Path Schedulers Performance on Cellular/Wi-Fi Multipath Video Streaming

Masayoshi Kondo*, Dirceu Cavendish**, Daiki Nobayashi**, Takeshi Ikenaga**
*Graduate School of Engineering, **Faculty of Engineering
Kyushu Institute of Technology
Fukuoka, Japan
e-mail: kondo.masayoshi146@mail.kyutech.jp {cavendish@ndrc, nova@ecs, ike@ecs}.kyutech.ac.jp

***Abstract*—Internet traffic nowadays is predominantly composed by video streaming. Moreover, most video streaming traffic is carried over Hypertext Transfer Protocol/Transmission Control Protocol (HTTP/TCP). Understanding TCP stack performance in transporting video streams has become paramount, specially in face of recent multipath transport protocol evolutions and multiple client device interfaces available. In this paper, we characterize path schedulers performance of streaming of video sessions over cellular and Wi-Fi access networks, the two most common and dominant wireless technologies in the market. We use network performance level as well as video quality level metrics to characterize multiple path schedulers and resulting network and application layers' interactions.**

Keywords*—*Video streaming; TCP congestion control; TCP socket state; Multipath TCP; Packet retransmissions; Packet loss.

I. INTRODUCTION

Internet data transmission relies heavily on transport protocols to pace data delivery to avoid network congestion and uncontrolled data losses. Transmission Control Protocol (TCP) has arguably become the most successful transport protocol of the Internet, supporting reliable data delivery for most diverse applications nowadays. In particular, streaming applications, the most dominant type of application in sheer volume, data transport quality is related with the amount of data discarded at the client due to excessive transport delay/jitter, as well as data rendering stalls due to lack of timely playout data. Transport delays and data starvation performance measurers depend heavily on how TCP handles flow and congestion control, as well as packet retransmissions.

More recently, the evolution of portable device hardware, in particular support of multiple high bandwidth interfaces, has prompted the development of multipath transport protocols, allowing, for instance, video streaming over multiple IP interfaces and diverse network paths. Multipath video streaming is advantageous because it not only increases aggregated device downloading bandwidth capacity, but also improves transport session reliability during transient radio link impairments. An important issue in multipath transport is selecting a path among various active networking paths (called sub-flows), which can be done on a packet by packet basis. A path packet scheduler is used for this purpose, and should be designed to prevent head-of-line blocking across various networking paths, potentially with diverse loss and delay characteristics. Head-of-line blocking occurs when data already delivered at the receiver is waiting for additional packets that are blocked at another sub-flow, potentially causing incomplete or late frames to be discarded at the receiver, as well as stream stalling. Interplay between path schedulers and TCP variants will untimately define streaming quality, hence we propose to analyze video performance vis-a-vis popular TCP variants and path schedulers.

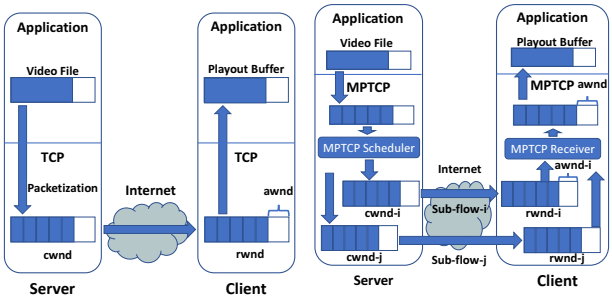
The paper is organized as follows. Related work is included in Section II. Section III describes video streaming transport over Transmission Control Protocol. Section IV describes recently proposed alternative path schedulers. Section V characterizes video streaming performance over Wi-Fi and cellular paths via network emulation, addressing performance evaluation of a default path scheduler, as well as alternative schedulers, working with popular TCP variants. Section VI summarizes our studies and addresses directions we are pursuing as follow up to this work.

II. RELATED WORK

Since the advancements of multipath transport protocols, several multipath transport studies have appeared in the literature, mostly focusing on throughput performance of data transfers over mobile networks (see [18] aggregate throughput studies and related work). Only recently, however, path scheduler research has been recognized as an important piece of multipath transport sessions performance. For instance, [13] has analyzed loss based congestion control TCP variants interactions with minimum Round Trip Time (RTT) default Multipath TCP (MPTCP) path scheduler, and showed how sender/receiver buffers dimentioning impacts throughput performance via inflation of sub-flow RTTs. Little research work, however, has focused on video performance over multiple paths. Recent multipath video streaming on ad-hoc network studies have appeared, motivated by vehicular communication in assisted driving systems. [2], for instance, introduces an interference aware multipath video streaming scheme in Vehicular Ad-hoc Networks (VANETs). Vehicle to vehicle throughput performance of video streams over multiple paths is evaluated, taking into account interference within neighbors, as well as shadowing effects onto Signal to Noise ratio, and data delay. Their goal is reliable transport of high quality video streams, minimizing video freezes and dropped frames, via link layer channel interference control, coupled with efficient routing strategies on ad-hoc vehicular networks. In contrast, we focus on video streaming over regular Internet paths,

