Optimal QoE-fair Resource Allocation in Multipath Video Delivery Networks

Optimal QoE-fair Resource Allocation in Multi-Path Video Delivery Networks

Gioacchino Manfredi, Luca De Cicco, *Member, IEEE*, Saverio Mascolo, *Fellow, IEEE*

***Abstract*—A steadily increasing number of users consume videos over the Internet. In current video platforms, players run a control algorithm that dynamically chooses the video bitrate to match the time-varying network bandwidth. Such an algorithm strives to 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 rather than to a quality-fair distribution of resources, possibly leading to user abandonment for those users receiving a lower quality. Therefore, we argue that well-designed video delivery networks should gracefully degrade the perceived quality equally for all users when resources become scarce. In this paper, we propose the Multi-Commodity Flow Problem (MCFP) optimization framework to address the issue of designing a QoE-fair optimal allocation strategy. We show how to make the resulting problem tractable for video platforms serving massive audiences. The performance of the proposed optimal fair resource allocation strategy is tested through realistic simulations involving thousands of concurrent users on two real networks by varying both the total load on the network and the system parameters.**

***Index Terms*—Optimal resource allocation; Massive video distribution; Multi-Commodity Flow Problem; Adaptive Video Streaming; Quality of Experience; Fairness**

I. Introduction and Background

A steadily increasing number of users prefer to consume video contents over the Internet rather than using classical TV broadcast channels. As a consequence, more than half of the global Internet traffic is today due to video contents [1]. To make their services profitable, on-line video content providers set out to increase the number of engaged users and prevent service abandonment. Towards this end, such services should be designed to provide users with the best possible *Quality of Experience (QoE)* given the constraints imposed by the particular user device and the network.

Leading video platforms (Netflix, YouTube, etc) base their services on control architectures that decouple the problem into two non-cooperating subproblems: (i) video services design and size 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 watch videos through players that run Adaptive BitRate (ABR) control algorithms 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 [2]. If, on one hand, this fully decoupled control approach has the advantage of being very simple to be implemented, on the other hand, it 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 concurrent users sharing the same network resources (i.e., network links) with a fair share of network bandwidth. However, this 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 a specific level of QoE by users with high resolution devices (f.i., Smart TVs) might be considerably larger compared to the bitrate needed by devices with low resolution (e.g. smartphones).

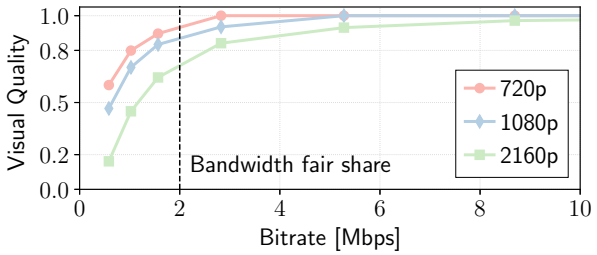


Fig. 1: Visual quality function of the video bitrate and the client screen resolution

To make an example, consider Fig. 1, which shows the measured visual quality of the same video as a function of the encoding bitrate obtained by clients with different screen resolutions. Let us suppose that three concurrent users with different screen resolutions (namely 720p, 1080p, and 2160p) request the same video and that the video flows share the same bottleneck link having a bandwidth equal to 6 Mbps. In such a case, the network bandwidth share obtained by each video flow is equal to 2 Mbps. As a result, the visual quality obtained by the three clients would be respectively equal to 0.9, 0.85, 0.7. We can conclude that low resolution devices will enjoy a better visual quality compared to high resolution 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. Consequently, video distribution networks, which allocate resources without taking into account the user's degree of satisfaction, cannot provide a fair level of QoE to users when the network resources become scarce. Hence, we argue that video service providers should implement a QoE-aware network resource allocation strategy (as opposed to QoS-aware strategies) to assign a differentiated

This work has been partially supported by the Italian Ministry of Education, Universities and Research (MIUR) through the MAIA project (no. ARS01_00353). Gioacchino Manfredi, Luca De Cicco, and Saverio Mascolo are with the Dipartimento di Ingegneria Elettrica e dell'Informazione (DEI) at Politecnico di Bari, Via Orabona 4, 70125, Bari, Italy Emails: name.surname@poliba.it

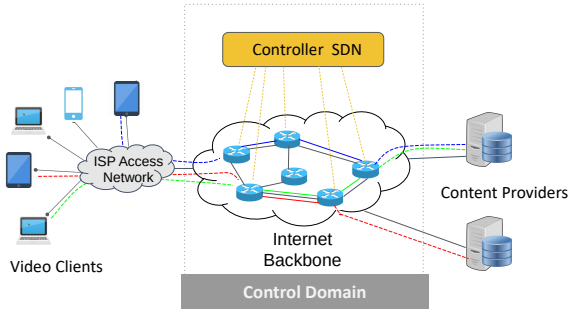


Fig. 2: The distribution network model considered in this work

network bandwidth to video flows sharing the same bottleneck with the objective of equalizing the video quality obtained by heterogeneous devices. To the purpose, an interaction between video and network provider is needed. The Software Defined Network (SDN) is a technology that allows such an interaction through a centralized control plane, as shown in Fig. 2. This control plane directly controls the network nodes, which in our case are the network switches.

This paper addresses the problem of designing a QoE-fair optimal resource allocation strategy through a constrained optimization problem to be implemented on the SDN of a generic distribution network made of programmable switches. Such a work will be carried out with the help of traffic engineering techniques based on *network slicing*. This technique consists of *slicing* the links in a network and then assigning them to subsets of the video flows, according to their characteristics. Computing the size of each slice is the objective of the optimization problem. 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 ([3], [4]); (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 a Multi-Commodity Flow Problem (MCFP) (Section III and IV), a traffic clustering approach is proposed to sensibly reduce the number of network slices and variables to make the resulting problem tractable for video distribution platforms serving a massive audience (Section IV). 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 performance 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 for different parameters, such as the total load on the delivery network and the number of clusters (Section V).

II. Related work

In this section, we provide an overview of the state of the art in resource allocation techniques employed in video streaming. In particular, we consider some works concerning centralized provider-side resource allocation strategies and then we turn to distributed client-side techniques. Several works take into account the quality perceived by users (QoE) when addressing resource allocation problems in video streaming. However, it is not an easy task to find an objective formula that measures a user's QoE or to define client classes. A definition of QoE and client classes is provided in [5], where the user's perception of the quality of a video stream includes pre-fetching delay, probability of rebuffering events, duration of such events, and so on. Client classes are characterized by flow sizes, arrival rates, and channel statistics while in our case users are classified by their device resolution only. Notice that this represents an advantage because no active measures on the network are needed to be fed to the optimizer.

In this work, we consider a model of QoE defined by the visual quality perceived by the user. This is done under the realistic assumption that the ABR algorithm is designed to avoid rebuffering events. Notice that several works employ more complete models of QoE including the impact of rebuffering events. In [6], for instance, the authors employ the SVR-QoE model and the NARX-QoE model. Both of them take into account the presence of rebuffering events and change the QoE function accordingly. Other models of QoE can be found in [7], [8]. In this paper, we have decided not to include client-side metrics such as the rebuffering ratio in the utility function to be maximized since this would have required a continuous feedback being sent from all the clients to the optimizer thus making the solution less scalable.

In [9], Ai et al. propose to solve the resource allocation problem as a 0-1 programming problem keeping into account the QoE and fairness among users. This approach entails a cross-layer design, i.e., the resource allocation is based on the information gathered from different layers of the network. The authors in [10] present an optimal QoE-aware scheduling for video segment selection in the framework of HTTP Adaptive Streaming (HAS). Such a technique also addresses rebuffering events avoidance and initial delay minimization. In [11], the authors provide a solution to optimize the QoE of multiple video streaming sessions. In fact, the bandwidth is not equally allocated among competing flows but its allocation takes into account the content complexity of the requested video and the playout buffer status of the individual clients. However, in the aforementioned works, both clients and videos are not aggregated, thus implying an extremely heavy computational load that may result unmanageable.

