Network-layer Fairness for Adaptive Video Streams

Ahmed Mansy
School of Computer Science
Georgia Institute of Technology
amansy@cc.gatech.edu

Marwan Fayed
Computing Science & Mathematics
University of Stirling
mmf@cs.stir.ac.uk

Mostafa Ammar
School of Computer Science
Georgia Institute of Technology
ammar@cc.gatech.edu

***Abstract*—Recent studies observe that competing adaptive video streaming applications generate flows that lead to instability, under-utilization, and unfairness in bottleneck link sharing within the network. Additional measurements suggest there may also be a negative impact on users' perceived quality of service as a consequence. While it may be intuitive to resolve application-generated issues at the application layer, in this paper we explore the merits of a network layer solution. We are motivated by the observation that traditional network-layer metrics associated with throughput, loss, and delay are inadequate to the task. To bridge this gap we present a network-layer QoS framework for adaptive streaming video fairness that reflect the video user's quality of experience (QoE). We begin first by deriving a new measure to describe user-level fairness among competing flows, one that reflects the dynamics between the video encoding and its mapping to a screen with a given size and resolution. We then design and implement our framework in VHS (VideoHome-Shaper) to evaluate performance in the home's last access hop where this problem is known to exist. Experiments using a variety of devices, O/S platforms, and viewing screens demonstrate the merits of using video QoE as a basis for fair bandwidth sharing.**

I. Introduction

Stability, utilization, and fairness (equal or proportional), have long been cornerstones of the Internet's design. Yet the dominant application on the Internet today, adaptive video streaming (a.k.a. DASH - Dynamic Adaptive Video Streaming over HTTP), has been shown to defy these long-held tenets. Adaptive streaming over HTTP [25] works with segments of video encoded at multiple bitrates. Segments are non-overlapping and of equal length. Clients intermittently request segments from the video server over HTTP. While downloading segments, clients estimate the available bandwidth to the server and switch between video bitrates.

At the server-side, DASH protocols resolve issues of scale, cost, and delivery. Recent studies suggest that server-side gains may be at the expense of the network and, in particular, the end users who are meant to benefit most. The negative effects of video streaming protocols on flow stability, efficiency, and fairness, are increasingly observable in the network and in the home. As the proportion of video traffic grows to 69% in 2017 [5], these behaviours may impact on the quality of delivery service from the content provider to the end user.

The network-level behavior of DASH systems has received significant attention (e.g. [6], [17], [21]). These studies show that network-level analysis of adaptive video streaming behaviors is challenging because segments length, their encoded bitrates, and the algorithms that switch between them, are left to the discretion of the implementation. As a consequence, when multiple video streaming clients (or flows) compete for bandwidth across a bottleneck link, the intermittent downloading and estimation causes (i) instability when switching between encodings, (ii) bottleneck link under-utilization, and (iii) disproportional shares of available bandwidth.

Fair sharing between multiple competing DASH streams, which is the focus of this paper, is particularly problematic: The on/off nature of flows leads to inaccurate client estimations of available bottleneck capacity, and results in potentially unfair demand by any accepted definition. 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 support, and the content provider [6], [21].

We posit the following explanation for the difficulty in controlling resource sharing for adaptive video streaming: *There exists no metric at the network level that reflects the user experience of streaming video at the application layer.* The traditional metrics of throughput, delay, and loss, appropriately describe the quality of short or long continuous flows. For example, the speed with which webpages may be retrieved or the quality of a real-time voice conversation may be predicted (or characterized) by the width, delay, or reliability of a connection. Many application protocols, decidedly separate from network layer protocols, can use these metrics to accurately estimate path resources and quality. By contrast streaming video flows are intermittent, periodic, and exhibit an on/off transfer pattern that is neither short nor long and continuous. The result is a disconnect between network and application in which estimates of available path resources over one interval are no indication of available resources in the next interval.

How, then, can competing streams be expected to behave fairly when there exists no reliable indication of network conditions? One approach is to standardize application-level behavior across the different implementations and commercial services. Standardization of application-layer characteristics violates the spirit of flexibility in the Internet's architecture and, more importantly, is not feasible given the strong competition among the DASH services. A second approach, explored in this paper, is to deploy network-level mechanisms to enforce application-aware fair resource sharing. We will argue that the

This work was supported in part by NSF grant NeTS 1409589.

ISBN 978-3-901882-68-5

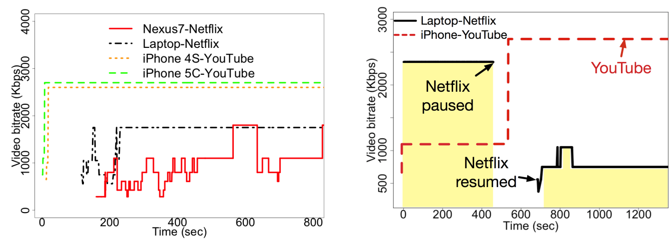
natural response at the network layer, which is to assign equal resources to each competing video, is actually unfair from an application viewpoint.

Alternatively, we take the perspective first suggested in [8] that equal bitrate, or any flow-rate definition of fairness, is ultimately unfair. In the case of streaming video, flow-rate fairness ignores user-level fairness. We would expect, for example, that small screen devices in the presence of contention get video streams of a lower rate of encoding than larger screen devices (including televisions). Since quality is determined by the appearance of the video on the screen, throughput definitions of fairness are inherently inappropriate since they fail to reflect user interactions.

In this paper we explore network-layer QoS for streaming video fairness that allows quality of experience (QoE) metrics. To our knowledge this is the first work to suggest user-level metrics for evaluating fairness in the network layer. We demonstrate the viability of user-level metrics first by introducing a QoE measure that reflects the dynamic between the video encoding and its mapping to screen size and density. Though quality assessment is an active area of research in the video community, we are unaware of any metrics in the literature that take screen size into consideration. Our measure is inspired by the bandwidth *utility* concept introduced in [24] and ensuing work in [9], [19], [23]. The use of a single metric resolves the need for per-video, per-resolution, utility functions stored in a database [13]. We use it to define *QoE max-min fairness* for a set of video streams sharing a network, that matches the *utility* of available bandwidth (and associated video bitrate) to screen size and density. We emphasize, however, that while this metric is grounded in user-perception arguments and correlates well with user experience, our framework can admit other metrics that other researchers may deem more appropriate for their specific purpose.

