# Distributed Rate Allocation in Switch-Based Multiparty Videoconference

Stefano D'Aronco
LTS4, Ecole Polytechnique
Fédérale de Lausanne (EPFL)
Switzerland
stefano.daronco@epfl.ch

Sergio Mena
Cisco Systems
Switzerland
semena@cisco.com

Pascal Frossard
LTS4, Ecole Polytechnique
Fédérale de Lausanne (EPFL)
Switzerland
pascal.frossard@epfl.ch

## ABSTRACT

Multiparty videoconferences, or more generally multiparty video calls, are gaining a lot of popularity as they offer a rich communication experience. These applications have however, large requirements in terms of both network and computational resources and have to deal with sets of heterogenous clients. The multiparty videoconferencing systems can be grouped in two classes. They are based either on expensive central nodes, called multipoint control units (MCU), with transcoding capabilities, or, on a peer-to-peer strategy where users help each other to distribute the different video streams. Whereas the first one requires an expensive central hardware, the second one depends completely on the redistribution capacity of the users, which sometimes might neither provide sufficient bandwidth nor be reliable enough. In this work we propose an alternative solution where we use a central node to distribute the video streams but at the same time we maintain the hardware complexity and the computational requirements of this node as low as possible. The proposed solution uses a distributed algorithm to allocate the users' rates in a Quality of Service (QoS) aware manner. The allocation algorithm is also extremely fast and is able to quickly reallocate the rates in case the conditions change. We have further implemented our solution in a network simulator where we show that our rate allocation algorithm is able to properly optimize users' QoS and adapt to dynamic changes in the system. We also illustrate the benefits of our solution in terms network usage and average utility when compared to a baseline heuristic method operating on the same system architecture.

## CCS Concepts

•**Networks** → **Application layer protocols;** *Network resources allocation;* •**Information systems** → **Multimedia streaming;** •**Mathematics of computing** → *Mixed discrete-continuous optimization;*

*MMSys'16, May 10-13, 2016, Klagenfurt, Austria*
© 2016 ACM. ISBN 978-1-4503-4297-1/16/05…$15.00
DOI: http://dx.doi.org/10.1145/2910017.2910595

## Keywords

Videoconference; rate allocation; QoS optimization

## 1. INTRODUCTION

Nowadays video conferencing applications are getting more and more popular, and this trend is expected to continue in the next years according to Cisco Visual Networking index [3]. These applications allow several users to communicate together using audio-video streams and thus to provide a rich communication experience. When the number of participants to the videoconference is large, the bandwidth requirements for sending the video data among all the endpoints might be extremely high. In the case of poor network resources it is therefore extremely important to optimize the rates of the video streams in order to provide a good Quality of Service (QoS) to the users. The video distribution among the users can be carried out using different architectures, each of them having some benefits and some disadvantages.

In the most naïve implementation all users send simultaneously their video flows directly to all the other users. If the number of participants is equal to $N$, then $N-1$ video flows share the download link of every endpoint and $N-1$ replicas of the source stream share the upload link. Since in many connections, such as asymmetrical digital subscriber line (ADSL), the upload capacity is usually much smaller than the download capacity, the main bottleneck of this architecture is usually the upload bandwidth of the endpoints. This is particularly true for large values of $N$. The simplest way to alleviate the aforementioned problem is to use a multipoint control unit (MCU) with transcoding capabilities. The MCU is an endpoint used as a communication bridge by the videoconference participants. In this case every sender sends its video only to the MCU, the MCU transcodes the video of each sender and forwards it to all the other participants [1]. Often transcodes of the same source stream are performed in order to adapt to the download bandwidth of the receivers. Since every source sends a single video flow to the MCU, this architecture alleviates the problem of the upload capacity of the endpoints. On the other hand, since the transcoding of all the streams is carried out by the MCU, this solution exhibits important scalability problems.

In order to alleviate the workload of the MCU some works proposed a peer-to-peer solution for the multiparty conference problem[5, 12, 6]. These works focus on the allocation of flows in a peer-to-peer system, which is composed of the videoconference participants, in order to maximize the overall QoS. The main idea is first to build a set of 2-hop trees using user's nodes as the central nodes of the trees, and as second step to optimize the sending rates of the trees. In [6] every user encodes its video in a single stream, and then users help each other to distribute all the streams by using multicast trees. This solution might not provide good performance in the scenario where the users' download links have heterogenous values. In [12] every user is required to encode $N-1$ video layers in order to perfectly match the requirements of the different download links. The problem with this second solution resides in the computational load required from every user for the encoding process when the number of participants is large. Moreover, although both of these methods do not require a central communication bridge, their performance strongly depends on the upload bandwidths of the users' endpoints, which, in a real scenario, might not provide sufficient bandwidth. Secondly, the rate optimization problem among the conference streams, and the congestion control (CC) problem are solved jointly. This strategy might not be the optimal choice for different reasons: it cannot guarantee optimal performance in case of network cross-traffic, and the speed of rate variations among the videoconference flows is strongly constrained by the stability of the adopted CC.

In this work we aim at designing a multiparty videoconference system that offers a good tradeoff between a fully centralized solution with transcoding capabilities and a complete peer-to-peer solution, which ultimately relies only on peers' resources. From the point of view of a videoconference service provider such an architecture is extremely appealing: it in fact provides a remarkable improvement in terms of service without the need of an expensive MCU hardware. The key points of the proposed solution are the following: i) a central node is used to enable application layer multicast communication, ii) the users can encode video streams at different rates, with the number of different rates depending on their computation and bandwidth capabilities, iii) the users' sending/receiving rates are computed distributively to achieve an optimized QoS, independently of the media congestion control algorithm in use.

The central node, also called the *switch node* in the remainder of the work, is used by the users as a communication hub. Its main functions are the following. First, it offers a video packet forwarding service. In this way every sender can implement a 2-hop application layer multicast tree for the video delivery. This is important in order to alleviate the possible limitations of the upload link bandwidth. Second, it provides a coordination service among the senders and receivers in order to reach an effective QoS aware rate allocation. The complexity of the central node is kept as low as possible due to scalability concerns. The computation of the layer rates, as well as the forwarding policies of the video packets, are computed distributively by the videoconference participants. Therefore, the computation requirements of the switch node are extremely low compared to the ones of an MCU with transcoding capabilities. The central node is thus a sort of "application layer" switch, hence the name *switch node*.

