SDN-enabled Adaptation of Videoconference Streams to Network Dynamics

Christelle Al Hasrouty, Mohamed Lamine Lamali, Damien Magoni, John Murphy

SDN-enabled Adaptation of Videoconference Streams to Network Dynamics

Christelle Al Hasrouty*†, Mohamed Lamine Lamali*, Damien Magoni*, John Murphy†
* University of Bordeaux – LaBRI, France
† University College Dublin – Lero, Ireland
{alhasrouty, mohamed-lamine.lamali, magoni}@labri.fr, j.murphy@ucd.ie

***Abstract*—Real-time interactive communications, such as videoconferencing, have strong QoS requirements. Adapting such communications to network dynamics needs network management capabilities that traditional networks cannot provide. Recently, SDN has shown to be a potential solution for solving these network adaptation challenges. In a companion paper written by our team, we have proposed a videoconferencing system using SVC multicast trees built over an SDN network. This system minimizes the overall bandwidth consumption and provides the highest possible QoS to the users although it is not adaptable to network dynamics happening over the lifetime of the communication. In this paper, we complement our system, by providing efficient algorithms able to modify the connections between the participants during a videoconference in order to timely react to network changes. The simulation results confirm the efficiency of our solution in terms of time processing and bandwidth savings.**

I. Introduction

Videoconferencing has become a mainstream communication method but often suffers from poor Quality of Experience (QoE). During a videoconferencing call, QoE is mainly affected by the heterogeneous participants' characteristics and their channel capacity. The wireless channel is a shared-access medium, and the available bandwidth varies with the number of hosts contending for the channel. In wireless networks, the available access bandwidth experiences quick modifications due to physical obstacles, mobility from one access network to another and channel blurring. An access link with a low bandwidth capacity can prevent the reception of a high-quality stream regardless of the participant characteristics. Traditional networks find it difficult to detect and adapt to the bandwidth variations in real time. Software Defined Networking (SDN) provides new ways to manage network bandwidth. SDN has a centralized view that allows the management of the network variations in a more efficient way.

In [1], our team proposes algorithms that use SDN to create multicast adaptive trees with Scalable Video Coding (SVC) [2] layers inside the network. The main goal of this work is to save the network bandwidth consumption while adapting to the participants' heterogeneity. However, the proposed algorithms do not consider the dynamic adaptation to time-evolving user access bandwidth. The only solution to use them in a dynamic context is to run them again when a change occurs. This may be costly in processing time and resources. In this paper, we complete the work in [1] and propose new algorithms with a lower complexity that require less processing time and resources while minimizing the bandwidth consumption in the core network. We evaluate these algorithms through simulations and compare them to the algorithms of [1]. The simulation results show that the processing time of the new solution is much lower than that of the other algorithms. Moreover, it achieves the same bandwidth saving. This suggests that the new solution is more adapted to a dynamic context and provides high reactivity and QoS to the users, while minimizing the controller resource consumption and the used bandwidth in the core network.

The paper is organized as follows: Section II describes the problem, presents a brief state of the art on bandwidth and calls variations in a dynamic environment, and places our solution in this context. Section III details the model and assumptions of our proposal and describes our algorithms together with their complexity. Section IV details our simulation methodology, the used parameters and the results of the evaluation of the different algorithms. Finally, Section V concludes the paper.

II. Multicast for Videoconference Management

A. Problem Statement

We consider multi-party video calls where all the participants send their video stream and receive those of the other participants. Thus, for each video stream, there are one sender and several receivers. The unicast method consists in building a path between each pair of participants while the multicast one consists in computing a tree from each sender to the receivers. A layering technique (SVC) for adapting video stream is available on each SDN switch in the core network. An adaptation of a video stream is a dropping of high-quality video layers while allowing lower quality ones. In [1], our team aims to minimize the bandwidth consumption in the core network by accurately computing the multicast trees and opportunely placing the SVC adaptation rules inside the network. The problem we address is double: How these methods behave in a dynamic context (bandwidth variation)? And how to design less costly and more reactive algorithms to adapt video streams when variations occur.