 ISBN: 978-1-61208-880-8

where link layer channel and route optimization opportunities are limited. [1] focuses at integrating application layer with transport protocol, by introducing a path-and-content-aware path selection approach which couples MPEG Media Transport (MMT) with multipath transport. They estimate path quality conditions of each subflow, and avoid sending I-frames on paths of low transport quality. A similar approach, where different sub-flows are utilized for segregating high priority packets of Augmented Reality/Virtual Reality streams has been introduced by Silva et al. [23]. In contrast, our previous and current work do not couple applications with multipath transport, rather focusing on generic path schedulers and TCP variants to deliver high quality video streaming. Recently, Ferlin et al. [7] have introduced a path scheduler based on a path head-of-line blocking predictor. They carry out emulation experiments of their proposed scheduler against minimum RTT default scheduler, in transporting bulk data traffic, Web transactions and Constant Bit Rate (CBR) applications. Hence, they use goodput, data transport completion time and packet delays as performance evaluation metrics. Kimura et al. [12] have shown throughput performance improvements using schedulers driven by path sending rate and TCP window space, on bulk data transfers. Xue et al. [26] has introduced a path scheduler based on estimation of the amount of data each path is able to transmit. They evaluated the scheduler's throughput performance on simulated network scenarios. Also, Frommgen et al. [9], consistent with [13] work, have shown that stale RTT information causes problems with path selection of small streams, such as HTTP traffic. Similar to [13] tail burst probing, the authors propose an RTT probing to improve transport latency and throughput performance of thin streams. In contrast, the schedulers evaluated in our work do not rely on path probing mechanisms, but on sender based passive path evaluation metrics to steer video traffic appropriately. Along the same lines, Dong et al. [6] have introduced a path loss estimation scheme to select paths that may be subjected to high and bulk loss rates. Although they have presented video streaming experiments, they do not measure application level stream performance, as we do. By contrast, in our previous works, we have introduced multipath path scheduling generic principles, which can be applied in the design of various path schedulers to specifically improve video stream quality. Using these principles, we have introduced in [14] Multipath TCP path schedulers based on dynamically varying path characteristics, such as congestion window space and estimated path throughput. In addition, in [15], we have also proposed to enhance path schedulers with TCP state information, such as whether a path is in fast retransmit and fast recovery states. Finally, in [16], we have introduced a novel concept of sticky scheduling, where once a path switch is executed, the scheduler stays with the new path until the path bandwidth resources become exhausted. In this work, for the first time we propose to evaluate a multitude of path schedulers, some of our proposal, over realistic Wi-Fi/Cellular multipath scenarios, focusing on video quality at application layer. Our evaluation include widely deployed TCP variants, exposing best TCP/path scheduler combinations.



(a) TCP (b) MPTCP

Figure 1: Video Streaming over TCP/MPTCP

III. Video Streaming Over TCP

Video streaming over HTTP/TCP typically sources video content from an HTTP server, where video files can be streamed from upon HTTP requests over the Internet to video clients. At the transport layer, a TCP variant provides reliable transport of video data over IP packets between the server and client end points. Figure 1(a) illustrates these video streaming components. As mentioned, HTTP server stores encoded video files. Triggered by a HTTP video request, a TCP sender is instantiated to transmit packetized data to the client machine, connected to the application by a TCP socket. At this TCP transport layer, a congestion window is used at the sender to control the amount of data injected into the network. The size of the congestion window ($cwnd$) is adjusted dynamically, according to the level of congestion experienced through the network path, as well as the space available for data storage ($awnd$) at the TCP client receiver buffer. Congestion window space at the sender is freed only when data packets acknowledged by the receiver arrive. Lost unacknowledged packets are retransmitted by the TCP layer to ensure reliable data delivery. At the client, in addition to acknowledging arriving packets, the TCP receiver informs the TCP sender about its current available space ($awnd$), so that $cwnd \leq awnd$ condition is enforced by the sender at all times. At the client application layer, a video player extracts data from a playout buffer, which drains packets delivered by the TCP receiver from its socket buffer. The playout buffer hence serves to smooth out variable data network throughput.

