Client-Driven Network-level QoE fairness for Encrypted 'DASH-S'

Junyang Chen[♭], Mostafa Ammar[‡*], Marwan Fayed[♮], and Rodrigo Fonseca[♭†],
[♭]Brown University, [‡]Georgia Institute of Technology, [♮]University of Stirling
{junyang,rfonseca}@cs.brown.edu, ammar@cc.gatech.edu, mmf@cs.stir.ac.uk

ABSTRACT

Adaptive video streams, when competing behind a bottleneck link, generate flows that lead to instability, under-utilization, and unfairness. Recent studies suggest this negatively impacts users' perceived quality of experience (QoE). Two general classes of solution exist. Client-side bitrate adaptation improves stability and may achieve flow-rate equality. However, operating in isolation, bitrate adaptation has no ability to establish equal end-user experience, or *QoE fairness*, among competing clients. Conversely, network services can manage bottleneck resources to achieve QoE fairness, yet the widespread use of HTTPS renders them ineffective. In this paper we step back to evaluate the issue with a broad and fully inclusive view of fairness, or equality, by any definition. Our focal contribution is a constructive argument that makes clear, given ubiquitous use of encrypted HTTP, *all notions* of QoE fairness or equality can only be achieved when clients and network collaborate. Furthermore, operating boundaries are identified with their features and limits. We then architect and implement client-Driven Video Delivery (cDVD), a proof-of-concept that provides a client-level API into the network, to explore these boundaries. cDVD measurements reinforce our argument and raise new opportunities for exploration.

CCS Concepts

•Networks → Network architectures; •Information systems → Multimedia streaming;

Keywords

DASH; Fairness; Quality of Experience; Performance

*M. Ammar supported in part by NSF grant NeTS 1409589.

†R. Fonseca supported in part by NSF grant NeTS 1320397.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Internet-QoE, Aug 22-26, 2016, Florianopolis, Brazil
© 2016 ACM. ISBN 978-1-4503-4425-8/16/08…$15.00
DOI: http://dx.doi.org/10.1145/2940136.2940144

1. INTRODUCTION

Users of the Internet have come to rely on the experience that derives from its stability, efficient utilization, and fairness. Mounting evidence suggests that adaptive video streaming, which most dominates in homes and in the Internet, defies these fundamental cornerstones of the Internet's design.

The streaming standard, Dynamic Adaptive Streaming over HTTP (DASH), is defined by the MPEG Foundation [19, 24]. The interactions between clients and the network in DASH systems have received significant attention (eg. [2,3,11,13,14, 17, 25]). These studies reveal the inherent challenges associated with delivering a high quality of experience to the user as the system scales. In particular, the on/off requests from multiple clients (or sessions) that compete for bandwidth across a bottleneck link cause (i) instability in the selected encoding, (ii) bottleneck-link under-utilization, and (iii) disproportional shares of available bandwidth.

Each manifests on users' quality of experience differently. Instability, for example, appears as images of varying quality over time. Under-utilization may prevent clients from requesting the best possible encoding for the user. Disproportional shares, or unfairness, is particularly problematic: Rather than equal or weighted share of capacity, selected bitrates can be determined by factors such as time of arrival, operating system support, and implementation [2, 17]. In total the complex interactions of DASH clients, intended to improve video delivery, *can negatively impact the quality of experience*.

One class of solution lies at the endpoints. Among clients a variety of bitrate adaptation algorithms are designed to estimate available bandwidth, and request the best possible encoding. Estimates may derive from segment downloads [11, 14] or, increasingly, inferred from buffer occupancy [10, 25]. Requests may then be staggered or scheduled to improve bitrate equality between competing clients. Server architectures may also be instrumented to improve system and end-to-end performance [15, 16, 20].

