Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

A Multi-Commodity Flow Problem for Fair Resource Allocation in Multi-Path Video Delivery Networks

Luca De Cicco * **Gioacchino Manfredi** * **Vittorio Palmisano** *
Saverio Mascolo *

* *Politecnico di Bari, Dipartimento di Ingegneria Elettrica e dell'Informazione, Via Orabona 4, Bari, Italy (e-mails: {name.surname}@poliba.it)*

Abstract: Video streaming services employ the Internet to distribute content to an ever-increasing number of concurrent viewers. The delivery architecture employed by leading video platforms requires players to run a control algorithm dynamically choosing the video bitrate to match the time-varying network bandwidth and avoid playback interruptions due to buffer underruns. Such an algorithm is generally designed to (selfishly) improve the quality individually perceived by users. Consequently, this control architecture leads, in the optimal case, to maximize the average quality perceived collectively by all users and not to a distribution of resources that is fair in terms of user perceived quality. We argue that video service providers should manage their delivery network to address fairness issues to gracefully degrade the perceived quality equally for all users when resources become scarce. Even though the general problem of providing a fair level of perceived quality does not scale with the cumbersome number of concurrent users, this paper shows that the Multi-Commodity Flow Problem (MCFP) optimization framework is a proper and efficient tool to address this open issue. First, we show how to cast the resource allocation problem to an MCFP and then we propose a strategy to make the resulting problem tractable for video distribution platforms serving massive audiences. The performance of the proposed optimal fair resource allocation strategy is assessed using realistic simulations involving thousands of concurrent video sessions on a real network topology by varying both the total load on the network and key system parameters.

1. INTRODUCTION AND BACKGROUND

An increasing fraction of users prefer to consume video content over the Internet instead of using classical TV broadcast channels. As a consequence, more than half of the global Internet traffic is today due to video. To make their services profitable, on-line video content providers aim at increasing the number of engaged users and preventing service abandonment. Towards this end, such services should be designed to provide users with the best possible *Quality of Experience* (QoE) given the constraints due to the user device and the network.

The control architecture employed today by all leading video platforms (Netflix, Youtube, etc) decouples the problem into two non-cooperating sub-problems: (i) video services control their delivery network to guarantee an optimal level of Quality of Service (QoS), by ensuring that parameters such as end-to-end network bandwidth, packet losses, and network latency meet specific requirements; (ii) concurrent users consume video through players that run *Adaptive BitRate* control 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. This fully decoupled control approach has the merit of being very simple to be implemented but has some important limitations. In fact, since no communication between users is available, ABR algorithms running at the players are designed to (selfishly) improve the individual QoE obtained. In addition, the architecture of current delivery networks is designed to provide to concurrent users sharing the same network resources (i.e., network links) a fair share of network bandwidth. The issue here is that such a QoS-fair distribution of network resources does not translate in equalizing the quality perceived by users. In fact, it is well-known that the video bitrate required to obtain the same level of QoE by users with large screen devices (f.i.,Smart TVs) might be considerably larger compared to the one needed by devices with small screens (e.g. smartphones). Consequently, video distribution networks that do not run an algorithm to allocate resources taking into account the obtained user quality cannot provide a fair level of QoE to users when the network resources become scarce. Hence, we argue that video services should implement a QoE-aware network resource allocation strategy (as opposed to QoS-aware strategies) to provide video flows sharing the same bottleneck a differentiated network bandwidth with the objective of equalizing the video quality obtained by heterogeneous devices.

[1] This work has been partially supported by the Italian Ministry of Economic Development (MISE) through the CLIPS project (no. F/050136/01/X32).

Several authors have addressed the issue of designing QoE-aware ABR controllers (see Yin et al. (2015); Cofano et al. (2018); De Cicco et al. (2019) and references therein). However, QoE-fair network resource allocation for video streaming has received far less attention and has been addressed only in a few recent papers (Georgopoulos et al. (2013); Cofano et al. (2016); Kleinrouweler et al. (2016)), all reporting the need of a *Video Control Plane* (VCP) to allow cooperation between clients and the delivery network. Georgopoulos et al. (2013) propose for the first time a solution to deliver a fair level of QoE to users by slicing shared bottlenecks through a Software Defined Networking (SDN) switch. Each video session is assigned to one network slice whose size is obtained by solving a max-min fairness problem. Cofano et al. (2016) design and systematically analyze the performance of an SDN-based VCP in the case of a single bottleneck.