A. Video Application and TCP Transport Interaction

As mentioned earlier, at the server side, the HTTP server transfers data into the TCP sender socket according to TCP $cwnd$ space availability. Hence, the injection rate of video data into the TCP socket is constrained by the congestion condition of the network path used, and thus does not follow the video variable encoding rate. On its turn, TCP throughput performance is affected by the RTT of the TCP session over a specific path, since only up to a $cwnd$ worth of data can be delivered without acknowledgements. Hence, for a given $cwnd$ size, from the moment a first packet is sent until the first acknowledgement arrives back, the TCP session throughput is capped at $cwnd/RTT$. As there are various TCP

congestion avoidance schemes regulating $cwnd$, according to the TCP variant, the size of the congestion window size is computed by a specific algorithm at the time of packet acknowledgement reception by the TCP source. Regardless of the variant, however, the size of the congestion window computed is capped by the available TCP receiver space $awnd$ communicated back from the TCP client, in order to ensure no receiver buffer overflow. At the client side, video data is retrieved from the TCP client socket by the video player into a playout buffer, from which data is delivered to the video renderer. Even though client playout buffer may underflow, if TCP receiver window empties out, the playout buffer never overflows, since the player will not pull more data into the playout buffer if space is not available during video rendering.

B. Multipath TCP

Multipath TCP is an Internet Engineering Task Force (IETF) transport layer protocol that supports data transport over multiple concurrent TCP sessions [8]. The multipath nature of the transport session is hidden from application layer by a single TCP socket exposed per application session. At the transport layer, however, MPTCP works with concurrent TCP variant sub-flows, each of which in itself unaware of the multipath nature of the application session. A path scheduler connects the application facing socket with transport sub-flows, extracting packets from the MPTCP socket exposed to the application, selecting a sub-flow, and injecting packets into a selected sub-flow TCP socket. MPTCP transport architecture is illustrated in Figure 1(b).

The most commonly used path scheduler, called default scheduler, selects the path with shortest RTT among paths with currently available congestion window space for new packets. Path schedulers may operate in two different modes: uncoupled, and coupled. In uncoupled mode, each sub-flow congestion window $cwnd$ is adjusted independent of the other sub-flow. In coupled mode, MPTCP couples the congestion control of the sub-flows, by adjusting the congestion window $cwnd_k$ of a sub-flow k according with current state and parameters of all sub-flows. Although several coupling mechanisms exist, we focus on Linked Increase Algorithm (LIA) [20], Opportunistic Linked Increase Algorithm (OLIA) [11], and Balanced Linked Adaptation algorithm (BALIA) [25]. We include also evaluation of various alternative uncoupled schedulers recently proposed.

IETF MPTCP protocol supports the advertisement of IP interfaces available between two endpoints via specific TCP option signalling. As IP option signalling may be blocked by intermediate IP boxes, such as firewalls, paths that cross service providers may require VPN protection so as to preserve IP interface advertizing between endpoints. Morever, both endpoints require MPTCP to be running for the establishment and usage of multiple transport paths. Notice that IP interfaces may be of diverse nature (e.g., Wi-Fi, cellular).

C. TCP variants

TCP protocol variants can be classified into delay and loss based congestion control schemes. Loss based TCP variants use packet loss as primary congestion indication signal, typically performing window regulation as $cwnd_k = f(cwnd_{k-1})$, hence being ack reception paced. Most f functions follow an Additive Increase Multiplicative Decrease (AIMD) window adjustment scheme, with various increase and decrease parameters. TCP NewReno [3] and Cubic [21] are examples of AIMD strategies. On the other hand, delay based TCP variants use queue delay information as the congestion indication signal, increasing/decreasing the window if the delay is small/large, respectively. Compound [22] and Capacity and Congestion Probing (CCP) [5] are examples of delay based congestion control variants. Most TCP variants follow a phase framework, with an initial slow start, congestion avoidance, fast retransmit, and fast recovery phases, regardless of the window adjustment congestion control used during congestion avoidance.