In contrast, in-network services are motivated by the inequality, or absence of fairness, between competing clients. The best case endpoint rate adaptation is per-flow bottleneck fairness [5]. Clients, servers, and CDNs, lack the global knowledge to properly allocate bandwidth. Further, in discussing notions of fairness, it is useful to distinguish between equal bitrate, and *equal utility*. Utility is critical when es-

tablishing *QoE-fairness*, which is distinct from conventional definitions of TCP-like fairness. Rather than equal shares of capacity, i.e. bits per unit of time, QoE fairness aims for equal utility. Bitrate utility varies, for example, with screen sizes, viewing distance, and cost of data plan [18]. Studies have shown that in-network controls, alongside new measures and algorithms, can allocate bottleneck resources in a QoE-fair manner [8, 18]. However, their deployments are rendered ineffective by the increasing use of encrypted HTTP [21].

In this paper we argue that, with the prevalence of DASH over encrypted HTTP – 'DASH-S' – client participation becomes *necessary*. It is well known [7, 22] that client-to-network interactions can simultaneously improve the quality of the former, and performance of the latter. The crux of our argument is that only the network, uniquely in posession of global views, can properly decide and enforce resource allocations among competing clients. However, encrypted traffic makes it infeasible for the network to identify and infer the needs of competing video streams. Clients, in contrast, are uniquely in posession of their per-bit utility. In more detail, our main argument is summarized as follows:

- Clients in the best case may achieve stability, and flow-rate (TCP-like) fairness; absent of a global view, Quality-of- any other measure of fairness is impossible.
- The Network, given a global view, is in the best position to ensure session-level equality or QoE fairness (and police rogue clients); *however, the prevalence and opaqueness of HTTPS preempts this possibility.*
- Ergo, high levels of QoE can *only* be achieved when the network is given insight by the clients.

Hence a strict separation between application and network is no longer viable for scalable streaming *encrypted* video. The validity of our argument is independent of the definition of utility, and of QoE. Without loss of generality, for our prototype we borrow the QoE metric from [18].

We have architected and implemented a prototype in cDVD, an in-network client-Driven Video Delivery service. Like its predecessors, cDVD embeds a software-defined networking (SDN) component [8], that draws allocations from a provably QoE fair algorithm and metric [18]. Separate from its predecessors, cDVD is characterized by a client-facing API using 'participatory networking' principles [7]. Clients use the API to drive the cDVD model, in contrast to models where traffic is monitored and then clients are informed post hoc.

Preliminary experiments with wide-area streams from the BBC [4] and a modified `dash.js` client reinforce our focal argument. In particular there are marked improvements *when clients actively disclose* utility to cDVD that is otherwise unavailable over encrypted HTTP. This is a necessary first step towards motivating content providers to participate.

Our work provides additional motivation, further reinforcing the case, for centralized controls first put forward in [16] and later instantiated in [20]. Though our position argues for network layer abstractions instead of for providers and CDNs, a similar set of questions emerges related to parameter spaces, interfaces, and incentives, reiterating their importance.

In summary we explore the design space for streaming video architectures over encrypted HTTP. We reason that any resource allocation with a goal beyond per-flow fairness or equality is only available when clients and network collaborate. Experiments with our cDVD reinforce this argument, and open new avenues for investigation.

2. DYNAMIC ADAPTIVE STREAMING

In Dynamic Adaptive Streaming over HTTP (DASH) [19, 24], video content is segmented and encoded at multiple bitrates over a range of quality. Segments are non-overlapping and have equal duration across encodings. Each video has an associated manifest file (`mpd`) that describes the available encodings, media segments, server locations, and other information that may be needed by the client. During playback clients intermittently request segments and simultaneously estimate available bandwidth. Requests are for the highest quality encoding subject to estimates of network resources. Outside of the DASH standard, `mpd` files, requests, and responses are increasingly encrypted over HTTPS and HTTP/2 [21].

The diversity of devices and network conditions makes it difficult to achieve high quality delivery. One reason is that the intermittent requests are anathema to TCP. Every new HTTP request is launched with stale TCP state from the previous request. This on/off behaviour prevents TCP from reaching equilibrium [11]. Rather than equal or weighted share of capacity, obtained bitrates appear to be determined by factors such as the time of arrival relative to other streams, the viewing platform and implementation, operating system supports, and the content provider [17].

