Boosting Application Performance using Heterogeneous Virtual Channels: Challenges and Opportunities

Talal Touseef[1], William Sentosa[1], Milind Kumar Vaddiraju[1], Debopam Bhattacherjee[2], Balakrishnan Chandrasekaran[3], Brighten Godfrey[1,4], Shubham Tiwari[2]

[1]UIUC, [2]Microsoft Research India, [3]Vrije Universiteit Amsterdam, [4]VMware

Abstract

Interactive networked applications require high throughput, low latency, and high reliability from the network to provide a seamless user experience. While meeting these three requirements simultaneously is difficult, there has been an emergence of *heterogeneous virtual channels* (HVCs) which support some subset of them at the expense of the others. For instance, URLLC sacrifices throughput to achieve low latency and reliability in 5G NR, and Wi-Fi 7 and other novel Internet architectures provide similar disparate types of service. Prior work either focuses on aggregating the bandwidth of these channels whilst neglecting their unique properties or fails to generalize in the sense of achieving high performance across different applications and channels. To utilize HVCs to their fullest, we argue that there are challenges and opportunities across the network, transport and application layers, and the application-transport interface of the network stack. In this work, we explore the trade-offs of these architectural choices in the context of web browsing and real-time video, and identify the constituting principles of a design that is general, performant, and deployable.

CCS Concepts

• **Networks** → **Network architectures**; **Mobile networks**; *Transport protocols.*

Keywords

Heterogeneous Virtual Channels, 5G, WTSN, Transport Layer

HotNets '23, November 28–29, 2023, Cambridge, MA, USA

ACM ISBN 979-8-4007-0415-4/23/11…$15.00
https://doi.org/10.1145/3626111.3628193

ACM Reference Format:
Talal Touseef[1], William Sentosa[1], Milind Kumar Vaddiraju[1], Debopam Bhattacherjee[2], Balakrishnan Chandrasekaran[3], Brighten Godfrey[1,4], Shubham Tiwari[2]. 2023. Boosting Application Performance using Heterogeneous Virtual Channels: Challenges and Opportunities. In *The 22nd ACM Workshop on Hot Topics in Networks (HotNets '23), November 28–29, 2023, Cambridge, MA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3626111.3628193

1 Introduction

Increasingly, networked applications are becoming more interactive, resulting in greater demands being placed on the network. For instance, extended reality (XR) applications require high reliability and a latency less than 20 ms to provide immersive experiences [7] and avoid simulator sickness [19]. Similarly, cloud gaming requires high throughput for a smooth visual experience and a latency less than 100 ms to continuously engage a player [35]. Even web browsing, a critical mobile application [43], is sensitive to latency with a mere 100 ms increase causing a 7% decrease in conversion rate for retail websites [6].

Simultaneously, both WAN and access network technologies have begun to incorporate *heterogeneous virtual channels*, i.e., a collection of channels which individually excel in *different* dimensions of performance — throughput, latency, or reliability. For instance, 5G NR supports high throughput via enhanced Mobile Broadband (eMBB) and low latency and reliability via Ultra-Reliable and Low-Latency Communication (URLLC) [20]. Wi-Fi is poised to offer a similar virtual channel of deterministic latency by employing the synchronization and scheduling tools [16, 17, 36] described in the IEEE 802.1 Time Sensitive Networking (TSN) standards [5]. Further, the advent of Wi-Fi 7 will allow the use of multiple links in parallel, especially in the contention-free 6 GHz band, to reliably transmit information [17]. Finally, WAN paths are also diversifying with the proliferation of networks such as cloud providers' private WANs, LEO satellite networks [8, 45, 46], and potentially novel internet architectures such as cISP [10] and SCION [50].

These heterogeneous virtual channels can be leveraged to improve application performance. However, existing solutions that utilize multiple paths either ignore the diversity of their properties or exploit them poorly. For example, MPTCP [47] distributes traffic across multiple paths but will congest a low bandwidth URLLC link due to its extremely low RTT value leading to poor application performance [42]. Similarly, IANS [23, 24] which uses a modified socket API [40] performs suboptimally as it only maps content (i.e., web object in [23] or video chunk in [24]) to a single channel. Solutions that do leverage heterogeneity are sub-optimal or narrow in scope. Consider DChannel [42], which steers individual packets between channels at the network layer. Although DChannel seeks to be general-purpose in the sense of not being tailored to a specific application, it fails to profit from the knowledge of application goals and message semantics thereby leaving significant room for improvement as we show later. Xlink [51], a cross-layer design, uses application information to steer packets, but its tight integration with the application makes it unusable in a broader context. Further, its design needs modification to work with HVCs as it was developed to utilize paths that were not starkly different from each other. Evidently, there is much scope for better utilization of HVCs.