More in detail, the operation of our system is the following. Based on the activity or importance of the users and the video characteristics every receiver computes the QoS optimized receiving rates (ideal rates) from all the other participants given his bandwidth constraints. Every sender then encodes and sends its video at different rates to the switch node in order to offer the best rates according to the ideal ones, and at the same time respect the upload capacity. The receivers then select from the current available encoding rates of all the senders, the ones that maximize their own QoS under the download capacity constraints. These three steps need to be evaluated only once in order to compute the rate allocation in steady conditions, and updated every when the status of the system changes. As a result, our solution can adapt quickly to network or participant condition changes, and limit potential drops of the users' QoS. We carefully evaluate the performance of our system in a network simulator (NS3) that replicates realistic network settings. We used the NADA CC [20] to send the media data streams and, on top of it, we performed the rate allocation described above. Our experiments show that our algorithm is able to properly allocate resources for optimizing the QoS of each user, and that it reacts quickly to changes in the system. We further compare the solution with a baseline algorithm which uses the same network architecture, and we show the benefits that can be achieved by the proposed solution.

Authors in [18] analyzed how some of the existing commercial solutions (specifically: google+, ichat and skype) implement multiparty videoconferences. The analysis showed that both, the fully peer-to-peer, with a single encoded stream per sender, and the server based solution, with multiple encoded streams, are used. Another commercial solution [9] uses a simple central communication bridge in order to enable a 2-hop application layer multicast trees for the video delivery. The users encode the video at a finite set of different rates and send them to a central node. The central node then decides which layers to forward to each receiver according to the download link bandwidth. Similarly to our system, this method offers a good compromise with a reliable central node to improve communication but with no strong computational requirements for transcoding. The problem of optimal rate allocation and transmission policy are however not known nor available for all the analyzed solutions. In [10] authors develop a multiparty system with an architecture similar to the aforementioned solution and to our proposed system. Analogously to our work, this study is also motivated by the advantage and efficiency of having a simple central node with no transcoding capabilities but simply able to apply different forwarding policies to different flows. In this work, in order to limit the network usage, the central node forwards to the endpoints only a subset of the videoconference flows. The forwarding decision is made according to the users' importance level (based on the audio activity). However, as for the aforementioned commercial solution, this study does not tackle the specific problem of the bitrate selection in the presence of heterogenous bandwidths among the videoconference participants. This is one of the gaps that we aim to fill in this paper.

The remainder of the work is composed as follows. The system model and the main quantities used in this work are briefly provided in section 2. In section 3 we describe more in detail the problem to solve and in section 4 we explain the proposed solution. In section 5 we discuss more in detail how the algorithm can be implemented. In section 6 we provide some simulation results of the implemented system. Finally conclusions and future work are given in section 7.

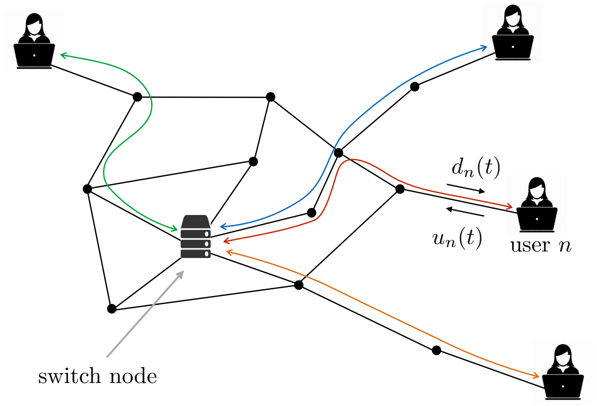

**Figure 1: System example.**

## 2. SYSTEM MODEL

We target a scenario where $N$ users participate to a videoconference. All the participants are both senders and receivers, thus the video of every user should ideally be received by all the other users. In the remainder of this work we use the words user and participant interchangeably.

The proposed system architecture is a hub topology with the hub node corresponding to a switch node. Fig. 1 depicts a simple topology example. Every user establishes a bidirectional connection with the switch node using a general CC algorithm suitable for real-time communications. Multiple streams of video data are sent from a single user node to the switch node, and vice versa. In our solution, all streams from the switch to a user share a single download session. Another session is active on the reverse path for uploading the user's video streams. In this way we can improve the responsiveness of the system when the rates of the video streams need to be modified. The idea of coupling several Real-time Transport Protocol (RTP) flows in order to gain more flexibility is not new, in fact, in the context of the RTP Media Congestion Avoidance Technique (RMCAT) working group [2], an internet draft [17] describes the benefits of coupling the congestion control of different RTP flows. Therefore, in our case, we use a single congestion control for all the streams. Without going into the details of the CC algorithm, we simply assume that the CC measures the experienced delay and loss ratio of the network path and converges to the maximum achievable sending rate on that path. We identify with $d_n(t)$ the download rate achieved by the CC from the switch to user $n$ at time $t$, and with $u_n(t)$ the upload rate of user $n$ at time $t$ to the switch node.

We assume that every user is able to encode its video into one or multiple streams at different bitrates. Encoding a larger amount of streams is obviously more hardware consuming. In this work we let the maximum number of encoded streams be different among the videoconference participants, so that we can emulate the possible hardware heterogeneity of the endpoints. Although the proposed framework can easily be adapted to the case of multiple descriptor coding (MDC) or multiple independent video streams, we consider the specific case of a Scalable Coding (SC), e.g., Scalable Video Coding (SVC) [16], which offers a compromise between adaptation to receiver bandwidth heterogeneity and overall resource requirements. In SC, a video is encoded using different layers, namely a base layer, and one or more enhancement layers. Users can increase video quality by stacking several enhancement layers on top of each other. The advantages of SC with respect to independent coding is illustrated by a simple example. Consider the case where one user wants to serve a video available at two rates, e.g., 0.5 Mbps and 1 Mbps. With SC the sender can encode the video progressively in two layers: one base layer of 0.5 Mbps; and an enhancement layer of 0.5 Mbps, the user will then send a total of 1 Mbps. Both layers are necessary at the receiver to decode the full quality stream. In the case of independent coding, the user needs to send a total of 1.5 Mbps, since the two streams are independent. It is easy to understand that if the total rate is an important constraint, SC allows a much more efficient bandwidth utilization. There is usually a small distortion penalty with SC coding, but we consider it to be negligible in this work. Finally, we indicate with $L_m$ the maximum number of different SC layers that the user $m$ can encode, and we denote with $r_m^l$ the rate required to decode the $l$-th layer of user $m$. Thus, $r_m^l$ is not the rate of the single $l$-th layer, but the cumulative rate of all the other base layers that are required to decode the layer $l$. The coding rate of layer $l$ can thus be written as $r_m^l - r_m^{l-1}$ for $0 < l \leq L_m$.