Separately, network-level analyses of adaptive video streaming behaviours is made difficult because segment lengths, encoding bitrates, and the algorithms to switch between them, are absent from the standard and left to the discretion of the implementation. Despite this omission, relevant details are contained in the `mpd` files that are transmitted once a video session is requested. Unfortunately, end-to-end encryption precludes in-network opportunities to glean such information.

This "HTTPS adoption impact chain" [21] motivates our exploration, and is the basis for arguing that QoE-fairness is only achievable if clients interact with the network.

3. TRANSITIONING FROM QoS TO QoE

In this section we evaluate *fairness* of competing DASH sessions over multiple possible architectures in an Internet environment where all traffic is encrypted. For any definition of QoE, and more generally all non-flowrate definitions of quality, we show that fairness can only be achieved when clients feed into the network.

Our argument is constructive, beginning with client-only solutions. A graphical summary of our argument is provided in Figure 1. The operating boundary between the network and clients in an encrypted environment is delineated by the vertical dotted line. Each greyed region is labeled with its corresponding subsection number below.

3.1 Bitrate Adaptation at Clients (e2e)

Streaming video clients execute a bitrate adaptation algorithm to select the best possible encoding. The adaptation may

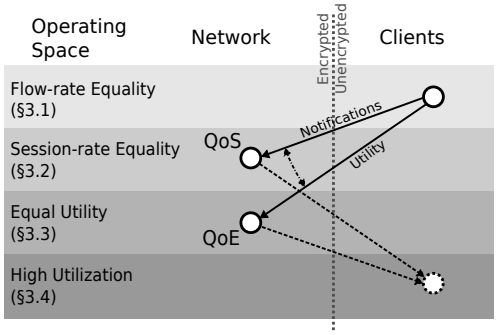


Figure 1: QoS- and QoE-fairness for encrypted streams requires client input. Conservative client adaptation makes high utilization unlikely without feedback.

be driven by bandwidth estimation and prediction (eg, [9]), or by using the occupancy of the playout buffer (eg, [25]). These algorithms have seen tremendous improvements in recent years, improving bottleneck utilization, stability, and better interaction with TCP. Server-side interventions and adaptations can also improve system performance and delivery [15, 20]. However even if unencrypted, end-to-end solutions are fundamentally limited when there are multiple independent video sessions sharing a bottleneck, as is increasingly the case in most highly connected homes.

Since video segments are requested with HTTP, bitrate adaptation algorithms are bound by TCP fairness. In the best possible case, client-side algorithms can achieve flow-based fairness, if all the estimation biases are controlled for [11]. However, only in very limited scenarios does this translate to session-level fairness, or QoE fairness. There are three main limitations of client-side bitrate adaptation:

First, clients that use multiple TCP connections for the same session increase their share of bandwidth [13]. Figure 2 measures the acquired bitrate between two DASH clients when one requests segments in parallel TCP connections. The current version of `dash.js` (1.6) requests segments in series. Measured against parallel requests from a Netflix client the gap in acquired bitrate is ∼3x. Even when compared to the previous version of `dash.js`, which requests segments in parallel, the serial client bitrate is neither stable nor fair.

Second, bitrate adaptation has no knowledge of competing clients, so QoE fairness – where the utility of a bit is not the same to each client – is impossible to achieve without network intervention.

Third, the absence of implementation standards leaves providers free to develop different adaptation algorithms, which means that more aggressive clients will capture most of the bandwidth [2, 3, 17].

This regime corresponds to region 1 in Figure 1. Together, these limitations suggest that any measure of quality, either at session- or experience-level, requires network-level interventions, since only the network has a global view of both available resources, and active sessions (when unencrypted).