In this work, we attempt to identify the principles and techniques that can be used for a design that can achieve high performance across a multitude of applications using a suitable combination of available HVCs. To do this, we evaluate solutions restricted to one layer of the network stack, or spanning multiple layers. First, at the network layer, we note that steering individual packets, different classes of which have different impacts on application performance, into channels with distinct properties leads to a powerful solution. DChannel [42] employs this approach and reduces page load time (PLT) for web browsing significantly. However, we show that it performs sub-optimally due to its completely application-agnostic design. We argue that packet steering can be extended by considering other trade-offs such as bandwidth vs. cost of usage and bandwidth vs. reliability. We also observe the utility of using channel information from underlying HVCs to improve steering performance.

Further, delay-dependent congestion control algorithms like BBR [14] may get confused as they misinterpret the latency spike caused by packet steering as congestion. This issue naturally leads to a transport layer solution that is aware of the latency difference of the HVCs. Operating at the transport layer also provides better knowledge about the reliability of different channels, and allows control and data packets to be steered more effectively. Modifying the application-transport interface to augment a steering solution with application-layer information in a cross-layer design further improves HVC use. For instance, awareness of flow or message sizes allows valuable channels to be used more or less aggressively depending on how much of the flow or message is remaining. Further, knowledge of packet and flow importance allows some to be prioritized and others ignored. In particular, we implemented a cross-layer scheme for real time video streaming that uses knowledge of application message priorities for steering to reduce the 95th percentile latency from 176 ms to 78 ms (2.26x improvement) compared to steering without this information. Such a gain is quite significant to real time video which forms a critical part of several interactive applications. This is particularly vital when multiple flows compete to use a resource-constrained HVC. We show that as few as two background flows, unimportant from an application or end user's perspective, can cause as much as a 138 ms increase in PLT for web browsing.

In Section 3, we discuss these various architectural choices and conclude that steering individual packets or segments, access to and use of easily available application information such as flow or packet priorities, and HVC information such as latency and packet drop rate form the essential components of high performance and yet general design.

Finally, several factors make us optimistic about the adoption of this direction of work. First, packet steering solutions at the network layer are feasible to deploy, requiring support only at a few points in the infrastructure. Second, applications prioritizing user experience can very reasonably be expected to adapt to a new transport layer API for improved performance as the deployment success of Xlink [51] and QUIC [29] indicates. Third, although URLLC [3] has been envisioned for niche applications, the standards provide considerable flexibility to accommodate new use cases [20]. Further, Wi-Fi 6 and 7 already include the physical layer features that enable deterministic latency and reliability, and as the design and development of other wireless TSN mechanisms is still ongoing, there is an opportunity to make them deployable.

With this paper, we hope to pique the community's interest and start a discussion about the simultaneous use of these HVCs, across and within different access networks, now available to enhance application performance.

2 Heterogeneous Virtual Channels

In this section, we discuss different types of HVCs in the case of 5G, Wi-Fi, and WAN designs.

2.1 5G NR

5G New Radio (NR) offers different modes of operation [20] to meet various network requirements: enhanced mobile broadband (eMBB) for high data rate, ultra-reliable low latency communication (URLLC) for low latency applications, and massive machine-type communications (mMTC) for massive connectivity. We will discuss eMBB and URLLC as HVCs with bandwidth and latency trade-offs.

eMBB is used as mobile broadband to support general mobile phone applications and provides high throughput but also incurs high latency. Based on a recent measurement of a commercial mmWave 5G network, eMBB can achieve a TCP throughput of up to 2 Gbps for download and 60 Mbps for upload [32]. However, under device mobility its packet round trip time (measured through probing) can be as high as 236 ms in its 98th percentile [42].

In contrast, URLLC sacrifices data rate for low latency. URLLC targets 0.5 ms of air latency between the client and the RAN with 99.999% reliability for small packets (e.g., 32 to 250 bytes) [4]. A number of optimizations are made in PHY and even the cellular core to do this. From the specifications, the end-to-end latency ranges from 2 to 10 ms and throughput from 0.4 to 16 Mbps [2]. Without a solution leveraging HVCs, URLLC is expected to serve only niche applications like autonomous driving due to its limited bandwidth.