Similarly to other works, [12, 6], we treat the video streams of the users differently based on their content. In a videoconference not all the video streams are equal, one user might be more active than the others, or the video content be more complex and require a higher encoding bitrate than the one of other senders for the same quality. In this case, where the rate allocation of the users is properly computed, a larger rate should be reserved to the most demanding users in order to maximize the overall QoS of the videoconference. In order to design our QoS aware rate allocation system, we need to define a mathematical model to measure the QoS of a user. We need to introduce two concepts: the importance of a participant and the utility curve. First we assume in this work every participant of the conference to be characterized by a certain role, or importance. We model the importance of every user using a real positive weight $w_n$. The higher the value of this quantity the more important the user. Second, we define $U_m(r_m)$ as the utility of receiving the stream of user $m$ at rate $r_m$, which roughly corresponds to the video quality of the decoded video stream of user $m$ at rate $r_m$. The concept of utility is largely used in rate allocation, and usually $U_m(r_m)$ is assumed to be a concave non-decreasing function of the rate. Ideally we would like to use a utility function that reflects the quality of the video. For example, if we use the Peak Signal-to-Noise Ratio (PSNR) as utility value, and we use the most simple classical model to fit the PSNR as a function of the rate [8] we obtain:

$$U_m(r_m) = \alpha_m \log(r_m) + \beta_m. \tag{1}$$

We can introduce now the total utility of the receiver $n$, modeled as a weighted sum of the utilities of the different streams it receives:

$$\mathcal{U}_n = \sum_{m=1, n\neq m}^{N} w_m \left(\alpha_m \log(r_m) + \beta_m\right) \tag{2}$$

It is worth noting two things about the above model. First, in reality, the coding efficiency of the SC decreases with the number of encoded layers for a given total encoding rate. Thus, in the utility function we should take into account the layer number $l$, and the utility should ideally decrease when $l$ increases. For the sake of simplicity we neglect this dependency in our model as the distortion penalty is negligible as long as the layers are large enough. It is however easy to include this behavior in the utility function for a more precise model if necessary. Secondly, we do not consider in this work the influence of the video complexity on the value of the utility for simplicity. In other words, $\alpha_m = 1$ and $\beta_m = 0$ in Eq. (2) for all the users. Nevertheless, this feature can easily be included in our framework, and actually other video quality metrics can be used, since the proposed method can work with any monotonically non-decreasing concave utility function, without altering the system design. In the rest of the paper, we show how to properly set the coding rates and select video layers to maximize the global QoS.

# 3. PROBLEM FORMULATION

Given the settings described in the previous section, the rate allocation problem for maximizing the QoS in the videoconference system can be stated as the following optimization problem:

$$\underset{z_n^{m,l},\, r_m^l}{\text{maximize}} \quad \sum_{n=1}^{N} \sum_{\substack{m=1,\\ n\neq m}}^{N} \sum_{l=1}^{L_m} z_n^{m,l} w_m \log(r_m^l) \tag{3}$$

$$\text{subject to} \quad \sum_{\substack{m=1,\\ n\neq m}}^{N} \sum_{l=1}^{L_m} z_n^{m,l} r_m^l \leq d_n \tag{4}$$

$$r_m^l \leq u_m \forall\ l, m \tag{5}$$

$$\sum_{l=1}^{L_m} z_n^{m,l} = 1 \quad \forall\ n \neq m \tag{6}$$

$$r_m^l \in [R_{\min}, R_{\mathrm{MAX}}] \tag{7}$$

$$z_n^{m,l} \in \{0,1\} \tag{8}$$

The above formulation is a mixed integer optimization problem, which aims at maximizing the overall QoS of the users, by selecting the video layers that every user has to receive and by allocating the rates of the different layers that each sender has to encode. The variables of the optimization problem are i) $z_n^{m,l}$, which correspond to decision variables for selecting which layer $l$ of user $m$ should be received by user $n$; and ii) $r_m^l$, which correspond to the rates of the different video layers. The objective function of the optimization problem corresponds to the sum of the receivers' utility functions defined in Eq. (2).

The first set of constraints (4) represents the download capacity constraints, which restrict the sum of the rates of the received layers to not be larger than the download capacity $d_n$. The second set of constraints (5) defines the limit on the upload bandwidth of the users; practically the largest cumulative layer rate of user $m$ has to be smaller than the upload capacity $u_m$. In (6) we impose that every receiver gets exactly one version of every source stream, where a version is given by a global cumulative rate. The next constraints define the limits on the values that the variables can take: the constraints (7) define some possible maximum and minimum encoding rates for each sender, while the constraints (8) limit the value of the decision variables to belong to the set $\{0, 1\}$.

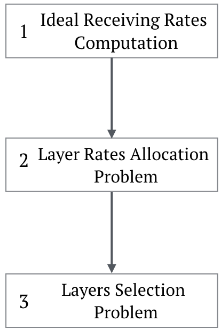
1 Ideal Receiving Rates Computation
2 Layer Rates Allocation Problem
3 Layers Selection Problem

**Figure 2: Steps of the proposed rate allocation algorithm.**

Note firstly that in the case where independent video coding is used instead of SC, the constraints (5) have to be changed. Since the encoding streams are independent in this case, the new constraints become $\sum_{l=1}^{L_m} r_m^l \leq u_m$. Finally note that the parameters of the above optimization problem, e.g., download/upload bandwidths and users' weights, are time-varying. The optimization needs to be reevaluated again when parameters change. The quick computation of the new optimal allocation is therefore fundamental to guarantee a good QoS to the users in dynamic conditions.

Due to the presence of the decision variables, which are discrete variables, the above optimization problem is extremely hard to solve. Thus, instead of solving the original problem we decompose the problem into smaller (tractable) subproblems, whose solutions lead to a suboptimal rate allocation for the videoconference. The subproblems are described in the next section.

# 4. RATE ALLOCATION ALGORITHM

The rate allocation problem is divided into three subproblems as shown in Fig. 2. In the first subproblem every receiver computes the ideal receiving rates for all the senders based on the download bandwidth constraint and the senders' importance. In the second subproblem, every sender chooses the rates of the different layers that will be made available to the receivers based on the ideal rates computed in the first subproblem. The third and last subproblem consists in the packing of the available streams for every receiver based on the download bandwidth and the available video layers in order to maximize the QoS. Solving these subproblems leads to some defined values for the optimization variables of the problem defined in (3)-(8). With respect to the original problem, the rate allocation is not guaranteed to be optimal, because of some assumptions that need to be made in order to solve the second subproblem (a more detailed discussion can be found in subsection 4.2). This solution method makes however the problem much simpler and allows us to solve it distributively. In the rest of this section, we provide details for each subproblem.