3.2 Session-layer Fairness via the Network

Rather than the flow-rate fairness to which client performance is bound, it is more appropriate to perceive the goal

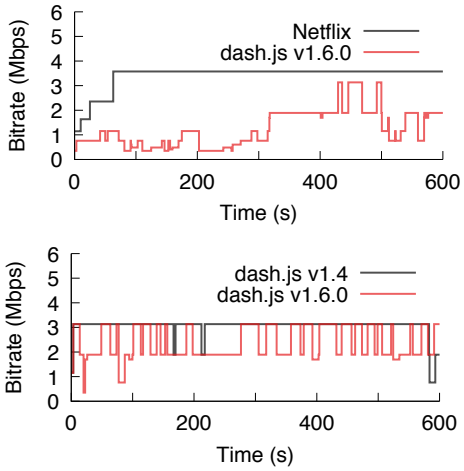


Figure 2: Netflix and dash.js 1.4 use parallel TCP flows, and get more bandwidth than dash.js 1.6, which does not.

of bitrate adaptation as session-rate equality. Session-layer fairness is an old idea in the Internet [12]. In the context of streaming video, a session is comprised of the set of TCP flows, generated by parallel HTTP client requests, for a single video (and associated audio) stream.

Absent encryption, two network functions combine for session fairness, traffic control and packet inspection. Traffic control is required to redirect all packets from a session's flows into a per-session queue. The packets' HTTP requests are inspected to identify the session to which they belong.

However, packet inspection is increasingly problematic. Traffic is increasingly encrypted over HTTPS [21]. In addition, all major Internet browsers have decidedly implemented HTTP/2 without unencrypted options. Packets associated with encrypted video sessions are opaque and indistinguishable from any other network traffic, rendering all in-network session-fair services ineffective. Figure 1 bridges this gap in the second region, where clients notify the network of flows to corresponding sessions, which re-enables traffic control.

Clients can additionally share a spectrum of information, trading off information exposure for more sophisticated resource allocation by the network. This leads to a set of requirements for providing QoE fairness.

3.3 Quality of Experience

The network is tasked with preserving classical notions of utilization, stability, and *fairness*. At the network layer, then, QoE-fairness reflects the experience to all users. This requires acknowledgement of bit *utility*, rather than bit equality [23]. We are aware of two studies in this direction [8, 18].

The QFF framework [8] established that software-defined networking mechanisms can be used to allocate and enforce resources based on utility. Utility functions derived from encoded resolutions are pre-computed and stored in a database. Implemented using OpenFlow, QFF monitors network traffic to capture client requests. Allocations are inserted into flow tables, and shown to improve utility.

Subsequently, a 'network-friendly' QoE utility measure based on screen size was designed for VHS [18], alongside

a maximally fair allocation algorithm. Implemented on a home router, VHS also monitors parallel HTTP requests to enforce session-level fairness. Real-world experiments conducted with Netflix and YouTube clients uniformly improved network utilization, stream stability, and QoE fairness.

These works show that the network is uniquely positioned to maximize QoE for all users in a fair manner. However, as is true of session identification, the opaqueness of encrypted payloads mean the network has no ability to infer experience nor identify utility. Thus, clients must participate with the network to establish QoE-fairness.

This observation leads third region in Figure 1: As is true of QoS, session-level QoE-fairness, by any definition, can only be achieved with client participation into the network. We reason this (i) resolves encryption barriers, while (ii) enabling content providers to keep control of otherwise proprietary details. The granularity of shared information, and type of exchange, offer differing levels of incentive to participate.

3.4 Client Feedback

The fourth region in Figure 1 begs the next question: Having reasoned that non-flow rate fairness needs client input,can the system be improved if feedback is returned to clients?

For example, DASH clients under-utilize network resources to accommodate variability by requesting video encodings that are below perceived available capacity. The current version of `dash.js` (1.6) limits requests to 0.9 of estimates; measurements of Netflix suggest requested encodings are 0.8 of available bandwidth [18]. Variability in a video stream can originate in either the encoded source media or the network. Clients already have visibility into the encoding, leaving network variability as the unknown. Feedback from the network could fill that gap. This suggests improvements in utilization, and higher quality encodings to the client.

QoE fairness demands that the encrypted HTTP boundary between network and clients be bridged. In the next section we build a client-facing in-network service to bridge this divide. Preliminary measurements validate our argument and raise opportunities for exploration.