2.2 Wi-Fi

Wi-Fi operates in the 2.4, 5, and now with Wi-Fi 6/6E, in the 6 GHz bands. A lot of latency in Wi-Fi comes from contention in the unlicensed spectrum. Several advances in the standards, some extending Ethernet's TSN tools to Wireless and others inherent to Wi-Fi, deal with this and have produced channels that provide low latency and high reliability at the cost of low bandwidth. With Wi-Fi 6, an access point (AP) can coordinate the uplink transmissions of multiple User Equipments (UEs) simultaneously using OFDMA leading to parallel use of the wireless medium and therefore lower latency. Further, with the application of 802.1AS (time synchronization) and 802.1Qbv (time-aware scheduling) to Wi-Fi, it is possible to synchronize UEs and APs within and across Basic Service Sets to classify and prioritize time-sensitive traffic. However, this requires a central controller, loses multiplexing gains with non-TSN traffic having to wait and restricts how many users can be supported.

Wi-Fi also introduced Multi-Link Operation (MLO), which allows UEs and APs to use links in the different frequency bands—2.4, 5, and 6—simultaneously [15, 25]. While this can be used to aggregate bandwidth, it can also be used for deterministic latency by directing time-sensitive traffic into the contention-free 6 GHz band. Further, it can be used to achieve reliability by replicating packets across the two links [25] at the expense of throughput. Currently, these are only envisioned for use in niche applications such as IoT and industrial automation in controlled environments [16, 36]. A key consideration is analyzing the trade-offs of TSN—unlike cellular, resources are not dedicated to a user and other users bear the cost of one's use of the low latency service. Another concern is making these commercially deployable. However, their design is still underway, and we believe there is much scope for making these services generally usable.

2.3 Wide Area Networks

Low-Earth Orbit satellite networks (e.g., SpaceX Starlink [45]) could offer much lower latencies than the terrestrial Internet by using space-based inter-satellite lasers [27, 38] operating at the speed of light, but with lower bandwidth than fiber due to radio up/downlink bottleneck. Alongside the terrestrial Internet, these could offer HVCs to users. HVCs may also materialize at CDN and content provider infrastructures to the extent that more novel network architectures (e.g., SCION [50] and cISP [10]) are widely deployed. CDN servers may, for instance, use a microwave-based ultra-low-latency network in addition to the conventional terrestrial fiber optic network [10]. While cISP's microwave links operating at the speed of light are much faster than optical fiber, their bandwidth and reliability are vastly lower. Similarly, SCION enables a host to learn of the multiple paths to a destination along with their (vastly different) performance characteristics [50], effectively transforming them into HVCs.

3 Leveraging HVCs Across Different Layers

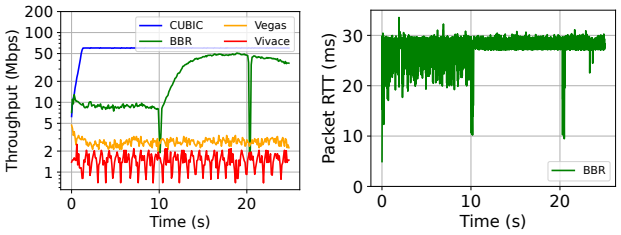
We discuss network, transport and application layer solutions to leveraging HVCs to boost application performance, and outline the opportunities and challenges at each layer.

3.1 Network Layer

HVCs can be leveraged at the network layer by steering IP packets into distinct channels without any application input. DChannel [42] is the state-of-the-art system that does this for HVCs (like eMBB and URLLC) balancing bandwidth and latency in cellular networks. It uses a heuristic to assess if the *reward* of sending a packet via a low bandwidth, low latency channel exceeds the *cost* of doing so. For eMBB and URLLC, it shows a considerable promise as it improves web browsing PLT by 16-40%. Implemented as a *shim* layer that intercepts packets transparently to both the application and the transport protocol, it is also very deployable.

Packet steering can be made more general by considering trade-offs other than bandwidth vs. latency. A latency vs. cost trade-off, where the lower latency channel costs more money per byte sent, is interesting, especially if it is provided by, say, cISP [10]. Bandwidth vs. reliability is another interesting trade-off. For instance, MLO (§2.2) can provide reliability in Wi-Fi by redundantly transmitting data across multiple channels, thereby sacrificing bandwidth. Even for HVCs with a bandwidth-latency trade-off, packet steering can likely be improved by using information about the underlying HVC from the MAC and PHY layers in wireless networks such as Wi-Fi [44] and cellular [48].

However, in assuming that each user gets a share of the wireless resources like in cellular networks, DChannel cannot readily be extended to Wi-Fi where the cost of sending one user's packets over a low latency channel is borne by others. Further, network layer packet steering solutions critically



(a) (b)

Figure 1: (a) Throughput achieved by CCAs with DChannel on two paths with a latency-bandwidth trade-off. Y-axis is in logarithmic scale. (b) Packet RTTs observed by BBR when using DChannel.

suffer from a lack of insight into application needs. Consequently, they end up steering packets that do not improve application performance to the costly low latency channel. Operating at higher layers of the stack (§3.2 & §3.3) can help address this deficiency. Packet steering can even *degrade* application performance by confusing transport protocols, which rely on packet delay and are unaware of the latency differences between different paths.