This paper addresses the problem of designing a QoE-fair optimal resource allocation strategy through a constrained optimization problem on a generic distribution network made of programmable switches with the help of traffic engineering techniques based on network slicing. The main novel aspects of this paper compared to the current state of the art are two: (i) a generic distribution network is considered instead of focusing only on the single bottleneck case as studied by several authors (Georgopoulos et al. (2013); Cofano et al. (2016)); (ii) it is shown that the *Multi-Commodity Flow Problem* (MCFP) optimization framework is an effective methodology to achieve a QoE-fair distribution of network resources. After casting the QoE-fair resource allocation problem to an MCFP (Section 2), a traffic clustering approach is proposed to sensibly reduce the number of network slices and variables in order to make the resulting problem tractable for video distribution platforms serving a massive audience (Section 3). Such a clustering approach assigns video sessions based on a proposed similarity metric that depends on the video visual quality. We implement the proposed resource allocation strategy in a realistic simulator to compare the performances obtainable when video content can be delivered using multiple network paths with those achievable in the single-path case. Finally, simulations assess the performance sensitivity to different parameters, such as the total load on the delivery network and the number of clusters (Section 4).

2. MULTI COMMODITY FLOW PROBLEM

In this section, we briefly review the *multi-commodity flow problem* (MCFP), the optimization framework that we leverage to design the proposed QoE-fair network bandwidth allocation strategy. The term *commodity* refers to a tuple composed of a source node, a destination node, and a volume that identifies the resources needed to satisfy the commodity. In the case of the network bandwidth allocation problem that we are considering, a commodity refers to a video session (or aggregate of video sessions) whose source node is the video server, the destination node represents the client consuming the video, and the volume represents the video bitrate required to obtain the maximum video quality. In general, the MCFP aims at maximizing a proper utility function with a set of constraints to allocate network resources in order to optimally satisfy all the commodities.

The following description of the MCFP employs the *link-path formulation* and the terminology introduced by Pióro and Medhi (2004). The delivery network is represented by a capacitated graph $G = (\mathscr{N}, \mathscr{E})$, where $\mathscr{N} = \{n_1, n_2, \ldots, n_N\}$ is the *node* set and $\mathscr{E} = \{e_1, e_2, \ldots, e_E\}$ is the *edge* set. Each edge or link $e \in \mathscr{E}$, which can be identified by a node pair, is assigned with a bandwidth capacity c_e. The commodities related to the delivery network can be represented by the set of *demands* $\mathscr{D} = \{1, 2, \ldots, D\}$, where each demand $d \in \mathscr{D}$ contains a source-destination node pair and the corresponding *traffic volume* H_d, i.e. the required network bandwidth for that demand. Furthermore, a demand d can be satisfied through a set of admissible paths $\mathscr{P}_d$ where each path $p \in \mathscr{P}_d$ connects the source node to the destination node of the demand. All the paths contained in $\mathscr{P}_d$ are computed off-line and represent the shortest paths connecting source node to destination node of demand d. Consequently, the demand volume H_d is split in *path flows* routed on paths belonging to $\mathscr{P}_d$, where each path flow is denoted with x_{dp} ($p \in \mathscr{P}_d$). The objective of the MCFP is to optimize such path flows. In this work, the nodes of the graph G identify the network switches, [2] whereas the links are partitioned in *bandwidth slices* whose number and size depend on the MCFP solution. Mo and Walrand (2000) introduce the α-fairness utility function that can be employed in the MCFP to size the bandwidth slices. In particular, we consider the following multi-path weighted α-fairness utility function:

