# QoE-fair Resource Allocation for DASH Video Delivery Systems

Luca De Cicco
luca.decicco@poliba.it
Politecnico di Bari
Bari, Italy

Gioacchino Manfredi
manfredi@uniecampus.it
Università e-Campus
Novedrate, Italy

Saverio Mascolo
mascolo@poliba.it
Politecnico di Bari
Bari, Italy

Vittorio Palmisano
vittorio.palmisano@poliba.it
Politecnico di Bari
Bari, Italy

## ABSTRACT

Services delivering videos to massive audiences are required to provide the users with a satisfactory Quality of Experience (QoE) to keep high engagement and avoid service abandonment. Adaptive BitRate algorithms (ABR) running in video players are designed to dynamically change the video bitrate to provide the best possible QoE given the user device features and the end-to-end network available bandwidth. Well-designed ABR algorithms strive to improve the individual QoE obtained by each user resulting, in the optimal case, in the maximization of the sum of QoE individually perceived by users. However, when resources are scarce, maximizing the sum of the QoE might result in favoring some clients at the expense of others which instead obtain poor QoEs with the possible consequence of service abandonment. Thus, we argue that video service providers should directly address fairness issues when designing their delivery networks so to gracefully degrade the QoE for all users when resources are scarce. This paper addresses this open issue and shows that the *Multi-Commodity Flow Problem* (MCFP) optimization framework is a proper methodology to achieve a QoE-fair distribution of the resources. The proposed solution is based on the *bandwidth reservation* approach that slices network resources and assigns *similar* video requests to the same network slice according to a proposed similarity metric dependent on video quality. Obtained results show that the proposed approach is able to achieve its goal and provide a fair level of QoE to heterogeneous clients.

## CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **Computing methodologies** → *Control methods.*

## KEYWORDS

Video delivery systems; Fairness; Quality of Experience; Traffic Engineering; Multi Commodity Flow Problem.

**ACM Reference Format:**
Luca De Cicco, Gioacchino Manfredi, Saverio Mascolo, and Vittorio Palmisano. 2019. QoE-fair Resource Allocation for DASH Video Delivery Systems. In *1st International Workshop on Fairness, Accountability, and Transparency in MultiMedia (FAT/MM '19), October 25, 2019, Nice, France.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3347447.3356753


*FAT/MM '19, October 25, 2019, Nice, France*

ACM ISBN 978-1-4503-6915-2/19/10…$15.00
https://doi.org/10.1145/3347447.3356753

## 1 INTRODUCTION AND BACKGROUND

The fraction of Internet traffic due to video content is steadily increasing and today accounts for more than half of the global traffic [6]. This phenomenon is mainly driven by a shift in the way users consume multimedia contents preferring Internet based video services (Netflix, Amazon Prime Video, Hulu, etc.) to classical TV broadcast channels. The architectural design choice that has made possible such a transition is the use of the standard HTTP protocol to deliver videos from servers to any video player supporting HTTP. Horizontal scalability is guaranteed by using Content Delivery Networks (CDNs) replicating video contents through surrogate servers which finally deliver the content to the user.

Video services are required to design their delivery systems to provide the users with the best possible Quality of Experience (QoE) in order to keep high engagement and avoid service abandonment. This problem is today addressed using a decoupled approach: the delivery network is properly designed and sized to guarantee that Quality of Service (QoS) related parameters such as end-to-end network bandwidth, packet losses, and network latency meet specific minimum requirements; video players run Adaptive BitRate algorithms (ABR) designed to dynamically select the video bitrate (and video resolution) from a discrete set $\mathscr{L}$ to provide the best possible QoE given the user device features and the end-to-end network bandwidth measured by the client and provided by the delivery network.

As a matter of fact, ABR algorithms are typically designed to improve the individual QoE obtained by each user resulting, in the optimal case, to a situation in which the sum of QoE individually perceived by users is maximized. However, when delivery network resources are scarce due to high load, maximizing the sum of the QoE might result in favoring some clients at the expense of others which instead obtain poor QoEs with the possible consequence of service abandonment. This is due to the fact that HTTP-based delivery networks operate a QoS-fair network bandwidth distribution among the flows, i.e. all users sharing a bottleneck are assigned with the same network bandwidth share. However, users with high resolution devices (f.i., Smart TVs) have larger requirements in terms of video bitrate compared to devices with small screens (smartphones) that typically require a lower video bitrate to obtain a satisfactory visual quality.

To make an example, consider Figure 1 that shows the measured visual quality as a function of the encoding video bitrate obtained by clients with different screen resolutions. Let us suppose that three concurrent users request the same video using clients with different screen resolutions (namely 720p, 1080p, and 2160p) and that the video flows share the same bottleneck link having a bandwidth equal to 6 Mbps. In such a case, the fair network bandwidth share obtained by each video flow is equal to 2 Mbps. As a result,the visual quality obtained by the 720p, 1080p, and 2160p clients would be

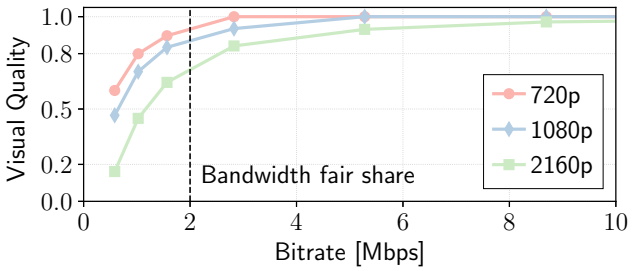



**Figure 1: Visual quality function of the video bitrate and the client screen resolution**

respectively equal to 0.9, 0.85, 0.7. We can conclude that small screen devices will enjoy a better visual quality with respect to large screen devices when provided with the same network bandwidth share. In other words, current video delivery networks cannot provide a *fair* level of QoE to users.

