Citation: Amend, M., Rakocevic, V. & Habermann, J. (2021). Cost optimized multipath scheduling in 5G for Video-on-Demand traffic. Paper presented at the 2021 IEEE Conference on Wireless Communications and Networking, 29 Mar - 1 Apr 2021, Nanjing, China. doi: 10.1109/WCNC49053.2021.9417415

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/27857/

Link to published version: https://doi.org/10.1109/WCNC49053.2021.9417415

Cost optimized multipath scheduling in 5G for Video-on-Demand traffic

Markus Amend
Deutsche Telekom
Darmstadt, Germany
markus.amend@telekom.de

Veselin Rakocevic
City, University of London
London, United Kingdom
veselin.rakocevic.1@city.ac.uk

Joachim Habermann
Technische Hochschule Mittelhessen
Friedberg, Germany
joachim.habermann@iem.thm.de

***Abstract*—This paper evaluates the limitations of existing scheduling algorithms when video-on-demand traffic is transported in multipath scenarios, and proposes a new scheduling algorithm called cost-optimized multipath (COM). The new algorithm is designed to decrease the mobile network operators' cost of the delivery of bursty video-on-demand traffic over multipath networks access. Local and Internet connected testbeds, as well as trials with real cellular customers have been deployed to analyse the video performance over MPTCP-based multipath. The results clearly demonstrate the impact the bursty nature of video-on-demand traffic has on the scheduling decisions in multipath scenarios, when traditional latency-based or cheapest-path-first schedulers are deployed. Based on the testbed and trial results, this paper presents the design of a new simple and scalable scheduling algorithm. The paper describes the typical use cases and shows preliminary testbed results, clearly demonstrating the cost benefits of the new algorithm, and indicating that the right balance between the user QoE and the operator cost can be achieved for the video traffic.**

***Index Terms*—Multipath, Scheduler, Video on Demand, cost optimization, MPTCP, Hybrid Access, 5G ATSSS**

I. INTRODUCTION

The tremendous monetary cost of creating and operating cellular networks drive the mobile network operators (MNOs) to search for more efficient ways for data transportation, paving the way for new developments in multi-connectivity networks. Within this concept, a number of multipath traffic delivery frameworks and protocols have been introduced, promising increased capacity and reliability by leveraging multiple, usually independent, path resources simultaneously. Multipath transportation also provides enhanced traffic engineering capabilities, and can significantly contribute in terms of performance metrics other than cost, primarily in terms of reliability and latency targets for many different services and applications.

Different standardization organizations have adopted multipath solutions at different OSI layers. For example, the Layer 2 solutions include [1] and [2], and the Layer 3 solutions include [3] and [4]. The steadily evolving multipath protocols at Layer 4 or even Layer 5 using congestion control for efficient multipath decisions are typically specified at IETF such as [5], others are for example MP-QUIC, CMT-SCTP and MP-DCCP.

However, for several reasons, the big breakthrough in the deployment of multipath network protocols is still outstanding, even if the working implementations of the cited protocols exist, such as GigaLTE, Apple's MPTCP implementation in iOS and Deutsche Telekom's Hybrid Access.

In the ongoing 5G evolution a game changer in respect to multipath deployment might be expected very soon. A new functionality called ATSSS (Access Traffic Steering Switching Splitting) found its way into the specification [6]. In a nutshell, ATSSS defines the combined usage of cellular connectivity and Wi-Fi between a mobile handset and the 5G mobile operator, whereas the latter builds the interface towards the Internet and converts between multipath transport and single path transport. MPTCP is the protocol of choice for traffic splitting in the first stage, probably complemented by MP-QUIC in future 3GPP releases.

It is becoming apparent from the involved parties in the standardization, that the ATSSS has broad support from mobile operators and mobile handset vendors, providing an important push for the standardisation and deployment of multipath network protocols.

The most obvious element of multipath frameworks, allowing traffic engineering, is the multipath traffic scheduler. The scheduler is responsible for the distribution of the traffic over the multiple paths. The design of the scheduling logic will typically be based on minimising the cost of the traffic delivery, where the cost can be defined either as monetary cost (e.g. the delivery over WiFi will always be cheaper than cellular access), or in terms of latency, throughput or reliability.

