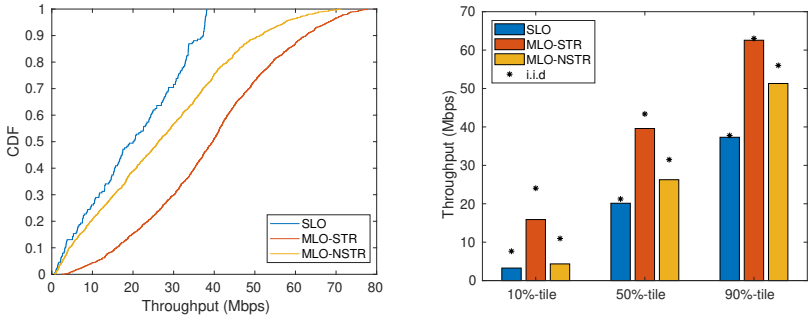
III. Origins of Throughput Gains

Owing to the availability of an extra radio interface at the AP, MLO is guaranteed to increase the BSS throughput with respect to single-link operation. However, the effectiveness of MLO in practice, including challenging scenarios such as crowded and rapidly changing environments, is tied to the instantaneous occupancy patterns of the links in use. In this section, we study this aspect by analyzing the effectiveness of each MLO policy in terms of the throughput gains attainable over a SLO baseline. Aiming for a general understanding, we consider all possible combinations for the statistical occupancy of the two available links, and study how the latter affects the attainable gains. In this section we consider that the AP is fully backlogged.

A. Throughput Distribution

MLO-STR operates interfaces independently, thus aggregating the available airtime from different links. MLO-NSTR however, synchronizes its secondary interface to the primary. One can thus expect MLO-NSTR to require links with similar activity patterns to achieve additive capacity. However, activity patterns on different links are likely to be completely independent due to the number of contenders, their traffic loads, and random access mechanisms.

Figure 7a shows the empirical CDF of the throughput achieved with each channel access mode across all channel occupancy combinations for our trace-based simulation. The SLO throughput increases proportionally to the airtime available, with the lowest value of 0.672 Mbps corresponding to the highest occupancy of 0.8, and the highest value of 38.1 Mbps to the lowest occupancy of 0.1. MLO-STR and MLO-NSTR have a different throughput distribution as a result of using two links, yet they function similarly. MLO-STR leverages the secondary link independently from the primary and thus offers a throughput improvement over SLO across the whole CDF, showing a remarkable $5\times$ gain in terms of 1%-worst throughput. MLO-NSTR behaves similarly to SLO in the lower tail, when it struggles to find simultaneous transmission opportunities on both links, and similarly to MLO-STR in the



(a) Cumulative distribution function of the throughput achieved by each of the three channel access modes.

(b) Throughput for each channel access method in terms of 10%, 50%, and 90%-tile using our trace-based simulator (bars). Values obtained analytically via an i.i.d model (stars) are also shown for comparison.

Fig. 7: Throughput statistics for each of the three channel access methods considered.

upper tail, when the low occupancy of the primary link allows frequent simultaneous transmissions.

We next evaluate the impact of time and channel statistical dependence on throughput achievable by the two MLO policies. To do so, we compare the throughput values obtained from the original traces against the those provided by a simple baseline model, the latter built under the assumption that the temporal activity of each channel is independent and identically distributed (i.i.d.), with an average occupancy value ρ matched from data.

Under such a baseline model, the mean throughput for the three access modes can be approximated as follows:

- SLO throughput is given by $\mathrm{Th}_{\mathrm{SLO}} = (1-\rho_1)L/T$, where ρ_1 is the occupancy of the primary link, L is the packet size (12,000 bits) and T (0.277 ms) is the packet transmission time (0.172 ms) plus a single DIFS and an average backoff duration assuming the link is sensed free all time (DIFS+$\frac{\mathrm{CW}_{\min}}{2}10$ μs+DATA+SIFS+ACK).

- MLO-STR throughput is computed as $\mathrm{Th}_{\mathrm{MLO-STR}} = (2 - \rho_1 - \rho_2)L/T$,
  where ρ_2 is the occupancy of the secondary link.
- MLO-NSTR throughput is computed as $\mathrm{Th}_{\mathrm{MLO-STR}} = (1-\rho_1)(2-\rho_2)L/T$,

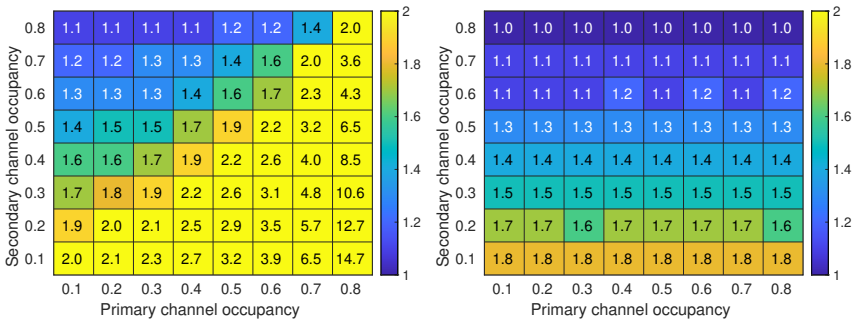
Figure 7b shows the throughput achieved by both our trace-based simulation and our i.i.d. model in terms of 10%, 50% and 90%-tile. For the 90%-tile, our baseline model matches the results closely, while for the 50% and 10%-tile the model predicts higher throughput. These differences for the 50% and 10%-tiles are explained by the frequent backoff interruptions that appear in our trace-based simulation when the links occupancy increases, an aspect not captured by the simple model. In the case of MLO-NSTR, in addition to the previous effect, the observed differences show that the model is also optimistic regarding the probability to find the two links idle since it does not either capture the existing temporal correlation —even if low— between links.

Findings: While MLO-STR is able to leverage the independent occupancy of multiple links to achieve better than additive capacity, MLO-NSTR only reaches near-additive capacity when the two links are almost completely idle, making it less suitable for crowded scenarios.

B. Throughput Gains and Spectrum Occupancy

The benefits of MLO compared to SLO are strongly affected by spectrum occupancy. Importantly, the occupancy of the primary and secondary link affect the MLO state machine quite differently as described in Section II. In light of the above, we conduct separate experiments according to the average per-link occupancy observed on each 1-second sample (as described in Section II-C).

Figure 8 shows the throughput of the two MLO modes normalized to the one attained by SLO, depicted as both numerical values and a heat map. First, we consider Figure 8a and MLO-STR opportunistic and asynchronous access. On the diagonal, both links have nearly identical occupancy and the gain compared to SLO is close to twofold, as expected since the two links are accessed independently. In contrast, when the secondary link has lower occupancy than the primary (bottom right



(a) Throughput gain of MLO-STR (b) Throughput gain of MLO-NSTR

Fig. 8: Average MLO throughput normalized to SLO. The upper bound of the colorbar is set to 2 to highlight ratios between 1 and 2.

of Figure 8a) the throughput gain of MLO-STR vs. SLO is highest, with a maximum factor of 14.7 when the primary and secondary links are busy 80% and 10% of the time, respectively. While the absolute throughput values are omitted from the figure, for this particular case MLO-STR achieves 40.5 Mbps, while SLO achieves only 2.76 Mbps.

Figure 8b depicts the results obtained for MLO-NSTR. Compared to MLO-STR, the gains are dramatically reduced and have a maximum of 1.8, due to the requirement that the primary link be unoccupied. In terms of absolute throughput values, when the primary and secondary links are respectively occupied 10% and 80% of the time, MLO-NSTR achieves a mean throughput of 39 Mbps. This value however plummets to a mere 5 Mbps if the primary/secondary link occupancy is reversed. That is, a busy primary link prevents MLO-NSTR from achieving high throughput, even when availing of an idle secondary link.

Findings: MLO-STR's independent link access can yield over $14\times$ throughput gains compared to SLO by overcoming a densely occupied link and taking advantage of a secondary sparse link. Conversely, the performance of MLO-NSTR is tied to the occupancy of the primary link, as the second link can only be accessed when the primary is available too. As a result, its throughput gain over SLO is at most twofold.

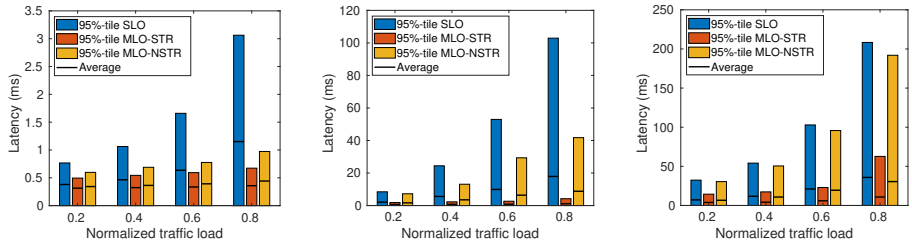
IV. Delay Experiments

A key performance objective of MLO is to improve delay performance, including both average and tail performance such as the delay distribution's 95th percentile. As the above throughput experiments demonstrated the critical role of link occupancy, we partition the study into two cases. First, we study the delay under the assumption that the two links used have similar occupancy, showing how both MLO access modes deal with a symmetrical increase in the occupancy of the two links and diminishing transmission opportunities. Then, we consider a secondary link statistically more occupied than the primary, and study how each MLO mode adapts to a reduced availability of its secondary link.

A. Symmetrically Occupied links

We begin by studying the case of *symmetric* link occupancy, in which both MLO links have channels with similar occupancy levels. In particular, we study epochs when both links have occupancy of about 10%, 40%, and 70%, which we denote as symmetric low, medium, and high occupancy. For SLO, the average throughput with fully backlogged traffic on the single link is 37 Mbps, 22 Mbps, and 6.8 Mbps, respectively. For MLO, we study the delay performance in the three spectrum occupancy cases as the traffic load increases. We consider four traffic loads of $\{0.2, 0.4, 0.6, 0.8\}$ times the SLO throughput, so that all access modes operate in a non-saturated regime, hence allowing to study their delay in comparable conditions.

Figure 9 shows the average and 95th percentile delay for all channel access modes and the different link occupancies. First, observe that when both links have 10% occupancy (Figure 9a), MLO has strikingly improved latency scaling with increasing traffic load compared to SLO. For example, at 20% traffic load, MLO-STR and MLO-NSTR offer a modest decrease in average delay compared to SLO of 17% and 9% respectively. In contrast, when the traffic load is 80%, MLO-STR and MLO-NSTR reduce the average delay by 69% and 62%. This scaling is even more pronounced when analyzing the 95th percentile of delay, in which MLO achieves up to a 78% delay reduction. Thus, for both



(a) 10% occupancy on both links (b) 40% occupancy on both links (c) 70% occupancy on both links

Fig. 9: Latency for symmetrically occupied links vs. normalized traffic load.

average and 95th percentile delay, the benefits of MLO are increasingly pronounced under higher traffic load. Indeed, in this case there are often multiple packets in the buffer such that both interfaces can be used. Moreover, with a relatively low link occupancy of 10%, both links are often available. Next, consider the case that both links have symmetrical medium (40%) occupancy (Figure 9b) and note the change in y-axis scale. Here, SLO's *average* delay increases by nearly an order of magnitude with increasing traffic (i.e., from 2 to 18 ms), whereas the 95th percentile delay increases much more rapidly, exceeding 100 ms. In contrast, MLO-STR can yield a striking order of magnitude reduction in 95th percentile delay compared to SLO. The reason is that with access to either or both links, MLO-STR realizes delay benefits unless both links are occupied.

Unfortunately, unlike MLO-STR, the benefits of MLO-NSTR over SLO are limited and mostly confined to how the average delay scales with traffic load. Indeed, MLO-NSTR can only gain access to the secondary link if the primary link is also idle, implying that the average delay is guaranteed to be lower than the average delay under SLO. However, the 95th percentile delays are triggered by long periods of occupancy of the primary link, thus making any availability of the secondary link during this time irrelevant. As a result, the 95th percentile delay under MLO-NSTR rapidly grows as the normalized traffic load increases.

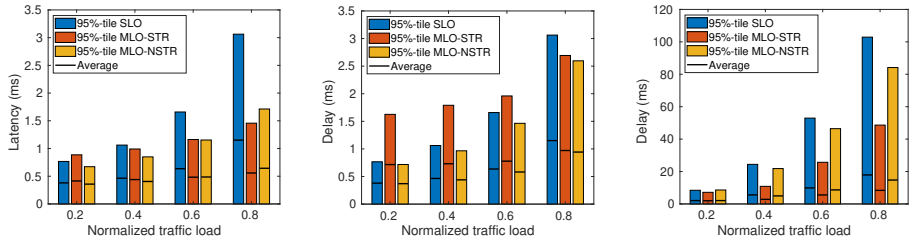
Lastly, when both links have high (70%) occupancy (Figure 9c), MLO-STR again has the most favorable 95th percentile delay scaling with traffic load, providing substantial reductions as compared to both SLO and MLO-NSTR. Nonetheless, at such high link occupancies, even MLO-STR has difficulty finding transmission opportunities on either link, so both mean and 95th percentile delays are increasing. Additionally, MLO-NSTR provides negligible benefits in both average and 95th percentile delay compared to SLO.

Findings: When both links have symmetric medium to high occupancy, MLO-NSTR fails to provide significant 95th percentile delay benefits compared to SLO. The key reason is that MLO-NSTR is only able to realize a benefit compared to SLO if both links are simultaneously unoccupied, an increasingly unlikely occurrence in this scenario. Fortunately, MLO-STR yields significant 95th percentile latency benefits (compared to both SLO and MLO-NSTR) even in the challenging regime of increasing occupancies and traffic. This is because MLO-STR can reduce packet waiting time even when it cannot simultaneously utilize both available links.

B. Asymmetrically Occupied links

We continue by employing the same normalized traffic loads as in the previous section, but consider epochs of asymmetric link occupancy. Between the two links, we present the case that the primary is the less occupied one, and gauge to what extent each MLO mode can exploit an extra (albeit busier) link to reduce the delay. We note that the opposite case—i.e., when the secondary link is on average less busy than the primary—favors both MLO modes, since SLO would always incur a high delay and both MLO modes would take advantage of a more idle secondary link.

Figure 10a depicts the case of a low (10%) primary and medium (40%) secondary link occupancy. As expected, MLO-NSTR always offers lower delay than SLO, with the highest benefits occurring under higher traffic loads. However, MLO-STR surprisingly incurs a higher average and 95th percentile delay than SLO for the lowest normalized traffic load of 0.2. Indeed, MLO-STR starts contention by initializing the



(a) {10%, 40%} occupancy. (b) {10%, 70%} occupancy. (c) {40%, 70%} occupancy.

Fig. 10: Latency for asymmetrically occupied links (indicated as {primary, secondary}) vs. normalized traffic load.

backoff counter as soon as a link is detected to be idle. Unfortunately, such a link may be occupied before the backoff timer expires, thus pausing the backoff counter. If the backoff is paused too often (or for long intervals), the packet could incur even higher delays than it would have if the other link—initially busy—had been selected.

In Figure 10b, this effect is exacerbated due to the even higher occupancy of the secondary link, as selecting an idle secondary link incurs the risk of the latter being occupied before the backoff counter expires. When this occurs, the 95th percentile delay can be twice as high as that with SLO, albeit still confined to below 10 ms. Nonetheless, MLO-STR average and 95th percentile delays grow at a lower rate than those of SLO as the traffic load increases. Indeed, MLO-STR can still take advantage of a secondary link (even when highly occupied) to reduce congestion and curb the latency when it is caused not only by the link occupancy patterns but also by the amount of traffic.

Finally, Figure 10c depicts primary and secondary link occupancy of 40% and 70%, respectively (note the different y-axis scale due to higher load). Similar to the symmetric cases of Figures 9b and 9c, MLO-STR scales well with increasing traffic load, keeping the average delay below that of SLO and decreasing the 95th percentile by up to a half. Compared to Figure 10b, because the primary link occupancy has increased, SLO delay increases more sharply with traffic load, whereas MLO-STR achieves the lowest delays for the same reasons as in the

symmetric case.

Findings: Asymmetric primary vs. secondary link occupancy radically transforms MLO performance. For MLO-STR, despite its uniformly superior performance in symmetrically occupied links, a secondary link that is much busier than the primary can lead to even higher delays than using SLO. This is owed to packets being suboptimally assigned to an interface before carrying out the backoff, with the latter likely to be interrupted on the busier link. This effect is exacerbated when the difference between link occupancies increases.

V. MLO-STR Latency Decomposition and Reduction

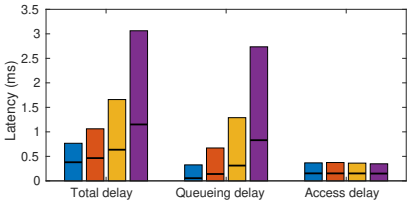
Given MLO-STR's superior throughput performance and delay performance under symmetrically occupied links, we proceed to study the origins of the surprisingly worse delay than SLO under asymmetric links. Moreover, we demonstrate how to overcome this limitation with a modification to MLO-STR.

A. Access and Queueing Delay

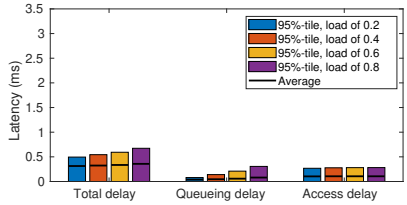
To study MLO-STR's latency performance, we decompose the total MAC delay into *(i)* the *queueing delay*, referring to the time elapsed since a packet arrives to the AP until it is allocated to an interface and *(ii)* the *access delay*, that spans the time between a packet being assigned to an interface and the beginning of its transmission, i.e., the contention time.

Figures 11a and 11b compare the delay under SLO and MLO-STR when using links with *symmetric occupancy* of 10%. Here, access delay remains largely unchanged between the two schemes across all traffic loads. Yet, the same does not hold for queueing delay, which remains low under MLO-STR for increasing load, whereas it rapidly grows for SLO, exceeding the access delay. Specifically, while for SLO the 95th percentile queueing delay can reach up to $9\times$ the access delay, MLO-STR curbs the waiting time and thus the total delay.

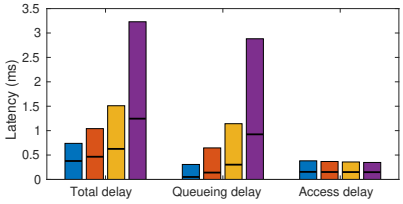
Figures 11c and 11d keep the primary link occupancy to 10% but now consider an *asymmetric* secondary occupancy of 70%. Although MLO-STR reduces the queueing delay with respect to SLO by availing



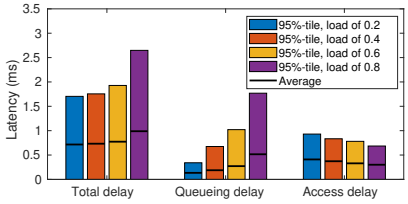
(a) SLO latency for occupancy {10%, 10%}.