To demonstrate this, we ran tests on Pantheon [49] with DChannel steering packets between two emulated HVCs with a latency-bandwidth trade-off: one has 50 ms RTT with 60 Mbps bandwidth and the other has 5 ms RTT and 2 Mbps bandwidth. These numbers respectively reflect 5G Lowband eMBB performance under movement [42] and URLLC. We tested three congestion control algorithms (CCAs) that adjust their sending rate based on packet delay—TCP BBR [14], TCP Vegas [13], and PCC Vivace [22]—with this setup. In addition to this, we also tested TCP CUBIC [28], a loss-based CCA which is far less sensitive to delay. Figure 1a presents the results of these experiments.

CUBIC, a loss-based CCA, utilizes the full throughput (60 Mbps) of the high bandwidth channel. All delay-dependent CCAs underutilize the link: BBR, Vegas, and Vivace achieve 26.5 Mbps, 2.73 Mbps, and 1.49 Mbps of averaged throughput respectively. This occurs as these CCAs misinterpret the resulting RTTs caused by packets switching HVCs.

To better understand this, we plot the measured RTT when running BBR across time in Figure 1b. The RTT varies quite a bit, confirming our suspicions of transport layer confusion caused by sudden changes in RTT. This is exacerbated by DChannel prioritizing steering control packets, including BBR's probes, to the low latency path. This produces an initial underestimate of the RTT followed by an ostensible spike upon reverting to the high latency path leading to the perceived congestion in the first 10 s of Figure 1a. At the 10 s mark, BBR drains the queue to get a better estimate of the minRTT. However, this estimate is lower than the RTT of the high bandwidth channel. As a result, BBR underestimates the Bandwidth Delay Product (BDP) and is not able to fully utilize the high bandwidth channel.

Hence, network layer steering may not be optimal, or even beneficial, depending on the application and the transport protocol. In §3.2 and §3.3, we argue for operating at the transport or application-transport layer to better use HVCs.

3.2 Transport Layer

Previous transport layer designs such as MPTCP [47] and MPQUIC [21] have considered multiple paths but ignored the heterogeneity of individual channels and focused primarily on aggregating bandwidth or supporting handover. Works which have focused on heterogeneous paths [26, 30] still try to aggregate bandwidth and do not fully exploit the space of actions enabled by paths as different as URLLC and eMBB. We argue that combining elements of these approaches into a transport layer design can lead to a more optimal and general solution. Such a transport layer would be aware of the existence of individual virtual channels and their properties, steer individual transport layer segments into them, and adapt congestion control to correctly interpret the resulting RTTs.

Steering of segments continues to be a core component of the solution even at the transport layer, similiar to DChannel's packet steering at the network layer. For low latency channels, this allows individual pieces of information that allow the end application to take some action to be accelerated and therefore improves performance. DChannel obtains a significant portion of its gains from accelerating ACKs and other control messages. However, if data is tacked onto the ACK, eMBB is preferentially used over URLLC due to the latter's low bandwidth, leading to sub-optimal performance. When individual segments are allocated to different virtual channels, an ACK can be separated from the rest of the message content and the two can be sent via different channels. Further, as it is the transport layer that fragments an application message, segments towards the end of a message can be selectively sent over a low latency path instead of the earlier segments to avoid head-of-the-line blocking. Finally, critical control packets that will cause significant performance degradation when lost can be steered into virtual channels with reliability guarantees.

In addition to the benefits of handling individual segments, awareness of the existence of HVCs will also allow the CCA to adjust its sending rate in a more informed manner. This will reconcile the control loops of congestion control and packet steering and resolve the issues of delay-dependent transport protocols described in §3.1. Further, being a transport layer solution, it pushes most of the complexity to the end host [39] and requires no support from the network itself unlike DChannel which needs a proxy at the packet gateway of the cellular core.

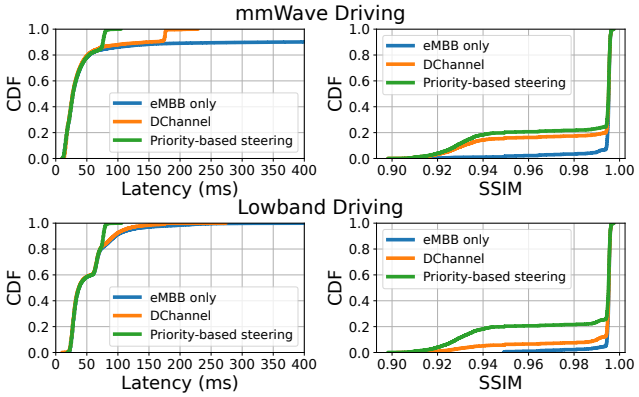


Figure 2: Latency and quality (SSIM) distributions of decoded frames for various steering algorithms for emulated 5G eMBB and URLLC[1].