A strict cost-based scheduling mechanism is known from Deutsche Telekom's Hybrid Access (HA), based on GRE [3]. Scheduler based on minimising the monetary cost metric prefers whenever possible DSL over the costly combined cellular 4G path. While this sounds reasonable, Fig. 1 - gained from live monitoring in an ISP network - demonstrates a disproportional use of costly cellular resources for Video-on-Demand (VoD) traffic even if the averaged traffic over time is below the one offered by DSL. The bursty nature of the video on demand traffic, together with the fact that the video-on-demand traffic is forecasted to be the dominating component of the Internet traffic, with more then 80 % share predicted in 2021 [7], motivates the development of new more efficient cost-based scheduling solutions.

This paper will firstly analyse the performance of the simple

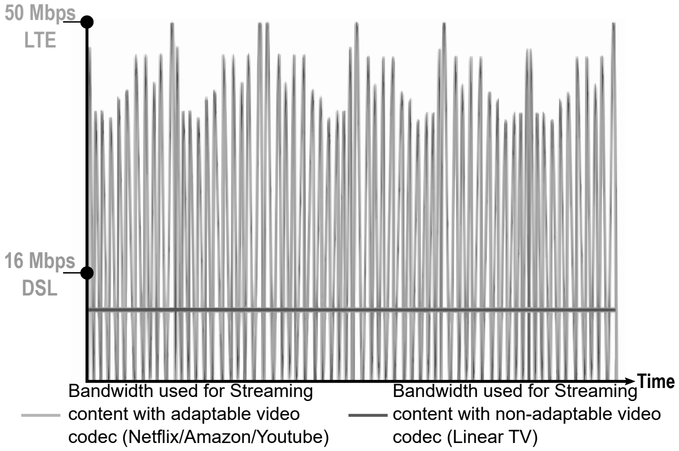


Fig. 1. Cost contradicting Video-on-Demand traffic in Hybrid Access with preferred DSL line over LTE access

cheapest path first scheduler in MPTCP based multipath. Testbed experiments and user trial results will be presented, helping us to identify the limitations of this and other scheduling algorithms for video traffic. Then design and case studies for the new cost-based scheduling algorithm will be presented together with the algorithm operation. Preliminary testbed results will then be shown, demonstrating the benefits of the new approach.

II. SCHEDULING IN MULTIPATH

Multipath scheduling as a method of traffic engineering is independent from a particular multipath protocol. Multipath TCP (MPTCP [5]) is used throughout this paper for evaluation purposes. MPTCP is available as open source Linux Kernel implementation[1] and is moreover a substantial part of the 5G ATSSS architecture. The use of several schedulers has been investigated in the past. While [8] and [9] give some overview about existing scheduler, [10], [11] and [12] propose specific designs goals, e.g. reduced head-of-line blocking. On the other hand the MPTCP reference implementation suggests the use of low latency preferred scheduler as *default*. A simple cost based scheduling mechanism which respects e.g. a path cost and can be reproducibly applied and evaluated is missing though.

Throughout this document, the monetary cost of using the available networks will be used as the main optimisation parameter, with the principle of having WiFi/DSL as the cheaper path compared to the cellular network path. In this section, we firstly present the concept, implementation and operation of a *cheapest-path first* scheduler, demonstrate its limitations, and then we proceed to introduce a new scheduler which is able to overcome these limitations.

The developed cheapest path first scheduler (*CPF*) for MPTCP schedules traffic on the cheapest available path, using permanent pre-defined access costs. When that path becomes fully saturated, other paths can be used, in ascending order of their cost. The trigger to switch from a prioritized path to a de-prioritized path is the exhausted TCP send window (`snd_wnd`).

To evaluate the performance of such a scheduler in MPTCP, two fully implemented MPTCP testbeds have been used. The local testbed setup (Fig. 2) comprises two x86 PCs equipped with Linux and MPTCP and two $1\,\mathrm{Gbps}$ capable Ethernet interfaces or a Wi-Fi link. Shaper devices within Link1 and Link2 let change the bandwidth, e.g. via `ethtool` to 10, 100 or $1000\,\mathrm{Mbps}$ full duplex or going through a separate Linux box using the `tc` command for fine granular bandwidth settings.

The basic relevant settings for all testbeds, except it is stated otherwise, are MPTCP version 0.90 - 0.95, a path manager enabling only one MPTCP subflow per link or access, TCP buffer settings according to [13] considering the expected sum bandwidth and Cubic congestion control. The respective traffic scheduler under test is always deployed on each side of the MPTCP termination points.