(b) MLO-STR latency for occupancy {10%, 10%}.



(c) SLO latency for occupancy {10%, 70%}.



(d) MLO-STR latency for occupancy {10%, 70%}.

Fig. 11: Access and queueing delay for SLO and MLO-STR.

of two interfaces, its access delay is higher than SLO, owing to a subset of packets being transmitted through the secondary link, which incurs a higher average occupancy. This latter effect may outweigh the former, resulting in higher total delay under MLO-STR than under SLO. It can also be observed that MLO-STR access delay decreases as traffic load increases. This is a side effect of the packet allocation process, explainable as follows. When a packet is assigned to the busier—but occasionally idle—secondary interface, it forces multiple subsequent packets through the primary interface as the only option. The latter creates an inherent trade-off in that for every packet assigned to the busier interface, multiple others are assigned to the interface having lower occupancy. As the traffic load increases, this phenomenon becomes more pronounced, leading to more packets being assigned to the lesser-occupied primary link, thus reducing the average and the 95th percentile access delay.

Findings: MLO-STR significantly reduces the time that packets spend waiting in the queue when compared to SLO, for both symmetrically

and asymmetrically occupied links. However, while the access delay is similar when the two links are symmetrically occupied, it can increase for MLO-STR when the secondary link is busier than the primary, sometimes outweighing the benefits of a reduced waiting time. These events of high access delay are the result of assigning the packet to an interface before the backoff starts. Indeed, a packet may be assigned to an interface just before a long link occupancy phase, pausing the backoff counter repeatedly and significantly delaying the packet transmission.

B. MLO-STR with Deferred Decision

We have shown that—and explained why—MLO-STR can lead to even higher delays than SLO in the case of links with different occupancies. To better understand the design space of MLO channel access, we define a minor variation—denoted *MLO-STR with Deferred Decision* (MLO-STR+)—which overcomes this limitation of MLO-STR by deferring the decision about which link to use until the end of the backoff countdown.

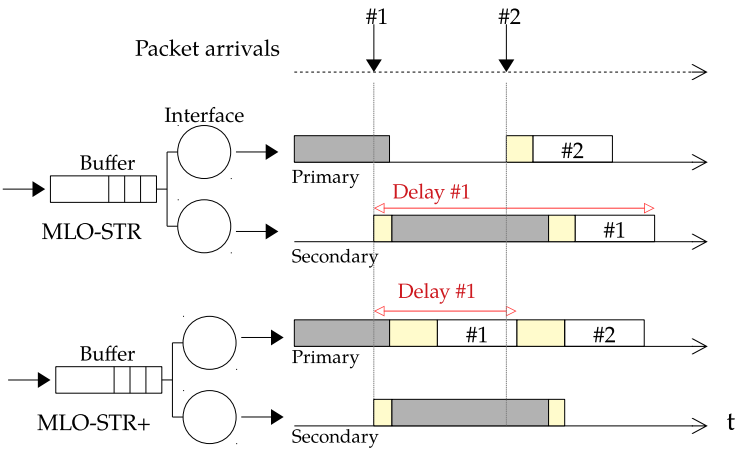
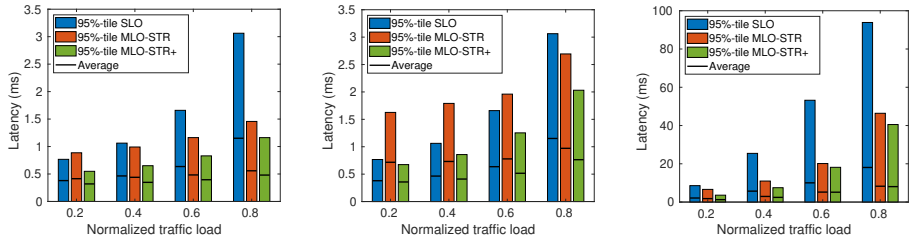


Fig. 12: Illustration of MLO-STR vs MLO-STR+. MLO-STR+ reduces the delay incurred by packet #1 at the expense of that incurred by packet #2.



(a) Link occupancy of $\{10\%, 40\%\}$. (b) Link occupancy of $\{10\%, 70\%\}$. (c) Link occupancy of $\{40\%, 70\%\}$.

Fig. 13: Delay for asymmetric links vs. variable normalized traffic load under SLO, MLO-STR, and MLO-STR+.

The main features of MLO-STR+ are *(i)* running as many backoff instances as radio interfaces while there are packets waiting for transmission, and *(ii)* allocating the first packet waiting to the interface with the backoff counter that expires first. This differs from MLO-STR, in which we allocate packets once links are idle, before running the backoff. Implementing MLO-STR+ requires only minor changes to the Wi-Fi MLO state machine: the ability to control when an interface can initiate, pause, and complete the backoff countdown without actually being allocated a packet. A comparison between MLO-STR and MLO-STR+ operations is illustrated in Figure 12, where it can be observed that the delay experienced by the first packet (the one exhibiting the worst-case delay in MLO-STR), is notably reduced with MLO-STR+ albeit at the expense of increasing the delay of the second packet.

Figure 13 shows the resulting average and 95th percentile delay for MLO-STR+, MLO-STR, and SLO. Observe that MLO-STR+ consistently outperforms both MLO-STR and SLO for both average and 95th percentile delay for all traffic loads and link occupancy combinations. Figure 14 depicts the breakdown of the waiting and access delay of MLO-STR+, showing that MLO-STR+ reduces the average and worst-case access delays while maintaining similar queueing delays than those in Figure 11c. Namely, the 95th percentile of both the access delay and the total delay are reduced by up to 60% when employing MLO-STR+

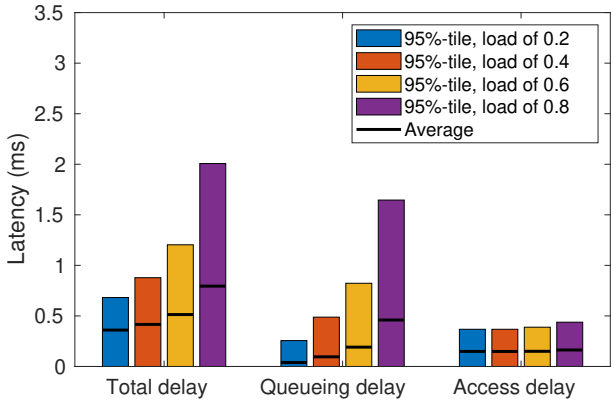


Fig. 14: Delay for MLO-STR+ with $\{10\%, 70\%\}$ occupancy.

instead of MLO-STR.

Findings: MLO-STR+ improves MLO-STR by delaying the link allocation of the packet at the head of the queue until one of the backoff counters expires, thereby leveraging up-to-date information on the link state, and ultimately improving the link allocation decision. The performance of MLO-STR+ also shows how MLO channel access can be improved by jointly controlling the operation of the multiple radio interfaces.

C. Packet jitter

Variance in the delay is an important metric to consider for next-generation networks. We have shown that MLO can be used to reduce network delay, and expect that it can also reduce the variability with which transmissions reach their target.

We analyse the packet jitter for the same cases shown in Figures 10b and 13b, by studying the standard deviation of the delay. Figure 15 shows the jitter for each access method. The jitter increases with traffic load for all access methods, except for MLO-STR, where decreases steadily. The reason for such a behavior in the case of MLO-STR is the high delays some packets suffer when transmitted through the secondary link, which

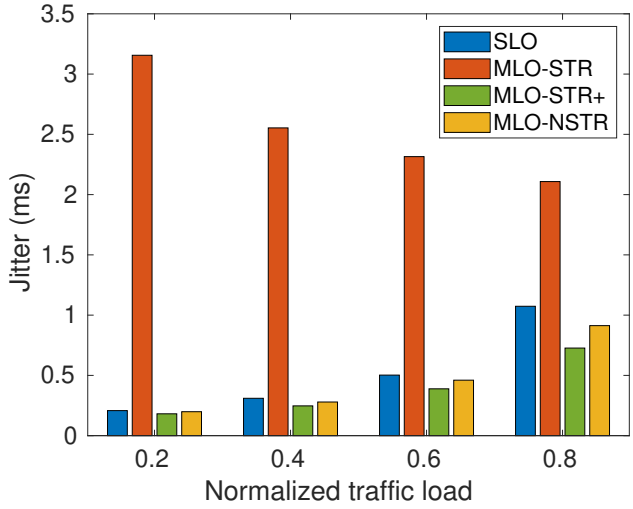


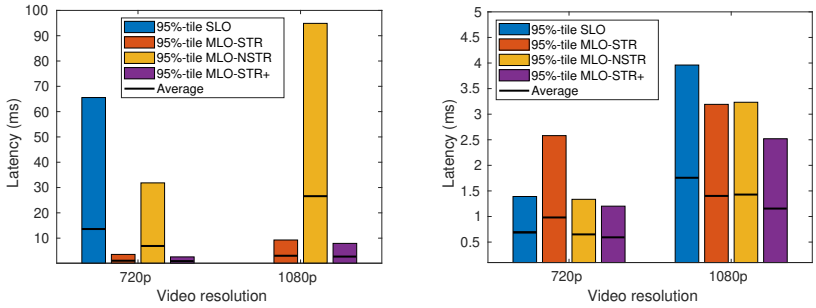
Fig. 15: Packet jitter for $\{10\%, 70\%\}$ occupancy.

at low traffic loads significantly differ from the low delay of the packets transmitted through the primary link. For a traffic load of 0.8, the only point in which SLO has a higher delay than MLO-STR (see Fig. 13b), MLO-STR still shows a jitter that is twice as high as the one for SLO. This high variability in MLO-STR delay further showcases its limitations in assigning packets to interfaces, and the negative effect of allocating packets to a busy secondary link. Similar as in the delay, MLO-STR+ is able to circumvent such a situation by deferring the decision about which link to use at the end of the backoff countdown.

Findings: MLO-STR incurs a high variability in its delay due to the blind allocation of packets to high occupied links, having jitter an order of magnitude higher than any other method at worst. MLO-STR+ and MLO-NSTR keep their jitter below that of SLO, thus offering more consistency in their operation.

VI. Delay Experiments with Real Traffic

The experiments presented so far have used over-the-air channel traces coupled with artificial traffic generated by a Poisson Process with fixed



(a) Primary 40% and secondary 40% occupancy

(b) Primary 10% and secondary 70% occupancy

Fig. 16: Delay vs. video resolution using Google Stadia traffic for (a) symmetric and (b) asymmetric link occupancy.

packet size and variable rate.

Here, we consider actual application traffic traces in order to jointly account for real traffic and real channels. In particular, we employ Google Stadia [24], a latency-sensitive cloud gaming application that streams videogames from Google's servers directly to a user's browser, and use it to generate packet size and arrival times. We consider a single gamer and collect traces corresponding to two different screen resolutions, namely 720p and 1080p, leading to average traffic loads of 10.4 Mbps and 22.1 Mbps, respectively. We use the same previous experimental methodology, generating the traffic (packet arrival time and size) according to the values extracted from the captured Google Stadia's traces. The main difference between Poisson traffic and Google Stadia's traffic is that, in the latter, packets arrive in periodic batches every 1/60 sec following the video frame rate, thus creating short congestion periods. More details about Stadia's traffic properties can be found in [24]–[26] where they have been also employed to generate traffic as done in this paper.

Figure 16a shows the delay performance for the same scenario as in Figure 9b, i.e., the one with symmetric link occupancy of 40%. To facilitate a comparison between Figures 16a and 9b, we note that a

normalized traffic load of 0.6 and 0.8 from Figure 9b is the closest to the traffic load of Stadia's 720p and 1080p resolutions respectively. For video at 720p, MLO-STR achieves a staggering order-of-magnitude reduction in the 95th percentile delay compared to SLO, keeping it well below 10 ms. The delay with MLO-NSTR is also consistently below that of SLO, although significantly underperforming MLO-STR. A resolution of 1080p can only be supported through MLO, with MLO-STR still guaranteeing a delay below 10 ms. Similar results were obtained with Poisson traffic, confirming the ability of MLO-STR to reduce the latency when the two links are symmetrically occupied regardless the traffic characteristics.

Figure 16b shows the delay performance for the asymmetric scenario considered in Figures 10b and 13b, i.e., that with primary and secondary link occupancy of 10% and 70%, respectively. The experimental results obtained using Stadia's traffic at 720p exhibit similar qualitative trends as those employing Poisson traffic. Indeed, MLO-STR increases the delay over SLO by 85.5%, a performance degradation even worse than the 68.6% delay increase observed under Poisson traffic.

Findings: Experiments using real application traces coupled with real channel occupancy traces confirmed our main findings originally obtained under Poisson traffic, namely: *(i)* MLO achieves significant delay reduction over SLO, and may enable new applications whose traffic load cannot otherwise be delivered in a timely manner; *(ii)* MLO-STR can suboptimally allocate packets to a secondary interface that is busier than the primary, occasionally yielding even higher delays than SLO; For MLO-STR, such an effect can be further exacerbated under the real-world traffic considered due to batch packet arrivals; and *(iii)* By deferring the decision on which interface to allocate a packet to, MLO-STR+ yields the lowest delay in all scenarios considered.

VII. Channel Bonding

In previous sections, MLO was using twice the bandwidth of SLO, which certainly contributes to the observed gains. In this section, we use channel bonding to equalize the amount of spectrum bandwidth used by SLO and MLO, so as to better study the MLO performance

gains stemming from contending over multiple narrow links vs. using a single wide one.

Channel bonding with preamble puncturing allows the use of multiple non contiguous 20 MHz channels. Each link uses an 80 MHz channel, containing four 20 MHz channels. From these, one is assigned as the primary, where backoff is performed, and all others are considered secondary. Prior to the backoff expiring, all secondary channels are checked, and all available channels are used for transmission (e.g., if three 20 MHz channels are open, transmission happens over 60 MHz). The same implementation of channel bonding with preamble puncturing used in [23] is considered, supporting also allocating multiple resource units to a single user as defined by IEEE 802.11be [7].

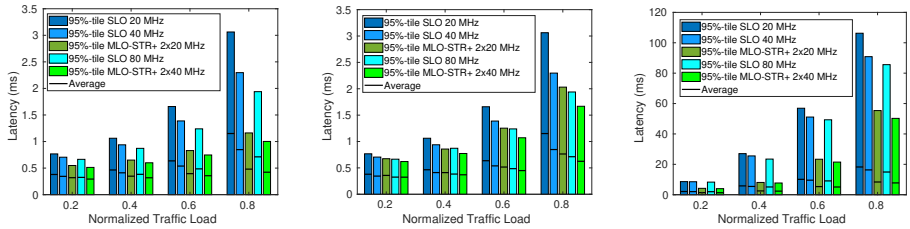
We employ the same channels previously used, 36 and 100, as the primary channels of each link, and the subsequent channels as the secondary channels. For SLO, we use 40 MHz links (channels 36 and 40), and 80 MHz (36, 40, 44, and 48) links, while for MLO, we use 20 MHz (36 and 100) and 40 MHz (36-40 and 100-104) links, respectively. We use the same traffic loads as in the previous sections.

A. Same bandwidth for SLO and MLO

Previously, we used two 20 MHz links for MLO, thus doubling the bandwidth of the SLO link. In this section we study if SLO can achieve similar gains as MLO by just doubling its channel bandwidth, and thus both schemes use the same amount of spectrum.

We focus on the asymmetric link case only, under the same conditions as in previous sections. To generate the wider links, we consider the adjacent channels from the same dataset sample used in previous experiments so as to keep the existing temporal correlation between them.

Figure 17 shows the average and 95th percentile delay for an increasing traffic load with different link bandwidths. As expected, using a link bandwidth of only 20 MHz leads to the highest delays for all schemes. Then, increasing the link bandwidth to 40 MHz leads to a delay decrease of up to 25% in SLO, and 17.8% in MLO-STR+. Comparing SLO performance at 40 MHz with MLO-STR+ using 2 links of 20 MHz



(a) Link occupancy of {10%, 40%}

(b) Link occupancy of {10%, 70%}

(c) Link occupancy of {40%, 70%}

Fig. 17: Latency for asymmetric links vs. variable normalized traffic load under different channel bandwidth.

each, we can observe that MLO-STR+ continues to outperform SLO. The same is observed for SLO 80 MHz and MLO-STR+ using 2 links of 40 MHz each. Moreover, we can observe that in Figures 17a and 17c MLO-STR+ with 20 MHz links achieves lower delays than SLO using a 80 MHz channel, i.e., MLO-STR+ improves SLO performance using half of the SLO bandwidth in total when the occupancy of the two links does not differ excessively. Otherwise, as shown in Figure 17b, this does not hold in the 10%-70% case since the secondary link is too busy, requiring to use the same bandwidth, i.e., two links of 40 MHz each, to outperform SLO.

Finding: The use of higher bandwidth links leads to a decrease in the delay for both SLO and MLO. However, using two links of lower bandwidth, each with its own backoff, still leads to better results than a single link with higher bandwidth, showing that MLO-STR+ achieves its performance mainly due to running multiple backoff counters instead of one.

B. Primary Channel Selection

Channel bonding performance can vary depending on the primary channel used [23]. By allowing SLO to use wider channels than MLO to keep the same total bandwidth, SLO has more opportunities to find an emptier primary channel that could lead to lower delays than MLO.

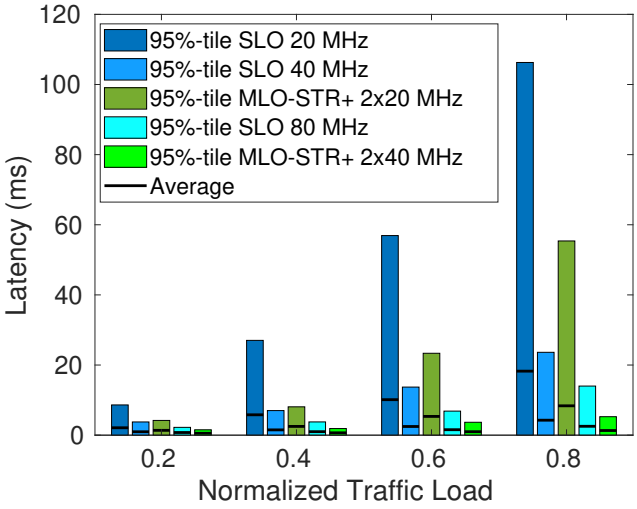


Fig. 18: Latency for occupancy of {40%, 70%} with dynamic primary channel selection

We use the same setup as in the previous section, considering 40 and 80 MHz links, but each link is now free to select the less occupied 20 MHz channel as its primary channel in every spectrum sample.