We then proceed to define DASH QoE max-min sharing of bandwidth on a constrained link. We show that a QoE max-min fair bitrate allocation does not always exist because video sessions have only a discrete set of bitrates that can be assigned. In lieu we define *QoE maximal fairness*, that is (i) easy to compute and (ii) is equal to max-min fair allocation when the latter exists. We then develop an algorithm to compute bitrate allocation that achieves QoE maximal fairness for the home network scenario.

We instantiate our framework by designing VHS (*Video-HomeShaper*). We implement VHS on a home router with OpenWrt firmware [3] and evaluate its performance in the home, the last access hop where this problem is known to exist [21]. VHS monitors outbound HTTP requests capturing those that identify Netflix and YouTube sessions initiated by clients connected to the router. Our video-bitrate fair allocation algorithm is integrated into VHS so that when streaming requests appear, or when active sessions terminate, the algorithm establishes new capacity allocations to reflect user-level fairness. Allocations are enforced via the Linux *traffic control* (tc), and assign adequate bandwidth to each client to obtain the desired video encoding bitrate.



(a) Four clients share a link of 8Mbps (b) Two clients share a link of 4Mbps

Fig. 1: Visualizing video bitrate *unfairness* when competing devices stream video from different streaming services

In our experiments we measure fairness, utilization, and introduce a metric to describe stability. We find that VHS almost always improves fairness and link utilization, even when compared against stochastic fair queuing. In all cases VHS improves stability of each video stream, thereby reducing the negative impact of fluctuating streams on competing sessions.

The rest of the paper is organized as follows. Additional background and motivation are presented in Section II. In Section III we introduce our QoE metric and its use at the network layer. Our VHS framework is presented in Section IV, followed by evaluation in Section V. Finally, before concluding remarks in Section VII, we address wider implementation challenges in Section VI.

II. Background and Motivation

In this section we give a brief background on adaptive HTTP streaming, utility max-min fairness, and video quality assessment metrics, and review some work in each topic.

A. Adaptive HTTP streaming

MPEG-DASH [2] is an ISO standard for adaptive HTTP streaming that defines media segmentation and representation at the server-side. In this scheme, video is split into non-overlapping segments of equal length. Each segment is then encoded at multiple bitrates. A client streams the video by intermittently requesting and downloading segments from the video server over HTTP. While downloading, clients also estimate the available bandwidth to the server to inform switches to other video bitrates.

Excluded from the standard are client implementation and bitrate adaptation techniques. This gives video streaming providers the flexibility to implement their own clients and adaptation mechanisms. It also makes difficult the performance prediction and analysis of multiple streaming clients when they compete for bandwidth. This challenge is further exacerbated when vendors implement their own rate limiting techniques at the video server [14].

Example behaviours are presented in Figure 1. Plots show the video bitrate obtained by a set of clients over time, when all clients are behind the same bottleneck home access link. In Figure 1a four clients, across three platforms, stream video from two services and are constrained by an 8Mbps access link. Measurements spanning more than 13 minutes show that

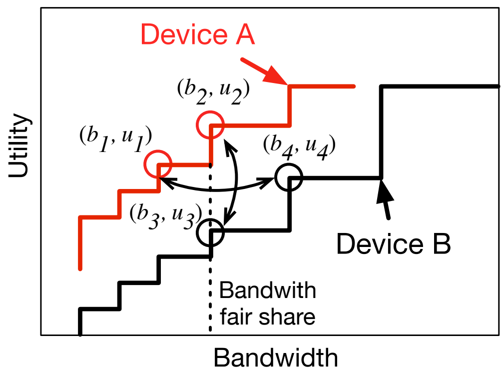


Fig. 2: Adaptive video utility function

iPhone YouTube clients fail to reduce their bitrate when Netflix clients later appear. One explanation may be that latecomers are penalized. This hypothesis is negated by bitrate measurements shown in Figure 1b, in which a Netflix client and a YouTube client are launched at the same time from behind a 4Mpbs access link. Over time, the YouTube client obtains a far greater share of capacity than its Netflix rival. Combined, these plots suggest that YouTube is overly aggressive and should be scaled back. We argue that explanations and solutions based on service-, platform-, and implementation-specific characteristics are misleading. We argue that requested video bitrates are determined by available network resources, making resource management a network-layer issue.

B. Utility max-min fairness

Bandwidth max-min fairness [7] maximizes the minimum bandwidth allocated to any flow, traditionally by aiming for equal share of the bandwidth to all connections bottlenecked links. This goal assumes that all flows in the network carry the same *utility* to the application. As shown above, this is a false assumption among competing adaptive video streams. In our investigation we adopt the position proposed in [24] that ties Internet service models to application *utility*. *Utility max-min fairness* was subsequently developed in [9], [19], [22], [23].

Consider a generic example described by Figure 2 that joins bandwidth and utility for two different devices. The step-nature of video bandwidth-utility functions are a consequence of the discrete sets of available encodings at the server (see sample real-world sets in Table I). Referring back to Figure 2, the vertical dashed line represents a bandwidth-fair allocation where $b_2 = b_3$. Clearly, the associated utility differs greatly as $u_3 << u_2$. By contrast, a non-equal bandwidth allocation can maximize the minimum utility of any single flow. In Figure 2, this *utility fairness* occurs at points (b_1, u_1) and (b_4, u_4). Additional technical details may be found in [20].

C. Video quality assessment

Utility must be described by some measure. And yet digital video usually suffers from a wide variety of distortions during encoding, compression, and reproduction that may result in a degradation of visual quality. To quantify visual quality subjective assessments are most often used. In this method, a video is viewed by a set of independent users. Each user evaluates the perceived video quality with a score from 1 to 5 with 1 being the worst quality. Scores are then averaged to give a Mean Opinion Score (MOS). While critical for evaluating video encoding quality, the cost, time, and human factors render MOS infeasible for use at network layers.