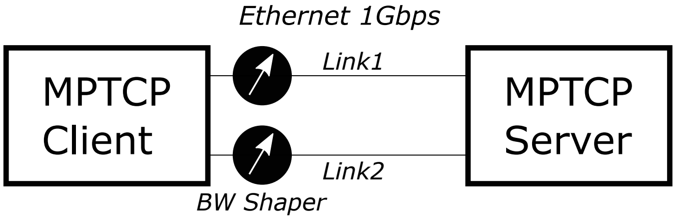


Fig. 2. Full controlled local MPTCP testbed

An Apache webserver running on the MPTCP server is used to simulate file transfers towards the MPTCP Client. In combination with hls.js[2] on client side, VoD transmission with the typical partial request of portions of the videos is produced.

The online testbed (Fig. 3) extends the one from Fig. 2 with access to the Internet and more realistic user equipment, resembling the 3GPP ATSSS multipath scenario. In this testbed a Proxy server located in the Internet acts as the MPTCP termination point for mobile phones (Google Pixel 2-4) or for the router and builds the interface towards the Internet, similar to [14]. In both scenarios, commercial LTE and fixed access

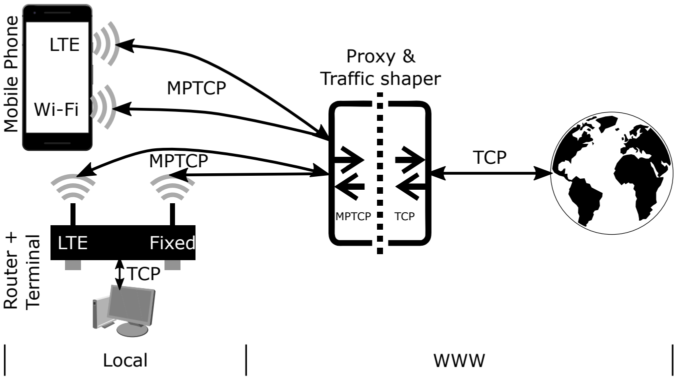


Fig. 3. MPTCP testbed for verification of online services

---

[1] http://multipath-tcp.org

[2] https://github.com/video-dev/hls.js/

(over Wi-Fi) is used to connect the respective user equipment (UE). An underlying UDP tunnel (not depicted for simplicity) between the UEs and the proxy over each access link ensures that any traffic passes through the Proxy server when traffic is exchanged with services in the Internet. Moreover, the UDP tunnel avoids MPTCP middlebox issues, which typically prevent MPTCP to work. The minimal lower MTU due to tunnelling is negligible from a performance perspective and anyhow irrelevant because all testing is applying the tunnel. A 800 Mbps downlink troughphut could be achieved over a combined commercial Wi-Fi and LTE on a Google Pixel 2 with 10 MB `tcp_wmem/rmem` set on UE and Proxy.

Verifying the *CPF* MPTCP scheduler reveals its functionality. With the local testbed, the responsiveness to path prioritization is demonstrated in Fig. 4, scheduling a 50 Mbps over a 10 Mbps Wi-Fi and an unlimited Ethernet link and toggling the priority every 20 s.

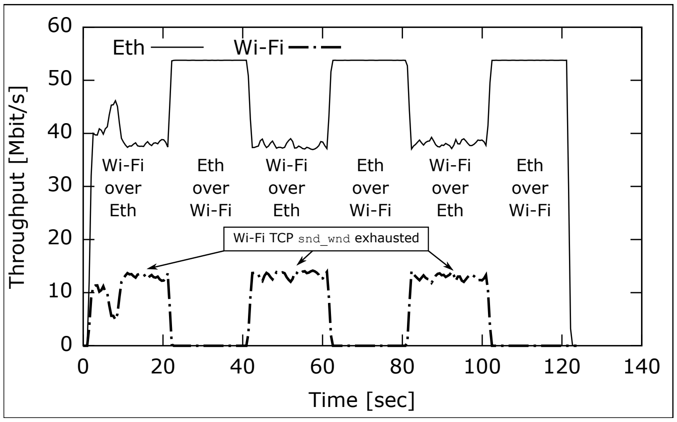
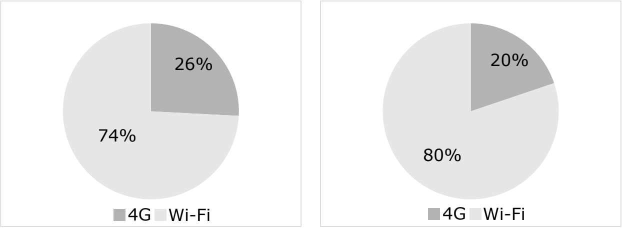