Traces	eMBB-only	DChannel	DChannel w. priority
Stat.	1697.3	1230.5 (27.5%)	1154.9 (32%)
Drv.	2334.3	1474.6 (36.8%)	1336.8 (42.7%)

Table 1: Web PLT (in ms) with small background traffic using emulated 5G lowband eMBB (stationary and driving traces) with URLLC.

While this solves the problems of a purely network-layer solution, it still leaves room for improvement. This comes from the fact that such an application-agnostic transport layer is unaware of what is important to an application and an end user and treats all flows equally. We discuss these shortcomings and some potential solutions in §3.3.

3.3 Application Layer

Although powerful, HVCs are resource constrained and packets need to be mapped to them judiciously. Input from applications can vastly improve a steering solution's efficacy as they understand their own performance and behaviour the best. For instance, Xlink [51] uses knowledge about the videos being transferred to selectively re-inject packets from earlier frames and streams to decrease buffering time at the start of short videos. Similarly, Socket Intents [40] provides the transport layer with application input about the nature of flows by extending the socket API. However, the former is not general-purpose, focusing only on on-demand video. The latter operates at the granularity of flows instead of packets. Both these elements—a general interface for information exchange and fine-grained steering—are vital to a solution that must leverage HVCs, especially when multiple flows and applications compete for resources.

Two key pieces of information that any application can provide relatively easily are message boundary and priority. These are particularly useful in case a low latency channel is available as they help identify which messages to accelerate and which to not. We demonstrate this for real-time video streaming and web browsing when HVCs making a latency-bandwidth trade-off are available. Here, a message is defined as a sequence of bytes which enable the receiving host to take some useful action once the *entire* sequence is received.

Real-time video streaming is an important component of multiple networked applications like video conferencing, cloud gaming, remote driving, XR, etc. It is time-sensitive, requiring both low latency and high frame quality. During network deterioration, receiving lower-quality frames on time rather than late high-quality frames helps preserve responsiveness and user QoE. In this experiment, we show that providing message priorities and boundaries to the steering algorithm helps outperform even the DChannel packet steering algorithm [42] at minimizing latency, at the cost of frame quality under network deterioration.

We used Scalable Video Coding (SVC) [41] to get messages with different priorities. In SVC, each frame is encoded with multiple spatial (quality) layers. The decoding of a higher layer depends on the successful decoding of all lower layers and the corresponding layer of the previous frame. Thus, the lowest layer (layer 0) is the most important and has the highest priority. It can be decoded on its own and the successful decoding of the higher layers depends on it. Each subsequent higher layer has a lower priority.

We modified the setup from [18] for our experiments. The video source was taken from the MOT17 dataset [31]. It was encoded using the VP9-SVC codec [1] with three layers with target bitrates of 400 kbps, 4100 kbps, and 7500 kbps respectively, resulting in a cumulative bitrate of 12000 kbps for the video. Every 33 ms (equivalent to 30 fps), the sender sent an encoded frame, i.e., the three layers as three separate messages in a single flow over the network as UDP packets. Upon receiving layer 0 of the frame, the receiver waited for 60 ms or for layer 0 of the next two frames to arrive before decoding the received frame. This waiting period helps strike the right balance between latency and quality. Without it, the receiver only ever decodes layer 0 frames and hence experiences very poor quality. In contrast, if it waits for too long, then it will get a very delayed higher-quality frame.

In our experiments, we used a modified DChannel shell to emulate eMBB and URLLC as our HVCs. We emulated URLLC with 5 ms RTT and 2 Mbps bandwidth. eMBB was emulated using the mmWave and Lowband driving traces from [42]. These traces have higher latency variations due to user equipment (UE) mobility and are interesting scenarios to evaluate different steering schemes in. We evaluated three different steering algorithms. The first (our baseline) only used eMBB. The second used DChannel [42] with no modifications. Finally, we implemented a new steering algorithm that maps packets to different channels based on the message they belong to and its priority. The sender embedded these

[1]The CDF of latency with mmWave driving is cut off on the right as the eMBB-only algorithm had an extremely long tail till 6400 ms.

priorities in the packets using a custom application header and prioritized layer 0 (the most important layer) over layers 1 and 2. Thus, the steering algorithm sent layer 0 over URLLC and layers 1 and 2 over eMBB.