Several objective quality assessment metrics have since appeared. Peak Signal-to-Noise ratio (PSNR) computes the average distortion between a compressed video and its lossless source. However, PSNR is known to have a weak correlation with perceived video quality [12]. Structural Similarity Index (SSIM) [26] is an objective metric that exploits the highly structured nature of images and strong dependence between spatially close pixels. Experiments have shown strong correlation between SSIM and MOS which represents ground truth for perceived video quality [26]. Despite their merits, we note that no existing metrics, pertaining to video nor network performance, naturally lend themselves to evaluating fairness between competing views.

Intuitively, one definition of *video fairness* is to associate the video encoded bitrate with screen size and viewing distance. Surprisingly, and to the best of our knowledge, no such metric exists. In the next section we introduce a new QoE-based metric to tie video bandwidth to the utility of the video bitrate. We then build upon this metric to establish utility max-min fairness for competing video flows.

III. Adaptive video QoE fairness

In this section we introduce a new video QoE metric as a function of the screen size, resolution, and viewing distance. We then introduce the home network model, define QoE *max-min* fairness, and introduce an algorithm to compute the set of fair bitrates. The overall framework is flexible and welcomes other metrics that researchers may deem more appropriate for their specific purpose. Our methodology follows from *max-min* utility allocation for multirate multicast networks [23], and establishes fairness for unicast adaptive video streams.

A. Device-dependent QoE metric

The visual quality of a video on a specific display is determined by a) video encoding algorithm and bitrate, b) the difference between video resolution and the physical screen resolution, and c) viewing distance from the screen. Among lossy video compression algorithms higher bitrates usually mean higher video quality. However this association is true only when the resolution of the display area is equal to the video resolution, i.e. when each pixel in the video maps to a single pixel on the screen. A video resolution that is lower than the physical screen resolution gets scaled up to match. Depending on the difference between the two resolutions, scaling to match will degrade the perceptual quality. As a consequence, the best achievable video quality exists when the video and the physical resolutions are identical. This suggests the use of the ratio between the two resolutions to measure quality degradation.

TABLE I: Different video profiles (resolutions and bitrates in Kbps) provided by Netflix and YouTube

Netflix	Resolution	320×240	384×288	512×384	512×384	640×480	720×480	1280×720	1280×720
	Bitrate	235	375	560	750	1050	1750	2350	3000
YouTube	Resolution	256×144	426×240	426×240	640×360	854×480	1280×720	1920×1080	
	Bitrate	190	260	380	750	1350	2750	5000	

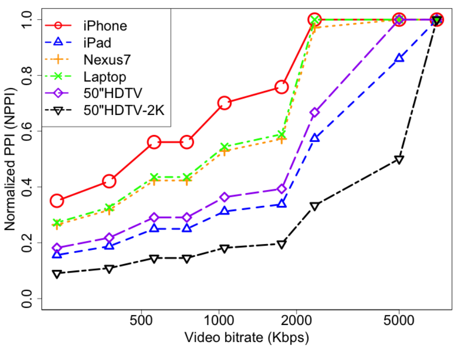


Fig. 3: Normalized PPD for different devices and video profiles

The Pixels-per-degree (PPD) metric captures both screen resolution and viewing distance [18]. PPD is defined as the number of pixels on the base of a triangle with a height of viewing distance and a one degree angle facing the base. PPD is computed as $PPD = d \times PPI \times \tan(\pi/180)$, where d is the viewing distance and PPI is the number of pixels-per-inch. It is defined as the number of pixels on the diagonal of the screen, and computed as $PPI = \frac{\sqrt{w^2+h^2}}{diag}$ where w, h are the width and height of the screen in pixels respectively and $diag$ is the length of the screen diagonal (in inches).

We use PPD to build a degradation metric, *normalized-PPD* (N-PPD). N-PPD expresses the relative quality of a video to the best achievable quality on the device. We define N-PPD as the ratio between video PPD and physical PPD. The video PPD is computed using width and height from the video resolution, while the physical PPD is computed using device parameters. Since there is no additional quality gain from playing a video of a higher resolution than the physical one, the maximum value of N-PPD is 1. Formally, N-PPD is computed as, $\text{N-PPD} = \min\left(\frac{PPI_v}{PPI_{phy}}, 1\right)$.

N-PPD values appear in Figure 3 for a sample set of devices, each with different screen sizes and resolutions. For each device, the set of N-PPD values are computed using video resolutions provided by Netflix [1] that appear in Table I. Figure 3 demonstrates the merit of N-PPD as bitrate increases: Smaller screen devices reach peak values faster with lower bitrates than do larger screen devices.

Finally, to account for the effect of encoding bitrate, we augment our degradation metric with a traditional objective QoE metric, namely SSIM. Studies show SSIM correlates better to MOS than does PSNR [26]. Our QoE metric in its final form is defined as $Q = \text{SSIM} * \text{N-PPD}$. We note that SSIM can be replaced by any other objective QoE metric.

[1] The Tamper Firefox plug-in can read encrypted Netflix manifest files.

B. *QoE max-min fairness*

We now develop QoE max-min fairness for a set of video streams sharing the home link. The more general case is developed in [20]. Consider N video streams sharing a home link of capacity C bits-per-second. Each video is available at the server in multiple bitrates for selection by the player based on player estimation of the available bandwidth. We label $R_{i,j}, j \in \{1, \ldots, M_i\}$ as the j^{th} video bitrate of the i^{th} stream, and assume that $R_{i,1} < R_{i,2} < \ldots < R_{i,M_i}$ for all streams without loss of generality. For each stream $R_{i,j}$, we define $Q_{i,j}$ as a value representing the quality of video as described in Section II-C, and $Q_i = \{Q_{i,j}\}$ as the set of all bitrates of video session i.