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

where $X_d = \sum_p x_{dp}$ is the total bandwidth (or total flow) allocated to demand d, $\boldsymbol{X} = [X_1, X_2, \ldots, X_D]^T$ is the vector of the total bandwidths for each demand, and w_d is a *weight* associated to the demand d. The value of $\alpha \in \mathbb{R}_+$ affects the balance between link utilization and fairness when equation (1) is maximized Mo and Walrand (2000). In fact, when $\alpha = 0$ it can be shown that the link utilization is maximized with no regards to the fairness among flows, whereas if $\alpha \to +\infty$, the resources are allocated in such a way that the minimum rate flow is maximized (max-min fairness problem). Finally, if $\alpha = 1$, the *Proportional Fairness* (PF) optimization problem is obtained (Nash Jr (1950)). The latter case represents a satisfying balance between fairness and link utilization, that is exactly what we seek to achieve in our problem setting. For this reason the proportional fair case, i.e. $\alpha = 1$, is considered. The resulting MCFP multi-path weighted proportional fair optimization problem is:

$$\text{Maximize} \sum\nolimits_d w_d \log X_d \tag{2}$$

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

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

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

In fact, it is straightforward to prove that, for $\alpha = 1$, (1) becomes $U(\boldsymbol{X}) = \sum_d w_d \log X_d$. In (4), δ_{edp} represents

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

the link-path indicator and it is equal to 1 if the path p associated to the demand d uses the link e, otherwise it is set to 0. 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}using 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.

It is straightforward to show that Problem (2)-(5) is convex since the objective function is convex and the constraints are linear. Thus, the solution is represented by a unique global maximum that could be achieved either at one single point or at a convex set of feasible points (Bertsekas et al. (1992); Pióro and Medhi (2004)).

3. THE RESOURCE ALLOCATION STRATEGY

In the following we present the proposed control strategy to distribute network resources in such a way that a fair level of QoE is delivered to concurrent heterogeneous users. To the purpose, we show how to cast the MCFP (Problem (2)-(5)) to achieve the aforementioned goal. This includes designing the demand weights w_d so that the maximization of (2) results in a QoE-fair resource allocation. The idea that we pursue is that demand weights should relate to a utility function mapping the relationship between the network bandwidth assigned to a video session and the obtainable visual quality (Section 3.2).

3.1 Definitions

The DASH standard requires that each video v belonging to the video catalog $\mathscr{V} = \{v_1, \ldots, v_V\}$ is encoded into different representations or *levels* $l \in \mathscr{L}_v$ that can be identified 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. In practice, the ABR algorithm running at the client dynamically selects the video level $l \in \mathscr{L}_v$ that best matches the current available network bandwidth of the path connecting the user to the video server. In this paper, we make the reasonable assumption that the control algorithm selects a video level whose bitrate b matches on average the average end-to-end path bandwidth. This is a nonrestrictive assumption since all well-designed ABR algorithms are in practice implemented in this way (see for instance Cofano et al. (2018)).

Now, let us define a *video request* t as the couple (v, c), where $v \in \mathscr{V}$ and c is the *user class* belonging to the set $\mathscr{C} = \{c_1, c_2, \ldots, c_C\}$. Since the screen resolution is one of the most important parameters impacting the QoE, we propose to classify users based on such a parameter. Consequently, the terms "user class" and "user screen resolution" are used interchangeably in this work.

We next define the set $\mathscr{L}_t$ for each video request $t = (v, c)$, such tat $\mathscr{L}_t = \{l \in \mathscr{L}_v : r \leq c\} \subseteq \mathscr{L}_v$. In other words, $\mathscr{L}_t$ contains the levels of $\mathscr{L}_v$ whose resolution is less than or equal to c. Since we assume that clients having a screen resolution equal to c do not request video levels whose resolution is higher than c, then for a given video request t, the levels chosen by the ABR algorithm will be contained in $\mathscr{L}_t$.

It is now immediate to assign to each video request t its *reference level* $\bar{l}_t = (\bar{b}_t, c) \in \mathscr{L}_t$ as the representation with resolution c having the maximum bitrate $\bar{b}_t$.