This paper particularly addresses this issue and advocates that video service providers should design their delivery networks to provide the best trade-off between the average obtained QoE and the QoE fairness. To reach this goal, it is necessary to implement a strategy such that video flows sharing the same bottleneck are assigned with a differentiated network bandwidth guaranteeing the same video quality when network resources are scarce.

If, on one hand, the problem of designing QoE-aware ABR algorithms has been explored extensively in the literature [1, 7, 9, 11, 20, 28], QoE-fair delivery of videos has been addressed only recently in a few papers [2, 8, 10, 14, 17, 18], all advocating the need of a *Video Control Plane* (VCP) to allow cooperation between clients and the delivery network. In [14] authors address for the first time the problem of delivering a fair level of QoE to users. A unique shared bottleneck managed by a Software Defined Networking (SDN) switch is sliced programmatically. Each video session is assigned to one network slice whose size is obtained by solving a max-min fairness problem [3]. Recently, the MPEG-DASH community has proposed the *Server And Network Assisted DASH* (SAND DASH), which introduces the DASH Assisting Network Elements (DANEs) providing primitives to drive DASH video clients and suggest them the suitable video bitrate to select [17, 18, 24]. In [10] authors design and systematically analyze the performance of an SDN-based VCP managing a single bottleneck. The paper shows both the impact on performance due to different ABR algorithms at the clients and the use of two different allocation strategies: 1) the *network slicing* (or *bandwidth reservation*) strategy which assigns video flows to network slices whose size is determined by solving an optimization problem; 2) the *bitrate guidance* case which employs DANEs to guide video clients in the choice of the video level. The paper has shown that the bandwidth reservation strategy provides better results in terms of achievable video fairness.

This paper addresses the problem of designing a QoE-fair optimal resource allocation strategy on a generic distribution network using traffic engineering techniques based on network slicing. This work makes the key contributions described in the following. First, we consider the case of a generic distribution network instead of focusing only on the single bottleneck case as studied in [10, 14]. This is particularly important from the point of view of video platforms owning the distribution network (f.i., Google YouTube, Comcast). In fact, the possibility of programming each network element allows those platforms to reach near 100% resource utilization without degrading performances as shown in the seminal paper by Google [16]. Secondly, we show that the *Multi-Commodity Flow Problem* (MCFP) optimization framework [25] is a proper methodology to enforce a QoE-fair distribution of network resources. In particular, we first show how to cast our QoE-fair resource allocation problem to an MCFP (Section 3) and then we propose a traffic clustering approach to sensibly reduce the number of network slices and make the resulting problem tractable for video distribution platforms serving a massive audience (Section 4). Such clustering approach assigns video sessions based on a proposed similarity metric which is dependent on video quality. Finally, we conduct a simulation study to assess the sensitivity of the performances of the proposed resource allocation strategy with respect to the total load on the delivery network and the number of clusters (Section 5).

## 2 BACKGROUND ON MCFP

In this section we briefly describe the *multi-commodity flow problem* (MCFP) optimization framework using the terminology adopted in [25]. In such a framework, the term *commodity* refers to the tuple composed of a source node, a destination node, and a volume, i.e. the required resources to satisfy the commodity. In our case, a commodity identifies a video session where the source node is the video server, the destination node is the client, whereas the volume represents the video bitrate required to obtain the maximum video quality.

In general, the objective of MCFP is to assign network resources such that commodities are satisfied simultaneously in an optimal way through the maximization of a properly defined utility function under a set of constraints.

In the following we describe the MCFP using the *link-path formulation* [25]. The delivery network is modeled as a capacitated graph $G = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{n_1, n_2, \ldots, n_N\}$ is the *node* set and $\mathcal{E} = \{e_1, e_2, \ldots, e_E\}$ is the *edge* set. Each edge $e \in \mathcal{E}$, or link, is defined as a node pair and is assigned with a bandwidth capacity $c_e$. The set of *demands* is denoted with $\mathcal{D} = \{1, 2, \ldots, D\}$, where each demand $d \in \mathcal{D}$ is a source-destination node pair characterized by a *traffic volume* $H_d$. The traffic volume is defined as the required bandwidth for that demand. Moreover, each demand $d$ is provided with a set of admissible paths $\mathcal{P}_d$ composed of selected paths from source to destination node belonging to graph $G$. The demand volume $H_d$ is realized using paths from $\mathcal{P}_d$ by means of *path flows* $x_{dp}$ ($p \in \mathcal{P}_d$) which are the variables to be optimized by the MCFP. Finally, $\delta_{edp}$ denotes the link-path indicator, which is set to 1 if path $p$ for demand $d$ uses link $e$, 0 otherwise. Notice that in the link-path formulation, all the admissible paths $\mathcal{P}_d$ are pre-computed by finding the shortest paths from demand source to demand destination according to a defined cost.