4. DESIGN AND IMPLEMENTATION

In this section we describe the architecture and prototype implementation a client-Driven Video Delivery service in the network, cDVD. We use our implementation to evaluate our main argument, and find that it opens new research directions.

At a high level, cDVD enables one or more DASH clients to interact with network components to achieve fair QoS or QoE, following the reasoning in Figure 1. We present corresponding variations by making the interface between the two progressively more expressive. Our work is inspired by 'participatory networking' [7], where end-user applications interact with the network for their mutual benefit. Figure 3 shows the architecture, where there are three video sessions that traverse two routers controlled by the cDVD controller. Each session can comprise multiple flows. Clients interact with the cDVD controller through a client interface, and the controller, in turn, applies bandwidth control to the routers

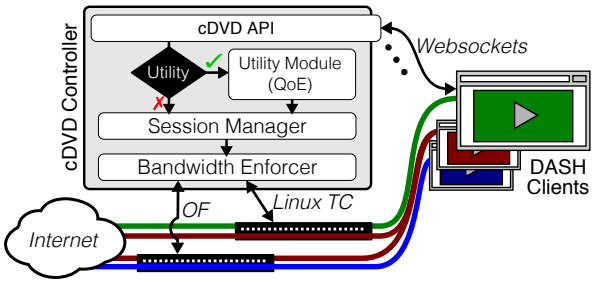


Figure 3: High-level cDVD architecture

via a suitable mechanism, such as OpenFlow or direct queue configuration. It is also possible for the controller to get feedback from the routers about the video sessions, although we leave this aspect for future examination.

We implement the cDVD client by modifying `dash.js`, the reference DASH client from the DASH Industry Forum [6]. We instrument the client to find and communicate with the cDVD controller. In our experiments, this communication can be unidirectional, with the client sending information about its video sessions; or bidirectional, with cDVD informing the client, for example, about its rate allocation.

While the architecture admits a more general implementation, in our prototype we make a few simplifying assumptions: one or more clients share an Internet connection through a single router (as in most homes); the router is their local DNS resolver; and, in most cases, the bottleneck for the clients' video sessions will be the link between the router and the provider. We also assume that there is only one simultaneous video session between each pair of client and server machines. This assumption is easily lifted, and made to facilitate session differentiation at the router without port numbers.

With each new video session, our modified DASH client attempts a websocket connection to a well-known domain name and port (we use `cdvd.local:9000`). If the controller replies, then the client operates in a cDVD mode, otherwise it operates as is normal. In our case, we configure the router's DNS so `cdvd.local` points to the controller.

The client communicates with the controller using a simple json API. The controller, in turn, can configure a bandwidth enforcer to rate-limit particular client flows. We implemented the controller and the bandwidth enforcer on a TP-Link TL-WR1043ND v1.8 running OpenWRT [1]. There is no requirement for co-location. The requirements are for clients to be able to find the controller, and for the bandwidth enforcement mechanism to sit on the path between the client and the server(s). The controller has a websocket server which interacts with the clients. Once it decides on the limits for each video session, it uses Linux `tc` to create one network queue per session, and to assign the TCP flow(s) corresponding with each session to the right queue.[1] In the case where the clients send their utilities (cf. §3.3), the controller allocates bandwidth to equalize QoE across sessions, otherwise it does a fair bandwidth allocation *across sessions*. We emphasize that

[1] Equivalently, an OpenFlow controller with a websocket-based northbound API, and bandwidth enforcement via QoS OpenFlow extensions, is functionally equivalent to our implementation.

cDVD is utility-agnostic, i.e. fair allocations by the Utility Module shown in Figure 3 are left to the implementation. To validate client interactions, the current implementation of cDVD borrows from [18] to determine utility and fairness.

5. EVALUATION

We use our implementation to evaluate the pitfalls and potential gains of the different regions in Figure 1. In our setup, the bottleneck link is 6Mbps. We have one or more DASH clients mimic different sized devices to evaluate a QoE context. Each requests the BBC DASH Testcard media [4] from the wider Internet, which comprises 13 video encodings, 2 audio encodings, and a duration of up to 1 hour.