## 4.1 Ideal Receiving Rate Computation

The first problem that needs to be solved consists in the calculation of the ideal sending rates for the participants in the videoconference. Each receiver $n$ has a set of ideal rates $\mathbf{x}_n$. These can be computed using the download bandwidth and the importance of the different participants. These ideal rates are the rates that match the download capacity of the receiver and maximize the receiver utility function as defined in (2). In mathematical terms the ideal rates of user $n$ are the ones that solve the underlying optimization problem:

$$\mathbf{x}_n = \underset{\{x_n^m\}}{\text{argmax}} \quad \sum_{\substack{m=1,\\ n\neq m}}^{N} w_m \log(x_n^m) \tag{9}$$

$$\text{subject to} \quad \sum_{\substack{m=1,\\ n\neq m}}^{N} x_n^m \leq d_n. \tag{10}$$

Assuming concave non-decreasing utility functions (e.g., logarithmic functions in our model), the solution of this optimization problem is unique and can be easily computed using basic convex optimization algorithms [4]. The constraint simply indicates that the sum of ideal rates has to be smaller than the download rate set by the congestion control of user $n$. This optimization problem has to be solved independently by every receiver. By merging the results of problem (9)-(10) for all the $N$ independent receivers we obtain $N-1$ ideal rates for every sender ( the user $n$ is not interested in receiving his own video). The next subsection describes how every sender selects $L_m$ encoding rates, $r_m^l$, out of the $N-1$ ideal rates requested by the receivers.

## 4.2 Layer Rate Allocation Problem

This is the most complex problem to solve and it corresponds to the rate allocation of the different video layers at each sender based on the ideal rates collected from all the receivers and on the upload link bandwidth. The optimal layer allocation is carried out similarly to [19, 11]. The method is based on the following intuition: if a receiver $n$ requests an ideal rate of $x_n^m$ for sender $m$, then the corresponding utility is certainly maximized if the sender $m$ encodes a video layer with a cumulative rate exactly equal to $x_n^m$. If a slightly lower rate is available for the receiver $n$ the utility will be surely smaller than the ideal one. On the other hand, if a slightly higher rate is available, then the overall utility is also going to be smaller. In fact, in order to select a higher rate, the receiver has to give up some bandwidth that is ideally planned to be used for other senders, which modifies the ideal (optimal) rate allocation and decreasing the overall utility. Using the above observations the sender is able to compute the rates of the different encoded layers to maximize performance of the receivers.

We first need to introduce a second concept of utility, namely the layer utility, written as:

$$U_{\text{layer}}(x_n^m, r_m^l) = \begin{cases} x_n^m / r_m^l & : x_n^m < r_m^l \\ r_m^l / x_n^m & : x_n^m \geq r_m^l \end{cases} \tag{11}$$

This function attempts to model the loss of utility when a receiver $n$ with ideal rate $x_n^m$ receives instead a video layer with cumulative rate equal to $r_m^l$. The choice of the utility function in Eq. (11) is not unique. It is however important that its maximum value is achieved when $x_n^m = r_m^l$, that it is non-decreasing for $0 \leq r_m^l \leq x_n^m$ and non-increasing for $x_n^m \leq r_m^l$. With this behavior, the utility function matches the above observations. If multiple layers are available, we assume that the user with ideal rate $x_n^m$ selects the video layer that maximizes the utility layer function $U_{\text{layer}}(x_n^m, r_m^l)$. If we consider a vector of $L_m$ layer rates $\mathbf{r}_m$, and a vector of $N-1$ ideal rates $\mathbf{x}^m$ that gathers the ideal rates $\{x_n^m\}$ computed by all receivers, we can estimate the overall layer allocation utility as:

$$\mathcal{U}_{\text{layer}}(\mathbf{x}^m, \mathbf{r}_m) = \sum_{n=1}^{N-1} \max_{r_m^l} U_{\text{layer}}(x_n^m, r_m^l) \tag{12}$$

We propose now to solve the layer rate allocation problem that maximize Eq. (12) by dynamic programming. The first step here is to discretize the possible values that the layer rates can take. The discretization step is a tradeoff between the computational complexity and the accuracy of the solution.

Then we make the following observation which permits to construct the dynamic programming algorithm. When a layer $l$ is added on top of the other $l-1$ layers, the only receivers that might be interested in switching to this new layer are the ones that are using the second highest layer $l-1$. Obviously, a user that is expected to choose a layer lower than $l-1$ is not be interested in selecting a new layer higher than $l-1$. This observation can be stated more formally as follows:

$$\mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, l, r_m^l) = \underset{r_m^{l-1}}{\text{maximize}} \Big\{ \mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, l-1, r_m^{l-1}) \tag{13}$$
$$+ \delta_l(\mathbf{x}^m, r_m^{l-1}, r_m^l) \Big\}.$$

$\mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, l, r_m^l)$ represents the optimal layer utility value when $l$ encoding layers are used with the cumulative rate of the last layer equal to $r_m^l$. Note that $\mathcal{V}^*_{\text{layer}}()$ and $\mathcal{U}_{\text{layer}}()$ represent both the total layer utility, but in terms of different input variables. The meaning of Eq. (13) is the following: the largest possible utility of having a layer $l$ with cumulative rate $r_m^l$ is the sum of two terms. The first term is the best utility that we have using $l-1$ layers with the highest cumulative rate of $r_m^{l-1}$. Whereas the second term is the utility gain of adding a layer $l$ at rate $r_m^l$. This layer can only affect the ideal rates with a rate larger than $r_m^{l-1}$, thus the possible increase in the utility $\delta_l(\cdot)$ depends only on the layer rates $r_m^{l-1}$, $r_m^l$ and obviously the ideal rates $\mathbf{x}^m$.

We explain now how we solve the problem of Eq. (13) using a dynamic programming procedure. We fix $l = 1$ and we evaluate $\mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, 1, r^1)$ for $r_m^1$ varying gradually from $R_{\min}, R_{\min} + \Delta_r, R_{\min} + 2\Delta_r, ..., R_{\text{MAX}}$. We then increase the value of $l = 2$, and we use the computed values of $\mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, 1, r_m^1)$ to calculate $\mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, 2, r_m^2)$ according to Eq. (13). We proceed in the described way until the maximum number of layers $L_m$ for sender $m$ is reached. We can then track back the optimal set of layer rates starting from the value of $\mathcal{V}^*_{\text{layer}}(\mathbf{x}^m, L_m, r_m^{L_m})$. Since the original problem requires that all the users receive the video of all the other participants, we force the first layer to be equal to the smallest ideal rate, in this way we can guarantee that every receiver is able to select at least the base layer from all the other participants.

We now briefly discuss the performance of the above method that selects the best set of video layers. In order to measure the overall layer utility we should know which layer is going to be chosen by every receiver. However, this information is not available. In fact, the selection of the rate for the sender $m$ made by receiver $n$ depends also on the final rate of the layers of all the other senders as all the streams compete for the same download capacity at the receiver side. In the above procedure we assume that the user $n$ selects the layer that leads to the highest layer utility, which may actually not be the choice that maximizes the real utility of the users. In order to make the algorithm distributed this approximation is however needed, since there is no way for the senders to know how the receivers select the layers. Nevertheless this method guarantees a fast video layer allocation, since it relies only on local information, and secondly, in general, receivers follow a policy close to the one assumed by the video layer rate allocation problem.