The importance of a Software Defined Networking (SDN) architecture is shown in [12], where a Video Control Plane (VCP) is introduced to allow cooperation between clients and the delivery network. In this scenario, the Dynamic Adaptive Streaming over HTTP (DASH) players are required to cooperate with the Service Manager to obtain an optimal bitrate while in our work clients are not involved in the optimal bandwidth distribution[1]. Additionally, in [12], each active player is assigned with an equal bandwidth share with no regard to the device resolution. In this paper, the network

[1]Clients are involved only at the beginning of the video session

bandwidth is not the same for each user, as it should be, since the type of video and device resolution are kept into account.

Samani and Wang [13] propose MaxStream, that is an SDN-based flow maximization framework based on two integer multi-commodity flow problem formulations: Most-flows IMCF, to select the maximum number of streaming sessions that improve the providers' revenue, and Maximum-ICMF, to select the paths maximizing the bitrate for the streaming sessions considered. In this case, there are some flows that will be rejected and therefore not all the session demands will be satisfied. Moreover, the authors consider only a single-path resource allocation strategy.

Multi-path routing is a long studied research problem. The reason why we consider multi-path traffic engineering instead of single-paths lays in routing robustness, low latency and load balancing for better performance [14]. A well-known property in multi-path routing optimization states that the maximum number of actually utilized paths is limited by the number of session demands (D) plus the number of links (L) [15]. Consequently, when dealing with multi-path routing problems, the optimal solution is achieved considering at most $D + L$ paths, where $D + L$ represents an upper bound.

In [3], Georgopoulos et al. 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 switch. Each video session is assigned to one network slice whose size is obtained by solving a max-min fairness problem. However, the work is limited to single bottlenecks and cannot scale to a large number of users.

We now focus on the relevant literature concerning client-side strategies to allocate resources in video streaming. Bentaleb et al. [16] describe the advantage of a client-side solution for the resource allocation problem in the network using SDN to obtain a higher scalability and per-client QoE. In [17], the same authors provide an improvement to communication overhead and client heterogeneity called SDNHAS, which is an intelligent streaming architecture helping HAS players to make efficient adaptation decisions. This work also presents a clustering of the clients that allows a large-scale network implementation yet not considering a grouping of videos. SDN is considered also in [18] along with a mixed integer linear program for optimal network resource allocation in live video streaming but without taking into account QoE-fairness.

In [19], a hybrid control system for video bitrate maximization, playback interruptions avoidance and video bitrate switches minimization is developed. Such a type of control affects positively the QoE since it optimizes the video bitrate and avoids rebuffering events in the case of a single bottleneck. The same authors in [4] compare different Adaptive Bitrate Algorithms (ABR) and analyse two possible allocation strategies: *network slicing* (or bandwidth reservation) and *bitrate guidance*. The first strategy assigns video flows to network slices whose size is determined by solving an optimization problem; the second strategy employs DASH Assisting Network Elements (DANEs) to guide video clients in the choice of the video level. The paper shows that the bandwidth reservation strategy provides better results in terms of achievable video fairness. However, no stress is given to the concept of client classes and video clustering. Additionally, QoE-fairness is not considered in the optimization problem.

Recently, machine learning techniques have been employed to address the problem of QoE-fairness. Altamimi et al. [20] propose a server-side QoE-fair rate adaptation method that uses Reinforcement Learning to select the best bitrate for each client. This approach implies a cooperation between the clients sharing a bottleneck link and their server, which modifies the Media Presentation Description (MPD) files to regulate the available bitrate at one client. In [21], a Q-learning based bandwidth allocation algorithm called Q-FDBA is implemented. The authors adopt a centralized approach based on SDN framework and test it on a single bottleneck with three players. Instead, our approach is implemented in the scenario of a realistic network with several thousands concurrent users.

It is worth to stress that the aforementioned works do not decouple the resource allocation from the ABR algorithms running at the clients as done in this work, but also consider the clients' buffer occupancy and the video level selection in the optimization problem.

III. The Multi-Commodity Flow Problem

The *multi-commodity flow problem* (MCFP) is the optimization framework that we propose to perform a QoE-fair network bandwidth allocation. The term *commodity* refers to a tuple containing a source node, a destination node and a volume, which identifies the resources needed to satisfy the commodity. In the case of the network bandwidth allocation problem considered here, a commodity identifies a video session where the source node is the video server, the destination node is the client, and the volume represents the video bitrate required to achieve the maximum video quality. In general, the MCFP has the aim of maximizing a properly defined utility function with a set of constraints in order to guarantee a network resource allocation in such a way that all the commodities are optimally satisfied.

The following description of the MCFP is given using the *link-path formulation* and terminology found in [15]. 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 $e \in \mathscr{E}$ is identified by a node pair and has a capacity c_e expressed in terms of bandwidth. The commodities related to the delivery network can be represented by the set of *demands* $\mathscr{D} = \{1, 2, \ldots, D\}$, where each demand $d \in \mathscr{D}$ identifies a source-destination node pair and the corresponding *traffic volume* H_d, i.e. the required network bandwidth needed by that demand. Furthermore, a demand d can be satisfied, i.e. it receives sufficient bandwidth, 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 the source node to the destination node of demand d. As a consequence, 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 the aforementioned path flows–by means of a proper utility function–while satisfying two constraints. The first imposes that all the commodities must be brought from their sources to their destination; the second requires that the total flow on each edge must not be greater than the maximum edge capacity.

In this paper, we assume without loss of generality that the graph is fully connected, i.e. there always exists a path connecting each possible pair of nodes in $\mathscr{N}$. Such nodes represent network switches in our problem, whereas edges identify links connecting a couple of switches[2]. Each link is divided in *bandwidth slices* of an appropriate size, whose number depends on the demands in the network. The size of bandwidth slices are computed by solving a multi-path weighted α-fairness optimization problem employing the following utility function [22]:

$$U(\boldsymbol{X}) = \begin{cases} \sum_d w_d \log X_d & \text{if } \alpha = 1 \\ \sum_d w_d \frac{X_d^{1-\alpha}}{1-\alpha} & \text{otherwise} \end{cases} \quad (1)$$

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. 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 α in the interval $[0, +\infty]$ [22]. In particular, when $\alpha = 0$, the link utilization is maximized with no consideration for 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. By setting $\alpha = 1$, the optimization problem results in the so called *Proportional Fairness* (PF) [23], which ensures a good trade-off between fairness and link utilization. Therefore, in this paper we explore the proportional fair case ($\alpha = 1$) and leave to future studies a performance comparison for different values of α. In the PF case, it results that $U(\boldsymbol{X}) = \sum_d w_d \log X_d$. Let us now derive the optimization problem that will be considered throughout this work. To this end, we will start with the single-path case, i.e. the easier case in which the demand volume H_d can be routed only on one possible path connecting the source node to the destination node; in other words $|\mathscr{P}_d| = 1 \; \forall \, d \in D$ and $X_d = x_d$. We will then pass onto the analysis of the multi-path optimization problem in order to carry out a comparison of the performance of the two problems in terms of QoE-fairness.

MCFP single-path weighted proportional fair optimization problem:

$$\text{Maximize} \sum\nolimits_d w_d \log x_d \quad (2)$$

$$\text{s.t.} \sum_d \delta_{ed} x_d \leq c_e, \; \forall e \in \mathscr{E} \quad (3)$$

$$x_d \leq H_d \quad (4)$$

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

In (3), δ_{ed} is the link-path indicator, which is equal to 1 if the demand d uses the link e, otherwise it is set to 0. The constraints (3) are imposed to respect the capacity of each link c_e, i.e. the sum of all the path flows x_d using link e should not exceed the capacity of that link. The last constraints (4) ensure that the total bandwidth x_d allocated for demand d is bounded by the demand traffic estimation given by H_d. This condition depends on the network traffic load: in the best case (no network bottlenecks) the total bandwidth x_d allocated to the demand d should be equal to the requested traffic volume H_d. However, in general, in the presence of bottlenecks, the total bandwidth x_d allocated to a demand is less than H_d. Notice that without this constraint, it could happen that a demand d is assigned with a bandwidth x_d greater than H_d, which would imply a waste of bandwidth.