B. Related Work

Using multicast to deal with multi-party (video) calls is a natural approach and is investigated since the development of multimedia telecommunication. Together with video layering techniques such as SVC, it provides mechanisms to manage

video calls while providing different quality streams to the participants. For example, some works [3]–[5] use these mechanisms to improve the participants' QoE. However the video layering adaptation in these works can only be performed at the application layer, and thus the core network bandwidth consumption cannot benefit from it. With the emergence of SDN, several works used the new paradigm to improve the participants' QoE and simplify the network management [6]–[8]. These works propose that the SDN controller directly manages the multicast tree construction and the video layering inside the network. However, while some of them minimize the bandwidth consumption but degrade the video quality stream of the participants, the others provide high quality video stream but at the price of higher bandwidth consumption. For example, the latter constructs a different multicast tree for each video layer, which is not optimal in term of bandwidth saving. In [1], our team proposes a solution to provide the best possible video quality to the participants while minimizing the bandwidth consumption in the core network. The proposed solution consists in constructing a tree from each sender to the receivers (the same tree for all the video quality layers) and then placing SVC adaptation rules (video quality degradation) on the SDN switches in the network. Two algorithms are proposed for the tree construction: *Shortest Path Tree* (SPT), which computes the shortest path tree between each sender and the other participants (the receivers), and *Minimizing Spanning Tree* (MST) which proceeds iteratively. MST first computes the shortest path between the sender and the first receiver (according to a specific ranking rule), then the shortest path between any node of the previous path and the second receiver, etc. At each step, it computes the shortest path between the current multicast tree and the next receiver. Both algorithms perform a very good bandwidth saving while providing to each participant the maximum video quality that it can receive. However, this method was not evaluated in a dynamic context, i.e., under access channel bandwidth variations, which is a realistic assumption in current networks.

C. Dynamic Aspects

There is an extensive literature on video calls under bandwidth variations. For example, [9] provides an architecture and several mechanisms to perform video streaming that adapts to congestion and bandwidth variations. It investigates both unicast and multicast methods. But the congestion control is performed at the application layer, and thus at the endpoints of the communication. The authors of [10] investigate the effect of congestion and bandwidth variation on a Skype video call. However, the case study includes only two hosts and, again, the adaptation mechanism is performed within the application layer. Some other works focus on the reactivity to bandwidth variation. For example, [11] proposes a cross-layer approach where the bitrate is directly estimated on the physical layer, allowing a quick adaptation to the new bandwidth. More recently, the new approach of *Pseudo analog video transmission*, which allows a direct adaptation of the video stream quality according to the channel noise, is investigated in [12], [13]. The drawback of these cited works on video call under bandwidth variation is that the video quality adaptation is at the endpoints. In the case of multicast approach, this has two consequences: i) If the video quality adaptation is performed at the sender, it will adapt to the video quality to the receiver with the lowest bandwidth. This means that the receivers with higher bandwidth cannot benefit from a quality corresponding to their access channel. ii) If the adaptation is performed at the receivers, the sender has to emit the highest quality stream through the multicast tree, which implies higher bandwidth consumption in the core network.

D. Proposed Solution

In the perspective of providing the highest possible video quality to each receiver while minimizing the bandwidth consumption, we first propose to evaluate the algorithm of [1] under random access bandwidth variations. It consists in running the algorithm that computes the multicast trees and replaces the adaptation rules at every bandwidth variation, which may be costly. We propose algorithms that do not rebuild the multicast trees but replace optimally the adaptation rules. This does not require any path computation but just requires moving upward or downward the rules in the trees. These methods are especially fast when the height of the multicast trees is small, thus improving considerably the complexity, the processing time and the reactivity. Our algorithms complete those of [1] which can still be used to construct the initial multicast trees.

III. System Model and Algorithms

A. Technical Assumptions

We assume that the network is SDN-enabled and that the controller manages the calls and has a global view of the network. All the switches can communicate with the controller and are supposed to be able to adapt SVC streams by dropping unused layers. We consider that there are at most 4 SVC stream layers as mentioned in Table I and a given layer can be used only if all the lower layers are also received. In addition, each participant accepts the highest number of layers allowable by its access downlink capacity. If a participant does not have enough bandwidth to receive any layer, the video call is rejected. Many participants can be connected wirelessly to the same node, which makes the access channel subject to variable communication conditions. This randomness in bandwidth availability can lead to a large heterogeneity of bandwidth between the participants.

B. Formal Model

The network is a graph $G = (V, E)$, where $|V| = n$ is the number of nodes and $|E| = m$ is the number of links. The set of participants is denoted by $\mathcal{P} = \{1, \ldots, p\}$. There is p multicast trees, one for each participant as a sender. Thus, a multicast tree T_i is associated to a sender s_i and a set of receivers $R_i = \{r_{i,j} \mid i \in \mathcal{P} \text{ and } j \neq i\}$. For simplicity, we consider that the participants are also nodes of the trees (the receivers as leaves and the sender as root).