Fig. 4. CPF - Toggle link cost

Another important starting point for this research has been the results of a real user trial study, conducted with 5 mobile customers of a MNO, re-use the online testbed with mobile phones in Fig. 3. The objective of this study was to evaluate the effect of *CPF* compared to the *default* MPTCP scheduler, based on lower latency (SRTT). Both schedulers were applied each over one month to every TCP communication transferring each a total of 71 GB (*default*) and 77 GB (*CPF*). Fig. 5 shows the reduction in the use of the costly LTE access when *CPF* is applied over the `SRTT` driven *default* one.



(a) Default

(b) CPF

Fig. 5. Customer trial - Downlink LTE share comparison CPF and default scheduler

III. LIMITATIONS OF CURRENT SCHEDULERS FOR VIDEO-ON-DEMAND TRAFFIC

While the testbed and trial results presented in section II show the benefits of the implementation of the CPF scheduler, limitations of such a scheduler when video-on-demand (VoD) traffic is present can also be observed. Fig. 6 outlines this in a simple test - a VoD transmission in the online testbed given in Fig. 3. The video transmission using a 6 Mbps DSL is compared with and without access to the LTE path. Applying *CPF* in the latter case when requesting a full HD (1920x1080) video[3], causes a significant share in the costly LTE access even when DSL is prioritized. It is important to note that in both scenarios the video runs smoothly. The fact that the CPF scheduler transmits around 90 % of the traffic over the costly LTE path, despite the fact that the LTE path should be of lower priority, presents a major limitation in terms of the application of this scheduler.

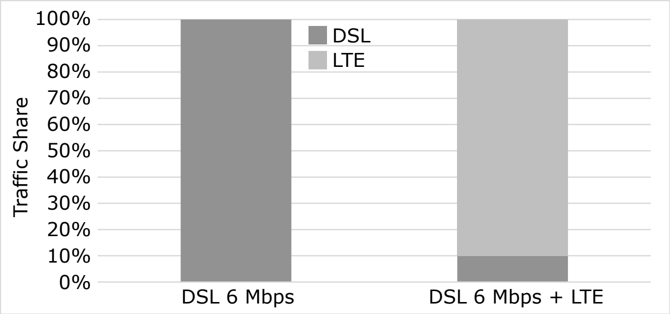


Fig. 6. Traffic share for smooth HTTP VoD streaming with 1920x1080 H.264 over 10min with MPTCP + CPF

This behaviour of the CPF scheduler has origin in the nature of the VoD traffic. The VoD typically exhibits a bursty traffic pattern (Fig. 1), using transport protocols like TCP or QUIC. Such protocols grab as much bandwidth as they can, which is also in the interest of the VoD operator, giving better overall service to the users and avoiding playback issues. On the receiver side, the playback application maintains a buffer storing the portions of the video until they are played. The new portions of the video are only transmitted if the buffer level has reached a low state. That avoids unnecessary transmission of the whole video while ensuring smooth playback, albeit it is responsible for the bursty traffic pattern as long as the bottleneck bandwidth is sufficient to transmit portions of the video faster than being played. Two issues appear to challenge the use of the *CPF* scheduler in this context. The first is the very fast dispatch of portions of the video data from the VoD service. This is most often the case at the proxy with significantly higher throughput then the bottleneck-bandwidth towards the consumer. The second issue is the time until a proper `snd_wnd` is built up on the primary path. Both of these issues lead even at higher DSL datarates with 50 Mbps or 100 Mbps to an increased LTE share, measured at ˜40 %

---

[3] https://hls-js.netlify.app/demo/?src=https%3A%2F%2Ftest-streams.mux.dev%2Fx36xhzz%2Furl_8%2F193039199_mp4_h264_aac_fhd_7.m3u8

and ˜$25\,\%$ respectively. One solution to this problem may be to increase the capacity of the cheaper access network using fiber or other technologies. The existing Data plan limitations, wide deployment of copper based infrastructures and Wi-Fi bottlenecks however often lead to situations where the capacity of the primary path is limited. Understanding this will also help to put the result of Fig. 5 much better in context, as they don't show a significant difference in the consumption of the costly resource.