The optimization problem shown above 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 ([24], [15]). In the Appendix, we use the theory of [15] to try and derive a closed form expression of the solution of Problem (2)-(4) and to understand the relationship between the optimization variables and the parameters involved. We start from the easier case of a single-path *uncapacitated* problem, i.e., when link capacities are to be sized, and then we consider the case when link capacities are given (*capacitated* problem).

Let us now analyse the more general case of a multi-path weighted proportional fair optimization.

MCFP multi-path weighted proportional fair optimization problem:

$$\text{Maximize} \sum\nolimits_d w_d \log(\sum_p x_{dp}) \quad (5)$$

$$\text{s.t.} \sum_d \sum_p \delta_{edp} x_{dp} \leq c_e, \; \forall e \in \mathscr{E} \quad (6)$$

$$X_d \leq H_d \quad (7)$$

where $X_d = \sum_p x_{dp}$ is the total bandwidth allocated for demand d. Also in this case, the constraints (6) 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. Constraints (7) ensure that X_d is bounded by the demand traffic estimation given by H_d.

It is straightforward to show that also Problem (5)-(7) is convex, thus implying the existence of a unique global maximum. However, in a multi-path scenario, it is not apparently possible to derive a closed-form solution in any case due to the complexity of the problem posed.

To clarify how the resource allocation works in multi-path video delivery networks, consider the basic delivery network shown in Fig. 3. The switch identifying the source node S has to satisfy two demands coming from the destination switch D: both demands have a volume of 50 Gbps. It can be seen that there are two available paths connecting node S to node D: $p_1 = \{S, 1, D\}$ and $p_2 = \{S, 2, D\}$. Thus, the MCFP could exploit both of them to transmit the video flows composing a

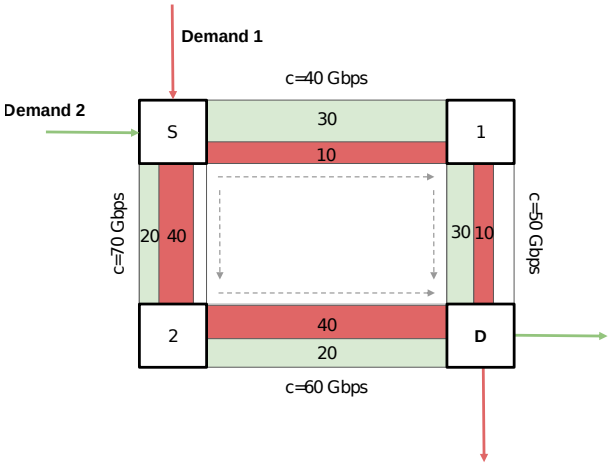


Fig. 3: Resource allocation in a sample network

demand. In this example, the solution of the MCFP for demand 1 consists of two slices: the first belonging to path p_1 with an assigned network bandwidth equal to 10 Gbps and the other to path p_2 with a bandwidth equal to 40 Gbps. The same occurs for demand 2, which obtains a 30 Gbps slice on path p_1 and 20 Gbps on path p_2. It is worth noticing that the MCFP chooses the appropriate path based on the bandwidth required by each flow composing a demand. In fact, the size of a slice associated to a particular path has to consider the available bandwidth of each link composing that specific path according to the constraint (6).

IV. The Resource Allocation Strategy

In the following, we introduce 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 adapt the MCFP (Problem (5)-(7)) to achieve such a goal. This also translates in designing the demand weights w_d so that the maximization of (5) results in a QoE-fair resource allocation. In order to compute such demand weights, we relate them to a utility function mapping the relationship between the network bandwidth assigned to a video session and the obtainable visual quality.

A. Definitions

Given a video catalog $\mathscr{V} = \{v_1, \ldots, v_V\}$, the DASH standard requires that each video $v \in V$ 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. Different videos can present remarkably different sets of encoding bitrate depending on the video content. In practice, at the client side, the ABR algorithm 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. Even though in this paper we do not focus on a specific ABR algorithm, 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

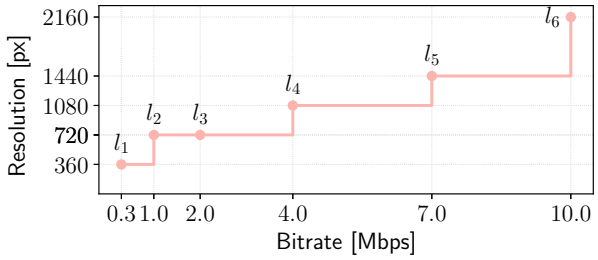


Fig. 4: Video level set representation in a *ladder* graph

assumption since all well-designed ABR algorithms are in practice implemented in this way (see for instance [25]).

The first important definition concerns a *video request* t, which is a couple (v, c), where $v \in \mathscr{V}$ and c is the *user class* belonging to the set $\mathscr{C} = \{c_1, c_2, \ldots, c_C\}$. The classification of users should be performed according to parameters having an impact on the obtainable QoE. Since the screen resolution is one of the most important parameters affecting the QoE, we propose to classify users based on their maximum screen resolution. It is important to point out that two devices with a different screen size such as a phone and a TV could have the same maximum screen resolution but their screen size can affect the QoE, especially when a video is streamed at low bitrates. In this paper, we make the assumption that each user in a client class has the same screen size, and leave to future studies the optimization of the QoE also wrt the screen size as highlighted in [26], [27], [28]. As a consequence, the terms "user class", "user screen resolution" and "user screen size" are used interchangeably in this work. Notice that, with this notation, a video request t denotes which video v a user having a client resolution c is willing to consume.

The set $\mathscr{L}_t$ for each video request $t = (v, c)$ is defined as $\mathscr{L}_t = \{l \in \mathscr{L}_v : r \leq c\} \subseteq \mathscr{L}_v$. In other words, $\mathscr{L}_t$ contains the levels of $\mathscr{L}_v$ whose resolution is not higher than c. 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 ABR algorithm will actually choose only the levels contained in the set $\mathscr{L}_t$. Notice that this is how ABR algorithms work in practice. Although they can choose among all possible video levels, those having resolutions higher than the user screen resolution are usually never selected mainly because it would increase the bandwidth consumption without producing a perceivable improvement in terms of visual quality.

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 $\bar{b}_t$, which indicates the minimum bitrate necessary to obtain visual quality equal to 1. As an example, let us consider a 4K video v (resolution 2160p) being encoded into six video levels $l = (b, r) \in \mathscr{L}_v$ as shown in Fig. 4, where $\mathscr{L}_v = \{$(0.3,360), (1,720), (2,720), (4,1080), (7,1440), (10,2160)$\}$ (the bitrate is expressed in Mbps). If we consider a video request $t = (v, 720\text{p})$, i.e. a user with a 720p screen requesting the video v, then the level set $\mathscr{L}_t$ and the reference level $\bar{l}_t$ would be respectively $\mathscr{L}_t = \{(0.3, 360), (1, 720), (2, 720)\}$ and $(2\,\text{Mbps}, 720\text{p})$.

Let us now define a *video session* as the tuple $(\mathrm{src}, \mathrm{dst}, t)$, where $\mathrm{src} \in \mathscr{N}$ is the switch the server delivering the requested video is connected to; $\mathrm{dst} \in \mathscr{N}$ 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 video sessions represented by the same tuple $(\mathrm{src}, \mathrm{dst}, t)$. In other words, a demand d contains all the video sessions from the same source node src to the same destination node dst associated to clients with the same video resolution c and requesting the same video content v. Consequently, if there are n_d video sessions with the same tuple $(\mathrm{src}, \mathrm{dst}, t)$, 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. 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 this case, if the constraint (7) 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 those demands will obtain a bandwidth share less than the bitrate associated to the reference level $\bar{l}_t$. This means that the ABR algorithm will select a video level with a lower resolution, thus obtaining a degraded video quality. In the following, we describe how to measure the visual quality as a function of the allocated network bandwidth share.