Fig. 18 shows the delay for different channel widths when the less occupied channel of each link is selected as primary channel. Comparing the results with Fig. 17c, where the primary channel was fixed for each link, we can observe a significant delay reduction. For SLO, the delay for 40 and 80 MHz is reduced by a factor 3.8 and 6.1, respectively, compared to the case where the primary channel is fixed. Similarly, for MLO, for a 40 MHz link, the delay is reduced by a factor 9.5. Overall, a significant gain in delay is observed for both SLO and MLO-STR+ in Fig. 18 by allowing dynamic primary channel selection. It is specially remarkable that SLO 40 MHz results in lower latency than MLO-STR 2x20 MHz. The reason is that SLO is able to leverage the existence of channel 40, with an occupancy lower than the one of channel 36.

Findings: Channel bonding and MLO operation both require careful selection of the channels used. A dynamic choice in the primary channel

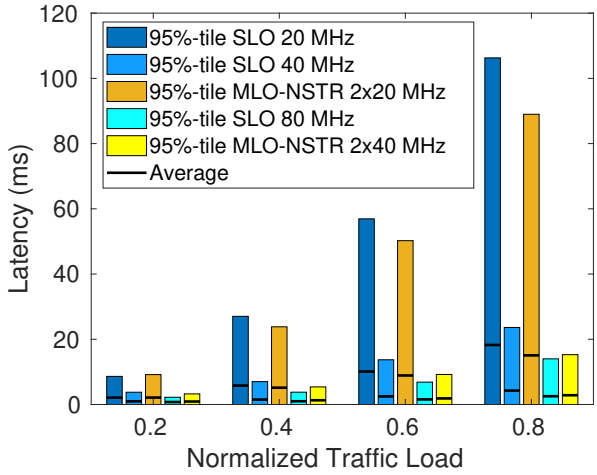


Fig. 19: Latency for occupancy of {40%, 70%} with dynamic primary channel selection

shows up to 6.1× and 9.5× lower delays in SLO (80 MHz) and MLO-STR+ (2x40 MHz) respectively. Remarkably, there are cases where SLO can outperform MLO-STR+ when the extra available choices to set the primary channel allow it to find a much emptier channel than the ones used by MLO-STR+. Moreover, and focusing only on MLO, allowing to dynamically select the best primary channel in each link multiplies the latency improvements obtained by contending over multiple independent links, yielding almost an order of magnitude decrease compared to a static choice.

C. Channel bonding and MLO-NSTR

We now focus on MLO-NSTR, and study its performance using channel bonding with primary channel selection in the same conditions as MLO-STR+ in previous section. As MLO-NSTR's behavior is very similar to channel bonding (both using primary and secondary hierarchies), we would expect NSTR and SLO behavior to be similar if the bandwidths used are equal. Figure 19 shows the MLO-NSTR delay when

the lowest occupancy primary 20 MHz channel is selected for each link. Much like Figure 18 for MLO-STR+, MLO-NSTR can greatly benefit from channel bonding, with its delay being reduced by a factor of up to 5.8× when switching from 20 MHz to 40 MHz links. However, SLO with a bandwidth of 80 MHz results in lower delays than 2 links of 40 MHz with MLO-NSTR, while MLO-STR+ still outperforms SLO. The reason is that SLO with an 80 MHz bandwidth uses 4 channels of 40% occupancy, while MLO-NSTR has a secondary link using channels with 70% occupancy, limiting the amount of times MLO-NSTR can transmit simultaneously, and the total effective bandwidth used.

Findings: The use of a single backoff counter in its primary link prevents MLO-NSTR to benefit from parallel transmissions on both links, thus achieving a similar delay reduction as SLO in the best case.

VIII. Related Work

Multiple Radios: The use of multiple radios or links has been studied for a variety of technologies and protocols. 5G Multi-Connectivity of a single device to multiple other technologies such as LTE and Wi-Fi has been investigated as a way to achieve higher capacity and meet ultra-reliability constraints [27]–[29]. In [30], the importance of link homogeneity is studied, showing that using links that are too far apart in terms of latency will result in no gains over single link communication. Multi-Path TCP has also been considered as a way to enable the use of TCP over multiple connections of Wi-Fi, 3G and LTE [31]–[34]. In [35], a framework for Multi-Path with multi-connectivity across different networks is presented, which also shows a decrease in bitrate when links have differences in their latencies. In [36], a Multi-Path implementation for QUIC is presented, also discussing the impact of heterogeneous links, and how adding high latency secondary paths can lead to worse performance.

The use of multiple radios focusing only on Wi-Fi deployments has been studied as well, seeking to improve reliability [20] and reduce latency [21] through cooperative links. Handoff delays can also be reduced by dedicating one radio to data exchange while another one serves management frames [37]–[39]. Insights learned through such works have

provided the basis for the current Wi-Fi standardization efforts to support multiple radios and Multi-link Operation in IEEE 802.11be.

Multi-Link Operation: Multi-Link Operation —as the key feature of IEEE 802.11be— has already received significant attention from the Wi-Fi community. The feasibility of the Simultaneous Transmission and Reception mode depending on the amount of cross-link interference is studied in [14], assessing the minimum spectral distance required between links, and showing that 100 MHz is enough to ensure proper packet reception, validating the 200 MHz separation between channels considered in our work. The evaluation of the impact that multi-link transmission has on the worst case latency for real time applications is shown in [18], showing that only using two links already leads to an order of magnitude delay reduction in the 90th percentile delay in some cases, which is confirmed by our results. STR delay for industrial and latency-bound settings is also investigated in [19], showing that a secondary link leads to halving the average delay and almost halving the worst case delay.

Coexistence and Traffic Differentiation: The interplay between SLO and MLO devices and its impact on latency is studied in [40] and [41]. NSTR coexistence with legacy devices is also studied in [42], [43]. The interplay between multiple STR nodes is studied in [44] Different implementations of Multi-Link Operation are studied in [12], [13], [45], analysing the impact that each of them has on the WLAN throughput, the latter finding a throughput increase of 200% and 80% for STR and NSTR over SLO, respectively, which aligns with our findings of MLO-STR with symmetrical occupancy, and MLO-NSTR for a primary of 10% occupancy . Finally, traffic allocation policies using multiple links are considered in [15], and an adaptation of EDCA with MLO NSTR devices can be found in [16] and in [46]. Real time application traffic is also discussed in [17], where different frequency resource allocation schemes for MLO are proposed.

Performance Analysis: Most of the cited references have studied MLO performance through simulations, although there are also some cases in which new analytical models are derived to study MLO performance. With the exception of [26] where the focus is placed on the delay analysis under finite load conditions, all other papers [12], [13],

[45]–[47] focus on the MLO efficiency under saturation (full-buffer) throughput conditions.

To the best of our knowledge, this paper is the first work studying the performance of Multi-Link Operation using real spectrum occupancy measurements. While our results confirm the potential latency gains of MLO seen in the state of the art, the use of real spectrum measurements offers new and unique insights on MLO performance otherwise not possible.

IX. Conclusions

To the best of our knowledge, this paper is the first experimental study of throughput and latency for MLO. Using a dataset containing real-world channel occupancy measurements in the 5 GHz spectrum, we studied throughput and latency performance of two MLO channel access modes, MLO-STR and MLO-NSTR. We demonstrated that MLO can enable new latency-sensitive applications, whose traffic load cannot otherwise be delivered in a timely manner through SLO. We showed that when both links are similarly occupied, both MLO modes can reduce the 95th percentile latency by nearly one order of magnitude. In contrast, with asymmetrically occupied links, we surprisingly found that MLO-STR, the mode with superior throughput performance, can sometimes yield higher worst-case latency than SLO. We proposed a *deferred decision* enhancement to MLO-STR that overcomes this limitation. We studied performance on traffic from a real delay-sensitive gaming application to couple real channel experiments with real application experiments. Finally, we showed how the gains attained by MLO can further grow when using channel bonding.

An important extension of this work would attempt at capturing the behavior of various MLO modes analytically, rather than via simulations, thereby allowing a more generalized comparison and wide reproducibility of the results. We refer the reader to [26] for a first attempt at the latter, and to [44], [45] for fresh summaries of the MLO standardization process. With Wi-Fi 7 defined and MLO up and running, beyond-802.11be technologies are expected to operate in new frequency bands [48] and/or augment the spatial reuse of the old ones through advanced

AP coordination [6]. Looking ahead, any new features being introduced in Wi-Fi 8 [49] should be conceived atop MLO, and their performance studied when paired with the latter.

REFERENCES

[1] Marc Carrascosa, Giovanni Geraci, Edward Knightly, and Boris Bellalta. An Experimental Study of Latency for IEEE 802.11 be Multi-link Operation. In *IEEE ICC 2022- IEEE International Conference on Communications*, pages 2507–2512. IEEE, 2022.

[2] Toni Adame, Marc Carrascosa-Zamacois, and Boris Bellalta. Time-sensitive networking in IEEE 802.11 be: On the way to low-latency WiFi 7. *Sensors*, 21(15):4954, 2021.

[3] Changhua Pei, Youjian Zhao, Guo Chen, Ruming Tang, Yuan Meng, Minghua Ma, Ken Ling, and Dan Pei. WiFi can be the weakest link of round trip network latency in the wild. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2016.

[4] IEEE P802.11be/D1.0 Draft 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 8: Enhancements for extremely high throughput (EHT), May 2021.

[5] Alvaro López-Raventós and Boris Bellalta. Multi-link Operation in IEEE 802.11 be WLANs. *IEEE Wireless Communications*, 2022.

[6] Adrian Garcia-Rodriguez, David Lopez-Perez, Lorenzo Galati-Giordano, and Giovanni Geraci. IEEE 802.11 be: Wi-Fi 7 Strikes Back. *IEEE Communications Magazine*, 59(4):102–108, 2021.

[7] Evgeny Khorov, Ilya Levitsky, and Ian F Akyildiz. Current status and directions of IEEE 802.11 be, the future Wi-Fi 7. *IEEE Access*, 8:88664–88688, 2020.

[8] Roger Pierre Fabris Hoefel. IEEE 802.11 be: Throughput and Reliability Enhancements for Next Generation WI-FI Networks. In *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–7. IEEE, 2020.

[9] Mao Yang and Bo Li. Survey and perspective on extremely high throughput (EHT) WLAN—IEEE 802.11 be. *Mobile Networks and Applications*, 25(5):1765–1780, 2020.

[10] David López-Pérez, Adrian Garcia-Rodriguez, Lorenzo Galati-Giordano, Mika Kasslin, and Klaus Doppler. IEEE 802.11 be extremely high throughput: The next generation of Wi-Fi technology beyond 802.11 ax. *IEEE Communications Magazine*, 57(9):113–119, 2019.

[11] Cailian Deng, Xuming Fang, Xiao Han, Xianbin Wang, Li Yan, Rong He, Yan Long, and Yuchen Guo. IEEE 802.11 be Wi-Fi 7: New challenges and opportunities. *IEEE Communications Surveys & Tutorials*, 22(4):2136–2166, 2020.

[12] Taewon Song and Taeyoon Kim. Performance Analysis of Synchronous Multi-Radio Multi-Link MAC Protocols in IEEE 802.11 be Extremely High Throughput WLANs. *Applied Sciences*, 11(1):317, 2021.

[13] Mao Yang, Bo Li, Zhongjiang Yan, and Yuan Yan. AP Coordination and Full-duplex enabled Multi-band Operation for the Next Generation WLAN: IEEE 802.11 be (EHT). In *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, pages 1–7. IEEE, 2019.

[14] Ilya Levitsky, Yaroslav Okatev, and Evgeny Khorov. Study on Simultaneous Transmission and Reception on Multiple Links in IEEE 802.11 be networks. In *2020 International Conference Engineering and Telecommunication (En&T)*, pages 1–4. IEEE, 2020.

[15] Álvaro López-Raventós and Boris Bellalta. IEEE 802.11 be multi-link operation: When the best could be to use only a single interface. In *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 1–7. IEEE, 2021.

[16] Hyunhee Park and Cheolwoo You. Latency Impact for Massive Real-Time Applications on Multi Link Operation. In *2021 IEEE Region 10 Symposium (TENSYMP)*, pages 1–5. IEEE, 2021.

[17] DV Bankov, AI Lyakhov, EM Khorov, and KS Chemrov. On the Use of Multilink Access Methods to Support Real-Time Applications in Wi-Fi Networks. *Journal of Communications Technology and Electronics*, 66(12):1476–1484, 2021.

[18] Gaurang Naik, Dennis Ogbe, and Jung-Min Jerry Park. Can Wi-Fi 7 support real-time applications? On the impact of multi link aggregation on latency. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[19] Guillermo Lacalle, Iñaki Val, Oscar Seijo, Mikel Mendicute, Dave Cavalcanti, and Javier Perez-Ramirez. Analysis of latency and reliability improvement with multi-link operation over 802.11. In *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, pages 1–7. IEEE, 2021.

[20] Nick Schwarzenberg, Albrecht Wolf, Norman Franchi, and Gerhard Fettweis. Quantifying the gain of multi-connectivity in wireless LAN. In *2018 European Conference on Networks and Communications (EuCNC)*, pages 16–20. IEEE, 2018.

[21] Yoshihisa Kondo, YOMO Hiroyuki, and Hiroyuki Yokoyama. A Low Latency Transmission Control for Multi-link WLAN. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–6. IEEE, 2020.

[22] Sergio Barrachina-Muñoz, Boris Bellalta, and Edward Knightly. Wi-Fi All-Channel Analyzer. In *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*, pages 72–79, 2020.

[23] Sergio Barrachina-Muñoz, Boris Bellalta, and Edward W Knightly. Wi-fi channel bonding: An all-channel system and experimental study from urban hotspots to a sold-out stadium. *IEEE/ACM Transactions on Networking*, 29(5):2101–2114, 2021.

[24] Marc Carrascosa and Boris Bellalta. Cloud-gaming: Analysis of google stadia traffic. *Computer Communications*, 188:99–116, 2022.

[25] Boris Bellalta. On the Low-latency Region of Best-effort Links for Delay-Sensitive Streaming Traffic. *IEEE Communications Letters*, 25(3):970–974, 2020.

[26] Boris Bellalta, Marc Carrascosa, Lorenzo Galati-Giordano, and Giovanni Geraci. Delay Analysis of IEEE 802.11be multi-link operation under finite load. *IEEE Wireless Communications Letters*, 2023.

[27] Roman Odarchenko, Rui Aguiar, Baruch Altman, and Yevgeniya Sulema. Multilink approach for the content delivery in 5G networks. In *2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, pages 140–144. IEEE, 2018.

[28] Subramanya Chandrashekar, Andreas Maeder, Cinzia Sartori, Thomas Höhne, Benny Vejlgaard, and Devaki Chandramouli. 5G multi-RAT multi-connectivity architecture. In *2016 IEEE International Conference on Communications Workshops (ICC)*, pages 180–186. IEEE, 2016.

[29] Nageen Himayat, Shu-ping Yeh, Ali Y Panah, Shilpa Talwar, Mikhail Gerasimenko, Sergey Andreev, and Yevgeni Koucheryavy. Multi-radio heterogeneous networks: Architectures and performance. In *2014 International Conference on Computing, Networking and Communications (ICNC)*, pages 252–258. IEEE, 2014.

[30] Marie-Theres Suer, Christoph Thein, Hugues Tchouankem, and Lars Wolf. Impact of link heterogeneity and link correlation on Multi-Connectivity scheduling schemes for reliable Low-Latency communication. In *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2020.

[31] Shuo Deng, Ravi Netravali, Anirudh Sivaraman, and Hari Balakrishnan. WiFi, LTE, or both? Measuring multi-homed wireless internet performance. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 181–194, 2014.

[32] Kien Nguyen, Yusheng Ji, and Shigeki Yamada. A cross-layer approach for improving WiFi performance. In *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 458–463. IEEE, 2014.

[33] Christoph Paasch, Gregory Detal, Fabien Duchene, Costin Raiciu, and Olivier Bonaventure. Exploring mobile/WiFi handover with multipath TCP. In *Proceedings of the 2012 ACM SIGCOMM workshop on Cellular networks: operations, challenges, and future design*, pages 31–36, 2012.

[34] Shiva Raj Pokhrel and Michel Mandjes. Improving multipath TCP performance over WiFi and cellular networks: An analytical approach. *IEEE Transactions on Mobile Computing*, 18(11):2562–2576, 2018.

[35] Markus Amend, Eckard Bogenfeld, Milan Cvjetkovic, Veselin Rakocevic, Marcus Pieska, Andreas Kassler, and Anna Brunstrom. A framework for multiaccess support for unreliable internet traffic using multipath dccp. In *2019 IEEE 44th Conference on Local Computer Networks (LCN)*, pages 316–323. IEEE, 2019.

[36] Tobias Viernickel, Alexander Froemmgen, Amr Rizk, Boris Koldehofe, and Ralf Steinmetz. Multipath QUIC: A deployable multipath transport protocol. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2018.

[37] Sunggeun Jin and Sunghyun Choi. A seamless handoff with multiple radios in IEEE 802.11 WLANs. *IEEE Transactions on Vehicular Technology*, 63(3):1408–1418, 2013.

[38] Kishore Ramachandran, Sampath Rangarajan, and John C Lin. Make-before-break mac layer handoff in 802.11 wireless networks. In *2006 IEEE International Conference on Communications*, volume 10, pages 4818–4823. IEEE, 2006.

[39] Vladimir Brik, Arunesh Mishra, and Suman Banerjee. Eliminating handoff latencies in 802.11 WLANs using multiple radios: Applications, experience, and evaluation. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 27–27, 2005.

[40] Marc Carrascosa-Zamacois, Lorenzo Galati-Giordano, Anders Jonsson, Giovanni Geraci, and Boris Bellalta. Performance and Coexistence Evaluation of IEEE 802.11 be Multi-link Operation. In *Proceedings of the 2023 IEEE Wireless Communications and Networking Conference* , pages 1–6, 2023.

[41] Shubhodeep Adhikari and Sindhu Verma. Analysis of Multilink in IEEE 802.11 be. *IEEE Communications Standards Magazine*, 6(3):52–58, 2022.

[42] Wisnu Murti and Ji-Hoon Yun. Multilink Operation in IEEE 802.11 be Wireless LANs: Backoff Overflow Problem and Solutions. *Sensors*, 22(9):3501, 2022.

[43] Nikolay Korolev, Ilya Levitsky, Ivan Startsev, Boris Bellalta, and Evgeny Khorov. Study

of Multi-link Channel Access without Simultaneous Transmit and Receive in IEEE 802.11 be Networks. *IEEE Access*, 2022.