A QoE allocation q_i is the QoE value allocated to session i such that $q_i \in \{Q_{i1}, \ldots, Q_{i,M_i}\}$. Define $\psi_i(q_i)$ as the video bitrate of session i with QoE value q_i, for example if $q_i = Q_{i,2}$ then $\psi_i(q_i) = R_{i,2}$. The N dimensional vector $\vec{q} = (q_1, \ldots, q_N)$ is a feasible QoE allocation if each video session is allocated a feasible QoE value, i.e. $q_i \in \{Q_{i1}, \ldots, Q_{i,M_i}\}$, and the total data rate does not exceed home link capacity, i.e., $: \sum_{i=1}^{N} \psi_i(q_i) \leq C$.

A feasible QoE allocation is max-min fair if it is not possible to increase the QoE of one session (i) while maintaining feasibility, and (ii) without reducing QoE of another session that has equal or lower QoE. Since each stream has only a set of discrete QoE values (corresponding to the available bitrates), QoE max-min fair allocations may not exist. Similar to [23], we use an alternative definition, *maximal fairness* (formally defined in [20], that is equal to max-min fairness if max-min fairness exists.

Maximally fair bitrate allocations may be computed in a greedy fashion. Initially, allocate each stream its lowest bitrate. Next, select and upgrade the stream with the lowest QoE value to the next higher bitrate if the new total of allocated bitrates do not exceed the link capacity. Repeat the previous step until no streams can be upgraded to higher bitrates. Additional details and the algorithm used to compute maximally fair bitrates may be found in [20].

IV. **VHS**: QoE FAIRNESS IN A HOME ROUTER

In this section we present our network-layer QoE-fair system, *Video Home Shaper (VHS)*. VHS is designed to be modular so that components may be distributed to accommodate scale. Our implementation focusses on the home environment for several reasons. First, the home is a closed environment where experimental variables may be controlled. Second, the home access link is one of few in the network that sees all competing video flows. Finally, potential gains may have the most immediate effect in the home where access links are often

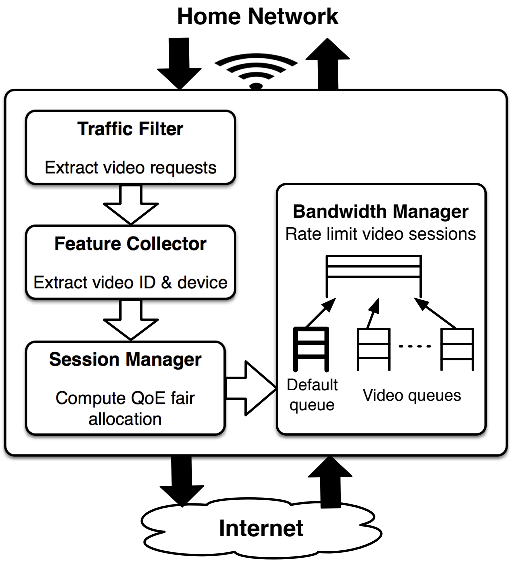


Fig. 4: Inside a home router: VHS design

the bottleneck. As the number of viewing screens increases with the proliferation of viewing devices, contention can only worsen at the access link. Source code is available at [4].

VHS is composed of four main modules, as shown in Figure 4. Each of the traffic filter, feature collector, session manager, and bandwidth manager modules is described below. A discussion of wider challenges for the community that emerged during implementation appears in Section VI.

Traffic Filter. This module identifies and captures HTTP requests of video streaming sessions. It filters out all non-HTTP traffic using the pair `(TCP, port 80)`. HTTP request headers are then matched to pre-defined patterns that represent video streaming services. In this context video requests are either the manifest file or a video/audio media segment. VHS currently matches patterns for Netflix and YouTube on PC, iOS, and Android clients[2]. Patterns for additional streaming services are easily added. When an HTTP video request is identified, it is forwarded next to the *feature collector* module.

Feature Collector. Upon receipt of a new HTTP video request, the feature collector module queries the *session manager* to determine the request as belonging to a new or existing session. The *feature collector* parses requests for new streaming session to identify the device type, video stream identifier, and video profile identifier. This module also parses manifest files (if available) and extracts all video profiles (bitrates and resolutions) within. Parsed information is then forwarded to the *session manager* where QoE-fair allocations are computed.

Session Manager. This module records all active streaming sessions and computes the QoE-fair bandwidth. New session requests are added to the *session table*. The session table is a hash table that stores all information about active video sessions. The key in the hash is the (*client IP, video identifier*) pair, where *client IP* is the local IP address of the client initiating the video request. The *video identifier* identifies all segments of a single streaming session downloaded from multiple sources.Timestamps of the last video request within each session are also recorded. Sessions with idle times that exceed a threshold are deleted (we used 50 seconds). The addition or deletion of a session triggers this modules main function: To compute a new set of QoE fair bitrates and forward them to the *bandwidth manager* module for enforcement.

[2]For each streaming service, HTTP requests differ on different platforms

```
tc qdisc del dev $IFACE root

tc qdisc add dev $IFACE root      handle 1:    htb default 30
tc class add dev $IFACE parent 1:  classid 1:1  htb rate 6000kbit ceil 6000kbit burst 30k

tc class add dev $IFACE parent 1:1 classid 1:2 htb rate 1000kbit ceil 6000kbit burst 30k
tc qdisc add dev $IFACE parent 1:2 pfifo limit 64

tc class add dev $IFACE parent 1:1 classid 1:3 htb rate 2500kbit ceil 3500kbit burst 30k
tc qdisc add dev $IFACE parent 1:3 pfifo limit 64

iptables -t mangle -A POSTROUTING -o $IFACE -j CLASSIFY --set-class 1:2
iptables -t mangle -A POSTROUTING -o $IFACE -s $SERVER_IP -d $CLIENT_IP -j \
        CLASSIFY --set-class 1:3
```

Fig. 5: Example Linux `tc` configuration generated by VHS

Bandwidth Manager. This module enforces QoE fair allocations computed by the *session manager*. The VHS *bandwidth manager* is implemented using Linux traffic control and `iptables` to allocate bandwidth to each video stream. Each stream is allocated a lower and an upper bandwidth value. The lower value is the guaranteed minimum bandwidth allocated for that session (further discussion in Section VI). The upper value represents its maximum allowable bandwidth, which is less than required to switch to the next higher video bitrate.