B. Measuring the visual quality

The proposed resource allocation strategy has the main goal of achieving a fair level of QoE among users. Such an objective is reached through the allocation of network resources using a multi-path approach. For this reason, a mapping between the allocated network bandwidth related to a video session and the achieved QoE is needed ([29], [30]). Such a mapping will be the reasonable base to design appropriate demand weights w_d that allow to solve Problem (5)–(7) 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 $\mathscr{V}$. 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 metadata.

The visual quality of a video $v \in \mathscr{V}$ is measured in the following way: for each video $v \in \mathscr{V}$, 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] The procedure is shown in Algorithm 1 and the output for a sample video is displayed in Fig. 1 in the case of a level set composed of 7 elements and a client class set $\mathscr{C} = \{720\mathrm{p}, 1080\mathrm{p}, 2160\mathrm{p}\}$, which contains some of the most common device resolutions. After fixing 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 IV-A. Then, for each video level $l \in \mathscr{L}_v$, the video quality is computed using a Full-Reference Video Quality (FRVQ) assessment tool such as, f.i., the Structural SIMilarity (SSIM) [31], the Peak Signal to Noise Ratio (PSNR), or the Video Multi-method Assessment Fusion (VMAF) [32], 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 is performed in Lines 6–7. In particular, to obtain the degraded video $\tilde{l}$ (Line 6), the video level l is up-scaled to the client device resolution. This reflects the way actual video streaming players typically work: if the user device has a given screen resolution (say 1080p) the ABR algorithm will select video representations characterized by a resolution up to the client screen resolution (i.e., 1080p in the example). The rationale is that rendering on the screen a video having a resolution higher than the screen resolution would not lead to an improved video quality. This is particularly evident in Fig. 1 where the visual quality (measured as VMAF score) is shown to saturate at 1 when the video representation has a resolution higher than the one of the user screen device. Then, the FRVQ assessment tool estimates the video quality by comparing the degraded video $\tilde{l}$ with the reference video $\bar{l}_t$ (Line 7). 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 provided 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.

Algorithm 1 Visual quality measurement for a video

```
1: for each client class c ∈ 𝒞 do
2:     t ← (v, c)
3:     Select reference level l̄_t from ℒ_t
4:     for each l ∈ ℒ_v do
5:         if l ∈ ℒ_t then
6:             l̃ ← Upscale l to c resolution
7:             Q_t(l) ← FRVQ(l̃, l̄_t)
8:         else
9:             Q_t(l) ← 1
10:        end if
11:    end for
12: end for
```

C. Demand weights computation

A proper computation of the demand weights w_d used in (5) is of high importance due to the fact that the solution of Problem (5)–(7) will result in the optimum QoE-fair (rather than

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

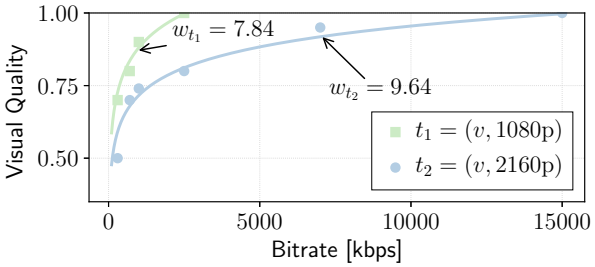


Fig. 5: Computation of weights

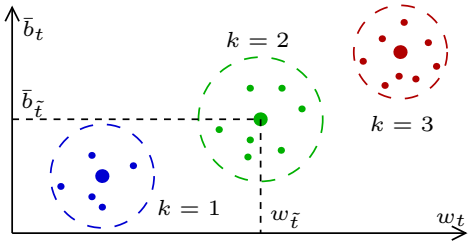


Fig. 6: Proposed clustering procedure

a throughput-fair) allocation of resources. From Section III, 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 in order to obtain a higher bandwidth for demands associated to users with high resolution screens and lower bandwidth for users with low resolution screens.

Notice that expression (5) is a weighted sum of logarithms, where the bandwidth slice X_d is the optimization variable while the weights are constant. This is due to the fact that the weights only depend on the demand d, which is not variable in the optimization problem. It is also important to stress that the weight w_d associated to a demand $d = (\text{src}, \text{dst}, t)$ 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 containing the same video request t, the weights associated to them will coincide, namely $w_{d_1} = w_{d_2} = w_t$.

Let us now define the couples (x_i, y_i) for $i = 1, \dots, L_t$ ($L_t = |\mathscr{L}_t|$) where $x_i = b_i \in \mathscr{B}_v$ and y_i is obtained through the mapping Q_t as described in Algorithm 1, 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 tuned. It is easy to show that this proposed procedure assigns weights that increase as the user device resolution becomes higher (Fig. 5), which is exactly what is needed to provide higher network bandwidth shares to users with high resolution.

D. Video clustering

The previous sections provide the necessary information to compute all the inputs to the optimization Problem (5)–(7). Recall that the goal of the optimization problem is to find the path flows x_{dp} such that the QoE-aware objective function (5) is maximized, i.e. the resources are distributed to provide a fair level of visual quality to heterogeneous users. In the multi-path case, each demand d is split 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 $(\text{src}, \text{dst}, t) \in \mathscr{N} \times \mathscr{N} \times \mathscr{T}$, it follows that $D = N \cdot (N-1) \cdot T$. 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 face 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 $\{\mathscr{V}_1^c, \dots, \mathscr{V}_K^c\}$ according to a clustering algorithm, with $\mathscr{K} = \{1, \dots, K\}$ the set of the video cluster indexes. Since K is a design parameter, it can be chosen such that $K \ll V$. As a consequence, we can 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 $t = (v, c)$, whose video v is 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 $(\text{src}, \text{dst}, \tilde{t})$, 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 $K \ll V$.

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 IV-C and $\bar{b}_t$ is the associated reference video level bitrate. Fig. 6 shows an example of how the couples $(w_t, \bar{b}_t)$ 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. 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 Fig. 6. Thus, the medoid of cluster k obtained for the user class c is representative of the traffic class $\tilde{t} = (k, c)$, i.e., of all the videos in that cluster. Therefore, it is easy to associate to each $\tilde{t}$ the weight $w_{\tilde{t}}$ and the bandwidth $\bar{b}_{\tilde{t}}$, which are the coordinates of the medoid. As an example, consider the cluster $k = 2$ in Fig. 6. The traffic class $\tilde{t} = (2, c)$ is associated with the weight $w_{\tilde{t}}$ and bandwidth $\bar{b}_{\tilde{t}}$ that are the coordinates of the medoid of cluster $k = 2$ (large green dot). It follows that changing the number of clusters K implies a trade-off between

[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.

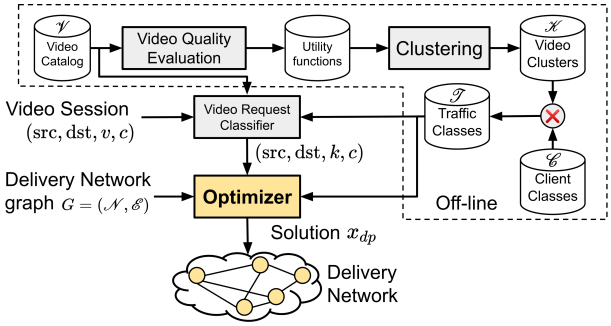


Fig. 7: The proposed Video Control Plane

the number of variables involved in the optimization problem and the obtainable QoE-fairness, as explained further on. It is important to point out that also other clustering algorithms have been considered (i.e., k-means and spectral clustering) but since they provide worse results, they have been omitted. Notice that the methodology proposed in this paper is in fact applicable with any clustering approach.

Fig. 7 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 IV-C are employed to 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 we have discussed so far. The solution is implemented through programmable network elements such as SDN switches.

V. Results

In this section we perform a clusterization of video requests in order to effectively implement the *QoE-Proportional Fair* (PF) multi-path optimization problem described in Section III. This allows us to carry out a performance evaluation of the proposed allocation strategy via simulations. In Section V-A we provide the general setting used for the experimental evaluations; in Section V-B we show the QoE fairness obtained with the proposed method and the corresponding average visual quality; finally, in Section V-C we give an overview of the time complexity of the algorithm employed.

A. General Setting

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.