In this paper nodes represent network switches, whereas edges identify links connecting a couple of switches[1]. Each link is divided in *bandwidth slices* of an appropriate size, whose number depends on the demands in the network. Bandwidth slices size are computed

[1]In the following, we will refer to nodes and SDN switches interchangeably as well as edges with links.

by solving a multi-path weighted $\alpha$-fairness optimization problem which employs the following utility function [21]:

$$U(\boldsymbol{X}) = \sum_d w_d \frac{X_d^{1-\alpha}}{1-\alpha} \tag{1}$$

where $\boldsymbol{X} = [X_1, X_2, \ldots, X_D]^T$ is the vector of the total bandwidths (or total flow) $X_d = \sum_p x_{dp}$ allocated to each demand $d$ and $w_d$ is a *weight* associated to the demand $d$. It has been shown that the maximization of (1) provides a balance between link utilization (which is related to the solution *efficiency*) and fairness by varying the scalar parameter $\alpha$ in the interval $[0, +\infty]$ [21]. In particular, if $\alpha = 0$ the link utilization is maximized disregarding the fairness among flows, whereas if $\alpha \to +\infty$, the flow assignment becomes *max-min* fair, i.e., the assignment allocates resources such that the flow obtaining the minimum rate is maximized. The setting $\alpha = 1$ results in the *Proportional Fairness* (PF) optimization problem [22], which provides a good balance between fairness and link utilization. For this reason, in this paper we explore the proportional fair case ($\alpha = 1$) and leave to future studies a performance comparison for different values of $\alpha$. In the PF case it is straightforward to show that (1) tends to $U(\boldsymbol{X}) = \sum_d w_d \log X_d$. We are now ready to present the optimization problem that we aim to solve in this paper.

Problem 1. *MCFP multi-path weighted proportional fair optimization problem:*

$$\text{Maximize} \sum_d w_d \log X_d \tag{2}$$

$$\text{s.t.} \sum_p x_{dp} = X_d \tag{3}$$

$$\sum_d \sum_p \delta_{edp} x_{dp} \le c_e, \ \forall e \in \mathscr{E} \tag{4}$$

$$X_d \le H_d \tag{5}$$

The constraints (4) are imposed to respect the capacity of the link $c_e$, i.e. the sum of all the path flows $x_{dp}$ insisting on the link $e$ should not exceed the capacity of that link. Constraint (5) ensures that the total bandwidth $X_d$ allocated for demand $d$ is bounded by the demand traffic estimation given by $H_d$.

## 3 THE PROPOSED VIDEO CONTROL PLANE

In this section, we describe the proposed Video Control Plane (VCP) to provide concurrent users consuming videos through heterogeneous devices with a fair level of QoE. To the purpose, we design the VCP to partition the delivery network links into a number of *slices* whose bandwidth values are determined by solving the MCFP (Problem 1 in Section 2) so that QoE is equalized among users.

In the following, we provide all the necessary information to specialize a generic MCFP to our particular case. Concerning the objective function we show how to properly compute the demand weights $w_d$ such that maximizing (2) corresponds to guaranteeing QoE-fair, rather than a throughput-fair, allocation of resources. To the purpose, we propose a procedure to compute demand weights based on the estimation of a mapping between the bitrate and the obtainable visual quality (Section 3.2).

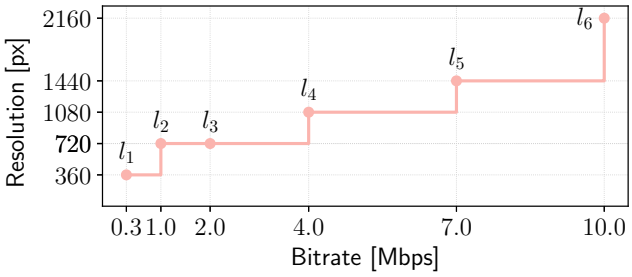



**Figure 2: Video level set representation in a *ladder* graph**

### 3.1 Definitions

**Video level set, reference level, video request.** According to the DASH standard, each video $v \in \mathscr{V} = \{v_1, \ldots, v_V\}$ in the *video catalog* is encoded into different representations or *levels* $l \in \mathscr{L}_v$ characterized by the couple $l = (b, r)$ where $b \in \mathscr{B}_v$ is the encoding bitrate and $r \in \mathscr{R}_v$ is the video resolution. Notice that different videos can have a significantly different set of encoding bitrate depending on the video content.

A *video request* $t$ is identified by the couple $(v, c)$, where $v \in \mathscr{V}$ and $c$ is the *user class* belonging to the set $\mathscr{C} = \{c_1, c_2, \ldots, c_C\}$. In this paper, users are classified based on their screen resolution, since this parameter has a key impact on obtainable QoE as discussed in the introduction (see Figure 1). Thus, in the following the terms "user class" and "user screen resolution" are used interchangeably. Notice that, with this notation, a video request $t$ denotes which video $v$ a user having a client resolution $c$ is willing to consume.

For each video request $t = (v, c)$ we define the set $\mathscr{L}_t$ containing all the levels of $\mathscr{L}_v$ such that their resolution is less than $c$, i.e. $\mathscr{L}_t = \{l \in \mathscr{L}_v : r \le c\}$. It is worth to stress that for a video request $t$ the ABR algorithm will choose video levels belonging to $\mathscr{L}_t$ since we are making the realistic assumption that a client having a resolution $c$ does not request video levels with a resolution higher than $c$.