We can now define a *video session* as the tuple $(\text{src}, \text{dst}, t)$ where: $\text{src} \in \mathscr{N}$ is the switch the server delivering the requested video is connected to; $\text{dst} \in \mathscr{N}$ is the switch the client is connected to; $t = (v, c)$ is the video request.

Finally, we are ready to define the *demand* d as the aggregate of the n_d video sessions identified by the same tuple $(\text{src}, \text{dst}, t)$. Consequently, the *demand volume* H_d is equal to $n_d \bar{b}_t$ where $\bar{b}_t$ is the bitrate of the reference level $\bar{l}_t$ defined above. In such a way, H_d can be interpreted as the minimum amount of network bandwidth that has to be allocated to the aggregate of the n_d video sessions composing the demand d so that each of these video sessions is served with a bandwidth share $\bar{b}_t$. It is straightforward to see that, in such a situation, if the constraint (5) is strictly verified (i.e., $X_d = H_d$), it results that all the video flows belonging to this demand will enjoy the maximum visual quality possible. Conversely, in cases when the delivery network is overloaded, it might occur that the solution of the MCFP leads to $X_d < H_d$ for some demands. In such cases, video sessions belonging to the demand d will obtain a bandwidth share less than the bitrate of the reference level $\bar{l}_t$.

3.2 Measuring the visual quality

The achievement of a fair level of QoE among users represents the main goal of the proposed resource allocation strategy. Such an objective is reached through the allocation of network resources in a multi-path fashion. For this reason, a mapping between the allocated network bandwidth related to a video session and the achieved QoE is needed (Fiedler et al. (2010)). Such a mapping will be the key to design appropriate demand weights w_d that allow to solve Problem (2)–(5) by allocating the network bandwidth based on the users' obtainable visual quality.

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.

The visual quality of a video $v \in \mathscr{V}$ is measured in the following way: for each level $l \in \mathscr{L}_v$, and user class $c \in \mathscr{C}$, a mapping denoted as $Q_t : \mathscr{L}_v \mapsto [0, 1]$ is computed, which relates the video level to the corresponding visual quality when the video is played on a device with resolution c.[3]

Therefore, a full-reference video quality assessment tool, such as the Video Multi-method Assessment Fusion (VMAF) (Li et al. (2016)), can be used to compute for each video level $l \in \mathscr{L}_v$ the corresponding visual quality belonging to the range $[0, 1]$ given a user class $c \in \mathscr{C}$.

3.3 Demand Weights computation

The importance of a proper computation of the demand weights w_d used in (2) lies in the fact that the solution

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

of Problem (2)–(5) will result in the optimum QoE-fair (rather than a throughput-fair) allocation of resources. From Section 2 we know that the larger the weight w_d the larger the assigned bandwidth slice X_d to the video flows belonging to demand d. It is then important to compute suitable weights that allow to obtain a higher bandwidth for demands associated to users with large screens and lower bandwidth to users with low resolution screens.

It should be stressed that the weight w_d associated to a demand $d = (\mathrm{src}, \mathrm{dst}, t)$ does not depend on the source and destination node, but only on the particular video v and the user class c, i.e., on the video request t. As a consequence, given two demands d_1 and d_2 associated to the same video request t, the weights associated to them will coincide, namely $w_{d_1} = w_{d_2} = w_t$.

Let us define the couples (x_i, y_i) for $i = 1, \ldots, L_t$ ($L_t = |\mathscr{L}_t|$) where $x_i = b_i \in \mathscr{B}_v$ and y_i is obtained through the mapping Q_t as previously described, i.e., $y_i = Q_t(l_i)$. It is possible to compute the weight w_t as the parameter of a fitting function. In particular, we propose to consider a least square problem fitting the data (x_i, y_i) through the function $y = a \cdot \log x$ having a as the unique fitting parameter. Then we impose $w_t = 1/a^\beta$ where β is a positive parameter to be properly tuned. It is easy to show that this proposed procedure assigns weights that increase as the user device resolution becomes higher, which is exactly what it is needed to provide users with high resolution with higher network bandwidth shares. We will experimentally show that the value of β affects the obtainable QoE and needs to be properly tuned.