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 proposed PF allocation strategy in a simulator composed of three modules implementing realistic scenarios of typical video distribution networks. The first module is the *video session generator* that, after receiving the network graph G, the video catalog $\mathscr{V}$, and the set of user classes $\mathscr{C}$ as inputs, randomly generates a configurable number of video sessions $(\mathrm{src}, \mathrm{dst}, t)$. The second module is the solver, that employs the CVXPY Python tool [33] to implement Problem (5)-(7) by making use of the *Splitting Conic Solver* (SCS)[5] [34]. Once the optimization problem is solved, the third module, called *QoE evaluator*, computes the obtained QoE for each video session $(\mathrm{src}, \mathrm{dst}, t)$ 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 [35]:

$$F = 1 - 2\sigma$$

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 are served exactly with the same visual quality.

We have chosen to use YouTube videos in order to build a realistic video catalog. In particular, we have built a video catalog of $\sim$200 heterogeneous videos all with a maximum resolution equal to 4K fetched from YouTube on which we have performed the visual quality measurement reported in Algorithm 1. It is worth to remark that we do not re-encode the videos fetched from YouTube. Thus, the bitrate ladder of a given video is the one set by YouTube at time of encoding. In particular, for each video we select the six video representations encoded at resolutions 360p, 480p, 720p, 1080p, 1440p, 2160p and made available by YouTube (an example of bitrate ladder is given in Fig. 4). We have employed the VMAF metric to compute the video-level/video-quality mapping Q_t as described in Section IV-B. 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 – belonging to the set $\mathscr{C} = \{720\mathrm{p}, 1080\mathrm{p}, 2160\mathrm{p}\}$. The *load* values considered to generate the video sessions range in the set $\{100, 200, 300, 400, 500\}$ Gbps while two networks have been chosen as delivery network topologies in order to make a comparison as well as a performance evaluation. The first network topology is the GARR network[7], composed of 61 switches and 73 links with an average capacity of $\sim$4 Gbps. We fixed 10 switches as server nodes and the remaining 51 as clients. The second network topology, the Abilene network, is smaller sized, being it composed of 11 switches and 14 links. In this case we set 4 switches as server nodes and 7 as

[5]https://github.com/cvxgrp/scs
[6]https://github.com/Netflix/vmaf
[7]http://www.topology-zoo.org/files/Garr201201.gml

switches used by the clients. Finally, the set of clusters is such that $K \in \{3, 5\}$, the set of paths is such that $P \in \{1, 2, 5\}$ and the weight parameter β can be chosen among the values in the set $\{1.1, 1.2, 1.3, 1.4, 1.5\}$.

B. QoE-Fairness vs Average Visual Quality

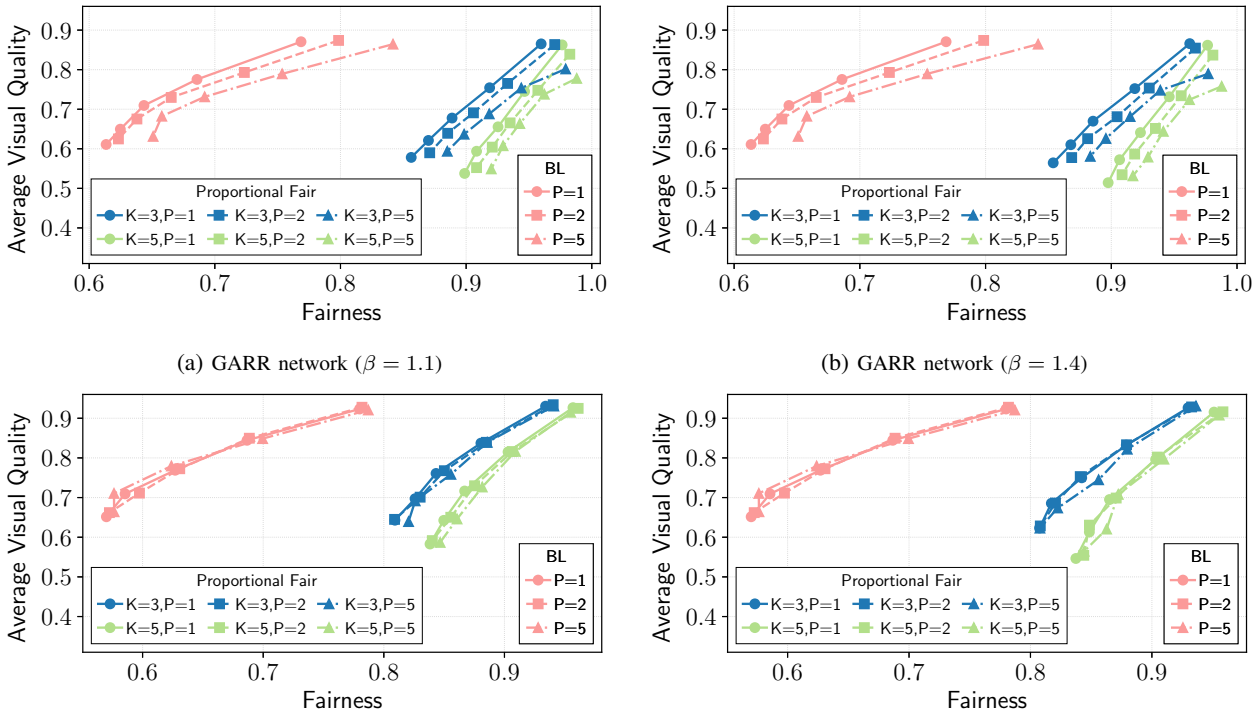
Let us start our investigation by comparing the obtained results on both the considered networks. Fig. 8 shows the trade-off between the average QoE obtained by the video sessions and the corresponding QoE-fairness when BL is employed or in the case of the proposed PF resource allocation strategy. It is worth stressing 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 denotes a specific number of paths P involved in the allocation and each marker on a line is representative of a specific load in $\{100, 200, 300, 400, 500\}$ Gbps. In the PF case, different colors indicate a different number of clusters K. Moreover, for space constraints, Fig. 8 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 to a lower visual quality. Consider Fig. 8a 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 it 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 is basically unchanged compared to the BL case. Although this result may appear unexpected, we need to keep in mind that we are referring to the *average* visual quality and that also in the BL case the QoE is maximized, but without considering the fairness. This leads to a situation in which, in the BL case, some users experience a high level of QoE while others have a low level of it. The average of all these values results similar to the average of the values obtained in the PF case, in which fairness is considered and therefore the QoE of users is more uniform. In this way, even though the average QoE levels may be similar, the fairness is not. 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 cases in the figures, where $K = 5$ clusters appears to be the best trade-off between average visual quality and QoE fairness.

Next, consider Fig. 8a and 8b related to the GARR network. By varying β from 1.1 to 1.4, the average visual quality slightly drops while the average fairness remains almost unchanged. This is due to the fact that the fairness among user classes is increasing–as will be better shown in the following–, and this implies a graceful degradation of the visual quality of all the users in the network. 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. Therefore, when passing from Abilene to a wider and more complex network such as GARR, the multi-path approach outperforms the single-path case.

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). Fig. 9 reports the CDF of the visual quality obtained by all video sessions grouped by user class, i.e. the maximum screen resolution of users, in the case of a 500 Gbps load (results for different loads are similar). It is important to point out that the resolutions reported in the figures are used only to identify the user classes. In other words, they represent the maximum screen resolution of users in a class, but the actual resolution of each user during playback can be lower. The figure shows that, regardless of the chosen network, when passing from $\beta = 1.1$ to $\beta = 1.4$ (Fig. 9b and 9c or Fig. 9e and 9f) the MCFP gives a better performance in terms of fairness. Moreover, the obtained fairness in the PF case with $\beta = 1.4$ is remarkably better than the BL case. In fact, the median value of the visual quality for clients with 720p, 1080p and 2160p resolution in Fig. 9a is respectively $0.45, 0.7, 0.8$, while for the PF case in Fig. 9c the values are $0.48, 0.5, 0.5$. As a result, the PF case with $\beta = 1.4$ guarantees a high level of fairness since all the users belonging to any user class will enjoy a similar visual quality. However, an increase of fairness will unavoidably imply an overall decrease in the average visual quality as shown in Fig. 8. The same considerations could be made for the Abilene network. It is worth stressing that the simulations for $\beta = 1.5$ are not shown because in the case of the GARR network the results in terms of visual quality fairness deteriorate compared to the case in which $\beta = 1.4$. In particular, for $\beta = 1.5$, clients with 720p and 1080p resolution obtain a worse visual quality than 2160p clients. However, in the Abilene network, the fairness improves, i.e., the three curves are more closely aligned, thus making $\beta = 1.5$ preferable to the previous values. For this reason, β is a parameter that has to be properly tuned according to the considered network topology.