TABLE I
SYSTEM CHARACTERISTICS

Parameter	Value(s)
Access links	LTE
SVC layers	L1: Scalable Constrained Baseline, ~90Kbps L2: Scalable Baseline, ~250Kbps L3: Scalable Constrained High, ~0.5Mbps L4: Scalable High, ~1Mbps
Videoconf. dedicated core link bandwidth	1Gbps

TABLE II
MODEL NOTATIONS

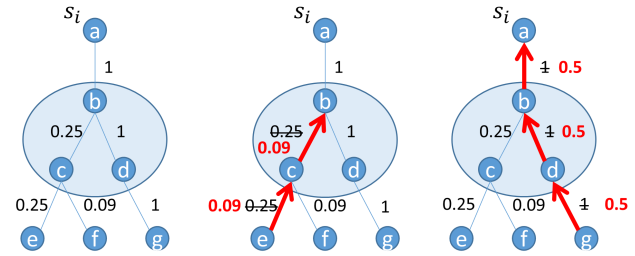
Notation	Definition
$B(r_{i,j})$	Downlink bandwidth of the receiver $r_{i,j}$
$B(s_i)$	Uplink bandwidth of the sender s_i
$B(x,y)$	Bandwidth of the link (x,y)
$b(s_i)$	Uplink bitrate of the sender s_i
$b_i(r_{i,j})$	Downlink bitrate of the receiver $r_{i,j}$ in the tree T_i
$b_i(x,y)$	Bitrate of the link (x,y) in the tree T_i
$C_i(r)$	Set of children of a node r in the multicast tree T_i
m	Number of links in the network
n	Number of nodes in the network
$\mathcal{P}$	Set of participants
p	Number of participants
$P_i(r)$	Parent of a node r in the multicast tree T_i
R_i	Set of receivers in the tree T_i
$r_{i,j}$	Receiver j in the tree T_i
Rule(b,b')	The adaptation of bitrate b to b'
s_i	Sender i
T_i	Multicast tree associated to the sender s_i

for any link specified by a pair of nodes (x,y), the bandwidth[1] (or capacity) is denoted by $B(x,y)$ and the bitrate (the actual used capacity in a specified tree T_i) is denoted by $b_i(x,y)$. In a tree T_i, the parent of a node x is denoted by $P_i(x)$ and the set of its children (if any) by $C_i(x)$. Note that each receiver has the same parent in all the trees, as it has only one access link. For sake of simplicity, $B(r)$ (resp $b_i(r)$) can denote the download bandwidth (resp. the bitrate in T_i) of the receiver r. Likewise, $B(s_i)$ (resp. $b(s_i)$) can denote the upload bandwidth (resp. bitrate) of sender s_i. At each node in the network and for each tree, there is a set of adaptation rules. This corresponds to SVC layer adaptation in the network. In a specified tree and on a node x, an adaptation rule (b,b') for a child c of x means that x receives a bitrate b but sends a bitrate $b' < b$ to c. SVC layers above b' are dropped before transmitting the video stream to c.

C. Algorithms

As said in section II, SPT and MST methods compute the multicast trees and place the adaptation rules at each node. We will use them in our algorithms in the initialization phase, i.e., when a call is established. However, when a bandwidth variation occurs, as recomputing the trees is costly, we opt for keeping them unchanged. But in order to guarantee that each participant receives the highest video quality allowed by its bandwidth, our algorithm recomputes the bitrates according to the following formulas:

[1]Note that it is possible that $B(x,y) \neq B(y,x)$ if the uplink and downlink bandwidths are not the same.



(a) Tree before bitrate change. (b) Bitrate change propagated. (c) Bitrate change propagated until the sender.

Fig. 1. Bitrate change propagation

For each sender s_i

$$b_i(r_{i,j}) \leftarrow \min\left\{\frac{B(r_{i,j})}{p-1}, B(s_i)\right\} \qquad (1)$$
$$b(s_i) \leftarrow \min\left\{B(s_i), \max_{r_{i,j}\in R_i} b_i(r_{i,j})\right\}$$

These formulas are used in [1] in order to set the definitive values of the bitrates. In a dynamic context, they should be used at each bandwidth variation. The first one ensures that a receiver $r_{i,j}$ gets the maximum among what is allowed by its bandwidth ($B(r_{i,j})$) divided by the number of streams it receives $(p-1)$ and the access uplink bandwidth of the sender ($B(s_i)$). The second one ensures that the sender emits the maximum needed video quality ($b(r_{i,j})$ for each receiver) and allowed by its uplink bandwidth. The basic principle of our algorithms is to always satisfy these constraints but with minimum computation. Thus, when a bandwidth variation occurs, the bitrates are recomputed. If there is a bitrate change at a receiver, this change is propagated and the adaptation rules are pushed upward the tree. In the same way, if a bitrate change occurs at the sender, the propagation is done downward.