The bandwidth manager launches and maintains a parent queue that is directly connected to the LAN interfaces, and a child queue that is connected to the parent queue. This starting child queue is the default path for all non-video streaming traffic. For each new streaming session, the bandwidth manager creates a new child queue with assigned upper and lower limits. It then adds a new rule in `iptables` to forward that session's streaming traffic to its associated queue. The appropriate child queue and `iptables` rule are deleted upon notification from the session manager that a streaming session has terminated. An example of the rules generated appears in Figure 5 for reference. This example launches a parent queue `classid 1:1` and two children queues `classid 1:2` and `classid 1:3`. Two new `iptables` rules are then defined: The first marks all packets to be forwarded to the default queue `1:2`, while the second marks all packets destined to `CLIENT_IP` to be forwarded to queue `1:3`.

Implementation Platform. Our VHS is implemented on a NETGEAR home router, model WNDR3700, running OperWrt firmware [3]. VHS is written in C++ and is 1500 lines of code, the bulk of which is devoted to monitoring and logging. Traffic monitoring between inbound and outbound traffic on the bridge interface between the WAN port and the internal ports on the router rely on the popular libpcap [1]. This enables VHS to detect traffic from clients connected to the router through both wired and wireless interfaces.

VHS instantiates our QoE-fair network layer architecture for the home environment. In the next section we evaluate its efficacy using measures of fairness, utilization, and stability.

V. Evaluation

In this section we evaluate VHS and the efficacy of QoE fairness in a real home setting. Space constraints restrict our presentation to four representative experiments. Results are assessed according to well known fairness and utilization metrics. We propose a new metric to assess levels of instability.

A. Metrics

Recall from Section I that multiple video clients, when competing for bandwidth, can suffer from *unfairness*, *inefficiency* (i.e. link under-utilization), and *instability* [6], [17], [21]. Though our main objective is to achieve QoE fairness we must ensure that VHS respects link utilization and bitrate stability. We formally define the three metrics as follows:

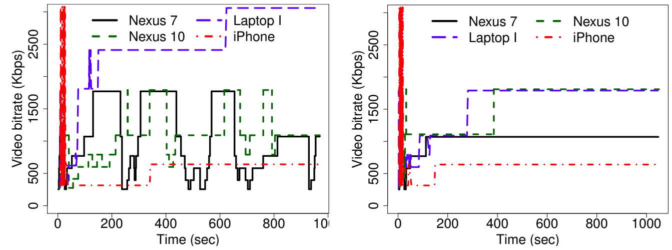
- **Fairness.** Based on Jain fairness [16], we define the QoE fairness index F_{QoE} as

$$F_{QoE} = \frac{\left(\sum_{i=1}^{N} q_i\right)^2}{N \sum_{i=1}^{N} q_i^2} \tag{1}$$

  where q_i is the QoE value of the bitrate allocated to video session i. The fairness index has a value between 0 and 1, with 1 being the best fairness and 0 being the worst.
- **Utilization.** Link utilization $U(t)$ at any time t is assessed by the standard definition as the percentage of the access link capacity C used to stream video regardless of any background traffic. Formally $U(t) = \frac{\sum_{i=1}^{N} r_i(t)}{C}$ where $r_i(t)$ is the bitrate of video stream i at time t.
- **Instability.** We define instability as the rate of video bitrate change among all streams over time. We use the average number of bitrate changes every 100 seconds as a good representative of the user experience. To the best of our knowledge there exist no standard metrics to describe the instability of a session. We propose the following indicator function: $I_i(t) = 1$ if the video bitrate of stream i changes between times $t, t+1$ or $r_i(t+1) \neq r_i(t)$, and $zero$ otherwise. Then for a streaming session of length T seconds, we compute instability as

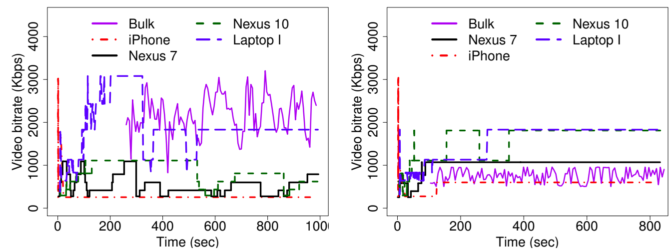
$$instability = \frac{\sum_{i=1}^{N} \sum_{t=0}^{T-1} I_i(t)}{T} \times 100 \tag{2}$$

  Frequent switching of video bitrate is known to hurt user video experience [11]; smaller *instability* values reflect more stable bitrates and hence improved experience.



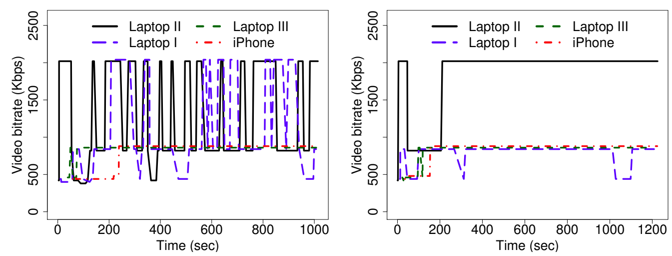
(a) Without VHS (b) With VHS

Fig. 6: Competing Netflix clients



(a) Without VHS (b) With VHS

Fig. 7: Netflix clients vs bulk web transfer.



(a) Original (b) VHS

Fig. 8: Competing YouTube clients.

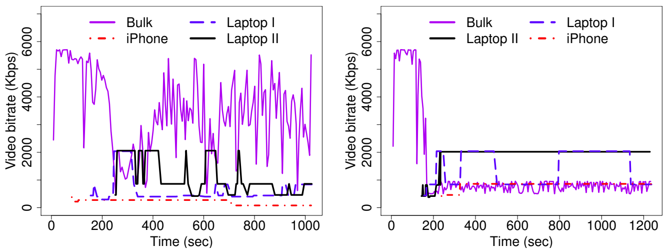
B. Experimental Setup

We conducted experiments using a range of devices, running streaming clients from Netflix and YouTube, over a home bottleneck DSL link. For each of the experiments described, the fairness, utilization, and instability metric are computed every second. Full experimental details are available in [20].