## 4.3 Layer Selection Problem

After the above rate allocation step every sender $m$ has allocated the rates of the $L_m$ layers using the ideal rates computed by the receivers in the first place. The receivers have now to select, based on the download capacity and the importance of the users, the layers they subscribe to. In this problem the input parameters of every receiver $n$ are the set of available rates from all the other senders $\{r_m^l\}$, the importance levels $w_m$ and the download capacity $d_n$ computed by the congestion control. This optimization problem can be stated as follows:

$$\mathbf{z}_n = \underset{\{z_n^{m,l}\}}{\operatorname{argmax}} \quad \sum_{\substack{m=1,\\ n\neq m}}^{N} \sum_{l}^{L_m} z_n^{m,l} w_m \log(r_m^l) \tag{14}$$

$$\text{subject to} \quad \sum_{\substack{m=1,\\ n\neq m}}^{N} \sum_{l=1}^{L} z_n^{m,l} r_n^{m,l} \leq d_n \tag{15}$$

$$\sum_{l=1}^{L_m} z_n^{m,l} = 1 \quad \forall\; m \neq n \tag{16}$$

$$z_n^{m,l} \in \{0,1\} \tag{17}$$

This problem is solved independently by every receiver $n$ and is similar to the one carried out in the first step. However, in the first subproblem the selection of the rates was continuous, whereas here the selection is discrete, since there exist only $L_m$ available rates for sender $m$. This problem is a constrained Integer Linear Programming (ILP) problem [15]; it can be solved exactly using a ILP solver. Alternatively, a suboptimal greedy solution can also be used if computation resources are poor. The greedy algorithm consists in simply selecting progressively the layers with the largest gain without violating the constraints capacity (the gain of a layer is defined by the incremental utility of that layer divided by its rate). Note that, as in the original problem, we force every receiver to select at least one layer from every sender. This solution is always feasible for problem (16)-(17) due to the choice of sending rates made in the previous section.

We finally recall that the global proposed algorithm is composed by all the three described subproblems evaluated in sequence, and secondly, that the three steps need to reevaluated if any of the input parameters, download/upload bandwidth or users' importance, change.

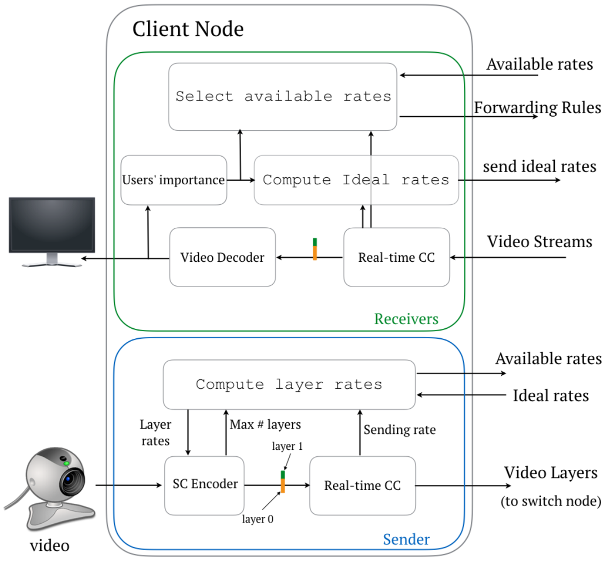

**Figure 3: Client application architecture.**

# 5. PRACTICAL IMPLEMENTATION

We describe now the details of the practical implementation of the proposed system. The global system is composed of two applications: the client node application and the switch node application. The client node application can be further divided in the receiver subpart and the sender subpart. Fig. 3 and Fig. 4 depict the internal structure of both applications, which are described in details below.

## 5.1 Client node

The client application can be seen as composed by two parts: the sender part and the receiver parts as shown in Fig. 3.

First, the receiver part is responsible for receiving the video streams and for solving problems described in Subsections 4.1 and 4.3. Upon receiving the video streams from the different users the decoder analyzes them in order to extract the importance of the participants, which is needed to solve the optimization problems. The importance extraction algorithm, which can simply consist in selecting the current speaker (see for example speech activity detection methods, e.g., [14]) goes however beyond the purpose of this work. The importance of the users and the download bandwidth, which is given by the CC algorithm, are then used as input for the computation of the ideal rates. They are computed every time the importance of the users or the download bandwidth change according to Subsection 4.1. If the new rates are different from the old ones the ideal rates are sent using a reliable protocol to the switch node, which eventually triggers an adaptation of the sending rates. The receiver part of the client application is also responsible for receiving the available rates from the switch node. The available rates, together with the importance of the users and the download bandwidth estimation, are used to select the layers to receive among the available ones, according to Subsection 4.3. The output of this optimization problem are the forwarding rules that the switch node has to apply in order to forward the right streams to the receiver. This optimization step is also executed every time any of the input variables

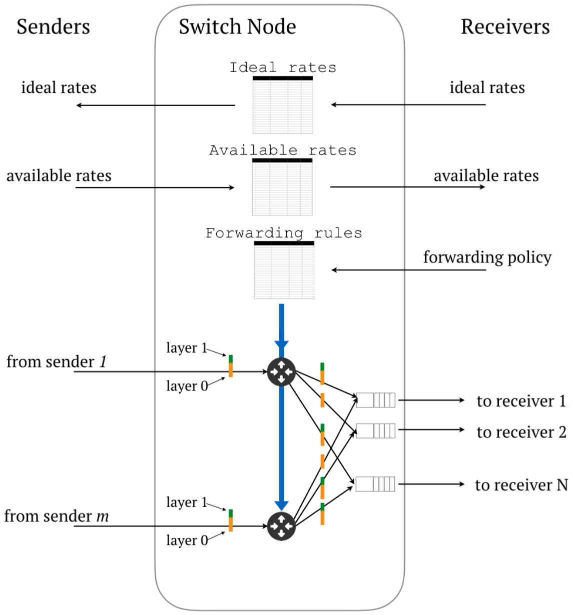

**Figure 4: Switch application architecture.**

change, and if the forwarding rules are different compared to the previous ones, they are also sent to the switch node for an update of the transmission policy.

Second, the sending part gets the video data from the camera, encodes the video at different layers and then sends these layers to the switch node. Simultaneously the sender part is responsible for the computation and update of the available rates. When new ideal rates are received from the switch node, they are used, together with the sending rate and the parameter that limits the maximum number of layers, to compute the rates at which to encode the layers. Note that when the rates of the layers change, it is important to forward this information to the receivers before effectively reallocating the rates. In fact the receivers need to reselect the rates they want to receive before these actually change. If this condition is not verified, the encoded new layer rates might not fit in the download bandwidth of the receivers. Consider for example that the rate of layer 2 will be increased from 800 kbps to 1 Mbps: the users that are downloading this layer might not have the needed extra-bandwidth to afford this new rate, resulting in packet losses. Therefore, the switch node, and then receivers, are notified in advance with the new rate values, but these rates become effective only after a period $T_{\text{eff}}$ in our implementation.