5.1 Client Participation

We summarize our experiments by the three measurements in Figure 4, corresponding to the first three regions in Figure 1. Figure 4a shows bitrate adaptation, alone, and serves as a baseline: Clients converge on their ∼2Mbps 'fair-share' streams with limited and varying degrees of success. Note the fluctuating client-generated estimates of available bandwidth (dashed lines), that they use to select encoding quality. We revisit this observation in Secion 5.2.

In Figure 4b cDVD and its API are activated to explore session-rate equality. Clients launch a video session with the provider and simultaneously notify cDVD, which is otherwise blind to HTTPS sessions. Given only notifications, cDVD allocates 2Mbps to each *session*. Client-generated bandwitdth estimates stabilize in response, causing clients to converge with greater success. The instability that appears in both Figures 4b and 4c, we believe, is due to upstream bottlenecks beyond our control and discussed further in Section 6.

Figure 4c shows that clients can also drive QoE-fairness. Upon launching a video session, clients communicate their *utility* via the cDVD API. For our purposes, and for validation, we implement VHS utility functions and algorithms [18]. Gaps between requested bitrates, and estimates indicative of cDVD allocations, reflect the conservative nature of DASH clients. Here, `dash.js` rejects video bitrates between 90-100% of estimates in favour of the next lower bitrate. This can be a substantial step downward, as can be seen in Figure 4c.

5.2 Client Feedback

Clients requests are necessarily conservative because of both network and encoding variability. Yet in Figure 4c, gaps between requested and allocated bitrates suggest a missed opportunity. We briefly consider whether feedback from the network enables clients to be reasonably less conservative.

Preliminary measurements appear in Figure 5, where clients request only those bitrates selected by cDVD. This represents the extreme from the range of feedback uses. Bandwidth estimates, indicated by the dashed lines, are ignored when making requests. Session-fair requests appear in Figure 5a, and contrast against the case where feedback to clients is absent in Figure 4b. Despite being unable to drop requested bitrates in response to upstream bottlenecks, session-fair rates requested in our measurements play with no re-buffering.

In Figure 5b clients request QoE-fair rates as instructed by cDVD. Here, bandwidth estimates were slightly below the allocated rates, corresponding with higher rates of re-buffering (2-7%). These lower estimates only occurred in wide-area experiments. They are explained by the way the client estimates bandwidth: After each requested segment, RTT is incorporated into the estimate alongside the transmission time. This suggests that cDVD may need to respect latency in allocations for the future.

In Figure 5c, the conservative nature of client requests is shifted to cDVD: cDVD increases bitrates by 10% before determining utility and selecting bitrates for clients. This experiment is otherwise identical to Figure 5b. Re-buffering falls to 0% in most streams and is consistently lower than in Figure 5b. All streams benefit from greater stability, with additional capacity allocated where there is greater utility.

6. DISCUSSION AND OPEN QUESTIONS

Our measurements mirror previous studies, which lend confidence to our results. The observation that the architecture can also be client-driven prompts additional questions.

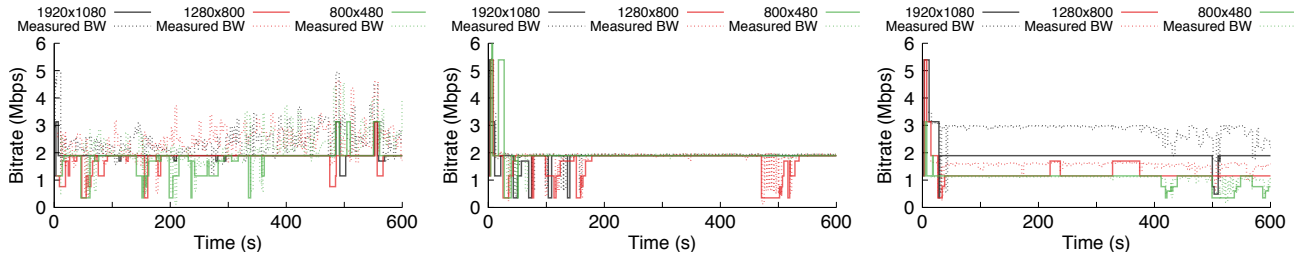
Incentives: The first question is whether incentives align. Incentives may be as simple as reducing encodings when requesting media over expensive mobile data plans. In general clients should be no worse off by participating, and ideally get a better experience. By informing the network of their sessions, clients can get sufficient, and stable, bandwidth allocations. At the other end, a cDVD controller could leave a portion of bandwidth for non-participating clients, and give no guarantees there. Clients should not gain an unfair advantage by lying about their bitrates, utilities, or sessions; cDVD is in a position to monitor and limit the damage a client can cause to others. These are important questions going forward.