Next, we define the *reference level* $\bar{l}_t = (\bar{b}_t, c) \in \mathscr{L}_t$ as the representation with resolution $c$ having the maximum bitrate $\bar{b}_t$. To make a concrete example, consider a 4K video being encoded into 6 video levels as shown in Figure 2 where each marker represents a video level $l = (b, r) \in \mathscr{L}_v$. Let us consider a video request $t = (v, 720\text{p})$. In such a case, it turns out that $\mathscr{L}_t = \{l_1, l_2, l_3\}$ and that the reference level $\bar{l}_t$ is equal to $l_3 = (2\,\text{Mbps}, 720\text{p})$.

**Video session.** A *video session* is defined as the tuple (src, dst, $t$) where: 1) src $\in \mathscr{N}$ is the switch the server delivering the requested video is connected to; 2) dst $\in \mathscr{N}$ is the switch the client is connected to; 3) $t = (v, c)$ is the video request.

**Demand and demand volume.** The *demand* $d$ is the aggregate of the $n_d$ video sessions characterized by the same tuple (src, dst, $t$). Consequently, the *demand volume* $H_d$ is equal to:

$$H_d = n_d \bar{b}_t \tag{6}$$

where $\bar{b}_t$ is the bitrate of the reference level $\bar{l}_t$ defined above. In other words, $H_d$ is the minimum amount of network bandwidth required to ensure that all video sessions belonging to the demand $d$ are served with a bandwidth share equal to the reference level $\bar{l}_t$.

**Algorithm 1** Visual quality measurement for a video $v \in \mathscr{V}$

1: **for** each client class $c \in \mathscr{C}$ **do**
2: $t \leftarrow (v, c)$
3: Select reference level $\bar{l}_t$ from $\mathscr{L}_t$
4: **for** each $l \in \mathscr{L}_v$ **do**
5: **if** $l \in \mathscr{L}_t$ **then**
6: $\tilde{l} \leftarrow$ Upscale $l$ to $c$ resolution
7: $Q_t(l) \leftarrow \text{FRVQ}(\tilde{l}, \bar{l}_t)$
8: **else**
9: $Q_t(l) \leftarrow 1$
10: **end if**
11: **end for**
12: **end for**

We expect that, if the constraint (5) is strictly verified ($X_d = H_d$), all the video flows of this demand will obtain the maximum visual quality.

**Link-path indicator** $\delta_{edp}$. As defined in Section 2, $\delta_{edp}$ is a binary variable that is equal to 1 if the path $p$ used to realize the demand $d$ uses link $e$. In practice, $\delta_{edp}$ is set based on the admissible path set $\mathscr{P}_d$ which, in turn, depends on the delivery network topology $G$.

## 3.2 Measuring the visual quality

Since the proposed VCP aims at allocating network resources to obtain a fair level of QoE among users, we need to define a mapping between the network bandwidth allocated to a video session and the achieved QoE [4, 5, 13, 26]. Such a mapping will be employed to define proper demand weights $w_d$ such that the bandwidth allocation resulting by solving Problem 1 is aware of the visual quality obtainable by users. Notice that the procedure described in the following should be performed off-line each time a video is added to the catalog. At the end of this procedure, we will obtain a number of mappings equal to the number of defined user classes for each video. The resulting mappings will be associated to the corresponding video as a metadata.

Algorithm 1 describes the procedure employed to assess the visual quality for a video $v \in \mathscr{V}$. In a nutshell, given a video $v \in \mathscr{V}$ the goal is to compute, for each $l \in \mathscr{L}_v$ and user class $c \in \mathscr{C}$, the mapping $Q_t : \mathscr{L}_v \mapsto [0, 1]$ expressing a relationship between the video level and the obtainable visual quality when playing the video on a device with a resolution $c$.[2]

The output of this procedure for a specific video is shown in Figure 1 in the case of a client class set $\mathscr{C} = \{720\text{p}, 1080\text{p}, 2160\text{p}\}$ with a level set composed of 7 elements. In particular, fixed the video $v$ and the client class $c$ (Line 2) we compute $Q_t(l)$ for each $l \in \mathscr{L}_v$ as follows (Lines 4–11). First, we select the reference video level $\bar{l}_t$ from the set $\mathscr{L}_t$ as described in Section 3.1. Then, for each video level $l \in \mathscr{L}_v$ the video quality is computed using a full-reference video quality assessment tool such as, f.i., the Structural SIMilarity (SSIM) [27], the Peak Signal to Noise Ratio (PSNR), or the Video Multi-method Assessment Fusion (VMAF) [19]. We assume that these metrics are normalized in the range $[0, 1]$. These tools estimate the visual quality by comparing each frame of a *degraded* video with the *reference* frames of the non-degraded video. This operation

[2]Recall that $t = (v, c)$ denotes the video request.