3.4 Video clustering

In this paper, we consider the multi-path case, where each demand d is realized by splitting it among a pre-computed number of available paths of the delivery network, i.e. $|\mathscr{P}_d|$. It follows that, in the multi-path case, the number of variables involved in the solution of the optimization problem is equal to the number P of all the possible paths available for each demand, i.e. if $\mathscr{P} = \{\mathscr{P}_1, \mathscr{P}_2, ..., \mathscr{P}_D\}$ where D is the cardinality of the demand set $\mathscr{D}$, then $P = |\mathscr{P}|$. Since a demand is defined as the triple $(\mathrm{src}, \mathrm{dst}, t) \in \mathscr{N} \times \mathscr{N} \times \mathscr{T}$, 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 size in the order of millions [4] it is easy to understand that the number of the video requests would make the cardinality D, and consequently P, too high and would result in an intractable optimization problem.

In order to tackle such an issue, we propose to act on the video catalog. The employed procedure is the following: 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. Let $\mathscr{K} = \{1, \ldots, K\}$ be the set of the video cluster indexes. Since K is a design parameter, it can be chosen such that $K \ll V$. We can now assign each video request $t = (v, c)$ to a *traffic class* $\tilde{t} = (k, c)$ where $k \in \mathscr{K}$ is the cluster the video v belongs to (i.e., $v \in \mathscr{V}_k^c$). In this way, the video requests $t = (v, c)$ having v mapped to the same video cluster $\mathscr{V}_k^c$ belong to the same traffic class $\tilde{t} = (k, c)$. After redefining the demand as the aggregate of video sessions having the same triple $(\mathrm{src}, \mathrm{dst}, \tilde{t})$, the cardinality of the new demand set will be equal to $N \cdot (N-1) \cdot K \cdot C$, that can be made manageable by properly setting $K \ll V$.

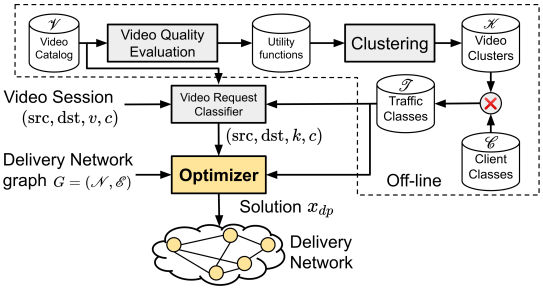


Fig. 1. The proposed Video Control Plane

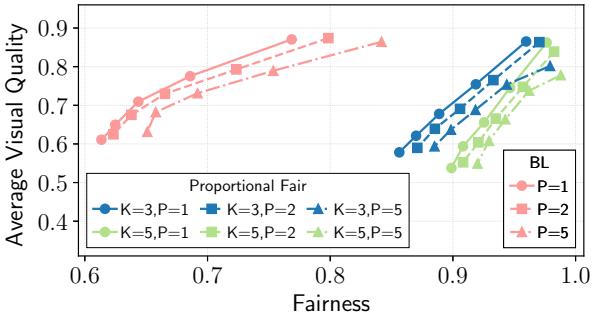
Let us consider all the video requests t having a user class equal to $c \in \mathscr{C}$. Each video request is associated to a couple $(w_t, \bar{b}_t)$ where w_t is the weight computed as discussed in Section 3.3 and $\bar{b}_t$ is the associated reference video level bitrate. Once a user class c is fixed, in order to obtain K clusters, the k-medoid clustering algorithm could be used. As a consequence, each video belonging to the cluster $\mathscr{V}_k^c$ will be identified with a point in the cluster k, which represents the *medoid* computed by the algorithm for each cluster $k \in \mathscr{K}$. Thus, for a specified user class c, the medoid associated to a cluster k will represent the whole traffic class $\tilde{t} = (k, c)$, i.e. all the videos in that cluster, through its coordinates $(w_{\tilde{t}}, \bar{b}_{\tilde{t}})$.