IV. MPTCP Path Schedulers

MPTCP scheduler selects a sub-flow to inject packets into the network on a packet by packet basis. As mentioned earlier, the default strategy is to select the path with shortest average packet delay, hereafter called LRF. Other path schedulers evaluated in this paper are as follows:

- **Low RTT First (LRF):** In low RTT first, the scheduler first rules out any path for which there is no space in its sub-flow congestion window ($cwnd$). Among the surviving paths, the scheduler then selects the path with small smooth RTT ($sRTT$). Smooth RTT is computed as an average RTT of recently transmitted packets on that sub-flow. Since on most TCP stacks each sub-flow already keeps track of its smooth RTT, this quantity is readily available.
- **Largest Packet Credits (LPC):** In largest packet credits scheduler, among the sub-flows with space in their congestion window $cwnd$, this scheduler selects the one with largest available space. Here available space consists of the number of packets allowed by current $cwnd$ size subtracted from the number of packets that have not been acknowledged yet.
- **Largest Estimated Throughput (LET):** In largest estimated throughput scheduler, among the sub-flows with large enough $cwnd$ to accommodate new packets, the scheduler estimates the throughput of each sub-flow, as $cwnd/sRTT$, selecting the one with largest throughput.
- **Greedy Sticky (GR-STY):** As it is the case of default scheduler (LRF), on the onset of a video streaming session, greedy sticky scheduler selects the path with smallest RTT. However, once a new path is selected, the scheduler stays on a new path for as long as there is available congestion window space, until the new path experiences congestion.
- **Throughput Sticky (TP-STY):** Similar to default scheduler (LRF), throughput sticky scheduler selects the path of lowest RTT. However, a new path is selected only if the throughput of the new path is larger than the throughput of the currently selected path.

 ISBN: 978-1-61208-880-8

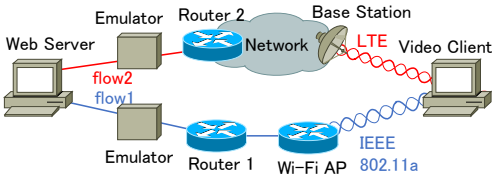


Figure 2: Video Streaming Emulation Network

TABLE I: EXPERIMENTAL NETWORK SETTINGS

Element	Value
Video size	409 MBytes
Video rate	5.24 Mb/s
Playout time	10 mins 24 secs
Video Codec	H.264 MPEG-4 AVC
MPTCP variants	Cubic, Compound, LIA, OLIA, BALIA
MPTCP schedulers	LRF, LET, LPC, GR-STY, TP-STY, TR-STY

TABLE II: EXPERIMENTAL NETWORK SCENARIOS

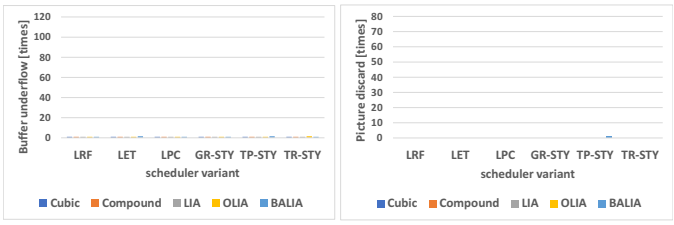
Scenario	Emulator	Path properties (RTT)
A- Baseline (LTE/Wi-Fi) no loss	LTE) Delay 0 ms Wi-Fi) Delay 20 ms	RTT 80 ms RTT 40 ms
B- TwoPath (LTE/Wi-Fi) no loss	LTE) Delay 0 ms Wi-Fi) Delay 30 ms	RTT 80 ms RTT 60 ms
C- TwoPath (LTE/Wi-Fi) Wi-Fi 6% loss	LTE) Delay 0 ms Wi-Fi) Delay 30 ms, Loss 6%	RTT 80 ms RTT 60 ms
D- TwoPath (LTE/Wi-Fi) no loss	LTE) Delay 0 ms Wi-Fi) Delay 40 ms	RTT 80 ms RTT 80 ms
E- TwoPath (LTE/Wi-Fi) Wi-Fi 6% loss	LTE) Delay 0 ms Wi-Fi) Delay 40 ms, Loss 6%	RTT 80 ms RTT 80 ms

- **Throughput RTT Sticky (TR-STY):** As with default scheduler (LRF), the path of lowest RTT is first chosen. However, in addition to requiring a larger throughput of a new candidate path as per TP-STY, in throughput RTT sticky scheduler, path switch requires also that a new path has smaller RTT than the current one.

LPC focuses on scenarios in which addresses a large RTT path has plenty of bandwidth, as compared to a shorter RTT path. LRF default scheduler may avoid this path due to its large RTT, regardless of having plenty of bandwidth for the video stream to flow unimpeded. LET addresses another scenario, in which a short path has plenty of bandwidth. Although the default scheduler may select this path due to its short RTT, if the short RTT has a smaller cwnd, LET will divert traffic away from this path prior to path congestion, whereas default scheduler will continue to inject traffic through it. Finally, the last three sticky schedulers attempt to reduce the number of path switches during transport session, in an attempt to reduce head of line blocking probability during the streaming session.