[44] Marc Carrascosa-Zamacois, Giovanni Geraci, Lorenzo Galati-Giordano, Anders Jonsson, and Boris Bellalta. Understanding Multi-link Operation in Wi-Fi 7: Performance, Anomalies, and Solutions. *arXiv preprint arXiv:2210.07695*, 2022.

[45] Cheng Chen, Xiaogang Chen, Dibakar Das, Dmitry Akhmetov, and Carlos Cordeiro. Overview and Performance Evaluation of Wi-Fi 7. *IEEE Communications Standards Magazine*, 6(2):12–18, 2022.

[46] Nikolay Korolev, Ilya Levitsky, Ivan Startsev, Boris Bellalta, and Evgeny Khorov. Study of Multi-link Channel Access without Simultaneous Transmit and Receive in IEEE 802.11be Networks. *IEEE Access*, pages 1–1, 2022.

[47] Nikolay Korolev, Ilya Levitsky, and Evgeny Khorov. Analytical Model of Multi-link Operation in Saturated Heterogeneous Wi-Fi 7 Networks. *IEEE Wireless Communications Letters*, 2022.

[48] Ehud Reshef and Carlos Cordeiro. Future Directions for Wi-Fi 8 and Beyond. *IEEE Communications Magazine*, pages 1–7, 2022.

[49] Lorenzo Galati-Giordano, Giovanni Geraci, Marc Carrascosa, and Boris Bellalta. What Will Wi-Fi 8 Be? A Primer on IEEE 802.11bn Ultra High Reliability. *arXiv preprint arXiv:2303.10442*, 2023.

An Experimental Study of Latency for IEEE 802.11be Multi-link Operation

Marc Carrascosa[⋆], Giovanni Geraci[⋆]
Edward Knightly[♭], and Boris Bellalta[⋆]

[⋆]*Dept. Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona*
[♭]*Dept. Electrical and Computer Engineering and Computer Science, Rice University, Houston, TX*

Abstract

Will Multi-Link Operation (MLO) be able to improve the latency of Wi-Fi networks? MLO is one of the most disruptive MAC-layer techniques included in the IEEE 802.11be amendment. It allows a device to use multiple radios simultaneously and in a coordinated way, providing a new framework to improve the WLAN throughput and latency. In this paper, we investigate the potential latency benefits of MLO by using a large dataset containing 5 GHz spectrum occupancy measurements. Experimental results show that when the channels are symmetrically occupied, MLO can improve latency by one order of magnitude. In contrast, in asymmetrically occupied channels, MLO can sometimes be detrimental and increase latency. This is a result of packets being assigned to an interface before carrying out the backoff, which is more likely to be interrupted on the busier link. We overcome this issue by allowing multiple backoffs to run in parallel, assigning the packet to the particular interface where the backoff expires first, which also achieves lower latency overall.

M. Carrascosa and B. Bellalta were supported by WINDMAL PGC2018-099959-B-I00 (MCIU/AEI/FEDER,UE) and Cisco.

G. Geraci was supported by MINECO's Project RTI2018-101040 and by a "Ramón y Cajal" Fellowship from the Spanish State Research Agency.

E. Knightly's research was supported by Cisco, Intel, and NSF Grants CNS-1955075 and CNS-1923782.

I. Introduction

The importance of wireless connectivity in a globalized society is unquestionable, and forced lockdowns reminded us how dependable Wi-Fi is. We resorted to Wi-Fi to be in touch with our loved ones, to make online purchases, and to get work done and keep the economy afloat. In a post-pandemic world, Wi-Fi technologies will be vital for accessing fair and remote-friendly education, medical care, and business opportunities in the unlicensed spectrum. There will be nearly 628 million public Wi-Fi hotspots by 2023 [1], one out of ten equipped with Wi-Fi 6 based on the IEEE 802.11ax amendment [2].

As the popularity of Wi-Fi grows, so does the demand for augmented data rates, higher reliability, and lower latency, driving the development of a new Wi-Fi 7 generation based on the IEEE 802.11be Extremely High Throughput (EHT) specification [3]–[7]. Despite its name, Wi-Fi 7 will be chasing much more than peak throughput. Indeed, the 802.11be Task Group acknowledges the need for lower delays to enable delay-sensitive networking use cases, including augmented and virtual reality, cloud computing, and cross-factory floor communications in next-generation enterprises [8]–[12].

In a quest for lower delays, one of the most disruptive features being proposed for 802.11be is Multi-Link Operation (MLO) [13]–[16]. In MLO, devices can make simultaneous use of different channels or bands, potentially allowing delay-sensitive traffic to be transmitted through multiple links to ensure its timely reception. With its standardization process being consolidated, and prompted by the increasing interest from the research community [17]–[20], a fundamental question arises as to whether and to what extent MLO can reduce Wi-Fi latency in real-world scenarios.

In this paper, capitalizing on over-the-air measurements of spectrum occupancy for the entire 5 GHz band recently collected [21], [22] and freely available in open source[1], we experimentally investigate the la-

[1]WACA dataset: https://github.com/sergiobarra/WACA_WiFiAnalyzer.

tency[2] performance of 802.11be MLO. Atop these traces, which include scenarios with high Access Point (AP) density and crowded environments and span multiple hours, we develop an emulation tool that fuses a Wi-Fi MLO state machine with the high-resolution spectrum measurements. Besides legacy Wi-Fi Single-Link Operation (SLO), we study the two MLO channel access modes currently under consideration by the IEEE 802.11be Task Group [3], [16]: (i) MLO-STR, where two radio interfaces are operated independently, and (ii) MLO-NSTR, where one interface acts as primary and the other as secondary. Our main contributions can be summarized as follows:

- We show that when using two links with statistically symmetrical occupancy, MLO reduces 95th percentile latency by up to an order of magnitude with respect to SLO by availing of a second radio interface.
- In contrast, we surprisingly discover that when using two links with asymmetrical occupancy, MLO-STR can sometimes worsen the latency performance with respect to SLO. In the worst case, we observe an increase of up to 112% in terms of 95th percentile latency.
- To overcome the aforementioned issue, we consider a minor variation of STR, denoted MLO-STR+, that allows to run in parallel as many backoff instances as interfaces. Then, MLO-STR+ simply allocates the first packet waiting for transmission to the interface whose backoff expires first. This way, STR+ guarantees same delay as or lower than SLO, with reductions of up to 70% in the best observed case.

II. Multi-radio Multi-link Operation

IEEE 802.11be considers two main channel access methods to support Multi-link Operation: Simultaneous Transmit and Receive (MLO-STR), and Non-simultaneous Transmit and Receive (MLO-NSTR) [3], [16]. We introduce them in the following, from the perspective of an AP equipped with two radio interfaces and thus able to operate on two different channels simultaneously:

[2]The terms latency and delay are used interchangeably throughout the paper.

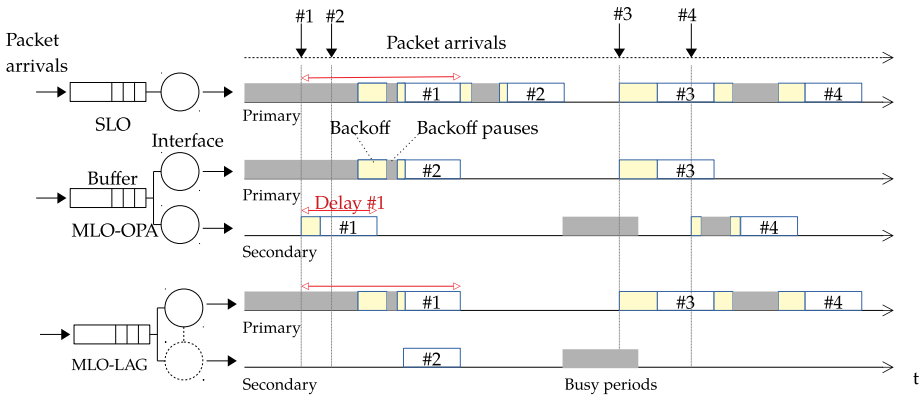


Fig. 1: Illustration of SLO, MLO-STR, and MLO-NSTR operations. Grey, yellow, and white bars respectively indicate occupied channels, random backoffs, and packet transmissions. Packet transmissions include both the data part and the corresponding ACK, as well as DIFS and SIFS inter-frame spaces.

- **MLO-STR:** The two radio interfaces operate independently and asynchronously, and a packet waiting for transmission is allocated to a radio interface as soon as the latter becomes available. If both radio interfaces are available, the packet is randomly allocated to either. Once an interface is allocated a packet, it starts channel contention by initializing a backoff instance.
- **MLO-NSTR:** One interface acts as primary, and the other as secondary. When there are packets waiting for transmission, the primary interface undergoes contention to access the channel through a backoff counter. Once the backoff counter reaches zero, packets are sent through both interfaces if the secondary one has been idle for at least a PIFS interval. Otherwise, only the primary interface is used to transmit.

Besides the MLO modes, IEEE 802.11be also considers the conventional Single-link Operation, where an AP is equipped with only one radio interface.

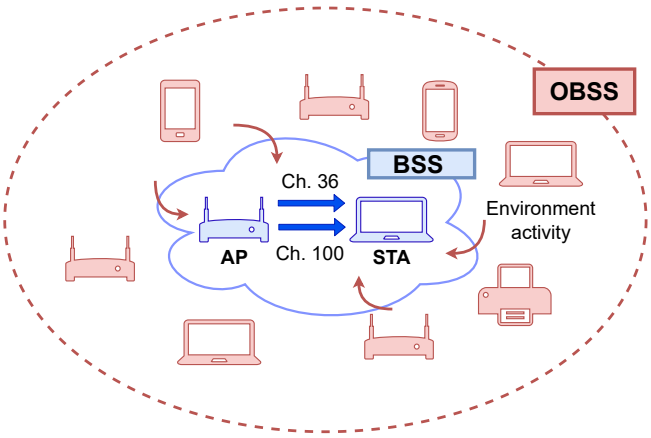


Fig. 2: Scenario considered. The WACA dataset is used to characterize the environment activity (red) observed by the target BSS (blue) on channels 36 and 100 in the 5 GHz band.

Figure 1 exemplifies SLO, MLO-STR, and MLO-NSTR operations. SLO follows default Wi-Fi operations, where packets are sequentially transmitted. In the case of MLO-STR, arriving packets are allocated to whichever interface becomes available first. This results in a significant delay reduction for packets #1, #2 and #4. In the case of MLO-NSTR, the secondary channel's dependence on the primary sometimes prevents efficiently using the two radio interfaces. As a result, and unlike MLO-STR, the delay for packets #1 and #4 cannot be reduced with respect to SLO.

III. Experimental Setup

In this work, we consider a target WLAN Basic Service Set (BSS) consisting of one AP and one station (STA), both equipped with two Wi-Fi interfaces each operating in the 5 GHz band on channels 36 and 100, respectively denoted *primary* and *secondary*. We refer to this BSS as MLO-BSS. On these channels, the target MLO-BSS observes the environment activity, i.e., the transmissions generated by Orthogonal Basic

Service Sets (OBSS). The MLO-BSS and OBSS under consideration are illustrated in Figure 2 in blue and red, respectively. For this setup, we consider the three modes of operation described in Section II, namely: (i) SLO, where only the primary channel interface is available; (ii) MLO-STR, where both interfaces are available and work independently; and (iii) MLO-NSTR, where both interfaces are available but usage of the secondary channel is conditioned on the primary also being unoccupied. For the above scenario, we consider downlink traffic, i.e., from the AP

Name	**Variable**	**Value**
Legacy preamble	$T_{\text{PHY-legacy}}$	20 μs
HE single-user preamble	$T_{\text{PHY-HE-SU}}$	52 μs
OFDM symbol duration	σ	16 μs
OFDM legacy symbol dur.	σ_{Legacy}	4 μs
Short InterFrame Space	SIFS	16 μs
DCF InterFrame Space	DIFS	30 μs
Slot time	T_0	10 μs
Service field	L_{SF}	32 bits
MAC header	L_{MH}	272 bits
Tail bits	L_{TB}	6 bits
ACK bits	L_{ACK}	112 bits
Frame size	L	12000 bits

TABLE I: Notation and Wi-Fi state machine parameters.

to the STA. We assume packet arrivals to follow a Poisson process, and transmitted packets to have a constant size of $L = 12000$ bits. Table I summarizes the main parameters used in the Wi-Fi state machine.

A. WACA Dataset

In order to evaluate the latency of 802.11be MLO in a real-world setting, we employ the WACA dataset, containing over-the-air measurements of the 5 GHz band occupancy that we have recently collected and made publicly available. This dataset was obtained by conducting extensive measurement campaigns on different days and in multiple locations, including a sold-out football stadium (F. C. Barcelona's Camp Nou). In this paper, we employ the football stadium measurements since they

range from completely idle to fully occupied channels. In the dataset, spectrum samples consist of 1 s of consecutive, 10 μs receive signal strength indicator (RSSI) measurements. We refer the reader to [21], [22] for further details on the dataset. Compared to [21], [22], in this work we have implemented a new Wi-Fi state machine, capable of (i) fully characterizing the temporal dynamics of the system under finite traffic loads, i.e., non-full buffer conditions, and (ii) supporting multiple Wi-Fi interfaces and packet buffers.

In what follows, we employ the FCB-WACA dataset to investigate how different combinations of primary and secondary channel occupancies affect the MLO-BSS performance. In particular, we assume that the MLO-BSS perceives the same spectrum activity as the one captured in the WACA dataset, and it contends for channel access accordingly. As for the OBSS, we adopt the same hinder interaction model as in [22], assuming that the OBSS sense the MLO-BSS channel access whenever this takes place and therefore defer their transmissions.

B. Trace-based Simulations Methodology

In order to study the effect of channel occupancy on latency, we partition the available traces in our dataset for both primary and secondary channels into different average channel occupancy regimes: $\{10\%, 20\%, \ldots, 90\%\}$. Then, we run each simulation as follows:

1) We select the occupancy regime of interest for the primary and secondary channels, e.g., 10% and 40%, respectively;
2) We combine uniformly at random one spectrum sample each for the primary and secondary channels;
3) For each spectrum sample pair and given a particular traffic load of interest, we compute the packet arrival times at the AP;
4) We execute the Wi-Fi state machine for SLO, MLO-STR, and MLO-NSTR access policies. The same packet arrival times are considered in all cases to allow a direct comparison.
5) We store the individual delay experienced by each packet over all spectrum samples.

For a fair comparison between SLO and MLO, we guarantee that all results are obtained in non-saturation conditions and thus we discard any simulations where less than 95% of all the transmitted packets are received.

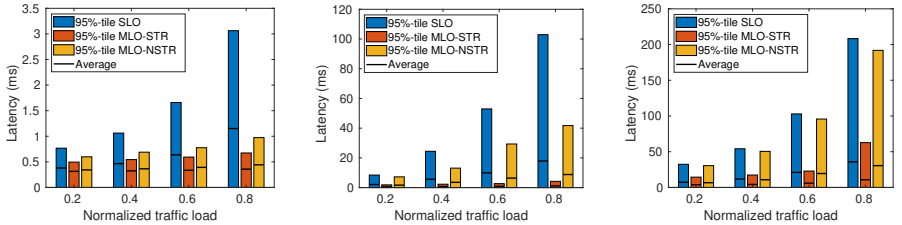
IV. Delay Performance

This section investigates the delay performance of both SLO and MLO modes for different combinations of channel occupancies and traffic loads.

To evaluate the gains of MLO in terms of delay, we consider the same traffic load for both SLO and MLO modes. In addition, the primary channel considered in the MLO mode is the same channel used in SLO. Therefore, we could expect that adding another channel in MLO mode, regardless of its occupancy, should yield lower delays.

A. Symmetrically Occupied Channels

Here we study the case of *symmetric* channel occupancies in which both MLO interfaces have channels with similar occupancy levels. In particular, we study the delay performance with pairs of channels in the ranges of 10%, 40%, and 70% occupancy. In those cases, the average full-buffer throughput under SLO is 37, 22, and 6.8 Mbps, respectively. For these three scenarios (symmetric low, medium, and high occupancy), we feed the Wi-Fi state machine with Poisson traffic and vary the intensity as a fraction of this SLO average full-buffer throughput, namely 0.2, 0.4, 0.6, 0.8. Figure 3 shows the average and 95th percentile delay for all channel access modes and the different channel occupancies. First, we observe that when both channels have 10% occupancy (Figure 3a), the three schemes have strikingly different scaling with increasing traffic load as MLO delay does not increase at the same rate as SLO delay. For example, at 20% traffic load, STR and NSTR offer a modest decrease in average delay compared to Single Link Operation of 17% and 9% respectively. In contrast, when the traffic load is 80%, STR and NSTR reduce the average delay by 69% and 62%. This scaling is even more pronounced analyzing the 95th percentile of delay, in which MLO



(a) 10% occupancy on both channels (b) 40% occupancy on both channels (c) 70% occupancy on both channels

Fig. 3: Latency for symmetrically occupied channels vs. variable normalized traffic load.

achieves up to a 78% delay reduction. Thus, for both average and 95th percentile delay, the benefits of MLO are increasingly pronounced under higher traffic load as in this case, there are often multiple packets in the buffer such that both interfaces can be used. Moreover, with a relatively low channel occupancy of 10%, both channels are often available.

Next, we consider the case that both channels have symmetrical medium (40%) occupancy (Figure 3b). Here, while SLO's *average* delay increases only modestly with traffic (i.e., from 2 to 18 ms), the 95th percentile delay delay increases much more rapidly, exceeding 100 ms. In contrast, STR can yield a staggering order of magnitude reduction in 95th percentile delay compared to SLO. The reason is that STR avails usage of two channels and can access either or both of them. STR, therefore, realizes delay benefits compared to SLO unless both channels are occupied.

Unfortunately, unlike STR, the benefits of NSTR over SLO are limited and mostly confined to how the average delay scales with traffic load. Indeed, NSTR can only gain access to the secondary channel if the primary channel is also idle, implying that the average delay is guaranteed to be lower than the average delay under SLO. However, the 95th percentile delays are triggered by long periods of occupancy of the primary channel, thus making any availability of the secondary channel

during this time irrelevant. As a result, the 95th percentile delay under NSTR rapidly grows as the normalized traffic load increases.