## 5.2 Switch node

The switch node is responsible for forwarding the video packets from the senders to the receivers and for the management and communication of the ideal rates and available rates, as illustrated in Fig. 4.

The switch application maintains a forwarding table where, for every receiver, a list of pair values (user identifier and layer identifier) indicates which layers should be forwarded to each user. Every time a video packet is received by the switch node, the application checks which receivers requested this pair of user-layer identifier and forwards the packet accordingly. Every sending stream, from the switch node to one of the receivers, has a sending buffer in order to temporarily accommodate packets that have to be sent. The buffers are drained according to the sending rate of the congestion control algorithm. These buffers help to handle situations where the download capacity changes suddenly, or where minor synchronization errors among the endpoints may temporarily generate a surplus of packets. In our particular implementation we simply use a droptail management policy for the output buffers [7]; however more advanced scheduling techniques can be used at this stage without affecting the design of the switch node. The maximum size of the buffers is set in order to limit the maximum queuing delay to 100 ms, when this limit is reached other incoming packets are discarded. It is worth noticing that packet losses at this stage are not taken into account by the congestion control, since the sessions of the congestion from the sender to the switch node, and from the switch node to the receiver(s) are completely decoupled.

In parallel to its forwarding process, the switch node receives the ideal rates computed by the receivers and the available video layer rates encoded by the senders, and stores these values in two different tables. The switch node can forward the ideal rates and available rates every time they change, or periodically, to limit the communication overhead. In our implementation we have chosen the following rules for the ideal/available rates forwarding procedure. The Switch node sends the most updated ideal rates to the senders every $T_{\text{eff}}$. The senders compute the new available rates and they send these new values to the switch node immediately. The switch node waits $T_{\text{eff}}/2$ s after it has sent the latest ideal rates in order to collect the new available layer rates from all the senders, and then it sends them to the receivers. Once the users receive the new available rates they know that they will become effective after $T_{\text{eff}}/2$ since the instant they received them ($T_{\text{eff}}$ since the senders received the new ideal rates, consistently with subsection 5.1) and they can update the forwarding rules accordingly. The choice of $T_{\text{eff}}$ has to provide both the possibility of a quick adaptation but also to avoid the waste of network resources caused by the communication overhead. Finally, when the switch node receives new forwarding rules, it modifies the entries in the forwarding rule table and at the same time applies the new policy for the new incoming packets.

# 6. SIMULATIONS

We study now the performance of the proposed algorithm. The system has been implemented in the network simulator NS3 [13]. The simulation results show the basic operation of the system and the benefits of the proposed solution over a heuristic and non-optimized rate selection method which, however, uses the same system architecture. The client and switch node applications described in Section 5 have been fully implemented in the network simulator. We use the NADA congestion control algorithm to send the media packets among the users, whose implementation corresponds to the one described in [20]. In the context of the RMCAT working group [2] the NADA algorithm has shown good performance as a valid CC for real-time media communication. It is worth noting, however, that the operation of the proposed optimization scheme is independent of the underlying

**Table 1: Settings for the 6 users scenario**

| | |
|---|---|
| Download capacity | [4 5 3 8 10 8] Mbps |
| Upload capacity | [2 2 1 3 10 3] Mbps |
| Maximum number of layers | [3 2 2 2 4 3] |

**Table 2: Settings for the 10 users scenario**

| | |
|---|---|
| Download capacity | [1.5 3 4.5 5 6.5 8 9.5 11 12.5 14] Mbps |
| Upload capacity | [0.5 0.8 0.8 1 1 1 1.5 1.5 1.5 1.5] Mbps |

CC algorithm that is used by the system.

A fundamental feature of our algorithm is to quickly adapt the rate allocation when the importance of the users changes. In order to stress this ability we mimic the change of the speakers during a conference and thus change of the importance parameters $w_n$. For the sake of simplicity we assume that only two levels of importance exist: namely speaker and listener; we assign to the speaker a weight $w_n$ equal to 3 and to all the listeners a weight equal to 1. More advanced distinction can however be made, for example the last speaker can have a higher importance in the case he is likely to speak again soon. The role of the user $n$ is in reality contained in the video-audio information coming from user $n$. In our implementation, we however include the weight of each user in the media packets that he sends. Receivers become aware of the importance of the other users simply by inspecting the incoming packets. Finally, the parameter $T_{\text{eff}}$ introduced in Section 5 is set equal to 1 s, as this value offers a good tradeoff between the responsiveness of the system and the limited overhead introduced by the communication of the ideal and available rate information.

In the first scenario we consider the case of 6 users that participate in a videoconference. The capacity of the upload/download links and the maximum number of encoding layers for the different users are listed in Table 1. We simply select some heterogeneous capacity values in order to emulate the diversity of the endpoints, and show the operation of the system. The speaker activity is shown in Fig. 5. In Fig. 6 we can observe the receiving rates of the users for the scenario under consideration. As can be seen the system quickly adapts the rates when the speaker changes and the overall receiving rates of the users are close to the maximum one without violating the download capacity. Since the system guarantees that all the users are always able to receive the video streams from all the other participants the user with the lowest download capacity, namely user 3, is the one that can use the channel more efficiently. However, due to the multiple layer encoding also the users with larger capacities are able to achieve a good link utilization.

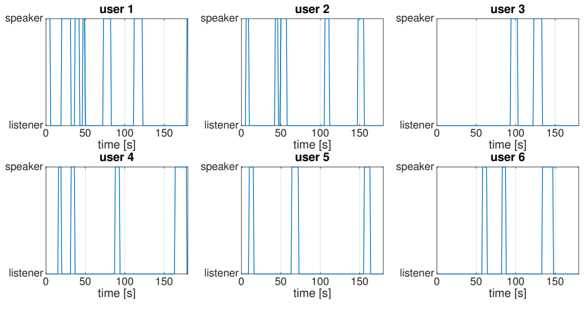

**Figure 5: Speakers activity for the 6 users scenario.**

Fig. 7 then shows the rate at which user 4 receives user's 5 video in the simulation described above. The dashed black line shows how the role of user 5 changes. The receiving rate of user 5 adapts to its importance in the conference and thus helps to improve the total utility of the videoconference. The delay between the weight variation and the rate variation is the time required for the algorithm to reallocate the rates. This amount of time depends on the choice of the parameter $T_{\text{eff}}$ and on the delay at equilibrium of the CC. In our case, with $T_{\text{eff}} = 1$ s and the NADA congestion control this delay is approximately in the order of 1.5 s. The rate variations that do not follow the activity curve are due to changes in the available rates caused by variations in the ideal rates of the other users.