Figure 2 shows the distribution of the latency and quality (SSIM) of the decoded frames for the two network traces. Priority-aware steering reduces the latency significantly compared to the other schemes when eMBB latency increases. For mmWave driving, priority-based steering reduces the 95th percentile latency by 1980 ms (26x) and 98 ms (2.26x) over the eMBB only scheme and DChannel respectively, while only reducing SSIM by 0.068 and 0.002 respectively. This reduction is because layer 0 (prioritized message) is sent over URLLC and the receiver always gets it within a narrow time bound as the URLLC latency is low and does not vary. However, DChannel, being unaware of message boundaries and priorities within a flow, treats each packet as a message boundary and tries to accelerate it. As a result, it does not send all the packets belonging to layer 0 through URLLC and performs worse than priority aware steering.

Web browsing with background flows. We now demonstrate the value of knowing information as basic as flow priorities when multiple flows compete. We load web pages while also running two background flows that do not contribute to the PLT. These background flows upload and download small JSON files—a common occurrence for mobile web-based apps that upload logs to the server and pre-fetch data not yet displayed to the user. In one case, packets from all three are steered into eMBB or URLLC HVCs with DChannel. Then, we supply DChannel with flow priorities to prevent the background flows from using the limited URLLC bandwidth.

For the experiment, we recorded 30 popular landing and internal web pages chosen randomly from the Hispar corpus [9] using the Mahimahi framework [33] and replayed them using Mahimahi with HTTP2 [52]. The client had two parallel paths – eMBB (using 5G Lowband stationary and driving traces from [42]) and URLLC (5 ms RTT and 2 Mbps bandwidth) – to the web server, and it ran the Chromium browser to load these pages. DChannel was used for steering with modifications to also support application input. We loaded each page 5 times and found its mean PLT based on the onLoad event [34]. Simultaneously, we spawned two background flows that use cURL to continuously upload (5 KB) and download (10 KB) JSON objects until the experiment was done. Before each fetch, we cleared both the browser and DNS caches. We used TCP CUBIC for this experiment.

Table 1 shows web PLT for three different steering policies. Both versions of DChannel improve PLT considerably over the eMBB-only case where all traffic is delivered over eMBB. Nonetheless, DChannel with priority performs the best, reporting a 6.2% to 9.5% of mean PLT improvement compared to the DChannel without any flow prioritization. This mainly stems from avoiding URLLC queue build-up caused by delivering the low-priority background traffic over the channel. As a result, the web page load traffic can steer more of its traffic to URLLC and receive greater acceleration.

In summary, these results present the significant benefits of incorporating minimal application level information into a solution to leverage HVCs.

4 Open Questions and Research Directions

As seen in §3, network layer steering for HVCs is the simplest but also the least powerful. Further, DChannel requires redesign to work well with different WLAN or WAN architectures. We find that solutions at the transport layer, maybe even with a new application-transport interface, are promising candidates and present some pertinent considerations.

Design. Solutions at the transport and application-transport layers can potentially be built using MPQUIC [21]. Its flexibility enables a transport design that sends ACKs from a high bandwidth path subflow to a low latency path, which we proposed in §3.2. As MPQUIC is based on QUIC, it can also accept application input (e.g., steam priority) which could help packet scheduling even though it can still operate without any.

Deployment. While deploying new transport protocols such as MPTCP and solutions supporting different QoS [11, 12, 37] has proven challenging, we are hopeful about the adoption of solutions leveraging HVCs due to several factors. First, the large scale deployment of QUIC [29] makes us optimistic about transport layer solutions designed in the user space. Further, Xlink's [51] success reveals the willingness of large service providers to modify their stack to improve the performance of interactive applications. Further, considering the stringent requirements of interactive applications [6, 19, 35], cross-layer designs are likely to benefit applications using even one channel [44, 48]. We hope the popularity of such a solution will in turn incentivize operators to make HVCs ubiquitous to tap into a ready market.

5 Conclusion

In this paper, we discuss the newly emergent HVCs, various architectures that leverage them to boost application performance and their attendant trade-offs. We demonstrate the effectiveness of cross-layer designs with real time video streaming and web browsing as examples. Finally, we explore some open questions and research directions with the hope that this paper can stimulate discussions within the community about better utilizing the potential of these HVCs.

Acknowledgments

We gratefully acknowledge helpful discussions with Javier Perez-Ramirez (Intel) and Zheng Cai (T-Mobile) and support from T-Mobile, Cisco, and the IBM-Illinois Discovery Accelerator Institute.