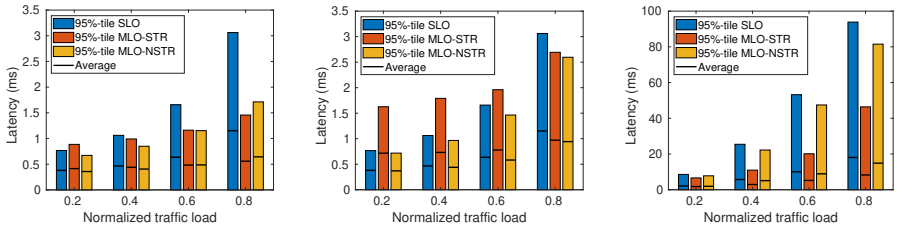
Lastly, when both channels have high (70%) occupancy (Figure 3c), STR again has the most favorable 95th percentile delay scaling with traffic load, providing substantial reductions as compared to both SLO and NSTR. Nonetheless, at such high channel occupancies, even STR has difficulty finding transmission opportunities on either channel, so both mean and 95th percentile delays are increasing. Additionally, NSTR provides negligible benefits in both average and 95th percentile delay compared to SLO.

Findings: When both channel occupancies are symmetrically medium to high load, NSTR fails to provide significant 95th percentile delay benefits compared to SLO. The key reason is that NSTR is only able to realize a benefit compared to SLO if both channels are simultaneously unoccupied, an increasingly unlikely occurrence in this scenario. Fortunately, STR yields significant 95th percentile latency benefits (compared to both SLO and NSTR) even in the challenging regime of increasing occupancies and traffic. This is because STR can utilize either available channel, and reduce the packets waiting time even if it cannot simultaneously utilize both available channels.

B. Asymmetrically Occupied Channels

Here, we employ the same normalized traffic loads from the previous section, but change the channel occupancy of our interfaces so that they lie in different ranges. Between the two channels, we always assume the primary to be the less occupied one. Note that the opposite case favors both MLO modes in this comparison: SLO would always incur a high delay, and both MLO modes would take advantage of a more idle secondary channel.

Figure 4a depicts the case of a low (10%) primary and medium (40%) secondary channel occupancy. As expected, NSTR offers deterministically lower delays than SLO, with the highest benefits occurring under higher traffic loads. However, STR surprisingly incurs a higher average



(a) Primary of 10% and secondary of 40%

(b) Primary of 10% and secondary of 70%

(c) Primary of 40% and secondary of 70%

Fig. 4: Latency for non-symmetrically occupied channels vs. variable normalized traffic load.

and 95th percentile delay than SLO for the lowest traffic load of 0.2. Indeed, STR starts contention by initializing the backoff counter as soon as a channel is detected to be idle. Unfortunately, such channel may be occupied before the backoff timer expires, thus pausing the backoff counter. If the backoff is paused too often (or for long intervals), the packet could incur even higher delays than it would have if the other channel—initially busy—had been selected.

In Figure 4b, this effect is exacerbated due to the even higher occupancy of the secondary channel, as selecting an idle secondary channel incurs the risk of the latter being occupied before the backoff counter expires. When this occurs, the 95th percentile delay can be twice as high as that with SLO, albeit still confined to below 10 ms. However, STR average and 95th percentile delays grow at a lower rate than those of SLO as the traffic load increases. Indeed, STR can still take advantage of a secondary channel (even when highly occupied) to reduce congestion and curb the latency when it is caused not only by the channel occupancy patterns but also by the amount of traffic.

Finally, Figure 4c considers the more symmetrical case of primary and secondary channel occupancy of 40% and 70%, respectively. Similar to the prior case of Figures 3b and 3c, STR scales well with the increasing traffic load, keeping the average delay below that of SLO and the de-

creasing the 95th percentile by up to a half. Compared to Figure 4b, the primary channel occupancy has grown, leading to a faster increase in the SLO delay vs. traffic load. However, STR is capable of leveraging both links and thus achieves lower delays.

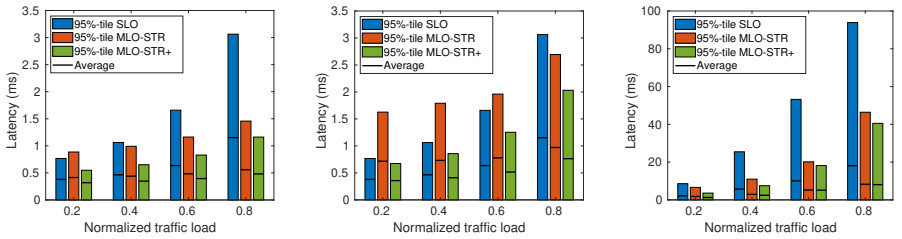
Findings: Channel occupancy is a crucial factor to account for when selecting a secondary channel in MLO mode. For STR, specifically, using a secondary channel that is much busier than the primary can lead to even higher delays than using SLO. This is owed to packets being suboptimally assigned to an interface before carrying out the backoff, with the latter likely to be interrupted on the busier channel. This effect is exacerbated when the difference between channel occupancies increases.

C. MLO-STR with parallel backoffs

We have shown that MLO-STR can lead to even higher delays than using SLO in the case of channels with different occupancies. To better understand the design space of MLO channel access, we now define a minor variation—denoted *Opportunistic MLO-STR* (MLO-STR+)—which nonetheless allows to overcome the limitations of MLO-STR and to evaluate the ultimate capabilities of MLO.

- **MLO-STR+:** When both interfaces are idle, one backoff instance is started on each. Packet allocation is deferred until either backoff counter expires, and the first waiting packet is allocated to the interface whose counter expires first. This approach differs from MLO-STR, where a packet is assigned to a channel as soon as the latter is idle, without waiting for its backoff counter to expire.

The main advantage of MLO-STR+ lies in the fact that, if one channel becomes occupied during the backoff, a transmission opportunity may be found on the other, avoiding unnecessarily delaying the waiting packet. In practice, implementing MLO-STR+ only requires a minor firmware update on the current Wi-Fi state machine: the ability to control when an interface can initiate, pause, and complete the backoff countdown without actually being allocated a packet.



(a) Primary of 10% and secondary of 40%

(b) Primary of 10% and secondary of 70%

(c) Primary of 40% and secondary of 70%

Fig. 5: Latency for non-symmetrically occupied channels vs. variable normalized traffic load. MLN-STR vs MLN-STR+.

Figure 5 shows the average and 95th percentile delay for the same cases studied in Figure 4. We still take SLO as the baseline and compare MLO-STR and MLO-STR+ modes. In Figure 5a, STR+ consistently outperforms STR and SLO in both average and 95th percentile delay, since packets are transmitted either at the same time as in SLO, or faster via the secondary interface. In Figure 5b, when the secondary channel has a 70% occupancy, we encounter the worst scenario for STR. In this case, STR selects the secondary channel when it undergoes a short idle periods. However, since the latter are typically followed by longer intervals of occupancy, the backoff counter often remains frozen, leading to 95th percentile delays more than twice as high as those with SLO. This shortcoming is avoided altogether by the proposed STR+, assigning a packet to either interface only after ensuring that the corresponding backoff counter has expired. Finally, Figure 5c depicts the case of 40% and 70% occupancy on the primary and secondary channels, respectively. As the former has increased, the SLO delay grows rapidly. STR already outperforms SLO in average and 95th percentile delay, and STR+ slightly reduces these values further.

Findings: MLO-STR+ improves over MLO-STR by delaying the allocation of the packet at the head of the queue until one of the backoff counters expires, allowing to leverage up-to-date information on the channel state, and thus to ultimately make better decisions.

V. Conclusions

In this paper, we provided an experimental study of latency for IEEE 802.11be MLO. Using the WACA dataset, which contains real-world channel occupancy measurements in the 5 GHz spectrum, we cast light upon the latency performance of two MLO channel access modes, namely (i) MLO-STR, where two radio interfaces are operated independently, and (ii) MLO-NSTR, where one interface acts as primary and the other as secondary.

We showed that when both channels are on average equally occupied, both MLO modes can reduce the 95th percentile latency by nearly one order of magnitude as they avail of a second radio interface. In contrast, in asymmetrically occupied channels, we surprisingly found the use of MLO-STR to be detrimental and cause even higher latency values than SLO. We define MLO-STR+ to show that this issue can be overcome by delaying the packet assignment until the expiration of the backoff, which also achieves lower latency overall.

References

[1] Cisco Annual Internet Report (2018–2023) White Paper, Mar. 2020.

[2] IEEE 802.11, "P802.11ax - IEEE Draft 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 Enhancements for High Efficiency WLAN," 2020.

[3] "IEEE P802.11be/D1.0 Draft 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 8: Enhancements for extremely high throughput (EHT)," May 2021.

[4] A. Garcia-Rodriguez, D. Lopez-Perez, L. Galati-Giordano, and G. Geraci, "IEEE 802.11 be: Wi-Fi 7 Strikes Back," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 102–108, 2021.

[5] E. Khorov, I. Levitsky, and I. F. Akyildiz, "Current status and directions of IEEE 802.11 be, the future Wi-Fi 7," *IEEE Access*, vol. 8, pp. 88 664–88 688, 2020.

[6] R. P. F. Hoefel, "IEEE 802.11 be: Throughput and Reliability Enhancements for Next Generation WI-FI Networks," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2020, pp. 1–7.

[7] M. Yang and B. Li, "Survey and perspective on extremely high throughput (EHT) WLAN—IEEE 802.11 be," *Mobile Networks and Applications*, vol. 25, no. 5, pp. 1765–1780, 2020.

[8] D. Lopez-Perez, A. Garcia-Rodriguez, L. Galati-Giordano, M. Kasslin, and K. Doppler, "IEEE 802.11be Extremely High Throughput: The Next Generation of Wi-Fi Technology Beyond 802.11ax," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 113–119, 2019.

[9] B. Bellalta, "On the Low-Latency Region of Best-Effort Links for Delay-Sensitive Streaming Traffic," *IEEE Communications Letters*, vol. 25, no. 3, pp. 970–974, 2020.

[10] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta, "Time-sensitive networking in IEEE 802.11 be: On the way to low-latency WiFi 7," *Sensors*, vol. 21, no. 15, p. 4954, 2021.

[11] M. Carrascosa and B. Bellalta, "Cloud-gaming: Analysis of Google stadia traffic," *arXiv preprint arXiv:2009.09786*, 2020.

[12] C. Deng, X. Fang, X. Han, X. Wang, L. Yan, R. He, Y. Long, and Y. Guo, "IEEE 802.11be Wi-Fi 7: New Challenges and Opportunities," *IEEE Communications Surveys Tutorials*, vol. 22, no. 4, pp. 2136–2166, 2020.

[13] I. Levitsky, Y. Okatev, and E. Khorov, "Study on Simultaneous Transmission and Reception on Multiple Links in IEEE 802.11 be networks," in *2020 International Conference Engineering and Telecommunication (En&T)*. IEEE, 2020, pp. 1–4.

[14] M. Yang, B. Li, Z. Yan, and Y. Yan, "AP Coordination and Full-duplex enabled Multi-band Operation for the Next Generation WLAN: IEEE 802.11 be (EHT)," in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2019, pp. 1–7.

[15] Á. López-Raventós and B. Bellalta, "IEEE 802.11be multi-link operation: When the best could be to use only a single interface," *CoRR*, vol. abs/2105.10199, 2021. [Online]. Available: https://arxiv.org/abs/2105.10199

[16] Á. López-Raventós and B. Bellalta, "Multi-link Operation in IEEE 802.11 be WLANs," *arXiv preprint arXiv:2201.07499*, 2022.

[17] T. Song and T. Kim, "Performance Analysis of Synchronous Multi-Radio Multi-Link MAC Protocols in IEEE 802.11 be Extremely High Throughput WLANs," *Applied Sciences*, vol. 11, no. 1, p. 317, 2021.

[18] G. Naik, D. Ogbe, and J.-M. J. Park, "Can Wi-Fi 7 Support Real-Time Applications? On the Impact of Multi Link Aggregation on Latency," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6.

[19] H. Park and C. You, "Latency Impact for Massive Real-Time Applications on Multi Link Operation," in *2021 IEEE Region 10 Symposium (TENSYMP)*, 2021, pp. 1–5.

[20] G. Lacalle, I. Val, O. Seijo, M. Mendicute, D. Cavalcanti, and J. Perez-Ramirez, "Analysis of Latency and Reliability Improvement with Multi-Link Operation over 802.11," in *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, 2021, pp. 1–7.

[21] S. Barrachina-Muñoz, B. Bellalta, and E. Knightly, "Wi-Fi All-Channel Analyzer," in *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*, 2020, pp. 72–79.

[22] S. Barrachina-Muñoz, B. Bellalta, E. W. Knightly *et al.*, "Wi-Fi Channel Bonding: An All-Channel System and Experimental Study From Urban Hotspots to a Sold-Out Stadium," *IEEE/ACM Transactions on Networking*, 2021.

Understanding Multi-link Operation in Wi-Fi 7: Performance, Anomalies, and Solutions

Marc Carrascosa-Zamacois[⋆], Giovanni Geraci[⋆], Lorenzo Galati-Giordano[♯], Anders Jonsson[⋆], and Boris Bellalta[⋆]

[⋆]*Universitat Pompeu Fabra (UPF), Barcelona, Spain*
[♯]*Nokia Bell Labs, Stuttgart, Germany*

Abstract

Will Wi-Fi 7, conceived to support extremely high throughput, also deliver consistently low delay? The best hope seems to lie in allowing next-generation devices to access multiple channels via multi-link operation (MLO). In this paper, we aim to advance the understanding of MLO, placing the spotlight on its packet delay performance. We show that MLO devices can take advantage of multiple contention-free links to significantly reduce their transmission time, but also that they can occasionally starve one another and surprisingly incur a higher delay than that of a well planned legacy single link operation. We examine and explain this anomaly, also putting forth practical workarounds.

I. Introduction

Heading out of a pandemic that made connectivity truly dependable, our appetite for data is stronger than ever. Myriad engineers behind the development of Wi-Fi, the technology carrying two thirds of all wireless data, relentlessly feed this hunger by crafting ever more clever amendments, defining new Wi-Fi generations one after another. At the

This work was supported in part by grants PID2021-123995NB-I00, PGC2018-099959-B-I00, PRE2019-088690, RTI2018-101040-A-I00, PID2021-123999OB-I00, and by the "Ramón y Cajal" program.

time of writing, Wi-Fi 6 and 6E are a commercial reality, the making of Wi-Fi 7 is nearing completion, and the definition of Wi-Fi 8 starts catalyzing the interest of tech giants and avid researchers alike [1]–[4]. Yet before debating or fantasizing about what Wi-Fi 8 should be, what will Wi-Fi 7 deliver?

The IEEE 802.11be amendment, expected to be at the heart of Wi-Fi 7, will remain loyal to its legacy—and to its very name: EHT, short for 'Extremely High Throughput'—by augmenting data rates through various upgrades ranging from wider bandwidths (up to 320 MHz) to higher modulation orders (up to 4096-QAM) [5]–[7]. But besides features boosting the nominal throughput, many experts point to multi-link operation (MLO) as the true paradigm shift Wi-Fi 7 will bring to the table. MLO will allow Wi-Fi devices to concurrently operate on multiple channels through a single connection, aiming to support applications demanding not only higher capacity but also lower delay [8]–[10].

Unlike merely multiplying the peak throughput gains provided by scaling up bandwidth and spectral efficiency, quantifying the advantages brought about by MLO in realistic scenarios is no straightforward endeavor. And while several works have recently made valuable attempts at studying how MLO performs in terms of throughput and delay [11]–[16], a deep and widespread understanding of the latter remains little more than wishful thinking. Indeed, the exact benefits on a device employing MLO for delay-sensitive applications and the effects on coexisting basic service sets (BSSs) hinge on the specific MLO implementation, with several being defined in 802.11be to trade off complexity and flexibility. Furthermore, as we will show in later sections, these benefits—or the lack thereof—highly depend on the traffic load, the surrounding environment, and the channel allocation strategy adopted.

In this paper, we shed light on the delay performance of STR EMLMR (standing for 'Simultaneous Transmit and Receive Enhanced Multi-link Multi-radio'), arguably the most flexible MLO mode, under varying traffic demand, congestion, and channel allocation strategies. We explain and quantify its main virtues with respect to legacy single-link (SL) as well as its caveats, also putting forth possible solutions to the latter. Our main takeaways can be summarized as follows:

- In scenarios devoid of contention, STR EMLMR exploits additional available links to perform multiple transmissions in parallel, proportionally reducing the channel access delay.
- In the presence of high load and contention, STR EMLMR devices frequently access multiple links thereby blocking contending neighbors, occasionally causing larger delays than those experienced with a static SL channel assignment.
- For consistent worst-case delay reduction, STR EMLMR may require more channels than contending BSSs and/or performing a clever channel assignment that entirely circumvents delay anomalies caused by sporadic BSS starvation.

Compared to existing work, the novelty and contribution of the present paper is at least threefold:

- We illustrate the intricate interactions MLO triggers between contending BSSs. We demonstrate how such interplay may turn out being benign or unfavorable, depending on the traffic load and channel allocation strategies.
- We identify, quantify, and explain, through novel results, the delay anomalies that may surprisingly arise when employing STR EMLMR in the presence of high load and contention. We also propose multiple solutions to circumvent such anomalies and we evaluate and compare their effectiveness.
- We provide a concise yet complete picture of the virtues and caveats of MLO by putting our findings into an even broader context.

II. A Primer on Multi-link Operation

In addition to legacy *SL* (Single-link) channel access as in IEEE 802.11ax, Wi-Fi 7 will allow MLO through single association, with channel contention and access performed independently for each link. The 802.11be amendment defines different MLO implementation flavors, with the main ones summarized as follows [8].

A. Multi-link Flavors

Enhanced Multi-link Single-radio (EMLSR): EMLSR enables a single-radio multi-link device (MLD) to listen to two or more links simultane-

ously, e.g., by splitting its multiple antennas, performing clear channel assessment and receiving a limited type of control frames. EMLSR supports opportunistic spectrum access at a reduced cost, as it requires a single fully functional 802.11be radio plus several other low-capability radios able only to decode 802.11 control frame preambles. Upon reception of an initial control frame on one link, EMLSR MLD can switch to the latter and operate using all antennas.

Enhanced Multi-link Multi-radio (EMLMR): For a MLD implementing EMLMR, all radios are 802.11be-compliant and allow operating on multiple links concurrently. EMLMR is further classified into two modes:

- *Non-simultaneous Transmit and Receive (NSTR) EMLMR*, where no simultaneous transmission and reception is allowed over a pair of links in order to prevent self interference at the MLD. The latter entails ensuring near alignment in the end time of physical layer protocol data unit that are simultaneously transmitted, so as to avoid that subsequent incoming responses on one link, e.g., ACKs, overlap with the remaining transmission on another link.
- *Simultaneous Transmit and Receive (STR) EMLMR*, where the above rule does not apply. In order to avoid uplink-to-downlink intra-device interference, operating STR EMLMR requires sufficient frequency separation between the channels used by different links and/or sophisticated self-interference cancellation capabilities. For instance, STR EMLMR with four links, each on an 80 MHz channel, could be implemented by using two channels each in the 5 GHz and 6 GHz bands, with a minimum channel separation of 160 MHz, and equipping MLDs with suitable radio-frequency filters.