Figure 1 gives an overview of the proposed resource allocation strategy and how it can be implemented in a Video Control Plane. In particular, after the visual quality evaluation of the video catalog, the fitting functions described in Section 3.3 perform the clustering procedure of the videos in order to obtain the traffic classes. Notice that these operations can be performed off-line since the video catalog and the user classes are always available. Then, a video request classifier associates each received video session to the corresponding traffic class and then the optimizer, on the basis of the demands defined as $(\mathrm{src}, \mathrm{dst}, k, c)$, the traffic classes, and the delivery network graph solves the MCFP.

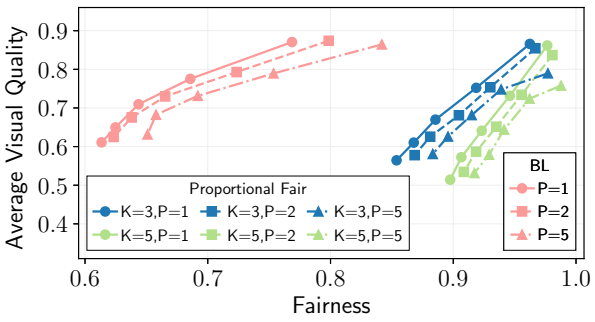
4. RESULTS

In this section we employ a clusterization of video requests and then we implement the *QoE-Proportional Fair* (PF) multi-path optimization problem described in Section 3 to carry out a performance evaluation of the proposed allocation strategy via simulations. The analysis is performed by varying three main parameters: the delivery network *load,* which represents the total traffic volume of concurrent video sessions, the number of paths P, i.e., the maximum number of paths that can be used to realize a specific demand from a source node, and the number of clusters K.

[4] In practice, video catalog of the order of millions or billions are possible for user providers distributing user-generated videos such as YouTube and Vimeo.



(a) $\beta = 1.1$



(b) $\beta = 1.4$

Fig. 2. QoE-Fairness vs Average Visual Quality

In order to prove the effectiveness of our proposed allocation strategy, we consider as the *baseline* (BL) the QoE-unaware allocation strategy that associates each video session to the same traffic class. It is important to notice that the BL case is the approach currently used by video delivery services, which are unaware of the heterogeneity of the user devices and video contents.

We have developed the PF allocation strategy in a simulator composed of three modules allowing to implement realistic scenarios of typical video distribution networks. The first module is the *video session generator*, which randomly generates a configurable number of video sessions $(\text{src}, \text{dst}, t)$having access to the network graph G, the video catalog $\mathscr{V}$, and the set of user classes $\mathscr{C}$. The second module is the solver, which employs the CVXPY Python tool (Diamond and Boyd (2016)) to implement problem (2)-(5) and which makes use of the *Splitting Conic Solver* (SCS) [5] (O'Donoghue et al. (2016)). Once the optimization problem is solved, the third module called *QoE evaluator* computes the obtained QoE for each video session $(\text{src}, \text{dst}, t)$composing the load. The resulting QoE depends on the bandwidth share assigned by the solver and the corresponding visual quality given by the Q_t mapping. Finally, we use the definition of *fairness* F among video sessions proposed by Hossfeld et al. (2017):

$$F = 1 - 2\sigma$$

where σ is the standard deviation of the QoEs obtained by concurrent video sessions. The maximum of the fairness index is 1, which is obtained only when concurrent video sessions obtain exactly the same visual quality.

The simulations have been carried out on a realistic video catalog that we have built by downloading ∼200 heterogeneous videos from YouTube. Notice that we have chosen the VMAF metric to compute the video-level/video-quality mapping Q_t as described in Section 3.2. The VMAF metric has been implemented by using the open-source tools released by NetFlix [6]. We have assumed that clients can belong to three possible user classes – which are representative of most common user devices – identified by the set $\mathscr{C} = \{720\text{p}, 1080\text{p}, 2160\text{p}\}$. The *load* values considered to generate the video sessions range in the set $\{100, 200, 300, 400, 500\}$ Gbps. The employed network topology is the GARR network [7], which is composed of 61 switches and 73 links with an average capacity of ∼4 Gbps. Finally, the set of clusters is such that $K \in \{3, 5, 10\}$, the set of paths is such that $P \in \{1, 2, 5\}$ and the weight parameter β belongs to the set $\{1.1, 1.2, 1.3, 1.4, 1.5\}$.