In order to evaluate the adaptivity of the system to bandwidth variations, we use the same settings of the previous scenario but at 100 s we change the receiving capacity of user 3, from 3 Mbps to 6 Mbps. After the capacity variation the congestion control detects that a higher bandwidth is available and it converges to the new equilibrium rate. At the same time the rate allocation algorithm modifies the ideal rates of user 3 and selects higher available layers, the adaptation can be observed in Fig. 8. It is worth noticing however, that the speed of increase of the total receiving rate depends on the congestion control that is used, while the responsiveness of the rate allocation depends completely on the proposed algortihm.

In the second scenario we increase the number of users to 10, and we impose the bandwidth of the upload and download links according to Tab. 2. In order to have a realistic setting we set the capacities of the links according to different speed tiers provided by a large internet service provider company[1]. In this case we perform different tests with an increasing number of maximum encoding layers. We varied the number of layers from 1 to 5, and this parameter was set to the same value for all 10 participants. We compare our algorithm to a baseline method built on heuristics for the layer rates allocation and selection. The baseline method uses the same network architecture of the proposed method, i.e., it uses a central node to allow application layer multicast communication. Moreover SC and NADA are used as well for the video encoding and as network CC respectively. This heuristic method however does neither take into account the ideal rates nor the role of the users. It operates as follows:

- Every sender encodes the video at the following possible rates: $[1/4\ 1/2\ 3/4\ 1/8\ 3/8] \cdot u_n$, where $u_n$ is the upload bandwidth of the user. If it is possible to encode only one layer the users select the first rates of the vector, if two rates are possible then the first and second rate of the vector are used and so on (note that the rates are not monotonic). The rates have been selected in order to offer a tradeoff between an efficient use of the upload capacity and have a good probability of guaranteeing that all the users are able to receive all the streams.

- The central node simply forwards to the receivers the

[1] available at: http://www.att.net/speedtiers

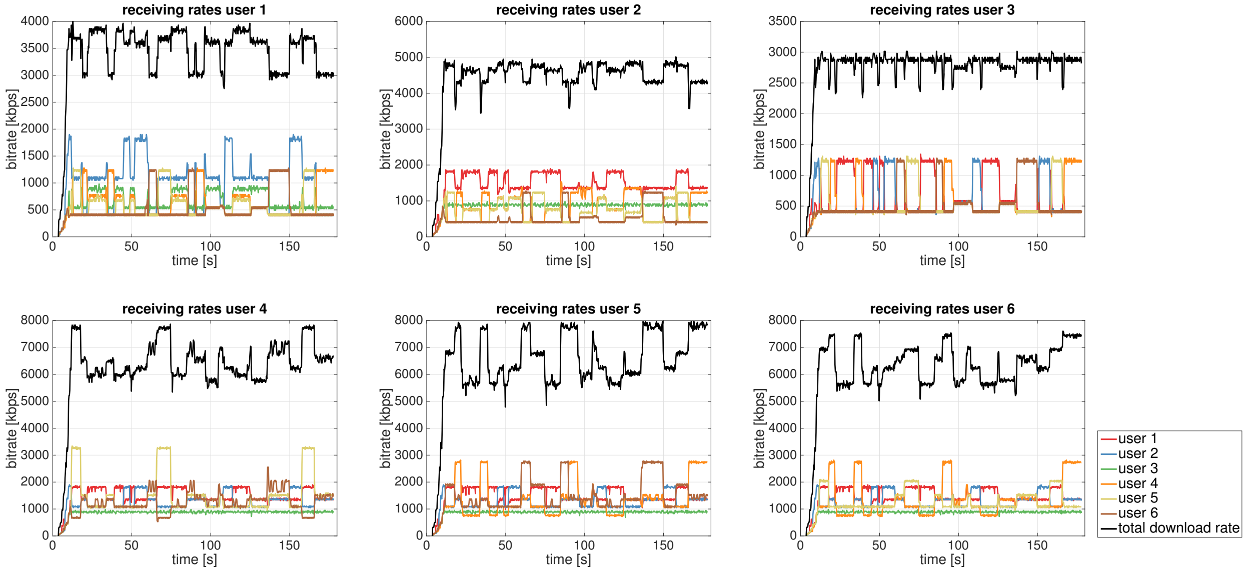

**Figure 6: Download rates of the videoconference participants.**

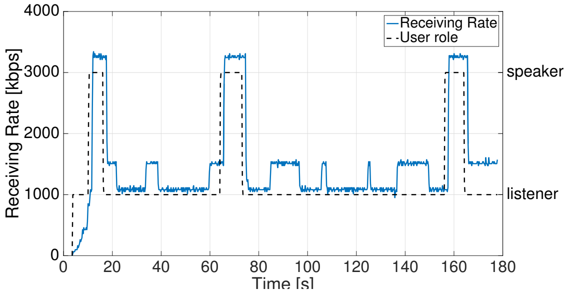

**Figure 7: Receiving rate of user 5 by user 4. And importance variation of user 4.**

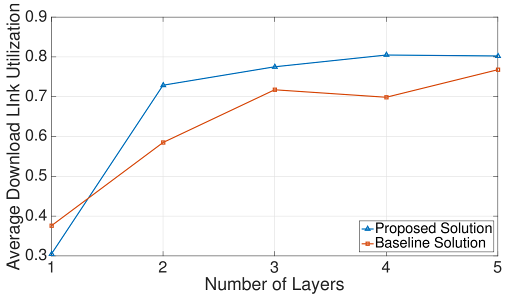

**Figure 9: Average download link utilization for the 10 users scenario, for different values of maximum encoding layers.**