The following four representative experiments have been selected for presentation. It is worthwhile emphasizing that in all of our experiments, *VHS and its underlying QoE-fair mechanisms ensure that smaller screen bitrates never exceed the bitrates of their larger screen counterparts.*

Competing Netflix clients on an iPhone, Nexus 7, Nexus 10, and Laptop I. Figure 6 shows bitrate allocations over time with VHS versus allocations without VHS. We can clearly see in Figure 6b that VHS improves the bitrate stability of the two Nexus tablets; this is an observation that persists through all of our experiments. VHS also enables the iPhone session to achieve a higher bitrate sooner in time.

Netflix clients with background traffic. Background traffic, labeled 'Bulk' in Figure 7, is generated via large file transfer



(a) Original (b) VHS

Fig. 9: YouTube clients vs bulk web transfer.

from a web server using Laptop III. We can see in Figure 7a that the file download consumes about $20 - 30\%$ of the bottleneck link, and appears to impact video bitrates when compared to Figure 6a. On the other hand, Figure 7b shows how VHS limits the file download rate to about 1Mbps, and still manages to allocate fair bitrates to all video streams.

Competing YouTube clients. In YouTube experiments we substitute laptops for Android devices. This is because Android has no direct support for adaptive streaming [21]. Bitrate allocations for the four devices are shown in Figure 8. Again, the erratic bitrates acquired in Figure 8a are made more stable by VHS in Figure 8b.

YouTube clients with background traffic. A bulk file download is injected as in Experiment 2. In a fashion that is even more pronounced than with Netflix clients, the bulk file download shown in Figure 9a consumes between 50% and 65% of the bottleneck bandwidth. By contrast VHS limits the bulk transfer bandwidth consumption through its default child queue; this can be seen in Figure 9b.

C. *Fairness*

We evaluate the overall fairness of bitrate allocations according to Equation 1. The cumulative distribution function of those values is plotted in Figure 10 for values with VHS and without (labeled 'Original'). Fairness provided by stochastic fair queueing (SFQ) is included in Figure 10a. SFQ appears to be marginally more fair than the original system; this is a trend that continues throughout and so is omitted from remaining plots. VHS maintains a fairness index greater than 0.94, exceeding original fairness indices in three experiments.

Interestingly, in our experiments consisting solely of competing YouTube flows (shown in Figure 10c), fairness indices are indistinguishable. This is an artifact of the way in which the fairness index is calculated, i.e. a system can be fair when flows are equally unstable. Referring back to Figure 8a we can see that two devices fortunate to receive their desired bitrates (Laptop III and iPhone), while the two remaining devices repeatedly switch between bitrates. Despite being fair, the instability leads to poor QoE for the user [11]. Our subsequent instability evaluations reveal that VHS resolves this gap.

D. *Utilization*

Utilization measurements for the four experiments appear in Figure 11. We remind our reader that we define utilization as the occupied portion of the bottleneck link capacity allocated to stream video, i.e. the capacity that is isolated from background traffic. By this definition VHS performance is more predictable, and better understood.

For example, Figures 11a and 11c suggest that VHS provides no real improvement in utilisation, and is sometimes slightly worse. This is expected because there is only a discrete set of bitrates for each video stream, and VHS is designed to provide QoE-fair bitrates. The corresponding Figures 6 and 8 show that without VHS, the competitive and intermittent behaviour of clients can cause erratic switching between bitrates. This tradeoff between bitrate instability and link utilization was also observed in previous work [6], [10].

In the presence of background traffic VHS improves utilization. In Figure 11d, for example, VHS streams consume 80% of the available bandwidth for over 50% of the time, compared to less than 40% consumption of the capacity for 70% of the time under the original system. This is because VHS isolates the capacity allocated to video streams from other background traffic (as described under 'Bandwidth Manager' in Section IV). Specifically, the intermittent TCP video streams are protected from being forced to back-off by the bulk transfer. Residual capacity, in our experiments 1Mbps, remains for the bulk transfer. The increase in utilization is a direct consequence of increases in stability that we present next.

E. *Stability*

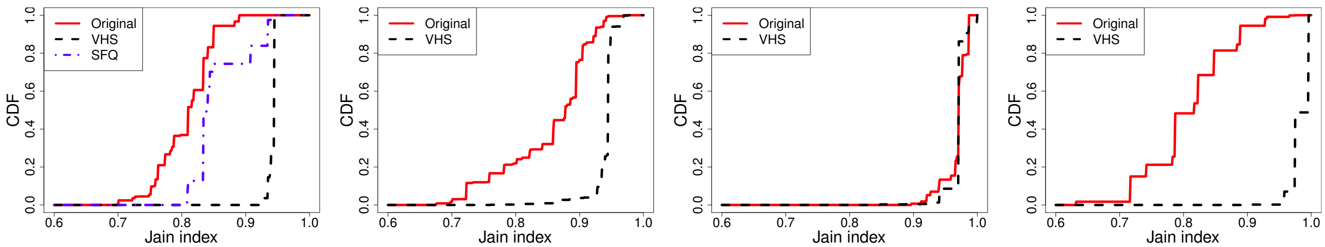
Frequent switching of video bitrates is known to hurt users' video experience [11]. In Figure 12 we plot values derived using our own *instability* metric in Equation 2; smaller values reflect greater stability, thereby contributing to user experience. In Figure 12a SFQ provides stability levels between the Original and VHS environments, albeit providing only marginal increases in fairness (as shown in corresponding Figure 10a).

In all our experiments VHS improves stability, in the best cases by factors of 6. In addition, we find that the values represented by our metric in Figure 12 are reflective of the bitrate measurements over time in corresponding Figures 6-9.