A remark is in order about the 'E' in EMLSR and EMLMR, standing for 'enhanced'. Indeed, non-enhanced versions of both have also been defined [8], summarized as follows:

- *MLSR*, where unlike EMLSR, clear channel assessment and control frame reception (and of course, data transmission/reception) can only be performed on one channel at a time, thereby limiting opportunistic link selection.
- *MLMR*, which compared to EMLMR only lacks extra capabilities to

dynamically reconfigure spatial multiplexing over multiple links. This difference is immaterial for the case studies of the present paper.

In the remainder of this article, we place the spotlight on STR EMLMR since it is the MLO operation mode that grants the highest degree of flexibility and requires the least amount of signaling, thus being the most likely to be adopted in first-wave Wi-Fi 7 commercial products. Unlike previous work devoted to the achievable throughput of STR EMLMR, we focus on its delay performance as we deem it crucial to support ever more proliferating real-time applications.

B. A Close-Up of STR EMLMR

As shown in Fig. 1, exemplifying STR EMLMR in action over two links, it turns out that this mode of operation can affect the packet delay in multiple ways, depending on the particular scenario at hand. In the following, we provide two examples that illustrate how STR EMLMR can respectively reduce and increase the delay with respect to legacy SL operations. For the latter (not shown), we assume an orthogonal channel assignment as a benchmark, with AP 1 and AP 2 operating on link 1 and link 2 only, respectively.

Delay reduction through STR EMLMR: Let us begin by focusing on the left hand side of Fig. 1, where AP 2 is inactive and AP 1 can take advantage of two available links by routing traffic to either as needed. In the example, packets #1 and #2 are aggregated and promptly transmitted over link 1. As for packet #3, which arrives during an ongoing transmission, a new backoff is started on link 2, followed by a transmission. Packet #3 thus enjoys a significant delay reduction compared to a legacy SL scenario, as in the latter it would have needed to wait for the ongoing transmissions on link 1 to be completed.

Delay anomaly in STR EMLMR: The right hand side of Fig. 1 illustrates a scenario where AP 1 and AP 2, both implementing STR EMLMR, contend for channel access. In this example, AP 1 aggregates packets #4 and #5 upon backoff expiration and transmits them over link 1. Meanwhile, more traffic arrives, namely packets #6 and #7 at AP 1 and packets #1 and #2 and AP 2. Since link 1 is occupied by AP 1, both AP 1 and AP 2 undergo contention for link 2, with the backoff

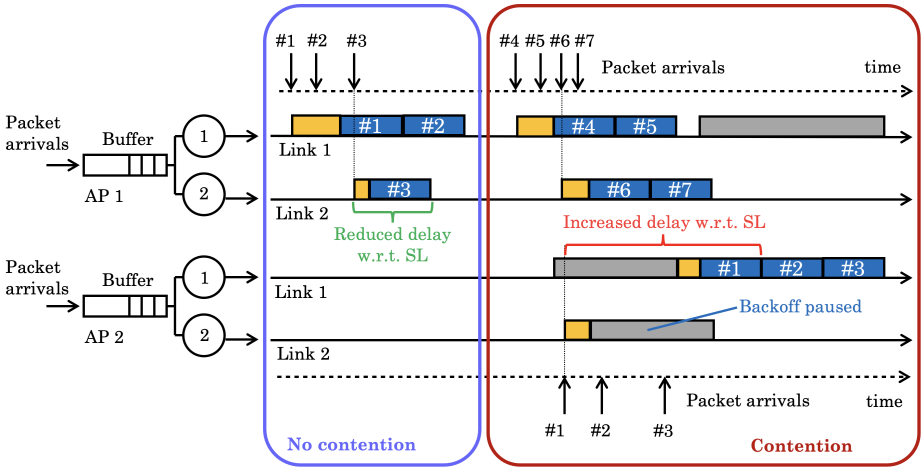


Fig. 1: Illustration of STR EMLMR operations and packet interactions over two links without (left) and with (right) contention. Grey, orange, and blue slots denote occupied channels, ongoing backoffs, and successful transmissions, respectively. Consecutive blue slots indicate aggregated packets. For illustration purposes, all transmissions are downlink and the corresponding ACKs are omitted. In the example, for AP 1, packet #3 experiences a lower delay than it would under SL operations. For AP 2 instead, packet #1 undergoes a higher delay than it would with SL.

for AP 1 expiring first. AP 1 thus aggregates and transmits packets #6 and #7 on link 2, thereby occupying both links concurrently. It is only after the transmission of packets #4 and #5 by AP 1 is completed that AP 2 can eventually aggregate and transmit all its queued packets on link 1. In the example, these packets experience a much higher delay than they would have under legacy SL operations. Indeed, with SL and a static channel allocation (e.g., AP 1 on link 1 and AP 2 on link 2), AP 1 would have not been able to occupy both links simultaneously, and therefore would have not temporarily forced AP 2 into starvation. We identify this as an *anomaly* of MLO, and will devote Section IV to its understanding.

As it can be seen through the above two examples, STR EMLMR is capable of taking advantage of multiple links to reduce the channel access time with respect to SL, but also to occasionally starve neighboring BSSs thereby increasing their delay. In the sequel, we will confirm and quantify these two phenomena through targeted simulation campaigns.

III. STR EMLMR in Contention-free Scenarios

We begin by considering a single, isolated BSS with one MLD station (STA) associated to an MLD access point (AP), and evaluate the delay performance of STR EMLMR in such a contention-free scenario. Without loss of generality, we focus on downlink traffic and assume Poisson arrivals with constant packet size of 12000 bits, 80 MHz channels, two spatial streams, and a modulation and coding scheme of 256-QAM 3/4 [8]. Packet aggregation is employed with the number of aggregated packets decided at the start of a transmission, up to a maximum of 1024. A buffer size of 4096 packets is employed, ensuring sufficient room for the maximum allowed number of aggregated packets. For this scenario, we study the effect of the traffic load on the packet delay under three schemes, namely: (i) *SL*, as in Wi-Fi 6, (ii) *STR EMLMR:2*, where the isolated AP can use two links at any time, and (iii) *STR EMLMR:4*, with four links available.

Packet delay: Fig. 2 shows the delay statistics with shaded curves ranging from 50%-tile to 99%-tile (i.e., median to 1%-worst) using SL and STR EMLMR with two or four links. Intuitively, as more links are available and can be accessed dynamically, a certain delay requirement can be met for proportionally higher values of the traffic load, i.e., while supporting a proportionally higher throughput. For instance, given a median delay of 1 ms, SL, STR EMLMR:2, and STR EMLMR:4 can roughly support up to half, one, and two Gbps, respectively. Similarly, given a certain traffic load, availing of extra links decreases the delay, albeit with diminishing returns. For instance at 0.5 Gbps, the three schemes incur 99%-tile delays of about 1.6, 0.7, and 0.5 ms. Nonetheless, depending on the traffic load, accessing multiple links may be the only way to prevent the delay from growing unbounded. E.g., a load of 1 Gbps exceeds the capacity of a single channel, thus SL incurs

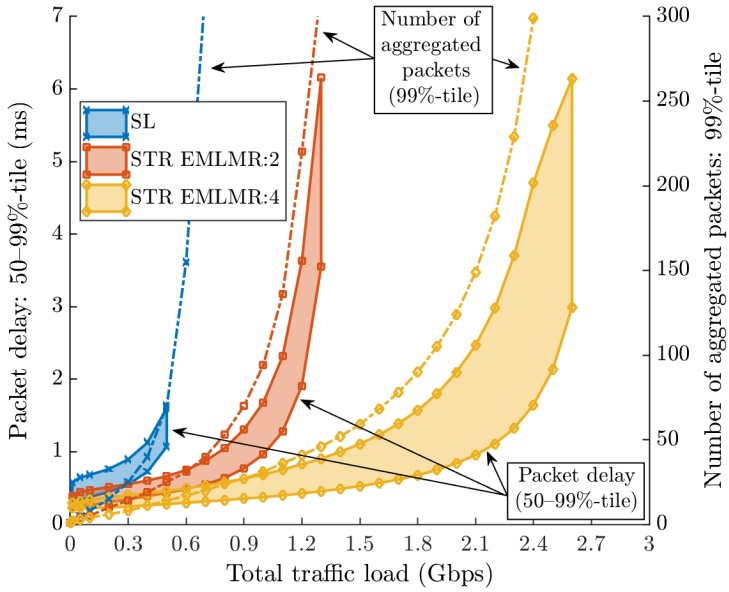
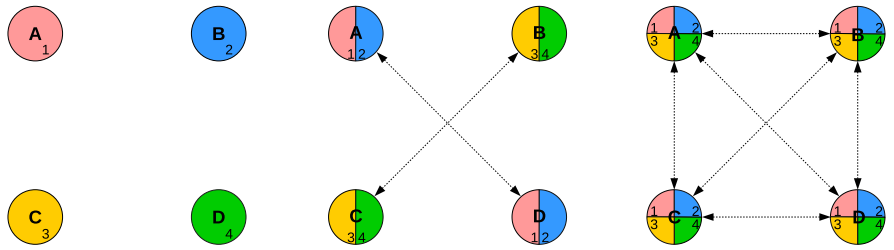


Fig. 2: Contention-free scenario: packet delay (spanning 50–99%-tile) and number of aggregated packets (99%-tile) vs. traffic load for SL, STR EMLMR:2, and STR EMLMR:4.

unbounded delay, whereas STR EMLMR:2 and STR EMLMR:4 keep the delay below 1.7 and 0.7 ms, respectively, 99% of the time.

Packet aggregation: Fig. 2 also displays the 99%-tile for the corresponding number of packets aggregated under each of the three schemes (dashed lines). Even at moderate loads, owing to its inability of using multiple links, SL experiences a higher buffer congestion and is forced to aggregate a much larger number of packets per transmission than STR EMLMR. The latter can instead parallelize access on multiple links, reducing the buffer congestion and thus the number of aggregated packets for each transmission.

Takeaway: In scenarios devoid of contention, STR EMLMR can exploits extra links—even across different frequency bands, something SL is not capable of—to operate on a wider bandwidth, and can therefore



(a) SL, a single 80 MHz channel exclusively assigned to each BSS and no contention.

(b) STR EMLMR:2, two 80 MHz channels per BSS, both shared with one more BSS.

(c) STR EMLMR:4, four 80 MHz channels per BSS, all shared with three more BSSs.

Fig. 3: Three modes of operation considered for a crowded scenario: (a) SL, (b) STR EMLMR:2, and (c) STR EMLMR:4. Colors and numbers refer to different channels, letters denote BSSs, and dashed arrows indicate contention between BSSs.

meet a certain delay requirement while supporting higher traffic loads (i.e., throughput) than SL.

While the above results are somewhat expected, they are in stark contrast to the delay anomaly experienced by MLDs in crowded scenarios, quantified in the next section.

IV. STR EMLMR in Crowded Scenarios

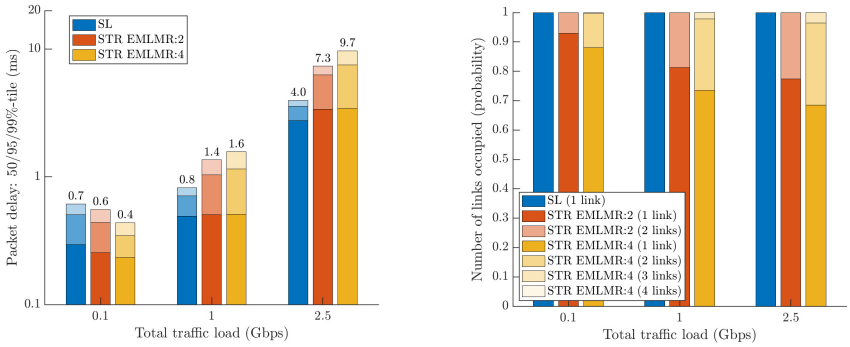
We now investigate when the delay reduction provided by STR EMLMR is maintained in the presence of contention, and when instead the delay is increased due to the starvation phenomenon, i.e., the *anomaly*, outlined in Section II-B. To this end, we turn our attention to a more crowded enterprise scenario with 4 BSSs as depicted in Fig. 3. Each BSS comprises one AP and one associated STA, all BSSs are in the coverage range of each other, and the whole system has a limited amount of resources, namely four orthogonal 80 MHz channels. All other parameters are kept the same as in the previous section. For this challenging scenario, we consider three possible modes of operation, each making a different use of the four available channels:

- *SL*, with a single channel exclusively assigned to each BSS, as illustrated in Fig. 3a, and no contention. Again, we take this mode as the baseline to assess STR EMLMR.
- *STR EMLMR:2*, as shown in Fig. 3b, where each BSS employs two channels and shares both with one more contending BSS.
- *STR EMLMR:4*, as shown in Fig. 3c, where all four BSSs employ and contend for all four channels.

Note that the above three arrangements assume statically assigning channels to BSSs according to a specific reuse scheme, and thus embody a hypothetical enterprise use case. In this section, we assume the same values of total traffic load as in Section III, but this time evenly spread among all BSSs, i.e., one quarter each. The scenarios in Section III (Fig. 2) vs. Section IV (Fig. 4) can thus be regarded as an asymmetric vs. symmetric distribution of the same total load between contending BSSs.

Packet delay: Fig. 4a shows the mean, 95%-tile, and 99%-tile delay using SL (Wi-Fi 6), STR EMLMR:2, and STR EMLMR:4 vs. the total traffic load. While the median is not significantly affected by the operation mode, the 95%- and 99%-tile delay is. For a relatively low load of 0.1 Gbps, the 95%-tile and 99%-tile delay is decreased by adding multiple links since there is negligible contention and STR EMLMR can quickly find and exploit extra transmission opportunities, as previously shown in Section III. However, once the load reaches higher values such as 1 Gbps and above, STR EMLMR worsens the 95%- and 99%-tile delay compared to SL, and four links incur a higher delay than two. These results stem from the anomaly illustrated on the right hand side of Fig. 1, and can be further explained by the interplay between multi-link contention and packet aggregation, detailed as follows.

Multi-link contention: In Fig. 4b we dig deeper into the delay anomaly by observing how STR EMLMR devices occupy the available links depending on their traffic load. The bars show, through different color opacity, the probability that an active BSS (i.e., with packets to transmit) will use a certain number of links concurrently. For a high traffic load of 2.5 Gbps, SL is limited to transmit on one link only, whereas STR EMLMR:2 employs a second interface 26% of the time, and STR



(a) Delay vs. traffic load for the three difference schemes. For each color, high/medium/low opacity respectively denote the 50/95/99%-tile delay.

(b) Probability for an active BSS to occupy a certain number of links concurrently vs. traffic load for the three different schemes.

Fig. 4: Crowded enterprise scenario: (a) delay and (b) number of links concurrently used by each active BSS vs. traffic load.

EMLMR:4 uses two or more interfaces 34% of the time. A remarkable consequence (not shown for brevity) is that, with STR EMLMR:4, each contending BSS finds all four links occupied simultaneously 24% of the time. These events cause a deferral of the backoff countdown and prevent access to any wireless channel. In other words, while SL mode allows—or better said, forces—each BSS to operate on its own dedicated link 100% of the time (Fig. 3a), whenever STR EMLMR BSSs use multiple links opportunistically they inevitably prevent at least another BSS from accessing at least one of its allocated channels (Figs. 3b and 3c).

Packet aggregation: Due to a higher contention, which results in longer backoff times, whenever a STR EMLMR device does succeed in accessing the channel, it must occasionally aggregate a larger number of queued packets as exemplified on the right hand side of Fig. 1. While not shown, we observed that for a load of 2.5 Gbps, switching from SL to STR EMLMR:4 decreases the median number of aggregated packets from 138 to 91, but it also increases its 99%-tile value from 207 to 317. The latter corresponds to occasional intervals of long channel occupancy

and undesirable delay anomalies.

Takeaway: In the presence of high load and contention, STR EMLMR devices frequently access multiple links, thereby occasionally blocking contending neighbors for long periods of time and causing larger delays than those experienced by legacy SL under a static orthogonal channel allocation.

V. Overcoming the Delay Anomaly

How to side-step the delay anomaly occasionally experienced by STR EMLMR in crowded environments? We now explore multiple practical options based on clever and/or extra channel assignment and compare their performance for the same enterprise scenario introduced in Section IV:

- *EMLSR:2*, detailed in Section II-A, with each MLD availing of two channels as in Fig. 3b but only equipped with one radio and thus only able to use one link at a time. This setup still requires a total of four channels.
- *STR EMLMR:1+1*, with each MLD using two links: one on a channel exclusively reserved (thus undergoing no contention) plus one on a channel shared with all other BSSs. This hybrid arrangement requires a total of five channels as opposed to the four required in Fig. 3b.
- *STR EMLMR:5*, with an overprovisioning of five links per MLD, each operating on a different channel, with all channels accessible by all four BSSs. Like the previous one, this setup requires a total of five channels, but it additionally requires five radio interfaces per MLD.

Fig. 5 displays the delay experienced by the three above approaches when compared to SL (Fig. 3a) and STR EMLMR:2 (Fig. 3b). We note how forcing each MLD to transmit on one link at a time with EMLSR:2 (purple) keeps the delay below or equal to that of SL (blue) across all values of load considered, while not increasing the total number of channels required. At a load of 0.1 Gbps, all approaches experience low delays, with STR EMLMR:5 (green) achieving the lowest. Indeed, the low contention arising in this regime makes it likely for a MLD to encounter multiple links available, and juggling up to five running backoffs further reduces the delay. Interestingly, as the load grows to

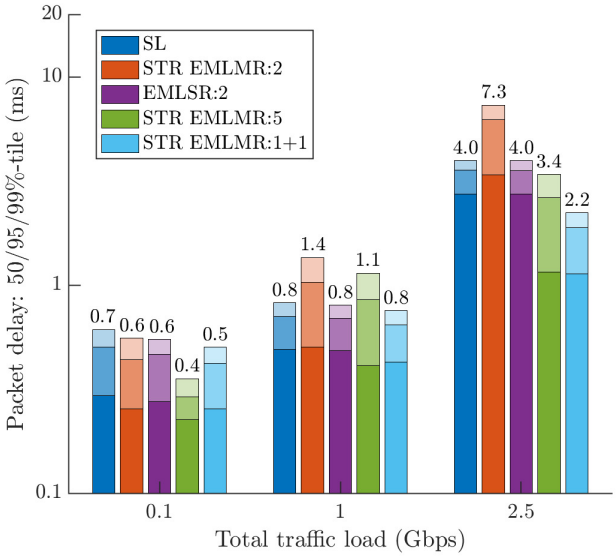


Fig. 5: Crowded enterprise scenario: delay vs. traffic load for difference schemes. For each color, high/medium/low opacity respectively denotes the 50/95/99%-tile delay.