Utility: Definitions remain open. In a cDVD-type of system, it is unclear whether all clients must adhere to an identical definition of utility, or if heterogenous definitions might be adopted with similar effect. cDVD also allows for exploration of *partial* utility, wherein protective clients may prefer to withhold details, and submit incomplete utility functions.

Feedback and Conservative Behaviour: Network feedback could add new dimensions to bitrate adaptation. Preliminary measurements in Section 5.2, for example, suggest it is possible to improve bottleneck utilization and client stability.

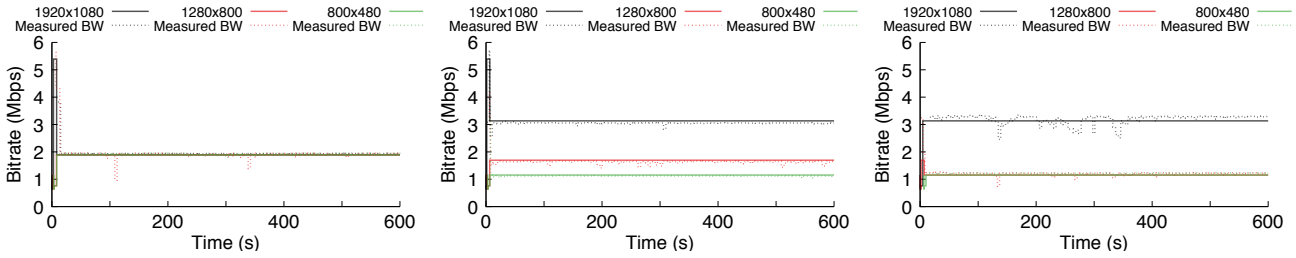
Upstream Bottlenecks: Our solution only controls the bandwidth down-stream from the control point. This may be sufficient in many homes, in practice. In general cases cDVD only provides upper bounds on the bandwidth to a session. Bitrate adaptation is a complement, and necessary fallback.

API Expressiveness and Scale: Alongside utility are questions regarding the API and scalability of a cDVD system. In this paper we used an ad-hoc API for different options between the client and cDVD, which would have to be standardized. Interesting questions arise, for example, if clients report upstream bottlenecks to cDVD and lend that bandwidth to other flows temporarily, until the bottleneck is resolved. Chains of cDVD along a path also warrant exploration.



(a) Flow-based fairness (R: 0, 0.51, 2.27%) (b) Per-session QoS (R: 1.7, 3.11, 1.51%) (c) Per-session QoE (R: 0.29, 0.27, 3.75%)

Figure 4: Transition from flow-rate equality to user-driven QoE (with respective rebuffering ratios R).



(a) Equal allocations (R: 0, 0, 0%) (b) VHS allocations (R: 7.4, 4.17, 2.16%) (c) VHS + 10% headroom (R: 0, 2.15, 0%)

Figure 5: Experimental feedback where cDVD *tells* clients which bitrate to select (with respective rebuffering ratios R).