The diagram in Fig. 7 presents the operation points (OP) of the *CPF* driven Hybrid Access (HA) and the single path low-cost access. The figure depicts QoE over the costly resource usage within a multi-access setup. Between the two extreme operation points, the QoE gain provided by multi-access will move from none (single low-cost access) to maximum (HA). Even if Fig. 7 cannot give us a concrete answer how to solve or mitigate non-beneficial utilization of costly resources, it helps to define the following principle design objective:

Moving the today's HA operation point within the solution space and minimizing the access to the costly resource while keeping the QoE at a sufficient level.

Reaching the optimal operation point for VoD traffic in multipath scenario can then be defined as the objective of the design of a new cost-based scheduler, which will be presented in the next section.

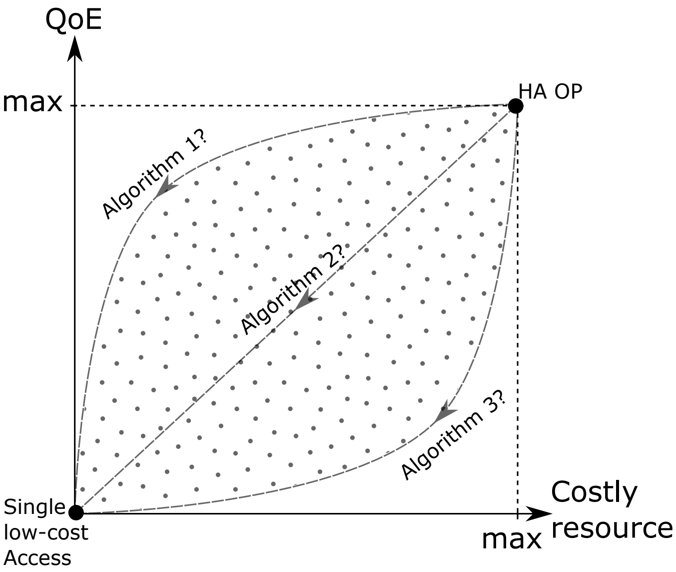


Fig. 7. Multi-connectivity operation point considering QoE and costly resource consumption

IV. COST-OPTIMIZED MULTIPATH SCHEDULER

To define the design goals of the new algorithm, we should take a look at the nature of traffic bursts in multipath scenarios, depicted in Fig. 8. We can note that:

1) The overflowing part requires a costly transmission, and as such should be avoided.
2) Between burst exists a "valley", leaving available capacity on the cheaper path unused.

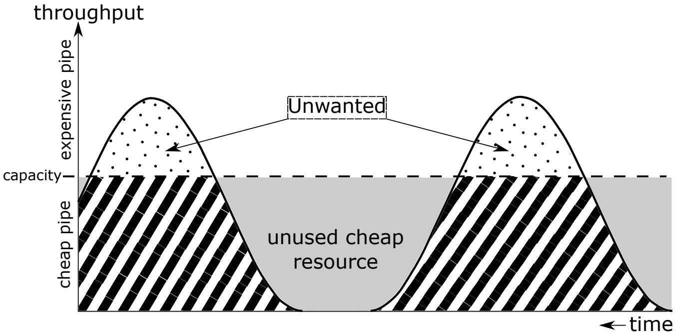


Fig. 8. Unwanted multipath operation when traffic burst overflow into costly paths

With this in mind, we can define design goals for the new algorithm as:

- the algorithm needs to be able to detect traffic bursts causing spurious demand in multi-connectivity scenarios
- the traffic needs to be scheduled according to the real application and customer needs as much as possible on the cheaper resources
- the algorithm needs to be generic, simple, and service-agnostic, requiring no service-specific support
- the algorithm must not decrease user QoE
- the real demand must not be affected

If we use C_{cheap} to denote the capacity of the cheaper resource and BW_{demand} to denote the total bandwidth demand, we can state the following design principle for the new algorithm:

If (1) *is true, then prevent access to the expensive pipe.*

$$\int_0^t C_{cheap} \geq \int_0^t BW_{demand} \tag{1}$$

We will call this new scheduler the *Cost Optimized Multi-path (COM)*. While (1) provides many challenges to overcome, including dimensioning the time interval, determining C_{cheap} and monitoring BW_{demand}, an idea presented in Fig. 9 can be used to design a working solution. Instead of monitoring capacities and demands, the T_{GAP} size can be used to identify the saturation information. To do this, the time gap between consecutive packets can be measured and compared against a threshold value $T_{GAPthresh}$. If the condition given in (2) is true, access to the expensive path is prevented for a time span of T_{Delay}, with the multipath scheduler sending the packets to the cheaper path only. As a consequence of this, bursts will be stretched over time using a higher share of the cheaper pipe.