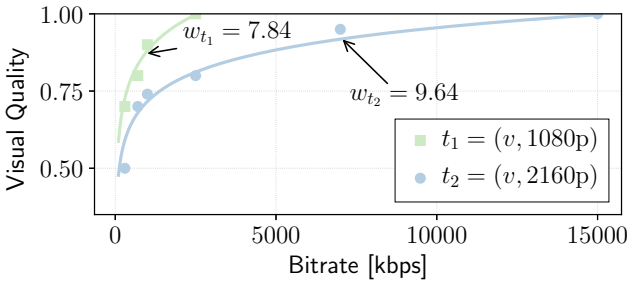



**Figure 3: Computation of weights**

is performed in Lines 6–7. The video level $l$ is first up-scaled to the reference video level $\bar{l}_t$ obtaining the degraded video $\tilde{l}$ (Line 6), and then a full reference video quality assessment tool estimates the video quality by comparing the degraded video $\tilde{l}$ with the reference video $\bar{l}_t$ (Line 7). Notice that this estimation process captures exactly what happens during video playback. In fact, the video player has to upscale the decoded video to the device screen resolution if the client screen resolution is higher than the video resolution served by the content provider, leading to a degradation in terms of perceived video quality and user QoE. Conversely, when the user is served with a video resolution equal to his device resolution, no upscaling is needed and the user perceives the best visual quality experience. This situation is taken into account by Line 9. In this case the video level $l$ does not belong to $\mathscr{L}_t$, i.e. if the resolution of $l$ is larger than that of the reference level $\bar{l}_t$, the video quality is set to 1.

## 3.3 Demand Weights computation

In order to guarantee that the solution of Problem 1 corresponds to achieving the optimum QoE-fair (rather than a throughput-fair) allocation of resources, we need to properly compute the demand weights $w_d$ used in (2). Recall that, as already mentioned in Section 2, it turns out that the higher the weight $w_d$ the higher the assigned bandwidth $X_d$ to the video flows belonging to demand $d$. Thus, weights should be computed in such a way that demands corresponding to users with large screens obtain higher bandwidth shares compared to users with small screens.

First, given a demand $d = (\text{src}, \text{dst}, t)$, notice that the weights $w_d$ do not depend on source and destination nodes, but only on the video request $t$, i.e. on the particular video and user class. Thus, two demands $d_1$ and $d_2$ characterized by the same video request $t$ will have the same weights $w_{d_1} = w_{d_2} = w_t$. Therefore, in the following we focus on the procedure to compute $w_t$.

Consider the mapping $Q_t$ computed as described in Algorithm 1 and the couples $(x_i, y_i)$ for $i = 1, \ldots, L$ where $x_i = b_i \in \mathscr{B}_v$ and $y_i = Q_t(l_i)$. We propose to compute the weight $w_t$ as the output of a least square problem fitting the data $(x_i, y_i)$ with the function $y = (\log x)/w_t$ having $w_t$ as the unique fitting parameter. Figure 3 shows an example of how weights of a video $v$ are computed for the two video requests $t_1 = (v, 1080\text{p})$ and $t_2 = (v, 2160\text{p})$. The markers in Figure 3 represent the $Q_{t_1}$ and $Q_{t_2}$ mappings, whereas the logarithmic fittings related to these two mappings are shown using continuous lines. The figure also shows that the computed weights using the proposed procedure are respectively $w_{t_1} = 7.84$ and $w_{t_2} = 9.64$. Thus, the value of weights obtained using this

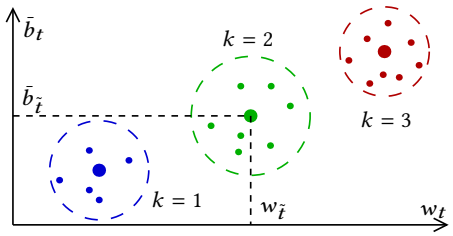



**Figure 4: Proposed clustering procedure**

methodology increases with the device resolution of the video request, which is exactly what we need to make sure that the optimal solution assigns clients with higher resolutions with higher network bandwidth shares.

## 4 CLUSTERING VIDEO REQUESTS

**Motivating video requests clustering.** In the previous section we have defined how to compute all the inputs to the optimization Problem 1. Recall that the goal of the optimization problem is to find the path flows $x_{dp}$ such that the QoE-aware objective function (2) is maximized. To simplify the following analysis consider the case of single-path, i.e., every demand is realized using a single pre-computed path of the delivery network. In such a case, it is immediate to observe that we only have one path flow $x_{dp}$ ($\mathscr{P}_d$ is a singleton) for each demand and thus $X_d = x_{dp}$. It follows that, in the single-path case, the number of variables involved in the solution of the optimization problem is equal to the number of all the possible demands $D$, i.e. the cardinality of the demand set $\mathscr{D}$. Since a demand is defined as the triple $(\text{src}, \text{dst}, t) \in \mathscr{N} \times \mathscr{N} \times \mathscr{T}$, it follows $D = N \cdot (N-1) \cdot T$. Now, recalling that a video request $t \in \mathscr{T}$ is defined as the couple $(v, c) \in \mathscr{V} \times \mathscr{C}$ , it turns out that the cardinality of $\mathscr{T}$ is equal to $V \cdot C$, i.e. the product of the video catalog size and the number of user classes. Thus, considering a video provider serving a catalog of $10^9$ videos it is easy to understand that the number of the video requests would make the cardinality of $D$ too high and would result in an intractable optimization problem.