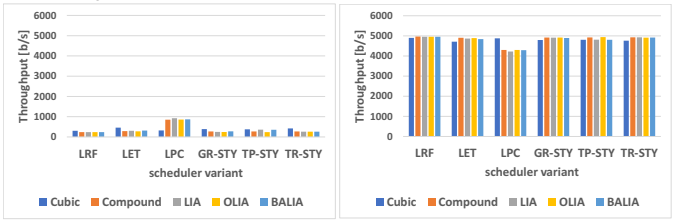
V. Path Schedulers over Wi-Fi & Cellular Paths

Figure 2 describes the network testbed used for emulating network paths with Wi-Fi and Cellular (LTE) wireless access links. An HTTP Apache video server is connected to two L3 switches, one of which directly connected to an 802.11a router, and the other connected to an LTE base station via a cellular network card via emulator boxes. In this paper, the emulator boxes are used to vary each path RTT, as well as inject controlled packet losses. The simple topology and isolated traffic allow us to better understand the impact of differential



(a) Buffer Underflow (b) Picture Discard

Figure 3: A- Baseline Scenario - Video Performance



(a) Throughput Cellular (b) Throughput Wi-Fi

Figure 4: A- Baseline Scenario - Throughput Performance

delays, packet loss, TCP variants, and path schedulers on streaming performance.

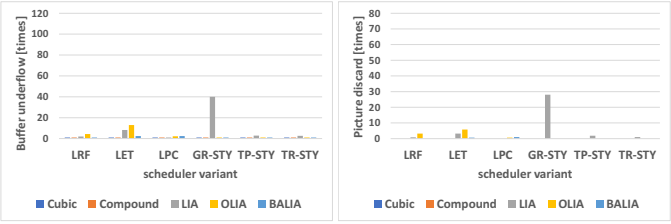
Network settings and scenarios under study are described in Tables I and II, respectively. Video settings are typical of a video stream, with video playout rate of 5.24 Mb/s, and size short enough to run multiple streaming trials within a short amount of time. Emulator boxes are tuned to generate various multiple path network conditions, and have been selected as per Table II to represent commonplace LTE/WiFi streaming situations at home. TCP variants used are: Cubic, Compound, LIA, OLIA and BALIA. Performance measures are:

- **Picture discards:** number of frames discarded by the video decoder.
- **Buffer underflow:** number of buffer underflow events at video client buffer.
- **Sub-flow throughput:** the value of TCP throughput on each sub-flow.

We organize our video streaming experimental results in network scenarios summarized in Table II): A- A Wi-Fi-Cellular (LTE) baseline scenario, where Wi-Fi path of good quality (RTT/loss) is predominantly used; B- A Wi-Fi-Cellular scenario, where a slightly larger Wi-Fi path delay causes cellular path to be used; C- A Wi-Fi-Cellular scenario, where a Wi-Fi link with medium delay suffers a 6 % packet loss degradation, representing user situation at which device is at the end of Wi-Fi range; D- A Wi-Fi-Cellular scenario, with a Wi-Fi path delay large enough to have cellular path predominantly being used; E- A Wi-Fi-Cellular scenario, where a Wi-Fi link with large delay also suffers a 6 % packet loss degradation.

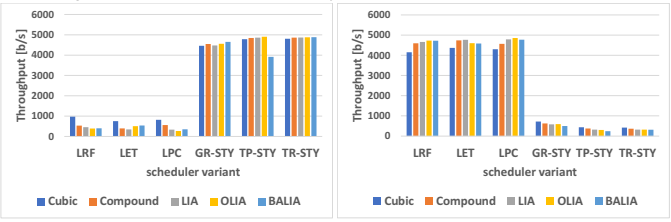
A. Baseline Scenario

Figures 3(a) and (b) report on video streaming buffer underflow and picture discard performance. Video performance is excellent for all TCP variants and path schedulers. Figures 4(a) and (b) report of Cellular and Wi-Fi throughput. We can see that Wi-Fi path is most used for all TCP variants and path schedulers.