2.5 Gbps, equipping each MLD with just two radios and operating STR EMLMR:1+1 (light blue) outperforms STR EMLMR:5, despite the latter employing as many as five radios per MLD. Indeed, delay reduction is owed not only to more channels and an increased system throughput, but also to circumventing the delay anomaly by guaranteeing one contention-free channel per BSS.

Overall, to consistently outperform SL, STR EMLMR may thus require a total number of channels larger than the number of contending BSSs, and therefore equipping MLDs with additional radios, self-interference cancellation capabilities, and ensuring a sufficient inter-channel spacing. For instance, operating both in the 5 GHz and 6 GHz bands could allow accessing five 80 MHz channels with a spacing of 160 MHz.

Takeaway: For consistent worst-case delay reduction, one may resort to STR EMLMR with more channels than contending BSSs and/or to performing a clever channel assignment that entirely circumvents delay anomalies caused by sporadic traffic starvation.

VI. Conclusions

Our study confirmed that in scenarios devoid of contention, STR EMLMR exploits extra links to transmit opportunistically, supporting significantly higher traffic loads (and therefore throughput) than SL while meeting strict delay requirements. Conversely, we discovered that in the presence of high load and contention, STR EMLMR devices frequently access multiple links, thereby blocking contending BSSs and occasionally causing larger delays than those experienced with a legacy SL operation with orthogonal channel assignment.

STR vs. NSTR EMLMR: Though we focused on STR EMLMR for brevity, NSTR EMLMR too may incur delay anomalies. Indeed, its required alignment of simultaneous transmissions comes at the expense of spectrum reuse efficiency, ultimately creating higher contention and further increasing the chances that a certain BSS is prevented from accessing any channel.

Symmetric vs. asymmetric load: While delay anomalies may arise under high traffic load across all contending APs, their likelihood and relevance are reduced when the traffic is unevenly distributed across contenders. Let us take the scenario studied in Section III as an extreme example, with all traffic handled by a single active BSS. In such cases, STR EMLMR can efficiently map asymmetric traffic loads to all available links, drastically reducing the delay with respect to SL operations.

Single vs. multiple radios: We observed that delay anomalies can be circumvented by employing EMLSR, which allows MLDs to opportunistically select a link among several but forces them to transmit on one at a time. Though EMLSR makes for a lower complexity than EMLMR to reduce the delay at low traffic loads, using one link at a time prevents MLDs from achieving higher throughputs than SL. While not shown in Fig. 2, a traffic load beyond 0.5 Gbps would eventually exceed the channel capacity of EMLSR, just as it does with SL. In this

regime, availing of multiple radios would be the only approach to scale up the throughput so as to guarantee bounded delays.

Static vs. dynamic channel allocation: We compared (i) a static channel assignment approach (SL, Fig. 3a), (ii) an entirely dynamic approach (STR EMLMR, Figs. 3b and 3c), and (iii) a hybrid approach that cleverly reserves a certain channel for each BSS while leaving one more for contention (STR EMLMR:1+1). We found the latter to be most effective at reducing worst-case delays, even more so than equipping MLDs with more radio interfaces (STR EMLMR:5). Indeed, while STR EMLMR:1+1 guarantees at least one contention-free link for each BSS, STR EMLMR:5 merely spreads contention out over all available links, reducing the likelihood of a delay anomaly but not necessarily overcoming it.

REFERENCES

[1] David López-Pérez, Adrian Garcia-Rodriguez, Lorenzo Galati-Giordano, Mika Kasslin, and Klaus Doppler. IEEE 802.11be Extremely High Throughput: The next generation of Wi-Fi technology beyond 802.11ax. *IEEE Communications Magazine*, 57(9):113–119, 2019.

[2] Adrian Garcia-Rodriguez, David López-Pérez, Lorenzo Galati-Giordano, and Giovanni Geraci. IEEE 802.11be: Wi-Fi 7 strikes back. *IEEE Communications Magazine*, 59(4):102–108, 2021.

[3] Ehud Reshef and Carlos Cordeiro. Future directions for Wi-Fi 8 and beyond. *IEEE Communications Magazine*, pages 1–7, 2022.

[4] L. Galati-Giordano, G. Geraci, M. Carrascosa, and B. Bellalta. What will Wi-Fi 8 be? A primer on IEEE 802.11bn Ultra High Reliability. *arXiv:2303.10442*, 2023.

[5] Evgeny Khorov, Ilya Levitsky, and Ian F Akyildiz. Current status and directions of IEEE 802.11be, the future Wi-Fi 7. *IEEE Access*, 8:88664–88688, 2020.

[6] Cailian Deng, Xuming Fang, Xiao Han, Xianbin Wang, Li Yan, Rong He, Yan Long, and Yuchen Guo. IEEE 802.11be Wi-Fi 7: New challenges and opportunities. *IEEE Communications Surveys & Tutorials*, 22(4):2136–2166, 2020.

[7] Mao Yang and Bo Li. Survey and perspective on extremely high throughput (EHT) WLAN—IEEE 802.11be. *Mobile Networks and Applications*, 25(5):1765–1780, 2020.

[8] Cheng Chen, Xiaogang Chen, Dibakar Das, Dmitry Akhmetov, and Carlos Cordeiro. Overview and performance evaluation of Wi-Fi 7. *IEEE Communications Standards Magazine*, 6(2):12–18, 2022.

[9] Álvaro López-Raventós and Boris Bellalta. Multi-link operation in IEEE 802.11be WLANs. *IEEE Wireless Communications*, pages 1–12, 2022.

[10] Boris Bellalta, Marc Carrascosa, Lorenzo Galati-Giordano, and Giovanni Geraci. Delay analysis of IEEE 802.11be multi-link operation under finite load. *IEEE Wireless Communications Letters*, 2023.

[11] Gaurang Naik, Dennis Ogbe, and Jung-Min Jerry Park. Can Wi-Fi 7 support real-time applications? On the impact of multi link aggregation on latency. In *Proc. IEEE ICC*, pages 1–6, 2021.

[12] Wisnu Murti and Ji-Hoon Yun. Multilink operation in IEEE 802.11be wireless LANs: Backoff overflow problem and solutions. *Sensors*, 22(9):3501, 2022.

[13] Guillermo Lacalle, Iñaki Val, Oscar Seijo, Mikel Mendicute, Dave Cavalcanti, and Javier Perez-Ramirez. Analysis of latency and reliability improvement with multi-link operation over 802.11. In *Proc. IEEE INDIN*, pages 1–7, 2021.

[14] Sharan Naribole, Srinivas Kandala, Wook Bong Lee, and Ashok Ranganath. Simultaneous multi-channel downlink operation in next generation WLANs. In *Proc. IEEE Globecom*, pages 1–7, 2020.

[15] Marie-Theres Suer, Christoph Thein, Hugues Tchouankem, and Lars Wolf. Adaptive multi-connectivity scheduling for reliable low-latency communication in 802.11be. In *Proc. IEEE WCNC*, pages 102–107, 2022.

[16] Taewon Song and Taeyoon Kim. Performance analysis of synchronous multi-radio multi-link MAC protocols in IEEE 802.11be extremely high throughput WLANs. *Applied Sciences*, 11(1):317, 2020.

Performance Evaluation of MLO for XR Streaming: Can Wi-Fi 7 Meet the Expectations?

Marc Carrascosa-Zamacois[♭⋆], Lorenzo Galati-Giordano[♭], Francesc Wilhelmi[♭], Gianluca Fontanesi[♭], Anders Jonsson[⋆], Giovanni Geraci[♯⋆], and Boris Bellalta[⋆]

[♭]*Nokia Bell Labs, Stuttgart, Germany*
[⋆]*Universitat Pompeu Fabra, Barcelona, Spain*
[♯]*Telefónica Research, Barcelona, Spain*

Abstract

Extended Reality (XR) has stringent throughput and delay requirements that are hard to meet with current wireless technologies. Missing these requirements can lead to worsened picture quality, perceived lag between user input and corresponding output, and even dizziness for the end user. In this paper, we study the capability of upcoming Wi-Fi 7, and its novel support for Multi-Link Operation (MLO), to cope with these tight requirements. Our study is based on simulation results extracted from an MLO-compliant simulator that realistically reproduces VR traffic. Results show that MLO can sustain VR applications. By jointly using multiple links with independent channel access procedures, MLO can reduce the overall delay, which is especially useful in the uplink, as it has more stringent requirements than the downlink, and is instrumental in delivering the expected performance. We show that using MLO can allow more users per network than an equivalent number of links using SLO. We also show that while maintaining the same overall bandwidth, a higher number of

M. Carrascosa and B. Bellalta were supported in part by grants MAX-R HEu-CL4-MAX-R-101070072, Wi-XR PID2021-123995NB-I00, and by MCIN/AEI under the Maria de Maeztu Units of Excellence Programme (CEX2021-001195-M). G. Geraci was supported in part by the Spanish Research Agency through grants PID2021- 123999OB-I00, CEX2021-001195-M, and CNS2023-145384, by the UPF- Fractus Chair, and by the Spanish Ministry of Economic Affairs and Digital Transformation and the European Union NextGenerationEU.

MLO links with narrow channels leads to lower delays than a lower number of links with wider channels.

I. Introduction

Extended Reality (XR) applications, which include Virtual Reality (VR) and Augmented Reality (AR), are growing in popularity as they unlock novel use cases across many domains, such as healthcare, industry, education and gaming. Most use cases are planned for indoor use, and thus Wi-Fi is expected to become the main technology to support them [1], [2], with most headsets including high-grade Wi-Fi capabilities[1], and services like *Steam Link*[2] allowing to stream games wirelessly from computer to headset. To deliver a good performance to the end user, XR traffic has stringent requirements in both application-level throughput, which can go over 100 Mbps, and delay, which needs to be well below 10 ms.

Wi-Fi struggles to provide delay guarantees: Wi-Fi's operation at the MAC layer is based on distributed channel access due to its operation in the unlicensed spectrum and the inherent requirement of using Listen Before Talk (LBT). For that reason, the contention among devices associated with the sharing of the same frequency channels has a direct impact on the delay experienced by the users, which deteriorates as the number of contenders increases. Further, XR applications have stricter requirements for the uplink (UL) delay, which is harder to control by the Access Point (AP) due to the spontaneous nature of such type of traffic.

IEEE 802.11be (Wi-Fi 7) [3], [4] is envisioned as an enabler for lowering network delay with Multi-Link Operation (MLO). A key feature of Wi-Fi 7, MLO allows a device to connect to multiple bands or channels through a single association and to transmit packets simultaneously over them, thus multiplying the available bandwidth by the number of radios on a device. Medium access is also independent for each radio, leading to more transmission opportunities and reduced contention as well [5].

[1]https://www.meta.com/help/quest/articles/headsets-and-accessories/oculus-link/connect-with-air-link/

[2]https://store.steampowered.com/app/353380/Steam_Link/

Wi-Fi's capability to support XR applications has been tested in [6], comparing the performance of wired and wireless deployments, and showing that, while Wi-Fi can achieve similar results than a wired connection, this only happens for devices that are close to the AP and with direct line of sight. In contrast, the tests in [6] also showed that a poor Received Signal Strength Indicator (RSSI) leads to inconsistent performance and lower frame rates. In [7], a setup with multiple VR users over Wi-Fi was studied, showing that scheduling the uplink transmissions (i.e., using OFDMA to improve multi-user contention) leads to worsened performance overall than just using DCF. In [8], user experience was studied for wired and wireless VR setups, also highlighting that a direct line of sight is necessary to ensure comparable Quality of Experience (QoE) between wired and wireless setups. Different types of XR applications were studied and classified in [9], [10], as well as their requirements and the possibility to cover them based on the current efforts done in 5G and Wi-Fi standardization. In [11], a performance evaluation model was proposed for edge-assisted XR applications, considering battery usage, end-to-end delay and handoff delays. In [12], the authors analyzed a multi-user VR setting deployed over Wi-Fi, and concluded that Quality of Service (QoS) enforced by the standard Enhanced Distributed Channel Access (EDCA) is insufficient to support the stringent delay and packet loss requirements of such a setting. They then proposed a new architecture to improve performance by separating the downlink and uplink using the 802.11ad/ay 60 GHz band and the 802.11ax 5 GHz band, respectively.

In the particular case of Wi-Fi 7's MLO, the work in [13] showed that adding a second link to traditional Wi-Fi can lead to order of magnitude gains in the 90th percentile delay for real-time applications. In [14], a dataset using real-world channel occupancy traces was used to test MLO performance, also showing a similar order of magnitude improvement over SLO in the 95th percentile delay. The performance of Wi-Fi 7 for AR applications was studied in [15], concluding that MLO can serve more users than SLO with equivalent bandwidth. None of these studies, however, have considered VR traffic characteristics and performance requirements in detail.

In this paper, we focus on realistic VR streaming applications, and study Wi-Fi 7's ability to effectively meet their stringent requirements. Unlike prior work, we study the impact of VR traffic on Wi-Fi 7 networks, which we test for both Single Link Operation (SLO) and Multi-Link Operation (MLO). Using simulation results, we showcase the relationship between different transmission parameters, namely the Modulation Coding Scheme (MCS) and channel bandwidth, and provide insights on their required configuration for achieving the desired performance for VR applications. Our analysis also studies the effect of an increasing number of users to better understand the limits of Wi-Fi 7 for VR traffic. Our main contributions are as follows:

- We provide an overview of VR gaming traffic based on real traces, and analyze the associated requirements defined by the Wi-Fi Alliance (WFA).
- We conduct an extensive performance evaluation based on several simulations and discuss the feasibility of Wi-Fi 7 for supporting VR traffic. We show that the number of VR users with MLO and N links is higher than N independent SLO APs.
- We show that for MLO, multiple narrow links provide a better support for VR applications than few but wider links, as the channel access delay—the most limiting factor—is significantly scaled down.

II. VR Gaming

A. VR Streaming Setup

Streaming VR gaming applications offloads the main tasks to a server, taking the computational load of rendering the video, audio and any other necessary data away from the Head Mounted Display (HMD). The rendered video and audio is then transmitted through the internet to the HMD, which acts as a client, and reproduces the incoming video to the user, also capturing the user inputs to send back to the server. This approach is also known as split-rendering VR.

In our setup, the server is connected to the AP directly via a 1 Gbps Ethernet link, and the HMD is connected to the AP wirelessly. Fig. 1

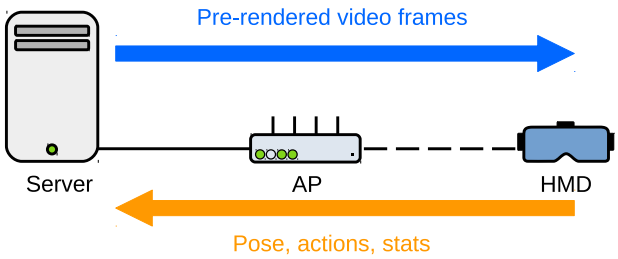


Fig. 1: VR streaming components.

shows the main components of VR gaming streaming in our particular setup.

B. VR Traffic Distribution

To analyze and replicate VR traffic, we use the traces from [7], which can be found in Zenodo as a dataset [16]. They were obtained using Air Light VR (ALVR), which was installed in both the server and HMD. ALVR allows the streaming of VR games over Wi-Fi, as well as gives the user control over several stream settings, such as resolution, refresh rate, codec used, and transport protocol (UDP or TCP). ALVR creates a bridge between the server and the HMD, transmitting audio, video, and tracking. Video is compressed at the server using either the H.264 or H.265 codecs. Audio is sent raw using Pulse-Code Modulation (PCM).

Tests were performed at different resolutions and refresh rates, using H.264 coding and UDP for the transport protocol. Wireshark was used to capture the traffic on the server. These captures, which have been replicated in our simulations, reflect the generation patterns for VR gaming. The traffic patterns are fully described in Section III-A. A comparison of the captures and our simulator output is shown in Fig. 2.

Downlink traffic: For the downlink (DL) traffic there are two types of packets: video and audio. Video is transmitted in batches of packets separated as a function of the frame rate. In this case, it is 90 frames per second (FPS), which leads to an interval of 11.11 ms between batches. At a 100 Mbps application rate, each batch contains an average of 96

video packets of 1448 bytes. The audio is transmitted at a different rate (25 ms) and in batches of 4 packets.

Uplink traffic: In the uplink (UL), we have information about the tracking, pose, and stream statistics. They also follow the video frame rate, with 3 packets of size 106 bytes per video frame, and a single one of 212 bytes, which we believe accounts for the pose and stats, respectively.

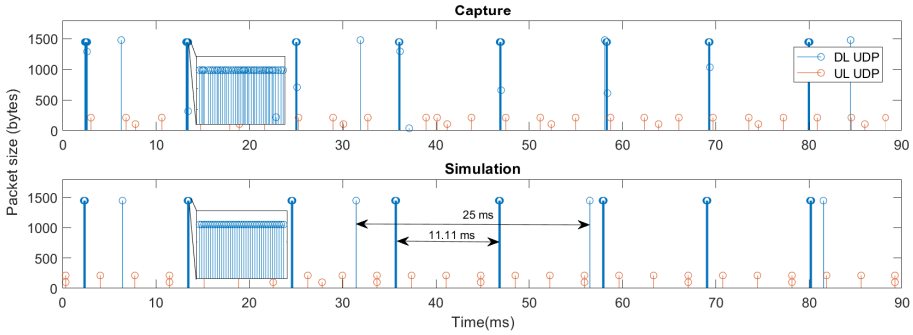


Fig. 2: VR traffic distribution obtained from the capture (top figure) and the simulation (bottom figure).

C. Throughput and Delay Requirements

VR content has strict throughput and delay requirements. It requires real-time rendering, meaning that a large amount of data needs to be transmitted constantly and consistently, and buffering is not possible. VR frame rates are high, ranging from 72 FPS to 120 FPS. This frame rate sets the pace at which traffic is generated, and so the higher the quality of the stream, the more frequent the transmissions. Video quality is also affected by its bitrate, which can range from 40 to 200 Mbps. The delay is particularly important as well, not only to deliver a good video experience, but to avoid that the user suffers dizziness. Finally, the rendered video in the downlink changes based on the inputs of the user, which are delivered by the uplink, thus it is important to protect the uplink so that the downlink is displaying the correct output in a timely

TABLE I: Reliability and delay requirements defined by the Wi-Fi Alliance for VR gaming.

Type of Traffic Stream	**Required Reliability (Percentile)**	**Maximum Recommended Delay (ms)**
Video frames (DL)	75th	5
	95th	10
	99.9th	50
Pose, IMU Controller inputs (UL)	90th	2
	99.9th	10

manner. In this work, we will look at the delay thresholds set by the Wi-Fi Alliance for VR gaming [17], which we summarize in Table I.