7. CONCLUDING REMARKS

In this paper we have argued that clients and network must collaborate to achieve QoE *fairness* when streams are encrypted. It is instructive to place cDVD in the greater context of QoE delivery methods. Our approach is dynamic and flexible, satisfying clients with heterogeneous levels of transparency; it is provider-independent, and uniquely positioned to provide QoE fairness in a world where HTTPS is ubiquitous. This paper provides additional motivation for, and further reinforces, the case for centralized control advanced by other works. It also opens more questions pertaining to the best architecture and means of deployment, as well as the right incentives and interfaces.

8. REFERENCES

[1] OpenWrt. http://openwrt.org.

[2] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. Begen. What happens when http adaptive streaming players compete for bandwidth? In *ACM NOSSDAV*, 2012.

[3] S. Akhshabi, A. C. Begen, and C. Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. In *Proc. Multimedia Systems (MMSys)*, 2011.

[4] BBC. Testcard Stream. http://rdmedia.bbc.co.uk, 2015.

[5] B. Briscoe. Flow rate fairness: Dismantling a religion. *ACM SIGCOMM Computer Comm. Review*, 37(2):63–74, Apr. 2007.

[6] dash.js player. http://http://dashif.org/software/.

[7] A. D. Ferguson, A. Guha, C. Liang, R. Fonseca, and S. Krishnamurthi. Participatory networking: An api for application control of sdns. In *Proc. ACM SIGCOMM*, 2013.

[8] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race. Towards network-wide qoe fairness using openflow-assisted adaptive video streaming. In *ACM FHcMN Workshop*, 2013.

[9] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari. Confused, timid, and unstable: picking a video streaming rate is hard. In *Proc. ACM IMC*, New York, NY, USA.

[10] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proc ACM SIGCOMM*, 2014.

[11] J. Jiang, V. Sekar, and H. Zhang. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *ACM CoNEXT*, 2012.

[12] P. Karbhari, E. Zegura, and M. Ammar. Multipoint-to-point session fairness in the internet. In *IEEE Infocom*, March 2003.

[13] R. Kuschnig, I. Kofler, and H. Hellwagner. Evaluation of http-based request-response streams for internet video streaming. In *Proc. ACM Multimedia Systems (MMSys)*, 2011.

[14] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran. Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale. *IEEE J. on Selected Areas in Communications*, 32(4):719–733, 2014.

[15] H. Liu, Y. Wang, Y. Yang, H. Wang, and C. Tian. Optimizing cost and performance for content multihoming. In *ACM SIGCOMM*, 2012.

[16] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A case for a coordinated internet video control plane. In *ACM SIGCOMM*, 2012.

[17] A. Mansy, M. Ammar, J. Chandrashekar, and A. Sheth. Characterizing client behavior of commercial mobile video streaming services. In *Proc. ACM MoVid*, 2014.

[18] A. Mansy, M. Fayed, and M. H. Ammar. Network-layer fairness for adaptive video streams. In *Proc. IFIP Networking*, 2015.

[19] MPEG. DASH. http://dashif.org/mpeg-dash.

[20] M. K. Mukerjee, D. Naylor, J. Jiang, D. Han, S. Seshan, and H. Zhang. Practical, real-time centralized control for cdn-based live video delivery. In *Proceedings of ACM SIGCOMM*, 2015.

[21] D. Naylor, A. Finamore, I. Leontiadis, Y. Grunenberger, M. Mellia, M. Munafò, K. Papagiannaki, and P. Steenkiste. The cost of the "s" in https. In *Proc. ACM CoNext*, 2014.

[22] P. S. Schmidt, T. Enghardt, R. Khalili, and A. Feldmann. Socket intents: Leveraging application awareness for multi-access connectivity. In *Proc. of ACM CoNext*, pages 295–300. ACM, 2013.

[23] S. Shenker. Fundamental design issues for the future internet. *IEEE J. Selected Areas in Communications*, 13(7):1176–1188, 1995.

[24] I. Sodagar. The mpeg-dash standard for multimedia streaming over the internet. *IEEE Multimedia*, 18(4):62–67, 2011.

[25] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman. BOLA: near-optimal bitrate adaptation for online videos. *CoRR*, abs/1601.06748, 2016.