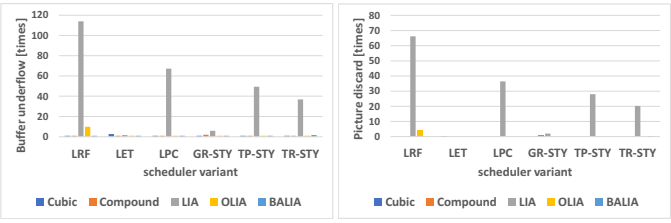
(a) Buffer Underflow (b) Picture Discard

Figure 5: B- Medium Delay Wi-Fi - Video Performance



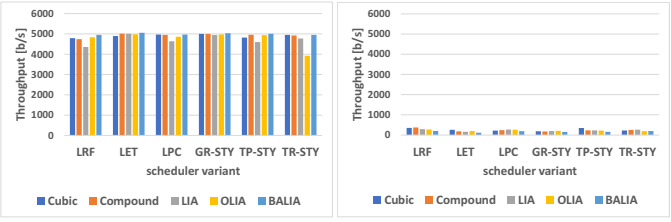
(a) Throughput Cellular (b) Throughput Wi-Fi

Figure 6: B- Medium Delay Wi-Fi - Throughput Performance



(a) Buffer Underflow (b) Picture Discard

Figure 7: C- Medium Delay&loss Wi-Fi - Video Performance



(a) Throughput Cellular (b) Throughput Wi-Fi

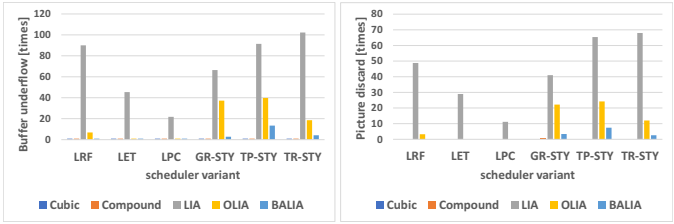
Figure 8: C- Medium Delay&loss Wi-Fi - Throughput Performance

B. Two path Medium Delay Wi-Fi Scenario

Figures 5(a) and (b) report on video streaming performance of Wi-Fi - Cellular network scenario with a medium Wi-Fi path delay. Even though most TCP variants and path schedulers perform well, LIA TCP variant under GR-STY scheduler results in video degradation, albeit not serious. Throughput performance in Figures 6 shows an that path schedulers drive the usage of one path versus the other, independent of the TCP variant. In particular, LRF (default), LET, LPC utilize Wi-Fi path mostly, whereas all sticky schedulers (GR-STY, TP-STY, TR-STY) use mostly the cellular path. This shows how sensitive path selection is to delay differentials.

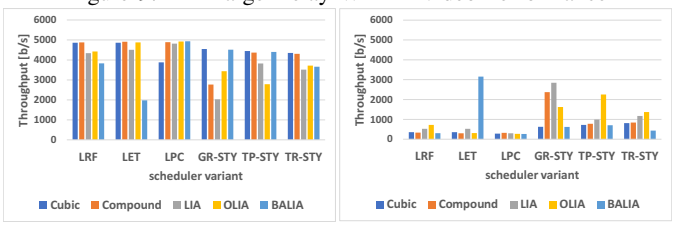
C. Two path Medium Delay&Loss Wi-Fi Scenario

Figures 7(a) and (b) report on video streaming performance of Wi-Fi - Cellular network scenario with a medium Wi-Fi path delay and 6 % packet loss. We notice a wide variety of performances vis a vis path scheduler/TCP variant combinations, which is expected because of loss impact on TCP



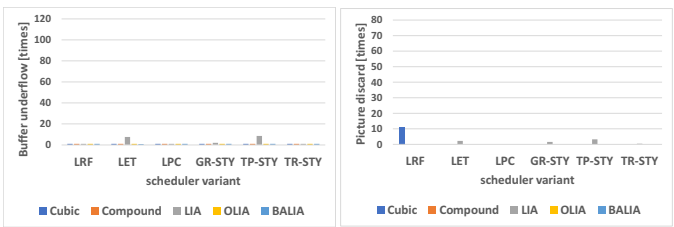
(a) Buffer Underflow (b) Picture Discard

Figure 9: D- Large Delay Wi-Fi - Video Performance



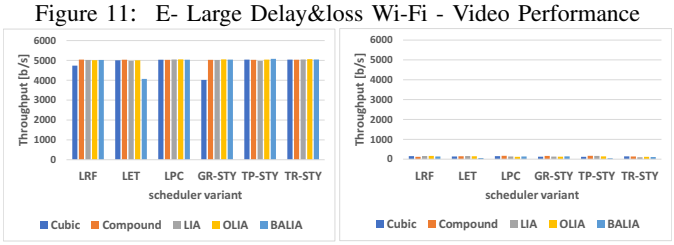
(a) Throughput Cellular (b) Throughput Wi-Fi