$$T_{GAP} \geq T_{GAPthresh} \tag{2}$$

To analyse this idea further, we can look at the following use cases:

- **VoD with** $BW_{demand} < C_{cheap}$
  Due to the original bursty nature of the video data, a time gap should be visible between the traffic bursts end-to-end, as long as no intermediate bottleneck disrupts this. If the multipath scheduler can monitor this, it can schedule

data to the cheaper path and access to the expensive path is not required.

- **VoD with** $BW_{demand} > C_{cheap}$
  Different to the use case above, demanding a higher throughput than the cheaper pipe can provide will constantly fill this path and no gap will be detected. Access is therefore given to the expensive path which is now responsible to drain the overflowing traffic.
- **VoD with adjustable demand**
  It is expected that this will also match the case when a VoD service dynamically adjusts the video resolution according to the available throughput. Such a situation will lead to either the first or to the second use case. At least an upgrade to a higher resolution should not be blocked, since the gap will become shorter or even vanish.
- **File download**
  A constant file download should not be affected at all. This kind of traffic is out of scope of this work since it already works with the *CPF* scheduler, as shown in Fig. 4. In the case the scheduler verifies a constant demand on the cheaper path, justified by the nature of a file download without gaps, access is provided to the expensive path. It also does not matter if the file download demand is below or above the capacity of the cheaper pipe, as the basic *CPF* principle kicks in.
- **Bottleneck before the scheduler**
  In the case the bottleneck is not the cheaper path and the bottleneck appears before the traffic reaches the multipath scheduler, the **file download** use case is applied.

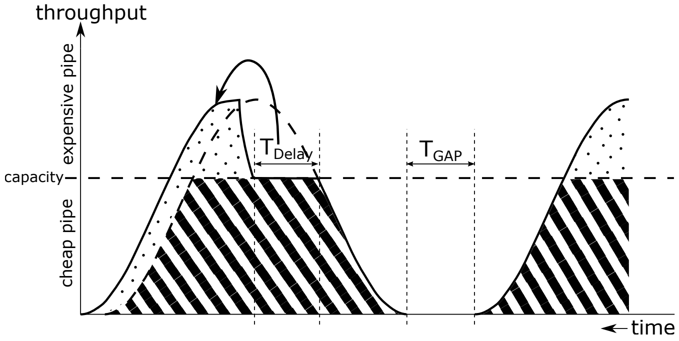


Fig. 9. COM - Practical idea to detect unsaturated link capacity based on the gap time T_{GAP} and T_{Delay} as measure to prevent spurious costly demand

The performance of the COM scheduler is evaluated in experiments using the MPTCP-based testbeds presented earlier in the paper. The experiments on the testbeds also server to determine the algorithm parameters, namely $T_{GAPthresh}$ and T_{delay} and potential dependencies on services or C_{cheap}.

V. EVALUATION

The performance impact of the *CPF* scheduler compared to the latency driven *default* MPTCP scheduler was already evaluated in section II. Similar tests were performed to compare the maximum achievable throughput of the *COM* scheduler. No measurable computational impact is caused by taking timestamps, comparing them, setting and verifying T_{delay}.

When QoE is considered for Video-on-Demand services, the good quality of experience is indicated by a smooth playback of video. Whenever the video playback is disrupted (frozen) a reduction of QoE is noticed. It is obvious that the focus on consumer QoE includes the network related transmission characteristics like loss, latency and jitter. If at least one of these fails to ensure the expected levels, the video playback starts stuttering. Furthermore, to validate the impact of the *COM* algorithm, its performance is compared to: 1) single path transmission over the cheaper path and 2) the *CPF* algorithm, both according to Fig. 7.