**From video requests to traffic classes.** To address this issue, we propose to employ the following procedure. For each user class $c \in \mathscr{C}$, we partition the video catalog $\mathscr{V}$ in a number $K$ of clusters $\{\mathscr{V}_1^c, \ldots, \mathscr{V}_K^c\}$ according to a clustering algorithm. We denote with $\mathscr{K} = \{1, \ldots, K\}$ the set of the video cluster indexes. Notice that $K$ is a design parameter that can be chosen freely such that $K \ll V$. Next, we associate to a video request $t = (v, c)$ a *traffic class* $\tilde{t}$ as the couple $(k, c)$ where $k \in \mathscr{K}$ is the cluster the video $v$ belongs to (i.e., $v \in \mathscr{V}_k^c$). Notice that with this procedure all the video requests $t = (v, c)$ having $v$ mapped to the same video cluster $\mathscr{V}_k^c$ are associated to the same traffic class $\tilde{t} = (k, c)$. If we now redefine the demand as the aggregate of video sessions having the same triple $(\text{src}, \text{dst}, \tilde{t})$ we end up with a demand set whose cardinality is now equal to $N \cdot (N-1) \cdot K \cdot C$ that can be made manageable by properly setting $K \ll V$.

**Clustering procedure.** Let us fix a user class $c \in \mathscr{C}$ and consider all the video requests $t$ having a user class equal to $c$. For each of these video requests, consider the couples $(w_t, \bar{b}_t)$ where $w_t$ is the weight computed as discussed in Section 3.3 and $\bar{b}_t$ is the associated reference video level bitrate. Figure 4 shows an example of how $(w_t, \bar{b}_t)$ couples are distributed for a specific user class $c$. Notice that each point in the figure represents a single video. Next, we employ the $k$-medoid clustering algorithm to form $K$ clusters as shown in Figure 4. As a result, each point in a cluster $k$ represents a video belonging to the cluster $\mathscr{V}_k^c$. Moreover, for each cluster $k \in \mathscr{K}$, the algorithm computes the *medoid*, which is represented with a large dot in Figure 4. Thus, the medoid of cluster $k$ obtained for the user class $c$ is representative of the traffic class $\tilde{t} = (k, c)$. Therefore, it is natural to associate to each $\tilde{t}$ the weight $w_{\tilde{t}}$ and bandwidth $\bar{b}_{\tilde{t}}$ that are the coordinates of the medoid. As an example, consider the cluster $k = 2$ in Figure 4. The traffic class $\tilde{t} = (2, c)$ is associated with the weight $w_{\tilde{t}}$ and bandwidth $\bar{b}_{\tilde{t}}$ which are the coordinates of the medoid of cluster $k = 2$ (large green dot).

Notice that the chosen number of clusters $K$ entails a trade-off between the resulting number of variables involved in the optimization problem and the obtainable QoE-fairness. In fact, the smaller the number $K$, the smaller the number of variables to be handled by the optimization problem. However, with a small $K$, the number of video sessions belonging to the same cluster will be large and the approximation of each of the associated points $(w_t, \bar{b}_t)$ to the cluster medoid $(w_{\tilde{t}}, \bar{b}_{\tilde{t}})$ may become poor.

## 5 RESULTS

In this section we carry out a performance evaluation of the proposed QoE-fair optimal resource allocation. In particular, we set the *QoE-Proportional Fair* (PF) optimization problem (Problem 1) using the definition of demands given in Section 4 which is based on a clusterization of video requests. The performance evaluation carries out a sensitivity analysis of the performances considering two key parameters: 1) the *load* on the delivery network, i.e., the total traffic volume due to concurrent video sessions; 2) the number of clusters $K$.

The performance gains obtainable using the proposed VCP are compared to the *baseline* (BL) QoE-unaware case which considers each video session to belong to the same traffic class. Notice that this is the typical approach employed by current video delivery services that do not take into account in network resource allocation the heterogeneity of user devices and video contents.

**The VCP simulator.** We have built a simulator in Python language implementing the VCP described in Sections 3 and 4. The simulator is composed of the following modules: 1) the video session generator, 2) the solver, 3) the QoE evaluator. In a nutshell, the video session generator generates a number of video sessions $(\text{src}, \text{dst}, t)$ based on the delivery network graph $G$, the video catalog $\mathscr{V}$, and the user classes set $\mathscr{C}$. The network nodes set $\mathscr{N}$ is partitioned in two subsets $\mathscr{N}_{\text{src}}$ and $\mathscr{N}_{\text{dst}}$. The server sending the video to the client is picked randomly in the set $\mathscr{N}_{\text{src}}$ (i.e., $\text{src} \in \mathscr{N}_{\text{src}}$), whereas the client is picked randomly from the set $\mathscr{N}_{\text{dst}}$ (i.e., $\text{dst} \in \mathscr{N}_{\text{dst}}$). The video request $t = (v, c)$ is generated by randomly selecting the video $v \in \mathscr{V}$ and the user class $c \in \mathscr{C}$. The solver module employs the CVXPY tool [12] to implement Problem 1. In particular, we use the *Splitting Conic Solver* (SCS)[3] [23]. Based on the generated video sessions and the delivery network topology, the solver derives the demands $d$. The demands are generated by properly aggregating the video sessions according to the defined traffic

[3] https://github.com/cvxgrp/scs

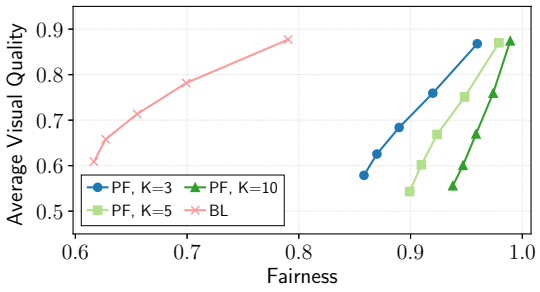
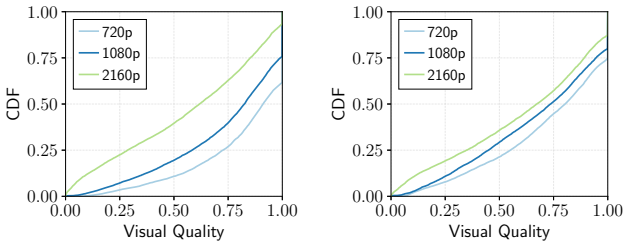