Figure 10: D- Large Delay Wi-Fi - Throughput Performance



(a) Buffer Underflow (b) Picture Discard

Figure 11: E- Large Delay&loss Wi-Fi - Video Performance



(a) Throughput Cellular (b) Throughput Wi-Fi

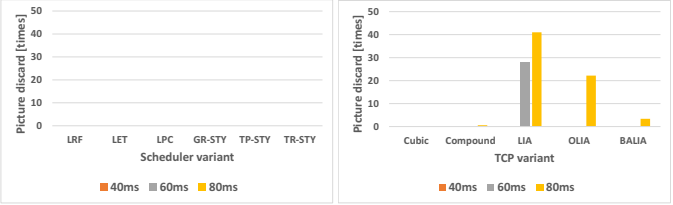
Figure 12: E- Large Delay&loss Wi-Fi - Throughput Performance

variants' performance and delay differential impact on path schedulers. We can identify, however, some trends. Firstly, GR-STY path scheduler delivers high video quality across all TCP variants. Noticeable also is Cubic consistent performance across all path schedulers. Finally, if taken together as a class, coupled TCP variants seem to incur video degradation across all schedulers.

Throughput performance in Figures 8 shows that a Wi-Fi packet delivery degradation pushes path utilization to mostly cellular path, regardless of the TCP variant. Good path schedulers design are expected to select high quality paths in both delay and loss path attributes.

D. Two path Large Delay Wi-Fi Scenario

Figures 9(a) and (b) report on video streaming performance of Wi-Fi - Cellular network scenario with a large Wi-Fi path delay. Even though most TCP variants and path schedulers perform well, LIA and OLIA TCP variants under all sticky



(a) TCP Cubic - Picture Discard (b) GR-STY - Picture Discard

Figure 13: TCP & Scheduler-Picture Discard Video Performance

schedulers results in video degradation. Throughput performance in Figures 10 shows that cellular link is mostly used, although some video traffic still goes through Wi-Fi path. We can see that sticky schedulers are responsible for streaming over the Wi-Fi path, despite its large RTT delay.

E. Two path Large Delay&Loss Wi-Fi Scenario

Figures 11(a) and (b) report on video streaming performance of Wi-Fi - Cellular network scenario with a large Wi-Fi path delay and 6 % packet loss. We again notice a wide variety of performances vis a vis path scheduler/TCP variant combinations. Firstly, all scheduler deliver similar video performance across all TCP variants. We believe this is because once the scheduler selects the cellular path, it continues to re-selecting it unless the path degrades, which does not occur in this scenario. Also, default scheduler operating with Cubic variant presents video degradation on picture discards. Throughput performance in Figures 12 shows that Cellular path is used predominantly during video streaming, due to Wi-Fi large delay and packet loss.

Finally, Figures 13(a) and (b) report on TCP Cubic variant and GR-STY scheduler impact on picture discard performance across multiple no loss Wi-Fi delay scenarios, respectively. TCP Cubic consistently deliver high performance, whereas GR-STY delivers high performance across all TCP variants except coupled LIA, OLIA and BALIA. A Cubic and GR-STY combo is our recommended variant/scheduler choice.

VI. Conclusion and Future work

We have provided an extensive analysis of path schedulers and TCP variants impact on video streaming performance over multiple paths. We have shown that some combinations of path schedulers with TCP variants negatively impact video streaming performance under certain network scenarios. In particular, we have shown video performance degradation for popular LIA and OLIA TCP variants, for which their congestion adjustment coupling slows down their recovery from packet losses. We are currently investigating if similar performance issues appear in 5G cellular links.

Acknowledgments

Work supported in part by JSPS KAKENHI Grant #20K11792, and National Institute of Information and Communication Technology (NICT).

References

[1] S. Afzal et al., “A Novel Scheduling Strategy for MMT-based Multipath Video Streaming,” In Proc. of IEEE Global Communications Conference - GLOBECOM, pp. 206-212, 2018.

[2] A. Aliyu et al., “Interference-Aware Multipath Video Streaming in Vehicular Environments,” In IEEE Access Special Section on Towards Service-Centric Internet of Things (IoT): From Modeling to Practice, Volume 6, pp. 47610-47626, 2018.

[3] M. Allman, V. Paxson, and W. Stevens, “TCP Congestion Control,” IETF RFC 2581, April 1999.

[4] Arzani et al., “Deconstructing MPTCP Performance,” In Proceedings of IEEE 22nd ICNP, pp. 269-274, 2014.

[5] D. Cavendish, K. Kumazoe, M. Tsuru, Y. Oie, and M. Gerla, “Capacity and Congestion Probing: TCP Congestion Avoidance via Path Capacity and Storage Estimation,” IEEE Second International Conference on Evolving Internet, pp. 42-48, September 2010.