Figure 2 shows the trade-off between the average QoE obtained by the video sessions and the corresponding QoE-fairness when BL is employed or in the case of the proposed PF resource allocation strategy. It is worth to stress that the considered fairness is obtained by computing the fairness metric F for each slice and then taking the average value of the fairness associated to all the slices. Each line represents a particular scenario where a different line style and marker denotes a specific number of paths P involved in the allocation and each point of a line is representative of a specific load. In the PF case, different colors indicate a different number of clusters K. Moreover, for space constraints, Figure 2 shows only the cases of $\beta = 1.1$ and $\beta = 1.4$. As it is clear from any of the figures, the average visual quality and the QoE fairness decrease as the load on the delivery network increases. This is expected since a higher load results in a lower allocated average bandwidth share per video session and consequently in a lower visual quality. Consider Figure 2a as an example: it shows that in the BL case, independently of the number of paths, the average visual quality is close to 0.9 when the load is 100 Gbps, then decreases to 0.8 for a 200 Gbps load and so on. The fairness presents values in the range 0.62-0.84 with a corresponding average visual quality in the range 0.6-0.88. However, the proposed PF approach proves remarkably better in terms of achieved QoE fairness for each considered number of clusters K and paths P. The visual quality presents a negligible deterioration compared with the BL case. Furthermore, as expected, the QoE fairness improves as K increases and, consequently, each line associated to a particular number of clusters and paths moves to the right and becomes steeper. Such considerations also hold for all the other values of β, where $K = 5$ clusters appears to be the best trade-off between average visual quality and QoE fairness.

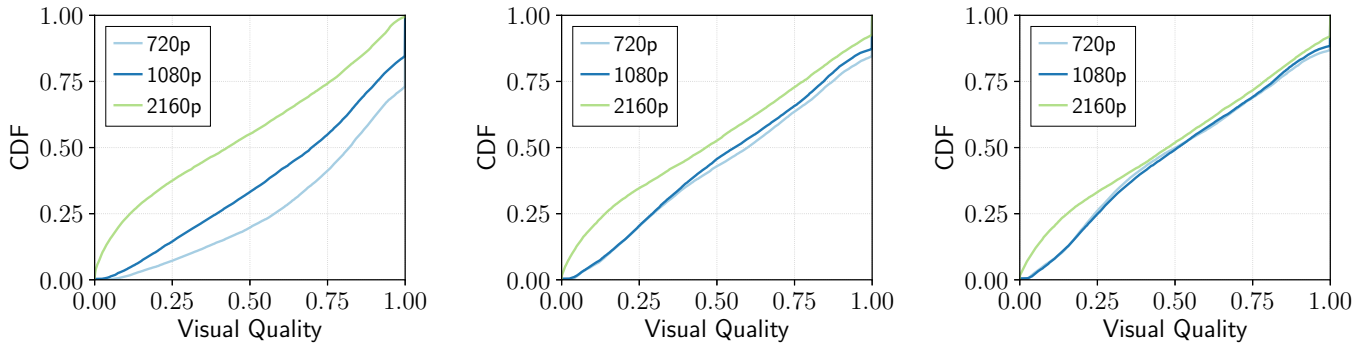
Next, consider Figures 2a and 2b. By varying β from 1.1 to 1.4 the average visual quality visibly drops while the average fairness remains almost unchanged. Indeed, the multi-path resource allocation is preferable with respect to the single-path case due to the possibility of exploiting more paths to realize a demand.