1) Change at a receiver: Figure 1 shows the propagation mechanism. Figure 1(a) depicts a multicast tree where node a is the sender and nodes e, f, and g are the receivers. The other nodes are SDN switches. The bitrate of each link is beside it. In Figure 1(b), because of a bandwidth change, the bitrate of (c,e) is no longer 250Kbps but 90Kbps. While neither e nor f need a stream of 250Kbps, the layer corresponding to this rate is dropped at b, the parent of c. In Figure 1(c), the bitrate of (d,g) changes from 1Mbps to 0.5Mbps. This change is propagated until b, but neither c nor d require 1Mbps, thus the bitrate of a is also adapted. This method adapts the bitrate upward from a receiver, but it is infrequent that it reaches the sender. It happens only in the case where the bitrate of the receiver and the sender are the same, and a change of the first one affects the second one. The goal of this algorithm is to minimize the computation task while

minimizing the consumed bandwidth. To do so, it places the adaptation rules that drop higher quality streams as close as possible to the sender, thus saving the bandwidth in the core network. Algorithm 1 formalizes this method. It takes as input

Algorithm 1: Relocate Up

```
1 Input: A tree T_i and a node r
2 b_max ← max_{1≤j≤k}(b_i(P_i(r), c_{i,j}) with c_{i,j} ∈ C_i(P_i(r))
3 b_min ← min(b_max, b(s_i))
4 if (b_min ≠ b_i(P_i(P_i(r)), P_i(r)) then
5     b_i(P_i(P_i(r)), P_i(r)) ← b_min
6     AdaptRule(P_i(r))
7     if P_i(P_i(r)) ≠ s_i) then
8         RelocateUp(T_i, P_i(r))
9 AdaptRule(P_i(r))
```

the receiver where a change occurs, then propagates the change recursively. At each step, it checks if all the children of a node $P_i(r)$ receive less than their parent (line 3). If it is the case, $P_i(r)$ does not need to receive its current bitrate from $P_i(P_i(r))$. The latter bitrate is adapted (line 5) and the adaptation rules at $P_i(r)$ are updated by Algorithm 3 (line 6) further described in this paper. The algorithm operates recursively (line 8) until it reaches the sender (line 7) or there is no need to go further, i.e., the bitrate received by $P_i(r)$ has not to be changed.

Complexity: In Algorithm 1, the number of recursive calls is bounded by the height of the multicast tree, which is itself bounded by the diameter of the network. The diameter is in $O(n)$ in the worst case, however, it is much smaller in random and realistic networks. It is well known that the diameter of an Erdős-Rényi[2] graph $G(n, \phi)$ is "almost constant" when ϕ is fixed, and the diameter of a scale-free graph is $\sim \log n / \log \log n$ [14]. At each recursive call, Algorithm 1 checks the bandwidth between $P(r)$ and its children. This suggests that this operation is in $O(p)$ (the number of children of any node in the tree is smaller than the number of participants). The same argument applies to the complexity of Algorithm 3 (AdaptRule). Assuming that the diameter of the network is in $O(\log n)$, the complexity of Algorithm 1 is in $O(p \log n)$.

2) Change at the sender: When the bitrates are recomputed and the value of $b(s_i)$ is modified. The method used is almost the same, except that the propagation is performed from the sender to the receivers. Again, the goal is to keep the adaptation rules as close as possible to the sender. The main difference between this case and the previous one is that the propagation is not only done on a branch of the tree but can occur on several branches down to the leaves (receivers). Algorithm 2 is performed when the sender's bitrate decreases. It starts at each sender's child. From each one of these nodes, if their own children receive more than the sender's bitrate (line 3), the children bitrates should be changed (line 4) and

[2]Erdős-Rényi graphs are usually denoted by $G(n, p)$ where p is the probability for each link to exist. We replace it by ϕ in order to avoid confusion with the number of participants.

Algorithm 2: Relocate Down

```
1 Input: A tree T_i and node r
2 foreach c_{i,j} ∈ C_i(r) do
3     if b_i(r, c_{i,j}) ≠ b_i(P_i(r), r) then
4         b_i(r, c_{i,j}) ← b_i(P_i(r), r)
5         AdaptRule(P_i(r))
6         RelocateDown(c_{i,j})
```

the adaptation rules updated (line 5). The algorithm performs recursively (line 6) until reaching a leaf. Unlike Algorithm 1 that starts from a leaf but infrequently reaches the sender, Algorithm 2 always reaches at least one receiver. Because there is always a receiver that receives the same bitrate as the sender (otherwise the sender could send a lower bitrate), and this receiver's bitrate must be updated.