[6] E. Dong et al., “LAMPS: A Loss Aware Scheduler for Multipath TCP over Highly Lossy Networks,” *Proc. of the 42th IEEE Conference on Local Computer Networks*, pp. 1-9, October 2017.

[7] S. Ferlin et al., “BLEST: Blocking Estimation-based MPTCP Scheduler for Heterogeneous Networks,” In Proc. of IFIP Networking Conference, pp. 431-439, 2016.

[8] A. Ford et al., “Architectural Guidelines for Multipath TCP Development,” IETF RFC 6182, 2011.

[9] A. Frommgen, J. Heuschkel and B. Koldehofe, “Multipath TCP Scheduling for Thin Streams: Active Probing and One-way Delay-awareness,” IEEE Int. Conference on Communications (ICC), pp.1-7, May 2018.

[10] J. Hwang and J. Yoo, “Packet Scheduling for Multipath TCP,” IEEE 7th Int. Conference on Ubiquitous and Future Networks, pp.177-179, July 2015.

[11] R. Khalili, N. Gast, and J-Y Le Boudec, “MPTCP Is Not Pareto-Optimal: Performance Issues and a Possible Solution,” IEEE/ACM Trans. on Networking, Vol. 21, No. 5, pp. 1651-1665, Aug. 2013.

[12] Kimura et al., “Alternative Scheduling Decisions for Multipath TCP,” IEEE Communications Letters, Vol. 21, No. 11, pp. 2412-2415, Nov. 2017.

[13] R. Lubben and J. Morgenroth, “An Old Couple: Loss-Based Congestion Control and Minimum RTT Scheduling in MPTCP,” IEEE 44th Conference on Local Computer Networks, pp.300-307, October 2019.

[14] Matsufuji et al., “Multipath TCP Packet Schedulers for Streaming Video,” IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM) , August 2017, pp. 1-6.

[15] Nagayama et al., “TCP State Driven MPTCP Packet Scheduling for Streaming Video,” IARIA 10th International Conference on Evolving Internet, pp. 9-14, June 2018.

[16] Nagayama et al., “Path Switching Schedulers for MPTCP Streaming Video,” IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM) , August 2019, pp. 1-6.

[17] R. K. P. Mok et al., “Measuring the Quality of Experience of HTTP Video Streaming,” Proc. of IEEE International Symposium on Integrated Network Management, pp. 485-492, May 2011.

[18] M. R. Palash et al., “MPWiFi: Synergizing MPTCP Based Simultaneous Multipath Access and WiFi Network Performance,” IEEE Transactions on Mobile Computing, Vol. 19, No. 1, pp. 142-158, Jan. 2020.

[19] Z. Lu, V. S. Somayazulu, and H. Moustafa, “Context Adaptive Cross-Layer TCP Optimization for Internet Video Streaming,” In Proc. of IEEE ICC 14, pp. 1723-1728, 2014.

[20] C. Raiciu, M. Handly, and D. Wischik, “Coupled Congestion Control for Multipath Transport Protocols,” IETF RFC 6356, 2011.

[21] I. Rhee, L. Xu, and S. Ha, “CUBIC for Fast Long-Distance Networks,” Internet Draft, draft-rhee-tcpm-ctcp-02, August 2008.

[22] M. Sridharan, K. Tan, D. Bansal, and D. Thaler, “Compound TCP: A New Congestion Control for High-Speed and Long Distance Networks,” Internet Draft, draft-sridharan-tcpm-ctcp-02, November 2008.

[23] F. Silva, D. Bogusevschi, and G-M. Muntean, “A MPTCP-based RTT-aware Packet Delivery Prioritization Algorithm in AR/VR Scenarios,” In Proc. of IEEE Intern. Wireless Communications & Mobile Computing Conference - IWCMCC 18, pp. 95-100, June 2018.

[24] H. Sinky et al., “Proactive Multipath TCP for Seamless Handoff in Heterogeneous Wireless Access Networks,” IEEE Transactions on Wireless Communications, Vol. 15, Iss. 7, pp. 4754-4764, 2016.

[25] A. Walid et al., “Balanced Linked Adaptation Congestion Control Algorithm for MPTCP,” IETF draft draft-walid-mptcp-congestion-control, 2014.

[26] Xue et al., “DPSAF: Forward Prediction Based Dynamic Packet Scheduling and Adjusting With Feedback for Multipath TCP in Lossy Heterogeneous Networks,” IEEE/ACM Trans. on Vehicular Technology, Vol. 67, No. 2, pp. 1521-1534, Feb. 2018.

 ISBN: 978-1-61208-880-8