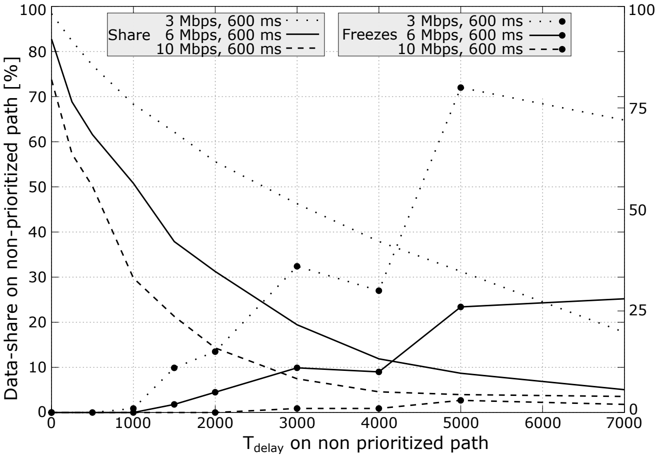


Fig. 10. COM - Costly share of different C_{cheap} over T_{delay} in a local testbed

Fig. 10 compares the amount of cost optimization (left y-axis) while monitoring the QoE (right y-axis) in terms of occurrences of video freezes. Both are depicted as function of T_{delay}, which indicates the time access to the more expensive path is denied after traffic burst is identified based on (2). T_{delay} is investigated in an interval $[0\,\mathrm{s}; 7\,\mathrm{s}]$ whereas $0\,\mathrm{s}$ means effectively *CPF* (no optimization) and $7\,\mathrm{s}$ points towards single path behaviour. In accordance with the local testbed Fig. 2, the MPTCP client/server setup is deployed, with the server providing the same 1080p/60fps video as in footnote 3, however locally based on hls.js. The measurement period corresponds to the video length of approximately $10\,\mathrm{min}$ and an average throughput of $8\,\mathrm{Mbps}$. The prioritized Ethernet link is shaped with means of `tc` in a bridging device to 3, 6 and $10\,\mathrm{Mbps}$, while $T_{GAPthresh}$ is fixed to $600\,\mathrm{ms}$. The secondary, non-prioritized path, is left unchanged at $1\,\mathrm{Gbps}$. In hls.js, the video segment size was set to $4\,\mathrm{s}$ and the buffer size on client side was set to two segments effectively lead to a pre-load of up to $8\,\mathrm{s}$ of video material. Measurements were only conducted once due to the local controlled nature of the testbed with a high expected reproducibility. Starting with the non-optimization ($T_{delay} = 0$) case, we can see for all tested C_{cheap} values a very high consumption of the de-prioritized resources roughly between $70\,\%$ and $90\,\%$. With an increasing T_{delay}, this share can be significantly reduced by multiple decades and that even with already very small T_{delay}. At the same time at around $1\,\mathrm{s}$ the total number of video freezes starts to grow. This is expected as the prioritized bottleneck path is prevented to aggregate a secondary path for some time. Nonetheless, comparing this with the freezes at high T_{delay}, which seem to be close to single path transmission, we can identify a wide period in between, which indicates both, a significant reduction of costly resource usage and an acceptable level of QoE with a clear benefit when no multipath is applied. In terms of the Fig. 7 operation point, a significant shift from the right to the left (costly resource usage) with a minor shift from the top to the bottom (QoE) can be identified.

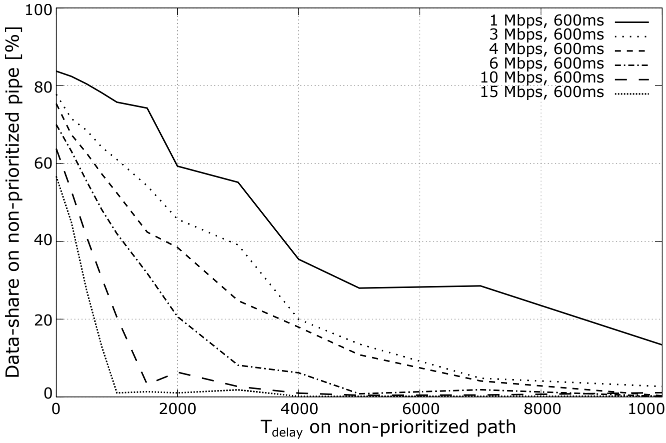


Fig. 11. COM - Costly share of different C_{cheap} over T_{delay} with YouTube

While in Fig. 10 *COM* was verified in a local testbed, Fig. 11 uses the router from the online testbed Fig. 3 to request a YouTube video[4] with auto-resolution limited to 1080p. Similar to the local testbed a significant offload of the costly resource is gained. Even with small T_{delay} values the LTE share can be brought down to almost zero for a $C_{cheap} \geq 6\,\mathrm{Mbps}$. An issue which appeared occasionally with YouTube was, that during the transmission of portions of the videos, control messages were exchanged which render the burstiness detection based on (2) useless.