C. Computation time

In this subsection, we provide some insights into the computation time required to solve the proposed optimization problem. Notice that, since ABR algorithms are independent and free to run at the clients, the optimizer can be run every Δt seconds, where usually Δt is significantly larger than a single video segment (which is in general in the order of 1-10s) and must be greater than the time spent to solve the optimization problem. In fact, the optimization loop can be seen as the *outer loop* that sets the bandwidth slice for video flows, while the *inner loop* is represented by the ABR algorithm that selects the video representation at a chunk time scale [4]. Regarding this architecture, named *cascaded control system*, a well-known practice used in controlling such systems entails separating the time scales at which each control loop works. The idea is to have the outer loop work at a higher sampling time and



(a) GARR network ($\beta = 1.1$)

(b) GARR network ($\beta = 1.4$)

(c) Abilene network ($\beta = 1.1$)

(d) Abilene network ($\beta = 1.4$)

Fig. 8: QoE-Fairness vs Average Visual Quality

the inner loop to run faster at a lower sampling time. This is motivated by the fact that, with such a control architecture, having two controllers working at around the same sampling time might provoke adverse effects on stability. Thus, we argue that a sampling time that is greater than 4-5 times the duration of a segment (∼25 seconds) is *required* to enforce the aforementioned separation of time scales.

Fig. 10 shows the computation time needed to solve the MCFP in the case of Abilene and GARR networks. As expected, the time complexity increases when passing from 3 to 5 clusters and from 100 to 500 Gbps for both the considered networks. Moreover, the time required to solve the optimization problem is higher (∼4x) in the case of the GARR network compared to Abilene due to the larger number of nodes of the GARR network. It is worth noting that the computation times are obtained using only one CPU core of an i7 workstation using an open-source tool, thus one can expect a significant speed-up using a commercial optimizer that supports multi-threading. Nevertheless, it is important to notice that, even in this non-ideal operating scenario, the obtained worst case for a large network is ∼50s which is in the order of ∼10 video chunks.

VI. 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.

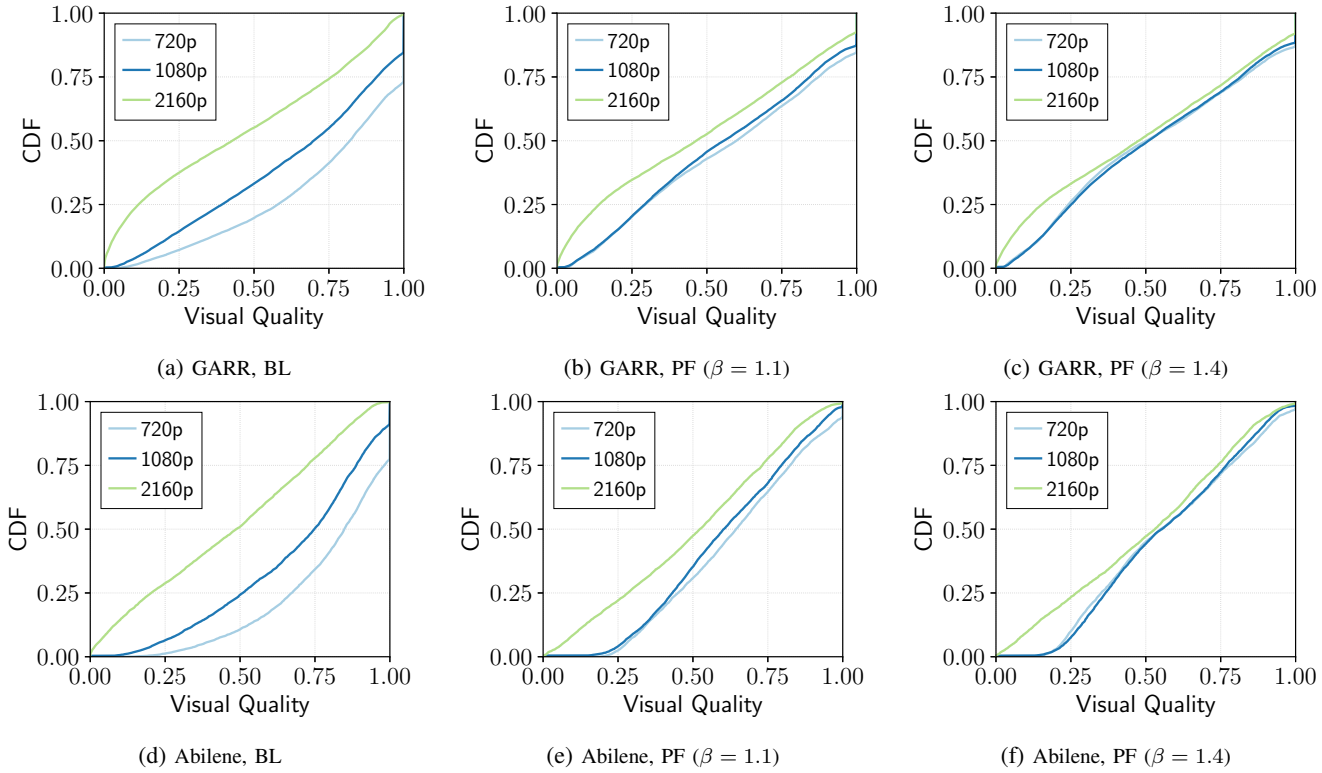
It is important to precise that the clustering process depends on the video catalog that may change many times and therefore a continuous update of the clusters should be guaranteed. To this end, future work will focus on an in-depth analysis of the video clustering procedure. Another interesting future development could be implementing a decentralized technique to solve the optimization problem to improve scalability-related issues as well as resiliency to computing failures, which could be caused by malicious users attacking the node where the optimizer is located. Furthermore, we plan to carry out this analysis with a more comprehensive QoE function, which not only depends on the visual quality, but also on rebuffering events and several other features that are still object of research. One last aspect worth being investigated is to see how the QoE changes according to the screen size of the device, which is independent of its video resolution and could be the cause of further QoE unfairness.

References

[1] V. Cisco, "Cisco visual networking index: Forecast and trends, 2017–2022," *White Paper*, 2018.

[2] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," *IEEE MultiMedia*, vol. 18, no. 4, pp. 62–67, 2011.

[3] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. 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*, 2013, pp. 15–20.



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

(d) Abilene, BL (e) Abilene, PF ($\beta = 1.1$) (f) Abilene, PF ($\beta = 1.4$)

Fig. 9: CDF of visual quality for different user classes and distribution networks in the case of a 500Gbps load

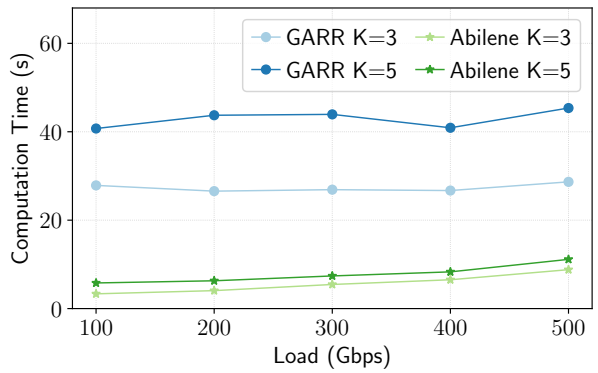


Fig. 10: Computation time required to solve the MCFP

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

[5] S. Poojary, R. El-Azouzi, E. Altman, A. Sunny, I. Triki, M. Haddad, T. Jimenez, S. Valentin, and D. Tsilimantos, "Analysis of qoe for adaptive video streaming over wireless networks," in *2018 16th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*. IEEE, 2018, pp. 1–8.