Let us now analyze in more detail the effect that the choice of the parameter β has on the QoE fairness in the single-path case (the multi-path approach gives similar results

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

[6] https://github.com/Netflix/vmaf

[7] http://www.topology-zoo.org/files/Garr201201.gml



(a) BL (b) PF ($\beta = 1.1$) (c) PF ($\beta = 1.4$)

Fig. 3. CDF of visual quality for different user classes

and is not shown for space constraints). Figure 3 reports the CDF of the visual quality obtained by all video sessions grouped by user class in the case of a 500 Gbps load [8] . The figure shows that, when passing form $\beta = 1.1$ to $\beta = 1.4$ (Figures 3b and 3c) the fairness is improved. Moreover, the obtained fairness in the PF case with $\beta = 1.4$ is remarkably better than the BL case. As a matter of fact, the PF case with $\beta = 1.4$ results in a high fairness since all the users belonging to any user class will enjoy a similar visual quality. It is worth stressing that, in the case of $\beta = 1.5$, results in terms of visual quality fairness deteriorate compared to the $\beta = 1.4$ case. In particular, for $\beta = 1.5$ clients with 720p and 1080p resolution obtain a higher visual quality than 2160p clients.

5. CONCLUSIONS

In this paper, we have proposed a Proportional Fair (PF) resource allocation strategy to equalize the QoE obtained by concurrent heterogeneous users for video delivery networks. 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 with the purpose of making the number of variables involved in the optimization problem manageable. The performance of the proposed PF allocation strategy has been compared to the case of a QoE-unaware allocation strategy, which is representative of the currently deployed video delivery networks. Simulation results show that the proposed PF allocation strategy is able to remarkably improve fairness among heterogeneous clients.

REFERENCES

Bertsekas, D.P., Gallager, R.G., and Humblet, P. (1992). *Data networks*, volume 2. Prentice-Hall International New Jersey.

Cofano, G., De Cicco, L., and Mascolo, S. (2018). Modeling and design of adaptive video streaming control systems. *IEEE Transactions on Control of Network Systems*, 5(1), 548–559.

Cofano, G., De Cicco, L., Zinner, T., Nguyen-Ngoc, A., Tran-Gia, P., and Mascolo, S. (2016). Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming. In *Proc. of the 7th ACM Multimedia Systems Conference*.

De Cicco, L., Cilli, G., and Mascolo, S. (2019). ERUDITE: A Deep Neural Network for Optimal Tuning of Adaptive Video Streaming Controllers. In *Proc. of the 10th ACM Multimedia Systems Conference*, 13–24.

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

Fiedler, M., Hossfeld, T., and Tran-Gia, P. (2010). A generic quantitative relationship between quality of experience and quality of service. *IEEE Network*, 24(2), 36–41.

Georgopoulos, P., Elkhatib, Y., Broadbent, M., Mu, M., and Race, N. (2013). 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*, 15–20.

Hossfeld, T., Skorin-Kapov, L., Heegaard, P.E., and Varela, M. (2017). Definition of QoE Fairness in Shared Systems. *IEEE Communications Letters*, 21(1), 184–187. doi:10.1109/LCOMM.2016.2616342.

Kleinrouweler, J.W., Cabrero, S., and Cesar, P. (2016). Delivering stable high-quality video: An SDN architecture with DASH assisting network elements. In *Proc. of the 7th ACM Conference on Multimedia Systems*, 4.

Li, Z., Aaron, A., Katsavounidis, I., Moorthy, A., and Manohara, M. (2016). Toward a practical perceptual video quality metric. *The Netflix Tech Blog*, 6.

Mo, J. and Walrand, J. (2000). Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, (5), 556–567.

Nash Jr, J.F. (1950). The bargaining problem. *Econometrica: Journal of the Econometric Society*, 155–162.

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

Pióro, M. and Medhi, D. (2004). *Routing, flow, and capacity design in communication and computer networks*. Elsevier.

Yin, X., Jindal, A., Sekar, V., and Sinopoli, B. (2015). A control-theoretic approach for dynamic adaptive video streaming over http. In *Proc. of ACM SIGCOMM '15*.

[8] Results for different loads are similar and not included due to space constraints.