References

[1] [n. d.]. VP9 Spatial SVC Encoder. https://chromium.googlesource.com/webm/libvpx/+/master/examples/vpx_temporal_svc_encoder.c. ([n. d.]).

[2] 2019. 3GPP TR 38.824 Release 16. https://www.3gpp.org/specifications-technologies/releases/release-16. (March 2019). [Last accessed on June 30, 2023].

[3] 3G PP. 2022. Release 16. (2022). "https://www.3gpp.org/specifications-technologies/releases/release-16" [Last accessed on 28 June 2023].

[4] 3rd Generation Partnership Project. 2017. *Study on Scenarios and Requirements for Next Generation Access Technologies.* Technical Report.

[5] IEEE 802.1. 2023. Time-Sensitive Networking (TSN) Task Group. (2023). https://1.ieee802.org/tsn/ [Last accessed on 27 June 2023].

[6] Akamai. 2017. Akamai Online Retail Performance Report: Milliseconds Are Critical. (2017). "https://www.akamai.com/newsroom/press-release/akamai-releases-spring-2017-state-of-online-retail-performance-report" [Last accessed on 12 June 2023].

[7] Fredrik Alriksson, Oskar Drugge, Anders Furuskär, Du Ho Kang, Jonas Kronander, Jose Luis Pradas, and Ying Sun. 2023. Future network requirements for extended reality applications. (2023). "https://www.ericsson.com/496150/assets/local/reports-papers/ericsson-technology-review/docs/2023/future-network-requirements-for-xr-apps.pdf" [Last accessed on 12 June 2023].

[8] Amazon. 2023. Project Kuiper. https://www.aboutamazon.com/what-we-do/devices-services/project-kuiper. (2023).

[9] Waqar Aqeel, Balakrishnan Chandrasekaran, Anja Feldmann, and Bruce M Maggs. 2020. On landing and internal web pages: The strange case of jekyll and hyde in web performance measurement. In *Proceedings of the ACM Internet Measurement Conference (IMC)*.

[10] Debopam Bhattacherjee, Waqar Aqeel, Sangeetha Abdu Jyothi, Ilker Nadi Bozkurt, William Sentosa, Muhammad Tirmazi, Anthony Aguirre, Balakrishnan Chandrasekaran, P Brighten Godfrey, Gregory Laughlin, et al. 2022. {cISP}: A {Speed-of-Light} Internet Service Provider. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. 1115–1133.

[11] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. 1998. *An architecture for differentiated services.* Technical Report.

[12] Robert Braden, David Clark, and Scott Shenker. 1994. Integrated services in the internet architecture: an overview. (1994).

[13] Lawrence S Brakmo, Sean W O'Malley, and Larry L Peterson. 1994. TCP Vegas: New techniques for congestion detection and avoidance. In *Proceedings of the conference on Communications architectures, protocols and applications*. 24–35.

[14] Neal Cardwell, Yuchung Cheng, C Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. 2017. BBR: congestion-based congestion control. *Commun. ACM* 60, 2 (2017), 58–66.

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

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

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

[18] Yongzhou Chen, Ammar Tahir, Francis Y. Yan, and Radhika Mittal. 2023. Octopus: In-Network Content Adaptation to Control Congestion on 5G Links. In *2023 IEEE/ACM Symposium on Edge Computing (SEC)*.

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

[20] Erik Dahlman, Stefan Parkvall, and Johan Skold. 2020. *5G NR: The next generation wireless access technology*. Academic Press.

[21] Quentin De Coninck and Olivier Bonaventure. 2017. Multipath quic: Design and evaluation. In *Proceedings of the 13th international conference on emerging networking experiments and technologies*. 160–166.

[22] Mo Dong, Tong Meng, Doron Zarchy, Engin Arslan, Yossi Gilad, Brighten Godfrey, and Michael Schapira. 2018. PCC vivace: Online-learning congestion control. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. 343–356.

[23] Theresa Enghardt, Philipp S Tiesel, Thomas Zinner, and Anja Feldmann. 2019. Informed Access Network Selection: The Benefits of Socket Intents for Web Performance. In *2019 15th International Conference on Network and Service Management (CNSM)*. IEEE, 1–9.

[24] Theresa Enghardt, Thomas Zinner, and Anja Feldmann. 2020. Using informed access network selection to improve HTTP adaptive streaming performance. In *Proceedings of the 11th ACM Multimedia Systems Conference*. 126–140.

[25] Juan Fang, Susruth Sudhakaran, Dave Cavalcanti, Carlos Cordeiro, and Cheng Chen. 2021. Wireless TSN with Multi-Radio Wi-Fi. In *2021 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE, 105–110.