**Figure 5: QoE-Fairness vs Average Visual Quality**




**(a) Baseline** **(b) Proportional Fairness, $K = 10$**

**Figure 6: CDF of visual quality for different user classes**

classes $\tilde{t}$ depending on the clustering parameter $K$. Next, after solving the optimization problem, the QoE evaluator module computes for each video session (src, dst, $t$) the obtained QoE based on the network bandwidth share allocated by the solver and $Q_t$ mapping the bandwidth share to the visual quality. Finally, the *fairness* $F$ among video sessions is computed by using the definition given in [15], i.e. $F = 1 - 2\sigma$ where $\sigma$ is the standard deviation of the QoEs obtained by concurrent video sessions.

**Simulation scenarios.** In order to perform a realistic performance evaluation, we have built a comprehensive video catalog composed of ~200 videos downloaded from YouTube characterized by different duration, content type (sports, news, music, cartoons), and video level sets. For each video, we computed the video-level/video-quality mapping $Q_t$ as described in Section 3.2. To the purpose, we have used the VMAF metric computed using the open-source tools released by NetFlix[4]. We have considered users belonging to the class set $\mathscr{C} = \{720p, 1080p, 2160p\}$ which are the typical screen resolutions of most common devices. For each simulation, the number of generated video sessions depends on the chosen *load* which varies in the set $\{100, 200, 300, 400, 500\}$ Gbps. The GARR network[5] has been employed as the delivery network topology that is composed of 61 switches and 73 links with an average capacity of ~4 Gbps. The server nodes set $\mathscr{N}_{\text{src}}$ is formed by the top-10 nodes having the highest total upstream capacity. For simplicity we assume that servers store the complete video catalog. We have carried out the performance evaluation by comparing the BL strategy with the proposed PF resource allocation when $K \in \{3, 5, 10\}$.

**The trade-off between average QoE and fairness.** Figure 5 shows the trade-off between the average QoE obtained by video sessions and the corresponding QoE-fairness when BL is employed or in the case of the proposed PL resource allocation strategy. Each line represents a particular scenario and each point of a line is representative of a specific load. For the PF case, different markers and color indicate a different number of clusters $K$. Notice that, for each curve in the figure, the average visual quality decreases together with the QoE fairness as the load on the delivery network increases. The fact that the average QoE decreases as the load increases is unavoidable, since the higher the load the lower the average bandwidth share per video session. For instance, in the BL case it results an average visual quality close to 0.9 for a load equal to 100 Gbps, decreasing to 0.8 in the case of a 200 Gbps load and so on in a decreasing fashion. In particular, the baseline strategy presents an average fairness varying in the range 0.62-0.78 and obtaining an average visual quality in the range 0.6-0.88. On the other hand, the figure clearly shows that the PF approach proposed in this paper exhibits remarkably better results in terms of QoE fairness for each of the considered number of clusters $K$ while retaining almost the same average visual quality obtained by BL. Moreover, as expected, the larger the number of clusters $K$ the better the results in terms of QoE fairness. As $K$ increases, lines move to the right and the slope becomes steeper, indicating that the QoE fairness gets insensitive to the network load. The best trade-off is obtained for the case of $K = 10$ clusters: the measured average visual quality is in the range 0.54-0.88, with a very high fairness confined in the very tight range 0.94-0.98.

**Video session QoE fairness.** In order to show why PF achieves a better fairness compared to BL, Figure 6 reports the CDF of the visual quality obtained by all video sessions grouped by the user class in the case of a 300 Gbps load[6]. In the BL case (Figure 6a), it is clear that clients with 720p resolution obtain a much higher video quality compared to 2160p users. In particular, the median value of the visual quality obtained by 720p, 1080p, 2160p clients is respectively equal to 0.91, 0.82, 0.61. In contrast, the proposed PF resource allocation strategy provides users with different screen sizes with comparable visual quality as the three CDFs are much closer to each other. In the case of PF the visual quality obtained by 720p, 1080p, 2160p clients is respectively equal to 0.80, 0.73, 0.67.

## 6 CONCLUSIONS

In this paper, we have proposed a Video Control Plane (VCP) to enforce a QoE-fair network resource allocation. To achieve such a goal, we have shown how to properly formulate a Multi-Commodity Flow Problem. Next, we have proposed a clusterization of video sessions such that the number of variables involved in the optimization problem becomes manageable. The performances of the proposed VCP have been compared to a QoE-unaware resource allocation strategy which is representative of the currently deployed video delivery networks. Simulation results show that the proposed VCP is able to improve fairness among heterogeneous clients.

## ACKNOWLEDGMENTS

This work has been partially supported by the Italian Ministry of Economic Development (MISE) through the CLIPS project (no. F/050136/01/X32). Any opinions, findings, conclusions or recommendations expressed in this work are the authors and do not necessarily reflect the views of the funding agency.

[4]https://github.com/Netflix/vmaf
[5]http://www.topology-zoo.org/files/Garr201201.gml
[6]Results for different loads are similar and not included due to space constraints.