[6] N. Eswara, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya, "Perceptual qoe-optimal resource allocation for adaptive video streaming," *IEEE Transactions on Broadcasting*, vol. 66, no. 2, pp. 346–358, 2019.

[7] C. Chen, X. Zhu, G. de Veciana, A. C. Bovik, and R. W. Heath, "Rate adaptation and admission control for video transmission with subjective quality constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 1, pp. 22–36, 2014.

[8] V. Joseph, S. Borst, and M. I. Reiman, "Optimal rate allocation for adaptive wireless video streaming in networks with user dynamics," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 406–414.

[9] Q. Ai, Y. Ji, L. Zhong, P. Wang, and F. Liu, "Qoe-based cross-layer resource allocation for video streaming in high speed downlink access," in *2012 8th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2012, pp. 1011–1016.

[10] R.-I. Chang, Y.-C. Liu, J.-M. Ho, Y.-H. Chu, W.-C. Chung, and C.-J. Wu, "Optimal scheduling of qoe-aware http adaptive streaming," in *IEEE TENCON 2015*. IEEE, 2015, pp. 1–4.

[11] S. Ramakrishnan, X. Zhu, F. Chan, and K. Kambhatla, "Sdn based qoe optimization for http-based adaptive video streaming," in *2015 IEEE International Symposium on Multimedia*, 2015, pp. 120–123.

[12] J. W. Kleinrouweler, S. Cabrero, and P. Cesar, "Delivering stable high-quality video: An SDN architecture with DASH assisting network elements," in *Proc. ACM Conference on Multimedia Systems*, 2016, p. 4.

[13] A. Samani and M. Wang, "Maxstream: Sdn-based flow maximization for video streaming with qos enhancement," in *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*. IEEE, 2018, pp. 287–290.

[14] P. Kumar, Y. Yuan, C. Yu, N. Foster, R. Kleinberg, P. Lapukhov, C. L. Lim, and R. Soulé, "Semi-oblivious traffic engineering: The road not taken," in *Proc. USENIX NSDI '18*, 2018, pp. 157–170.

[15] M. Pióro and D. Medhi, *Routing, flow, and capacity design in communication and computer networks*. Elsevier, 2004.

[16] A. Bentaleb, A. C. Begen, and R. Zimmermann, "Sdndash: Improving qoe of http adaptive streaming using software defined networking," in *Proc. ACM Multimedia*. ACM, 2016, pp. 1296–1305.

[17] A. Bentaleb, A. C. Begen, R. Zimmermann, and S. Harous, "Sdnhas: An sdn-enabled architecture to optimize qoe in http adaptive streaming," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2136–2151, 2017.

[18] A. Erfanian, F. Tashtarian, A. Zabrovskiy, C. Timmerer, and H. Hellwagner, "Oscar: On optimizing resource utilization in live video streaming," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 552–569, 2021.

[19] G. Cofano, L. De Cicco, and S. Mascolo, "A hybrid model of adaptive video streaming control systems," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 6601–6606.

[20] S. Altamimi and S. Shirmohammadi, "Qoe-fair dash video streaming using server-side reinforcement learning," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 16, no. 2s, pp. 1–21, 2020.

[21] J. Jiang, L. Hu, P. Hao, R. Sun, J. Hu, and H. Li, "Q-fdba: improving

qoe fairness for video streaming," *Multimedia Tools and Applications*, vol. 77, no. 9, pp. 10 787–10 806, 2018.
[22] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on networking*, no. 5, pp. 556–567, 2000.
[23] J. F. Nash Jr, "The bargaining problem," *Econometrica: Journal of the Econometric Society*, pp. 155–162, 1950.
[24] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-Hall International New Jersey, 1992, vol. 2.
[25] G. Cofano, L. De Cicco, and S. Mascolo, "Modeling and design of adaptive video streaming control systems," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 1, pp. 548–559, 2018.
[26] N. Eswara, K. Manasa, A. Kommineni, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya, "A continuous qoe evaluation framework for video streaming over http," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3236–3250, 2017.
[27] D. Ghadiyaram, J. Pan, and A. C. Bovik, "Learning a continuous-time streaming video qoe model," *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2257–2271, 2018.
[28] A. Floris, L. Atzori, G. Ginesu, and D. D. Giusto, "Qoe assessment of multimedia video consumption on tablet devices," in *2012 IEEE Globecom Workshops*. IEEE, 2012, pp. 1329–1334.
[29] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *IEEE Network*, vol. 24, no. 2, pp. 36–41, 2010.
[30] 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)*, 2015, pp. 2019–2023.
[31] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
[32] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, "Toward a practical perceptual video quality metric," *The Netflix Tech Blog*, vol. 6, 2016.
[33] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.
[34] 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*, vol. 169, no. 3, pp. 1042–1068, June 2016. [Online]. Available: http://stanford.edu/ boyd/papers/scs.html
[35] T. Hossfeld, L. Skorin-Kapov, P. E. Heegaard, and M. Varela, "Definition of QoE Fairness in Shared Systems," *IEEE Communications Letters*, vol. 21, no. 1, pp. 184–187, Jan 2017.

APPENDIX

This section makes use of the theory in [15] to clarify the relationship between the optimization variables and the parameters involved in Problem (2)-(4). We start from a single-path *uncapacitated* problem, i.e., when link capacities are to be sized, and then we consider our case, i.e., when link capacities are fixed (*capacitated* problem).

Proposition (Uncapacitated case): Suppose that the link capacities c_e of the network are not given a priori but need to be sized and that we want to set the total network capacity to a value Z, i.e., $\sum_e c_e = Z$, where Z is a constant.If constraints (4) are always satisfied, then denoting with $\boldsymbol{x}^* = [x_1^*, x_2^*, \dots, x_D^*]^T$ the optimal solution of Problem (2)-(4) and with $F(\boldsymbol{x})$ the objective function in (2), i.e., $F(\boldsymbol{x}) = \sum_d w_d \log x_d$, it results that

$$F(\boldsymbol{x}^*) = (\log Z)\sum_d w_d - \sum_d w_d \log(\sum_e \delta_{ed}) + \sum_d w_d \log w_d - (\log(\sum_d w_d))\sum_d w_d$$

$$x_d^* = \frac{Z}{(\sum_e \delta_{ed})}\frac{w_d}{(\sum_d w_d)}$$

Proof. Let us start by noticing that in the optimal case, the inequalities related to (3) become equalities, so it follows that $\sum_e \sum_d \delta_{ed} x_d = Z$. If we then suppose that the constraints in (4) are always satisfied, the Lagrangian function associated to the optimization problem is

$$L(x, \sigma) = -\sum_d w_d \log x_d + \sigma(\sum_e \sum_d \delta_{ed} x_d - Z)$$

where σ denotes the Lagrangian multiplier. The function can be rewritten as follows

$$L(x, \sigma) = \sum_d ((\sigma \sum_e \delta_{ed}) x_d - w_d \log x_d) - \sigma Z$$

In this way, we can define the dual objective function:

$$W(\sigma) = \min_{x \geq 0} L(x, \sigma)$$

By fixing σ, differentiating w.r.t. $x_d(\sigma)$ and setting the resulting derivatives equal to zero, we obtain:

$$\frac{w_d}{x_d(\sigma)} - \sigma \sum_e \delta_{ed} = 0$$

that produces

$$x_d(\sigma) = \frac{w_d}{\sigma \sum_e \delta_{ed}}$$

If we substitute the previous expression in $L(x, \sigma)$, it results that

$$W(\sigma) = \sum_d (w_d - w_d \log \frac{w_d}{\sigma \sum_e \delta_{ed}}) - \sigma Z$$

In order to compute the Lagrangian multiplier σ, we differentiate $W(\sigma)$ and find the stationary point.

$$\sum_d \frac{w_d}{\sigma} - Z = 0 \Rightarrow \sigma^* = \sum_d \frac{w_d}{Z}$$

from which $F(\boldsymbol{x}^*)$ and x_d^* follow.