Interestingly the relationship between instability, fairness, and utilization, is less clear. Consider, for example, the corresponding plots in Figures 10, 11, and 12: Overall, VHS appears to provide improved and consistent levels of performance. However, knowing any one or two values of fairness index, utilization, and instability, seems in no way to indicate remaining values. We suggest this is an artefact of the metrics themselves, and their inability to represent quality of experience when viewed in isolation.

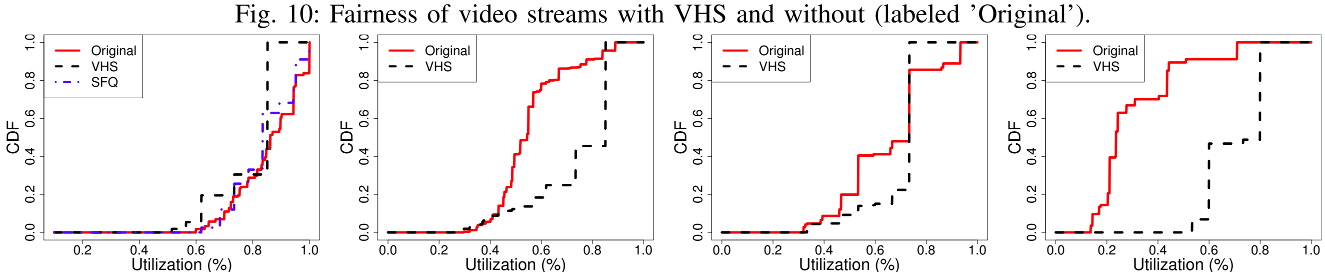
VI. Wider Deployment Challenges

In the home VHS overcomes a wider set of deployment challenges that emerged during implementation. Any similar real system, irrespective of location, needs to obtain details about the video source and the viewing device. In lieu of absent standards and alternatives, our architecture relies on information contained in HTTP requests. This, in addition to other potential challenges, are discussed in further detail below.



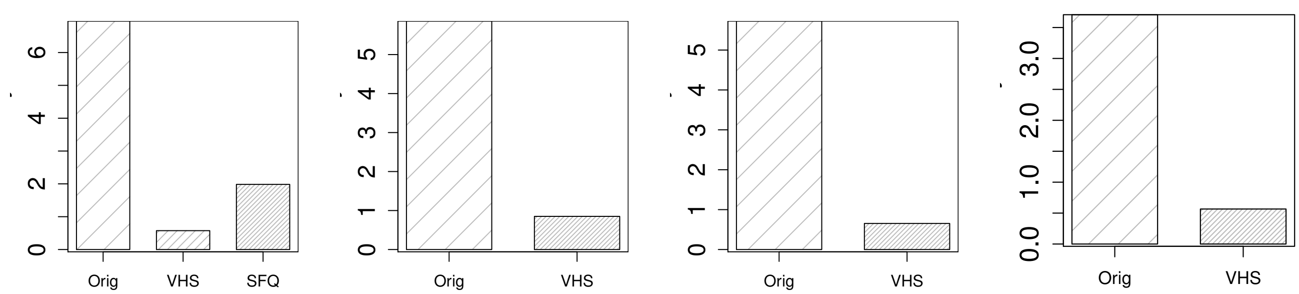
(a) Netflix clients (b) Netflix Clients vs. Bulk (c) YouTube Clients (d) YouTube Clients vs. Bulk

Fig. 10: Fairness of video streams with VHS and without (labeled 'Original').



(a) Experiment 1 (b) Experiment 2 (c) Experiment 3 (d) Experiment 4

Fig. 11: Utilisation of access link with VHS and without (labeled 'Original').



(a) Experiment 1 (b) Experiment 2 (c) Experiment 3 (d) Experiment 4

Fig. 12: Instability of video streams with VHS and without (labeled 'Original'). Lower values are better.

A. *HTTP Traffic Monitoring*

A network layer QoE fair system must keep track of all active video streaming sessions: (i) the device, screen, or PPI in use, and (ii) the video manifest file that lists the content provider's set of available bitrates and resolutions. This information is embedded in HTTP packets that can be inspected for adaptive video stream requests. Our VHS implementation demonstrates feasibility by identifying video flows for most popular streaming services (e.g. Netflix and YouTube) from their HTTP request format. HTTP traffic monitoring presents several challenges both for users and network operators. We proceed by addressing the technical concerns, and leave non-technical challenges related to privacy, etc., to better hands.

Deep packet inspection (DPI) is required on all HTTP traffic. Use of DPI may raise two concerns. First, DPI is known to be expensive and requires special hardware to handle large volumes of data at line speed. In the home, our implementation demonstrates DPI is technically feasible for a home router. Since only HTTP requests need to be inspected, candidate packets are easily identified by the `(TCP, dstport 80)` pair. HTTP request packets can be further distinguished from `ACKs` by filtering based on a packet length condition. The second concern may pertain to user privacy. A home router solution keeps all data within the home. Though no less relevant, third party monitoring in the wider network presents challenges that are necessarily beyond the scope of this work.

Access to manifest files is a crucial component of the system. As a session begins, a video client downloads a manifest file that includes a list of sources, target IP addresses and URLs for video segments, their resolutions, and bitrates. Some commercial services encrypt their manifests, while other services do not. We argue that MPEG-DASH [2] use of unencrypted XML manifest files should be standard in the future. For services that choose to encrypt manifest files we propose an unencrypted portion containing video bitrates and resolutions in plain view.

Screen Granularity Data establishes the viewing screen PPI, required to compute our QoE metric, N-PPI. The sets of video encodings and resolutions are available via the manifest. However, information about the screen, at the granularity required, are currently limited to inference. Specifically, an HTTP request contains the viewing platform, but omits screen size and resolution. For example, it is easy to identify an iPhone or an iPad but not the version or size of the device. Also, it is possible to identify a set-top-box but not the size and resolution of the TV connected to it. Thus, we propose that video players include PPI or other appropriate information about the streaming device in the HTTP request headers.

B. Enforcing Video bitrates via Bandwidth

Adequate bandwidth must be allocated to each session that could consist of multiple flows, to constrain clients to stream at their designated bitrate. In practice, an over-allocation is necessary to compensate for clients that under-estimate available bandwidth.