## REFERENCES


[1] Zahaib Akhtar, Yun Seong Nam, Ramesh Govindan, Sanjay Rao, Jessica Chen, Ethan Katz-Bassett, Bruno Ribeiro, Jibin Zhan, and Hui Zhang. Oboe: Auto-tuning Video ABR Algorithms to Network Conditions. In *Proc. of ACM SIGCOMM '18*, 2018.

[2] Abdelhak Bentaleb, Ali C Begen, and Roger Zimmermann. Sdndash: Improving qoe of http adaptive streaming using software defined networking. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1296–1305. ACM, 2016.

[3] Dimitri P. Bertsekas, Robert G. Gallager, and Pierre Humblet. *Data networks*, volume 2. Prentice-Hall International New Jersey, 1992.

[4] S. Canale, F. Delli Priscoli, S. Monaco, L. Palagi, and V. Suraci. A reinforcement learning approach for qos/qoe model identification. In *Proc. of 34th Chinese Control Conference (CCC)*, pages 2019–2023, 2015.

[5] Silvia Canale, Federico Cimorelli, Francisco Facchinei, Raffaele Gambuti, Laura Palagi, and Vincenzo Suraci. Profiled QoE based network controller. In *Proc. of 23rd Mediterranean Conference on Control and Automation (MED)*, pages 1085–1091, 2015.

[6] VNI Cisco. Cisco visual networking index: Forecast and trends, 2017–2022. *White Paper*, 2018.

[7] Maxim Claeys, Steven Latré, Jeroen Famaey, and Filip De Turck. Design and Evaluation of a Self-Learning HTTP Adaptive Video Streaming Client. *IEEE Communications Letters*, 18:716–719, 04 2014.

[8] Giuseppe Cofano, Luca De Cicco, and Saverio Mascolo. A control architecture for massive adaptive video streaming delivery. In *Proc. of 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming*, VideoNext '14, pages 7–12, 2014.

[9] Giuseppe Cofano, Luca De Cicco, and Saverio Mascolo. Modeling and design of adaptive video streaming control systems. *IEEE Transactions on Control of Network Systems*, 5(1):548–559, 2018.

[10] Giuseppe Cofano, Luca De Cicco, Thomas Zinner, Anh Nguyen-Ngoc, Phuoc Tran-Gia, and Saverio Mascolo. Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming. In *Proc. of the 7th ACM Multimedia Systems Conference*, 2016.

[11] Luca De Cicco, Giuseppe Cilli, and Saverio Mascolo. ERUDITE: A Deep Neural Network for Optimal Tuning of Adaptive Video Streaming Controllers. In *Proc. of the 10th ACM Multimedia Systems Conference*, pages 13–24, 2019.

[12] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[13] Markus Fiedler, Tobias Hossfeld, and Phuoc Tran-Gia. A generic quantitative relationship between quality of experience and quality of service. *IEEE Network*, 24(2):36–41, 2010.

[14] Panagiotis Georgopoulos, Yehia Elkhatib, Matthew Broadbent, Mu Mu, and Nicholas Race. Towards network-wide QoE fairness using openflow-assisted adaptive video streaming. In *Proc. of the 2013 ACM SIGCOMM workshop on Future human-centric multimedia networking*, pages 15–20, 2013.

[15] T. Hossfeld, L. Skorin-Kapov, P. E. Heegaard, and M. Varela. Definition of QoE Fairness in Shared Systems. *IEEE Communications Letters*, 21(1):184–187, Jan 2017.

[16] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: Experience with a globally-deployed software defined wan. In *Proc. of ACM SIGCOMM 2013*, pages 3–14, 2013.

[17] Jan Willem Kleinrouweler, Sergio Cabrero, and Pablo Cesar. Delivering stable high-quality video: An SDN architecture with DASH assisting network elements. In *Proc. of the 7th ACM Conference on Multimedia Systems*, page 4, 2016.

[18] Jan Willem Kleinrouweler, Britta Meixner, and Pablo Cesar. Improving video quality in crowded networks using a DANE. In *Proc. of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 73–78, 2017.

[19] Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara. Toward a practical perceptual video quality metric. *The Netflix Tech Blog*, 6, 2016.

[20] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. Neural Adaptive Video Streaming with Pensieve. In *Proc. of ACM SIGCOMM '17*, aug 2017.

[21] Jeonghoon Mo and Jean Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, (5):556–567, 2000.

[22] John F Nash Jr. The bargaining problem. *Econometrica: Journal of the Econometric Society*, pages 155–162, 1950.

[23] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, June 2016.

[24] Stefan Pham, Patrick Heeren, Daniel Silhavy, and Stefan Arbanowski. Evaluation of shared resource allocation using sand for abr streaming. In *Proc. of the 10th ACM Multimedia Systems Conference*, pages 165–174, 2019.

[25] Michal Pióro and Deep Medhi. *Routing, flow, and capacity design in communication and computer networks*. Elsevier, 2004.

[26] F. Delli Priscoli, C. Gori Giorgi, S. Monaco, A. Pietrabissa, and V. Suraci. Future Internet Architecture: Control-based Perspectives related to Quality of Experience (QoE) Management. In *Proc. of 34th Chinese Control Conference (CCC 2015)*, 2015.

[27] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[28] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. In *Proc. of ACM SIGCOMM '15*, aug 2015.