[26] Simone Ferlin, Özgü Alay, Olivier Mehani, and Roksana Boreli. 2016. BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks. In *2016 IFIP networking conference (IFIP networking) and workshops*. IEEE, 431–439.

[27] Jeff Foust. 2021. SpaceX adds laser crosslinks to polar Starlink satellites. https://spacenews.com/spacex-adds-laser-crosslinks-to-polar-starlink-satellites/. (2021).

[28] Sangtae Ha, Injong Rhee, and Lisong Xu. 2008. CUBIC: a new TCP-friendly high-speed TCP variant. *ACM SIGOPS operating systems review* 42, 5 (2008), 64–74.

[29] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. 2017. The quic transport protocol: Design and internet-scale deployment. In *Proceedings of the conference of the ACM special interest group on data communication*. 183–196.

[30] Yeon-sup Lim, Erich M Nahum, Don Towsley, and Richard J Gibbens. 2017. ECF: An MPTCP path scheduler to manage heterogeneous paths. In *Proceedings of the 13th international conference on emerging networking experiments and technologies*. 147–159.

[31] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. 2016. MOT16: A Benchmark for Multi-Object Tracking. (2016). arXiv:cs.CV/1603.00831

[32] Arvind Narayanan, Eman Ramadan, Jason Carpenter, Qingxu Liu, Yu Liu, Feng Qian, and Zhi-Li Zhang. 2020. A First Look at Commercial 5G Performance on Smartphones. In *Proceedings of The Web Conference*.

[33] Ravi Netravali, Anirudh Sivaraman, Somak Das, Ameesh Goyal, Keith Winstein, James Mickens, and Hari Balakrishnan. 2015. Mahimahi: Accurate record-and-replay for HTTP. In *Proceedings of the USENIX Annual Technical Conference (ATC)*.

[34] Jan Odvarko. 2007. HAR 1.2 Spec. (2007). Retrieved June 30, 2023 from http://www.softwareishard.com/blog/har-12-spec/

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

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

[37] Maxim Podlesny and Sergey Gorinsky. 2008. RD network services: differentiation through performance incentives. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*. 255–266.

[38] Eric Ralph. 2020. SpaceX Starlink 'space lasers' successfully tested in orbit for the first time. https://www.teslarati.com/spacex-starlink-space-lasers-first-orbital-test/. (2020).

[39] Jerome H Saltzer, David P Reed, and David D Clark. 1984. End-to-end arguments in system design. *ACM Transactions on Computer Systems (TOCS)* 2, 4 (1984), 277–288.

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

[41] Heiko Schwarz, Detlev Marpe, and Thomas Wiegand. 2007. Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. *IEEE Transactions on Circuits and Systems for Video Technology* 17, 9 (2007), 1103–1120. https://doi.org/10.1109/TCSVT.2007.905532

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

[43] M Zubair Shafiq, Lusheng Ji, Alex X Liu, Jeffrey Pang, and Jia Wang. 2012. Characterizing geospatial dynamics of application usage in a 3G cellular data network. In *2012 Proceedings IEEE INFOCOM*. IEEE, 1341–1349.

[44] Sanjay Shakkottai, Theodore S Rappaport, and Peter C Karlsson. 2003. Cross-layer design for wireless networks. *IEEE Communications magazine* 41, 10 (2003), 74–80.

[45] SpaceX. 2023. Starlink. https://www.starlink.com/. (2023).

[46] Telesat. 2023. Telesat: Global Satellite Operators. https://www.telesat.com/. (2023).

[47] Damon Wischik, Costin Raiciu, Adam Greenhalgh, and Mark Handley. 2011. Design, Implementation and Evaluation of Congestion Control for Multipath TCP.. In *NSDI*, Vol. 11. 8–8.

[48] Xiufeng Xie, Xinyu Zhang, and Shilin Zhu. 2017. Accelerating mobile web loading using cellular link information. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. 427–439.

[49] Francis Y Yan, Jestin Ma, Greg D Hill, Deepti Raghavan, Riad S Wahby, Philip Levis, and Keith Winstein. 2018. Pantheon: the training ground for Internet congestion-control research. In *2018 USENIX Annual Technical Conference (USENIX ATC18)*. 731–743.

[50] Xin Zhang, Hsu-Chun Hsiao, Geoffrey Hasker, Haowen Chan, Adrian Perrig, and David G. Andersen. 2011. SCION: Scalability, Control, and Isolation on Next-Generation Networks. In *2011 IEEE Symposium on Security and Privacy*.

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

[52] Torsten Zimmermann, Benedikt Wolters, Oliver Hohlfeld, and Klaus Wehrle. 2018. Is the Web ready for HTTP/2 server push?. In *Proceedings of the 14th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*.