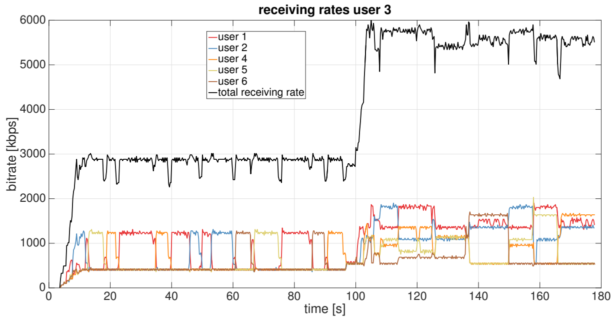

**Figure 8: Adaptivity of the algorithm to capacity variations.**

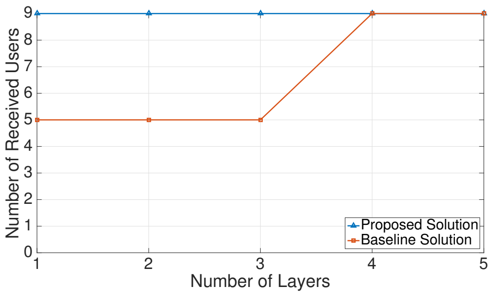

**Figure 10: Number of received streams by the user with the smallest download capacity.**

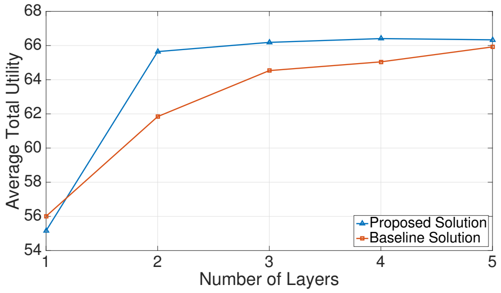

**Figure 11: Average user total utility for the 10 users scenario, for different values of maximum encoding layers.**

largest possible set of users, at the highest possible encoding rates, while respecting the download capacity.

We simulate a videoconference of 180 s with different time-varying speakers. We then compute the time average bandwidth utilization of the download link of every user. We also compute the average total utility of every user over time as defined in Eq. (2). Finally we average both metrics among the 10 participants of the conference. In Fig. 9 we show the average download link utilization. Our method has a slightly higher channel utilization but not for all the values of the maximum number of layers. The baseline solution is able to have a higher channel utilization when only one layer is available, but it is not able to guarantee that the user with the smallest download capacity receives the video streams of all the other participants, while this is a mandatory constraint for the proposed solution. This can be seen in Fig. 10, where we plot the number of received users for the participant with the smallest capacity. When the rate equal to $u_n/8$ becomes available (i.e., the number of available layers is 4), all the users are able to receive streams from all the participants also for the baseline solution. In the proposed solution this condition is always achieved at the price of a slightly lower channel utilization for some cases. The total utility, is finally illustrated in Fig. 11. At the beginning the heuristic solution has a higher average utility but cannot guarantee that all the users receive the video stream of all the other participants. When more layers are available the proposed method always outperforms the heuristic solution. Obviously with different layer rates selection, the baseline solution results would be different: smaller layers could guarantee that all the users receive all the video streams, but it would decrease the channel utilization. Our method actually allows to select the video rates in such a way that, independently of the scenario, all the users can achieve both, a discrete channel utilization and a good QoS. Moreover, the proposed method offers an adaptive rate allocation that takes into account the environment conditions in order to maximize the QoS.

## 7. CONCLUSIONS

In this work we propose a solution for designing a videoconference system that adapts dynamically to the network and user dynamics. We use a central switch node in order to allow application layer multicast communication among the videoconference participants. The central node is however kept as simple as possible in order to guarantee the scalability of our selection. The users, which are able to encode their own video at different bitrates, follow a distributed rate allocation mechanism in order to compute the encoding rates of the video streams. The distributed algorithm is able to rapidly allocate the rates and to adapt quickly to new environment conditions. The adaptation speed depends on the policy used to forward the new ideal/available rates and on the round trip time of the connections (which depends on the congestion control algorithm). The new method has been implemented in a network simulator and its performances have been compared with a baseline solution. The results show the benefits in link utilization and QoS that our adaptive solution can provide. The proposed system is able to allocate the rates of the videoconference in order to guarantee a good QoS to the users and guarantee that all the users are able to receive the video streams of all the other participants for any value of the link bandwidth. The responsiveness of the proposed solution allows a fast adaptation to condition changes, contributing to increase the QoS. As future work we will focus on improving the solution by using not only one, but a set of switch nodes to redistribute the packets. We aim at having a set of multiple switch nodes, which simply apply forwarding policies that are still computed by the users' endpoints without requiring any intelligent service by the switch nodes.

## Acknowledgments

This work has been supported by the Swiss Commission for Technology and Innovation under grant CTI-13175.1 PFES-ES.

## 8. REFERENCES

[1] Cisco video and telepresence architecture design guide. *Cisco*, 2012.

[2] RTP Media Congestion Avoidance Techniques (RMCAT). IETF Working Group, 2012.

[3] The zettabyte era: Trends and analysis. *White Paper, Cisco*, 2015.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[5] M. Chen, M. Ponec, S. Sengupta, J. Li, and P. A. Chou. Utility maximization in peer-to-peer systems. In *ACM SIGMETRICS Performance Evaluation Review*, 2008.

[6] X. Chen, M. Chen, B. Li, Y. Zhao, Y. Wu, and J. Li. Celerity: a low-delay multi-party conferencing solution. In *ACM international conference on Multimedia*, 2011.

[7] D. E. Comer. *Internetworking with TCP/IP*, volume 1. Pearson, 2006.

[8] M. Dai, D. Loguinov, and H. Radha. Rate-distortion modeling of scalable video coders. In *International Conference on Image Processing (ICIP)*, 2004.

[9] A. Eleftheriadis. SVC and video communications. *White Paper, Vydio*, 2011.

[10] B. Grozev, L. Marinov, V. Singh, and E. Ivov. Last n: relevance-based selectivity for forwarding video in multimedia conferences. In *25th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video.* ACM, 2015.

[11] J. Liu, B. Li, Y. T. Hou, and I. Chlamtac. On optimal layering and bandwidth allocation for multisession

video broadcasting. *IEEE Transactions on Wireless Communications*, 2004.

[12] M. Ponec, S. Sengupta, M. Chen, J. Li, P. Chou, et al. Multi-rate peer-to-peer video conferencing: A distributed approach using scalable coding. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2009.

[13] G. F. Riley and T. R. Henderson. The ns-3 Network Simulator Modeling and Tools for Network Simulation. In *Modeling and Tools for Network Simulation*. Springer Berlin Heidelberg, 2010.

[14] K. Sakhnov, E. Verteletskaya, and B. Simak. Dynamical energy-based speech/silence detector for speech enhancement applications. In *World Congress on Engineering*, 2009.

[15] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Inc., 1986.

[16] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the h. 264/avc standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9), 2007.

[17] M. Welzl, S. Islam, and S. Gjessing. Coupled congestion control for RTP media. Internet-Draft, 2015.

[18] Y. Xu, C. Yu, J. Li, and Y. Liu. Video telephony for end-consumers: Measurement study of google+, ichat, and skype. In *ACM Conference on Internet Measurement Conference*, 2012.

[19] Y. R. Yang, M. S. Kim, and S. S. Lam. Optimal partitioning of multicast receivers. In *IEEE International Conference on Network Protocols*, 2000.

[20] X. Zhu et al. NADA: A unified congestion control scheme for real-time media. Internet-Draft, 2015.