VI. FUTURE WORK

Since the results shown in this paper are very promising, the authors aim to follow this approach and evaluate and optimize the *COM* algorithm further. In the next steps they will elaborate the algorithm under different conditions and match the algorithm design to all design goals. Further experiments will be conducted to identifying optimal values for $T_{GAPthresh}$ and T_{delay} across a range of C_{cheap} values, and QoE measurements will be done for adaptive VoD. Operation of the scheduler with a mix of data and video traffic will also be analysed, as well as operation in multipath transport solutions other than MPTCP (e.g. MP-DCCP). Volatile network scenarios will be analysed, including possible trials with real users. These next steps allow a more detailed discussion of the *COM* algorithm with its advantages and limitations, as well as its position in the set of existing multipath schedulers.

VII. CONCLUSION

This paper analyses the challenge of transporting bursty video-on-demand traffic in multipath network access scenarios. Testbed experiments and user trial results are shown demonstrating the limitations of currently available scheduling algorithms in terms of the costly usage of cellular network access. The paper presents design goals of a cost-based scheduling algorithm which will simultaneously reduce the cost of multipath use for network operators and also retain the QoE levels required by the end-users. The new algorithm called Cost Optimized Multipath (COM) is presented, and preliminary evaluation results are shown, demonstrating the ability of the new algorithm to outperform the existing scheduling algorithms.

REFERENCES

[1] K. Sklower, B. Lloyd, G. McGregor, D. Carr, and T. Coradetti, "The ppp multilink protocol (mp)," Internet Requests for Comments, RFC Editor, RFC 1990, August 1996.

[2] ITU-T, "Itu-t recommendation g.998.2 : Ethernet-based multi-pair bonding," ITU-T, Tech. Rep., 2005.

[3] N. Leymann, C. Heidemann, M. Zhang, B. Sarikaya, and M. Cullen, "Huawei's GRE Tunnel Bonding Protocol," RFC 8157, RFC Editor, RFC 8157, May 2017.

[4] M. Bednarek, G. B. Kobas, M. Kühlewind, and B. Trammell, "Multipath bonding at layer 3," in *Proceedings of the 2016 Applied Networking Research Workshop*, ser. ANRW '16. New York, NY, USA: ACM, 2016, pp. 7–12.

[5] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch, "Tcp extensions for multipath operation with multiple addresses," Internet Requests for Comments, RFC Editor, RFC 8684, March 2020.

[6] 3GPP, "System architecture for the 5G System (5GS)," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 23.501, 07 2020, version 16.5.0.

[7] Cisco, "Cisco visual networking index: Forecast and methodology, 2016–2021," Ciso, Tech. Rep., June 2017.

[8] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental evaluation of multipath tcp schedulers," in *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*, ser. CSWS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 27–32.

[9] O. Bonaventure, M. Piraux, Q. D. Coninck, M. Baerts, C. Paasch, and M. Amend, "Multipath schedulers," Internet Engineering Task Force, Internet-Draft draft-bonaventure-iccrg-schedulers-00, Mar. 2020, work in Progress.

[10] H. Wu, O. Alay, A. Brunstrom, S. Ferlin, and G. Caso, "Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2295–2310, 2020.

[11] S. Ferlin, O. Alay, O. Mehani, and R. Boreli, "Blest: Blocking estimation-based mptcp scheduler for heterogeneous networks," in *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, 2016, pp. 431–439.

[12] Y.-s. Lim, E. M. Nahum, D. Towsley, and R. J. Gibbens, "Ecf: An mptcp path scheduler to manage heterogeneous paths," in *Proceedings of the 2017 ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '17 Abstracts. New York, NY, USA: Association for Computing Machinery, 2017, p. 33–34.

[13] F. Zhou, T. Dreibholz, X. Zhou, F. Fu, Y. Tan, and Q. Gan, "The performance impact of buffer sizes for multi-path tcp in internet setups," in *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, 03 2017, pp. 9–16.

[14] O. Bonaventure, M. Boucadair, S. Gundavelli, S. Seo, and B. Hesmans, "0-RTT TCP Convert Protocol," RFC 8803, Jul. 2020.

---

[4] https://youtube.com/watch?v=aqz-KE-bpKQ