Let us now take into account constraints (4). In this case we get

$$x_d(\sigma) = \begin{cases} \frac{w_d}{\sigma \sum_e \delta_{ed}} & \text{if } 0 < \frac{w_d}{\sigma \sum_e \delta_{ed}} \leq H_d \\ H_d & \text{if } \frac{w_d}{\sigma \sum_e \delta_{ed}} > H_d \end{cases}$$

The threshold value for σ is

$$\bar{\sigma}_d = \frac{w_d}{H_d \sum_e \delta_{ed}}$$

Therefore, $x_d(\sigma)$ can be rewritten as

$$x_d(\sigma) = \begin{cases} \frac{w_d}{\sigma \sum_e \delta_{ed}} & \text{if } \sigma \geq \bar{\sigma}_d \\ H_d & \text{if } 0 \leq \sigma < \bar{\sigma}_d \end{cases} \quad (8)$$

Let us now sort all $\bar{\sigma}_d$'s in a non-decreasing order and if some of the elements are equal, we keep just one of them until we get the sequence $(s_1, s_2, \dots, s_n)$ s.t. $s_1 < s_2 < \dots < s_n$, where we set $s_1 = 0$ and $s_n = +\infty$. Then, we can form $n-1$

intervals $[s_1, s_2], [s_2, s_3], \ldots, [s_{n-1}, s_n)$ and for each of them we can define two disjoint sets of demands:

$$F_j = \{d : x_d(\sigma) = \frac{w_d}{\sigma\gamma_d} \text{ for } \sigma \in [s_j, s_{j+1}]\}$$
$$U_j = \{d : x_d(\sigma) = H_d \text{ for } \sigma \in [s_j, s_{j+1}]\}$$

where $\gamma_d = \sum_e \delta_{ed}$ and $F_j \cup U_j = \{1, 2, \ldots, D\}$ for $j = 1, 2, \ldots, n-1$. For each j identifying the interval $[s_j, s_{j+1}]$, the set F_j contains all the demands whose associated flow x_d is such that $x_d \leq H_d$ when $\sigma \in [s_j, s_{j+1}]$. On the contrary, U_j is the set of the demands whose flows satisfy $x_d \geq H_d$ when $\sigma \in [s_j, s_{j+1}]$. This allows us to partition the demands in each interval according to constraints (4) and as a result, in each $[s_j, s_{j+1}]$, the dual function can be written as follows:

$$W(\sigma) = \sum_{d \in F_j} (w_d - w_d \log \frac{w_d}{\sigma\gamma_d}) - \sigma(Z - \sum_{d \in U_j} \gamma_d H_d) - \sum_{d \in U_j} w_d \log H_d$$

This function is continuous and differentiable. In fact, in the interval $[s_j, s_{j+1}]$, the first derivative is

$$W'(\sigma) = Z - \sum_{d \in U_j} \gamma_d H_d + \sum_{d \in F_j} \frac{w_d}{\sigma} \tag{9}$$

For any point s_j, $j = 2, \ldots, n-1$, and given the expression in (8), it results that for all demands d changing set from U_j to F_j and vice versa in s_j, the equations $H_d = \frac{w_d}{s_j\gamma_d}$ hold. Then, when (9) is evaluated in s_j, d belongs both to U_j and F_j and the terms $\gamma_d H_d$ and $\frac{w_d}{s_j}$ cancel each other. The same happens for $W'(\sigma)$ associated to the interval $[s_{j-1}, s_j]$. Therefore, the left and right derivatives of the dual function are equal in each point s_j, $j = 2, \ldots, n-1$, thus implying the differentiability. The dual function is not differentiable twice since the second derivative is in general discontinuous at the ends of the intervals because it is equal to

$$W''(\sigma) = -\sum_{d \in F_j} \frac{w_d}{\sigma^2}$$

Nevertheless, this implies the concavity of the function, which is differentiable. As a consequence, the maximum can be computed as $W'(\sigma) = 0$, $\sigma \in [0, +\infty)$ and the resulting stationary point σ^* can only belong to one of the intervals. Only in such an interval, the resulting stationary point of the dual function associated to that interval actually belongs to the interval. This property does not hold for the stationary points computed in the other intervals. In fact, for each other interval $[s_j, s_{j+1}]$, the resulting σ^* does not belong to the interval. Therefore, the stationary point is given by:

$$\sigma^* = \begin{cases} \sum_{d \in F_j} \frac{w_d}{Z - \sum_{d \in U_j} \gamma_d H_d} & \text{if } F_j \neq \emptyset \\ \text{any } \sigma \in [0, +\infty) & \text{if } F_j = \emptyset \\ & \text{and } Z = \sum_{d \in U_j} \gamma_d H_d \\ \text{does not exist} & \text{if } F_j = \emptyset \\ & \text{and } Z \neq \sum_{d \in U_j} \gamma_d H_d \end{cases}$$

Once we get σ^*, we can compute the optimal demand flows through (8). □

When the link capacities are fixed, it is no longer guaranteed that the optimization problem admits an optimal solution s.t. all the constraints in (3) are equalities. In this case, no closed-form expression of the solution is available. On the other hand, if an optimal solution exists s.t. all the constraints in (3) are equalities, then we can prove the following.

Proposition (Capacitated case): Suppose that the link capacities c_e of the network are given a priori. If the constraints (4) are always satisfied and there exists an optimal solution $\boldsymbol{x}^* = [x_1^*, x_2^*, \ldots, x_D^*]^T$ to Problem (2)-(4) s.t. all the constraints in (3) are equalities, then it results that

$$x_d^* = \frac{w_d}{\sum_e \delta_{ed}\sigma_e(c_1, \ldots, c_{E_p}, w_1, \ldots, w_D, \delta_{11}, \ldots, \delta_{E_pD})}$$

with

$$\sum_e \delta_{ed}\sigma_e(c_1, \ldots, c_{E_p}, w_1, \ldots, w_D, \delta_{11}, \ldots, \delta_{E_pD}) > 0$$

where $e \in \mathscr{E}_p$, $\mathscr{E}_p$ is the set of all the edges that serve at least one demand, $E_p = |\mathscr{E}_p|$, and $\sigma_i(c_1, \ldots, c_{E_p}, w_1, \ldots, w_D, \delta_{11}, \ldots, \delta_{E_pD})$ means that each Lagrangian multiplier σ_i, $i = 1, \ldots, E_p$ depends in general on the link capacities c_e, $\forall\, e \in \mathscr{E}_p$, on the demand weights w_d, $\forall\, d \in \mathscr{D}$ and on δ_{ed} $\forall\, e \in \mathscr{E}_p, d \in \mathscr{D}$. Moreover it is clear that the higher the weight w_d associated to a demand d, the higher the bandwidth allocated to that demand by the optimization problem.

Proof. Since now we have E_p constraints given by (3), which we consider to be equalities by hypothesis, the Lagrangian function becomes:

$$L(x, \sigma) = \sum_d \left(\sum_e \sigma_e \delta_{ed} x_d - w_d \log x_d \right) - \sum_e \sigma_e c_e$$

where the σ_e's denote the Lagrangian multipliers. By fixing all the σ_e's, differentiating w.r.t. $x_d(\sigma)$ and setting the resulting derivatives equal to zero, we obtain:

$$x_d(\sigma) = \frac{w_d}{\sum_e \delta_{ed}\sigma_e}$$

for each demand $d \in \mathscr{D}$.

By substituting $x_d(\sigma)$ in $L(x, \sigma)$ and differentiating w.r.t. each σ_e, we obtain a system with E_p equations in E_p unknowns. Such a system admits a solution–for the hypotheses made in the Proposition–represented by the σ_e's, $e = 1, \ldots, E_p$, that depend on the link-path indicators, on the demand weights and on the link capacities. Obviously, it must result that

$$\sum_e \delta_{ed}\sigma_e > 0, \quad \forall\, d \in \mathscr{D}$$

Notice that if we include constraints (4), there is no possibility of finding a closed-form expression of the solution. □