Under-estimation is a wide-spread practice. Consider that a video player requests a bitrate based on its estimate of the available bandwidth. Available bandwidth is computed as a function of the download rate of one or more of the recently downloaded video segments[3]. Unfortunately there is no standard specification,i.e. each player implements its own estimation method. Clients are known to request or switch to bitrate r only if an estimate of the available bandwidth is no less than $(1+\alpha)r$ with unknown and proprietary $\alpha > 0$.

To complicate matters, recent work in [21] observes different α values for the same streaming service on different platforms. We envision three potential solutions.

1) Inference of α via experimentation and measurement for all streaming services on all popular devices. This is currently a tractable exercise given the fixed set of bitrates (and resolutions) for most of streaming services. In the longer-term inference will become a challenge as α values change, or new services and devices emerge.
2) A control channel at the video player that can be used by the network controller to communicate the desired bitrate. The client will be expected to honor the network decision and commit to streaming that bitrate unless there is notification of a new bitrate.
3) Agreement among content providers and standardization of an α value, or set of values.

VHS demonstrates that these challenges are surmountable in the home. Solutions on wider scales remain as open problems.

VII. Conclusion

In this paper we explored network layer quality of service to enable fair sharing among competing dynamic adaptive video streams that reflects users' quality of experience.

We subscribe to the notion that, for the dominant adaptive video streaming applications, any flow-rate definition of fairness ignores application-layer fairness. Among streaming video, service based on measures of throughput, error, and latency are inherently unfair since they fail to reflect the quality of the user experience on the viewing interface. We bridged this gap by proposing a QoE measure that can be used to guide bandwidth allocation among competing video streams. Our measure takes an industry standard metric that encapsulates viewing screen attributes. It then normalizes this metric to reflect the relative quality of a video encoding on that screen.

[3]Bitrate adaptation based of the growth rate of the player buffer was suggested in [15], however, bandwidth estimation remains dominant

We used this metric to define and establish network-layer QoE fairness. We then designed and built VHS on a home router to evaluate QoE fairness. Experiments using a variety of devices, O/S platforms, and viewing screens demonstrate the merits of tying QoS to QoE. In all cases, utilization and fairness indices of the bottleneck link under QoE allocations most often improved, and were unchanged in the worst case. As an added bonus measurements revealed reduced instability across all flows as a result of our QoE-fair bandwidth sharing.

References

[1] Libpcap. http://www.tcpdump.org.
[2] MPEG-DASH. http://dashif.org/mpeg-dash.
[3] OpenWrt. http://openwrt.org.
[4] VHS source code. http://www.cc.gatech.edu/grads/a/amansy/.
[5] Cisco VNI: Forecast and methodology, 2012–2017, May 2013.
[6] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. Begen. What happens when http adaptive streaming players compete for bandwidth? NOSSDAV '12, pages 89–94. ACM, 2012.
[7] D. P. Bertsekas and R. G. Gallager. *Data Networks*. Prentice Hall, 1992.
[8] B. Briscoe. Flow rate fairness: Dismantling a religion. *ACM SIGCOMM Computer Communication Review*, 37(2):63–74, Apr. 2007.
[9] Z. Cao and E. Zegura. Utility max-min: an application-oriented bandwidth allocation scheme. In *INFOCOM '99*, pages 793–801, 1999.
[10] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang. A scheduling framework for adaptive video delivery over cellular networks. MobiCom '13, pages 389–400, 2013.
[11] N. Cranley, P. Perry, and L. Murphy. User Perception of Adapting Video Quality. *International Journal of Human-Computer Studies*, 64(8), 2006.
[12] A. Eskicioglu and P. Fisher. Image quality measures and their performance. *IEEE Trans. on Communications*, 43(12):2959–2965, 1995.
[13] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race. Towards network-wide qoe fairness using openflow-assisted adaptive video streaming. ACM FHcMN Workshop, 2013.
[14] M. Ghobadi, Y. Cheng, A. Jain, and M. Mathis. Trickle: Rate Limiting YouTube Video Streaming. In *Proceedings of the 2012 USENIX Conference on Annual Technical Conference*, 2012.
[15] T.-Y. Huang, R. Johari, and N. McKeown. Downton abbey without the hiccups: Buffer-based rate adaptation for http video streaming. In *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking*, FhMN '13, pages 9–14, 2013.
[16] R. Jain, M. Chiu, and W. Hawe. A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems. *ACM Transactions on Computer Systems*, 1984.
[17] J. Jiang, V. Sekar, and H. Zhang. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. ACM CoNEXT, 2012.
[18] H. Knoche, J. D. McCarthy, and M. A. Sasse. Can small be beautiful?: Assessing image resolution requirements for mobile tv. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*, MULTIMEDIA '05, pages 829–838, New York, NY, USA, 2005. ACM.
[19] C. Liu, L. Shi, and B. Liu. Utility-based bandwidth allocation for triple-play services. In *Fourth European Conference on Universal Multiservice Networks, 2007. ECUMN '07*, pages 327–336, Feb 2007.
[20] A. Mansy. *Network and End-host support for HTTP Adaptive Video Streaming*. PhD thesis, Georgia Institute of Technology, 2014.
[21] A. Mansy, M. Ammar, J. Chandrashekar, and A. Sheth. Characterizing client behavior of commercial mobile video streaming services. In *MoVid*, 2014.
[22] S. Sarkar and L. Tassiulas. Fair allocation of discrete bandwidth layers in multicast networks. In *INFOCOM*, pages 1491–1500, 2000.
[23] S. Sarkar and L. Tassiulas. Fair allocation of utilities in multirate multicast networks: a framework for unifying diverse fairness objectives. *IEEE Transactions on Automatic Control*, 47(6):931–944, Jun 2002.
[24] S. Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communications*, 13(7):1176–1188, 1995.
[25] I. Sodagar. The mpeg-dash standard for multimedia streaming over the internet. *IEEE Multimedia*, 18(4):62–67, 2011.
[26] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.