III. System Model

A. VR Traffic Characterization

VR traffic is periodic, defined mainly by its total traffic load and the frame rate used by the VR application. As feedback from the HMD is continuously transmitted to the server, a constant video bitrate is generated without exception. We match these main characteristics in our simulation: the frame rate ϕ sets the inter-arrival time Δ of the video data, with downlink packet batches separated by $\Delta = \frac{1}{\phi}$ secs. The video bitrate ρ sets the size of the downlink video batches (in packets per batch), which corresponds to $N_{\text{batch}} = \Delta\frac{\rho}{\text{L}}$, where L is the video packet size.

B. Channel Access

We consider two main modes of operation (represented in Fig. 3):

- **Single Link Operation (SLO):** Current Wi-Fi operation, used as our baseline. APs and STAs connect through a single link and then perform backoff to access it. Packets are transmitted sequentially.
- **Multi-Link Operation STR (MLO):** MLO Simultaneous Transmit and Receive (STR)[3] allows APs and STAs to connect through

[3]For the remainder of the paper, we use MLO to refer to MLO STR operation.

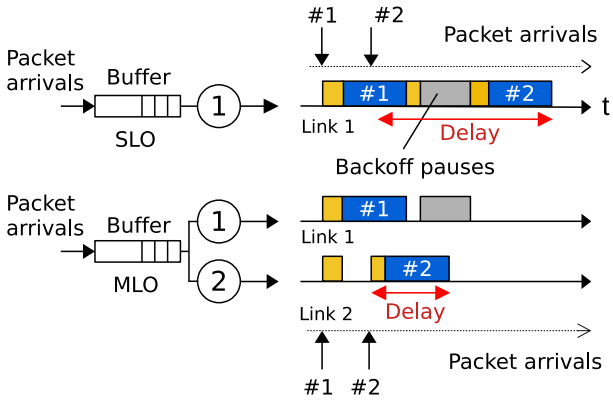


Fig. 3: Channel access modes: SLO (top) and MLO (bottom).

multiple channels at the same time. Each link uses an independent backoff timer, thus packets can be transmitted opportunistically through both links. Fig. 3 shows packet #2 arrives at the buffer once packet #1 is in the middle of being transmitted through link 1. Backoff is then performed in the second link, and the packet is transmitted at the same time as packet 1, reducing the delay in comparison to the sequential transmission in SLO.

C. Scenario

We consider a single Basic Service Set (BSS) Wi-Fi network. It consists of one AP and K VR stations. The VR server has a direct cabled connection to the AP. We consider Wi-Fi 7 modulation and coding schemes (up to 4096-QAM), and path loss at 5 GHz band is modeled considering the 802.11ax residential scenarios [18]. Simulation parameters can be found in Table II.

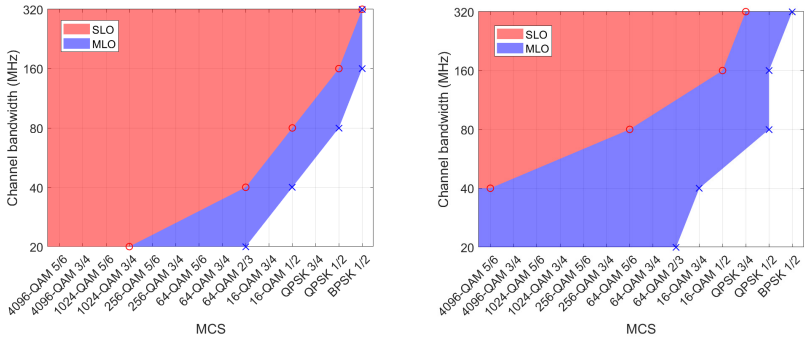
IV. Performance Evaluation

A. Required MCS and Channel Bandwidth

We start our study by verifying the minimum combination of MCS and channel bandwidth required for delivering a good experience to the

TABLE II: Simulation parameters.

Name	Value
Channel bandwidth	20, 40, 80, 160, 320 MHz
Transmission power	23 dBm
Clear channel assessment	-82 dBm
Spatial streams	2
PER	10%
Max. packet aggregation	1024 A-MPDU
Buffer size	5000 packets
Iterations	100 seeds
Simulation time	10 seconds



(a) Downlink (75%-tile of 5 ms) (b) Uplink (90%-tile of 2 ms)

Fig. 4: Minimum MCS to accomplish Wi-Fi Alliance thresholds for different channel widths.

end user according to Wi-Fi Alliance specification for XR gaming [17], defined in Table I. To do so, we compare the ability of Wi-Fi 7 MLO with respect to previous generation SLO only. We consider a single AP and STA operating at different channel bandwidths, from 20 MHz to 320 MHz. For each bandwidth, different MCS values are evaluated to find the combinations that meet a good user experience.

Fig. 4 compares the results obtained with latest Wi-Fi 7 MLO and SLO-only devices for both downlink (Fig. 4a) and uplink (Fig. 4b).

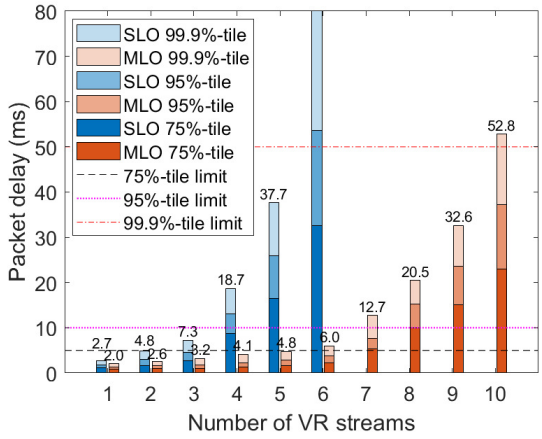
For SLO, the uplink is far more restrictive than the downlink and a minimum bandwidth of 40 MHz associated to an extremely high MCS, i.e., 4K QAM, is required. In the downlink, VR requirements can be met even with 20 MHz and 1024-QAM with coding rate 3/4. For all other bandwidth configurations, the UL in SLO requires much higher MCS values than the downlink, and even with 320 MHz, the lowest MCS cannot achieve the 2 ms requirement at 90th percentile. On the contrary, for MLO the disparity between uplink and downlink is much lower, generally requiring similar MCS values, which are also lower than the ones required by SLO.

Takeaway: The UL requirements are harder to meet than the DL despite having a much lower traffic load. MLO offers a clear advantage over SLO even in situations where the total used bandwidth is comparable with SLO (e.g., SLO using 160 MHz and MLO using two links of 80 MHz), as MLO implicitly relieves the UL/DL self-contention by taking advantage of transmitting over multiple links. In addition, by demanding lower MCS with the same total bandwidth, MLO provides a better flexibility with respect to SLO for VR streaming in more challenging propagation conditions, such as being far away from the AP or not in direct line of sight.

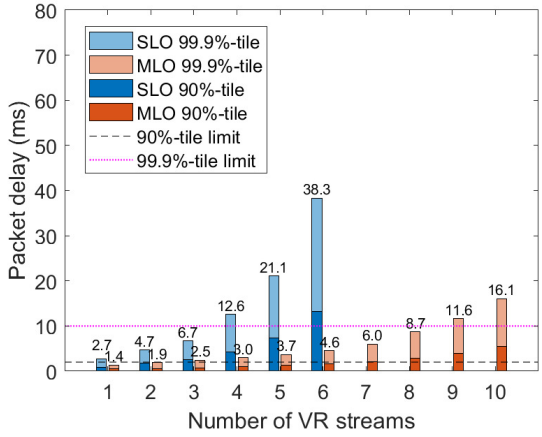
B. Increasing Number of VR Users with MLO

We now study the capacity of MLO to serve a certain number of VR streams and compare it to legacy SLO. We set a single AP transmitting multiple VR streams of 100 Mbps and 90 FPS to end users. All STAs have the same MCS of 1024-QAM 5/6 over 80 MHz channels.

Fig. 5a shows the 75th, 95th and 99.9th percentiles of the packet delay suffered by DL traffic, for both SLO and MLO, and for an increasing number of VR streams. The dashed lines highlight the DL delay thresholds for each percentile (as defined in Table I). We can observe that SLO's delay increases much faster than MLO's, allowing three streams before exceeding the 75th percentile threshold of 5 ms (black dashed line). MLO comfortably allows up to six streams, and offers delay improvements at lower loads (e.g., for three streams, SLO has a 99.9% delay of 7.3 ms, while MLO has a delay of 6 ms for six streams).



(a) Downlink packet delay



(b) Uplink packet delay

Fig. 5: Packet delay as number of users increases. The straight lines indicate the respective Wi-Fi Alliance requirements for DL and UL.

Generally, the number of extra streams that can be added with MLO is directly proportional to the extra number of links enabled. However, in all cases, MLO guarantees a lower delay in both DL and UL compared to SLO while supporting twice the number of VR streams, showing that

the opportunistic nature of MLO offers a slight improvement to network delay.

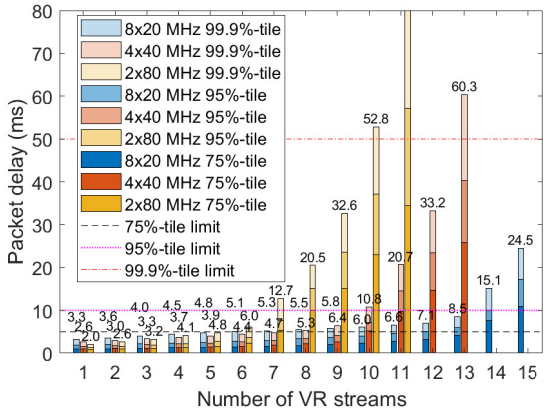
Similar to the DL, Fig. 5b shows the 90th and 99.9th percentiles for the packet delay at the UL. It can be observed that for SLO, the UL does not allow more than two users, exceeding the 2 ms threshold for the third user, once again showing that the UL limits SLO connections. In contrast, MLO can sustain six users, the same number as in the downlink. Additionally, the 99.9th percentile delay for six MLO users is 4.6 ms, which is lower than the 4.7 ms achieved by SLO with two users. This indicates that even if we had two SLO networks to match the bandwidth, SLO would only support four VR users, while MLO allows for an extra 50%.

Takeaway: The MLO advantage over SLO is a consequence of the increase in the number of independent channel access instances over the available links, rather than the increased bandwidth, allowing MLO to sustain more VR users than SLO deployments in non-overlapping bands using the same total bandwidth.

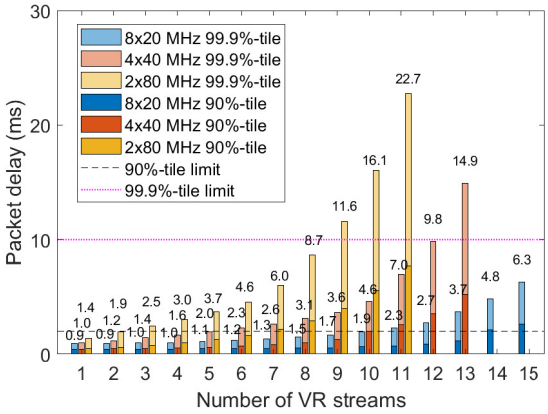
C. Configuring MLO for VR applications

We now attempt to further increase the number of VR users in the network by reducing contention, distributing the same bandwidth over a different number of links, thus increasing the opportunities for streams to be transmitted in parallel. MLO configurations of two, four and eight links of 80 MHz, 40 MHz and 20 MHz are used respectively, maintaining overall bandwidth used, but spreading it over an increasing number of links.

Fig. 6a shows the downlink packet delay for all configurations. We can observe that adding more links, even if each has lower bandwidth, can increase the number of VR users supported. With two links we get to serve up to six users, with four links we can serve up to nine users, and eight links allow supporting up to thirteen users. Note that if we focus on the cases where all configurations meet the requirements, for up to three users, two links of 80 MHz result in lower delays overall. Then from four to eight users, four links of 40 MHz is the best option. Beyond nine users, it is better to use eight links.



(a) Downlink



(b) Uplink

Fig. 6: Packet delay for different configurations of links and bandwidth.

Fig. 6b shows the uplink packet delay for all three configurations, in which we can observe that increasing the number of links improves the delay for all cases. These results show there is a clear trade-off between the bandwidth used per channel and the links-per-user ratio. When we have few users but many links, as the downlink arrives in batches, most

links end up being idle while a subset are used for transmitting all the data. In these cases, a higher bandwidth allows for higher data capacity and lower transmission times. Once the number of users increases, we have a higher chance of transmitting simultaneously on all the links, and having fewer links results in increased waiting times in the queue. If we have four users and two links, we can only support up to two users simultaneously. Once there are packets from more users than links, some packets must wait for the ongoing transmission to finish before being transmitted, thus leading to increased delay.

The uplink behaves differently due to two reasons: the first is the lower traffic load required per STA, and the second is the timing between packets. As uplink packets do not arrive in batches, the buffer does not fill as quickly as the downlink, resulting in minimal aggregation. Transmissions are always short, thus not benefiting from higher capacity, and having more links allows all packets to be sent as soon as possible.

Takeaway: There is a trade-off in the DL between channel access opportunities and transmission time (more independent links vs. more bandwidth per link). This is also affected by the number of VR users in the network. Under the assumption of using the same total bandwidth, the number of configured links should be large enough to guarantee that the delay-sensitive UL transmissions are not blocked by channel access contentions and, at the same time, maintain a sufficient bandwidth to support the high DL throughput demand of VR applications.

V. Conclusions

In this paper, we modeled real VR traffic traces to test Wi-Fi MLO capabilities to support the stringent requirements of VR traffic in terms of MCS-bandwidth pairs and number of links. We showed that MLO can support VR applications with lower bandwidth and lower MCS than SLO, providing more robustness over a wider range of propagation conditions. We also showed that MLO offers lower delays due to increasing the number of independent channel access instances and that using an equivalent number of links, MLO allows an extra 50% of users per network over SLO. In order to accommodate a higher number of VR users, a proper configuration of links and channel bandwidth is required.

Spreading the same bandwidth over more links can allow for more users to contend in the network without exceeding delay requirements, but for a lower user count, having an excess of links may not result in any gains.

In future work, we intend to further dive into ways to utilize MLO to further increase delay gains, as well as study coexistence between MLO and SLO devices. Wi-Fi 8 [2] will also bring even further improvements that could be used to drive this type of content, such as Multi-AP coordination, allowing to reduce contention between VR users associated to different APs.

REFERENCES

[1] E. Oughton, G. Geraci, M. Polese, V. Shah, D. Bubley, and S. Blue, "Reviewing wireless broadband technologies in the peak smartphone era: 6G versus Wi-Fi 7 and 8," *Telecommunications Policy*, vol. 48, no. 6, p. 102766, 2024.

[2] L. Galati Giordano, G. Geraci, M. Carrascosa, and B. Bellalta, "What will Wi-Fi 8 be? A primer on IEEE 802.11bn ultra high reliability," *arXiv preprint arXiv:2303.10442*, 2023.

[3] "IEEE Draft 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: Enhancements for Extremely High Throughput (EHT)," *IEEE P802.11be/D3.0, January 2023*, pp. 1–999, 2023.

[4] A. Garcia-Rodriguez, D. López-Pérez, L. Galati-Giordano, and G. Geraci, "IEEE 802.11be: Wi-Fi 7 Strikes Back," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 102–108, 2021.

[5] M. Carrascosa-Zamacois, G. Geraci, L. Galati-Giordano, A. Jonsson, and B. Bellalta, "Understanding multi-link operation in Wi-Fi 7: Performance, anomalies, and solutions," in *Proc. IEEE PIMRC*, 2023, pp. 1–6.

[6] M. Jansen, J. Donkervliet, A. Trivedi, and A. Iosup, "Can My WiFi Handle the Metaverse? A Performance Evaluation Of Meta's Flagship Virtual Reality Hardware," in *Companion of the 2023 ACM/SPEC International Conference on Performance Engineering*, 2023, pp. 297–303.

[7] C. Michaelides, M. Casasnovas, D. Marchitelli, and B. Bellalta, "Is Wi-Fi 6 Ready for Virtual Reality Mayhem? Aa Case Study Using One AP and Three HMDs," *Authorea Preprints*, 2023.

[8] G. Gonçalves, P. Monteiro, M. Melo, J. Vasconcelos-Raposo, and M. Bessa, "A comparative study between wired and wireless virtual reality setups," *IEEE access*, vol. 8, pp. 29 249–29 258, 2020.

[9] A. Hazarika and M. Rahmati, "Towards an evolved immersive experience: Exploring 5G- and beyond-enabled ultra-low-latency communications for augmented and virtual reality," *Sensors*, vol. 23, no. 7, p. 3682, 2023.

[10] I. F. Akyildiz and H. Guo, "Wireless communication research challenges for extended reality (XR)," *ITU Journal on Future and Evolving Technologies*, vol. 3, no. 1, pp. 1–15, 2022.

[11] A. Mallik, J. Xie, and Z. Han, "A Performance Analysis Modeling Framework for Extended Reality Applications in Edge-Assisted Wireless Networks," *arXiv preprint arXiv:2405.07033*, 2024.

[12] J. Ahn, Y. Y. Kim, and R. Y. Kim, "Virtual reality-wireless local area network: Wireless connection-oriented virtual reality architecture for next-generation virtual reality devices," *Applied Sciences*, vol. 8, no. 1, p. 43, 2018.

[13] G. Naik, D. Ogbe, and J.-M. J. Park, "Can Wi-Fi 7 support real-time applications? On the impact of multi link aggregation on latency," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[14] M. Carrascosa-Zamacois, G. Geraci, E. Knightly, and B. Bellalta, "Wi-Fi multi-link operation: An experimental study of latency and throughput," *IEEE/ACM Transactions on Networking*, 2023.

[15] M. Alsakati, C. Pettersson, S. Max, V. N. Moothedath, and J. Gross, "Performance of 802.11 be Wi-Fi 7 with multi-link operation on AR applications," in *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2023, pp. 1–6.

[16] C. Michaelides, M. Casanovas, D. Marchitelli, and B. Bellalta, "VR Gaming Dataset - Multi-users tests," https://doi.org/10.5281/zenodo.8169785, 2023.

[17] Wi-Fi Alliance, "Wi-Fi delivers immersive VR gaming," https://www.wi-fi.org/system/files/VR_Gaming_Highlights_20231215_0.pdf, 2023, last accessed 17/06/2024.

[18] IEEE 802.11 TGax, "TGax Simulation Scenarios," https://mentor.ieee.org/802.11/dcn/14/11-14-0980-16-00ax-simulation-scenarios.docx, 2015, last accessed 17/06/2024.