Complexity: Algorithm 2 browses all the tree in the worst case. Assuming that the diameter is in $O(\log n)$, the size of the tree is at most $O(p \log n)$, which is also the complexity of Algorithm 2.

Algorithm 3: Adapt rules

```
1 Input: A tree T_i and node r
2 foreach c_{i,j} ∈ C_i(r) do
3     if b_i(r, c_{i,j}) = b_i(P_i(r), r) then
4         Delete rule if exists
5     else
6         Rule(r, c_{i,j}) ← (b_i(P_i(r), r), b_i(r, c_{i,j}))
```

3) Adapting the SVC Downsizing Rules: When incoming and outgoing bitrates are changed at a node r, the adaptation rules should be updated. Algorithm 3 takes as input a node r and, for each one of its children, deletes the old rule if it exists and replaces it by the correct one, i.e., the pair (incoming bitrate into r, outgoing bitrate from r to its child). On any node, there is at worst one rule per child. The number of children of any node is smaller than the number of participants, this gives a complexity of $O(p)$.

4) The General Algorithm: Algorithm 4 is performed in the SDN controller. The call is first established by using algorithms MST or SPT of [1]. Then at each event the controller reacts and adapts the multicast trees. We consider four events:

- $B(r) \downarrow$: The downlink bandwidth of a node r decreases. This can lead to a change of $b_i(r)$ and $b(s_i)$ for each tree T_i because of formulas (1). In this case, the bitrates are recomputed and the new bitrate of r is propagated in all the trees (except the one where it is the sender) using Algorithm 1. Thus the complexity of processing this event is in $O(p^2 \log n)$.
- $B(s_i) \downarrow$: The Uplink bandwidth of a node s_i decreases. This can affect the bitrate of s_i and those of the receivers, but only in the tree T_i where s_i is the sender. The change is propagated using Algorithm 2 on the access node of s_i, which is its only child. The complexity is that of Algorithm 2, i.e., $O(p \log n)$.

- $B(s_i) \uparrow$: The uplink bandwidth of a node s_i increases. Again, this can impact the bitrates of s_i and the receivers $r_{i,j}$. After recomputation, if the receivers' bitrates do not change, there is no need to propagate the new bitrate of s_i, because no receiver can get a higher bitrate. Otherwise, the new bitrates are propagated in T_i using Algorithm 1. Likewise the case $B(r) \downarrow$, the complexity is in $O(p^2 \log n)$.
- $B(r) \uparrow$: The downlink bandwidth of a node r increases. This case is more complex since, according to formulas 1, it can impact all the bitrates in all the trees (except the one where r is the sender). For each tree, there are two possible cases. i) The sender's bitrate is not affected, in this case, the new downlink bitrate of r is propagated upward. ii) The sender's bitrate is affected, in this case it can, in turn, affect the other receivers' bitrates. This case is similar $B(s_i) \uparrow$. The complexity of processing this event is in $O(p^3 \log n)$.

Algorithm 4: The general algorithm

```
Perform MST or SPT to create a video conference
while an event occurs do
    if Downlink bandwidth of a receiver r decreases (B(r) ↓) then
        Recompute all the bitrates
        foreach Multicast tree T_i do
            RelocateUp(T_i, r)
    if Uplink Bandwidth of a sender decreases (B(s_i) ↓) then
        Recompute the uplink bitrate of s_i
        RelocateDown(c) where {c} = C(s_i)
    if Uplink Bandwidth of a sender increases (B(s_i) ↑) then
        Recompute the uplink bitrate of s_i
        Recompute the downlink bitrates of each r_{i,j}
        foreach receiver r_{i,j} of T_i do
            if the bitrate of r_{i,j} increased after recomputation
              then
                RelocateUP(T_i, r_{i,j})
    if Downlink bandwidth of a receiver r increases (B(r) ↑) then
        Recompute all the bitrates
        foreach Tree T_i do
            if the bitrate of s_i increases after recomputation then
                foreach receiver r_{i,j} of T_i do
                    if the bitrate of r_{i,j} increased after
                      recomputation then
                        RelocateUp(T_i, r_{i,j})
            else
                RelocateUp(T_i, r)
```

Note that the number of participants is much smaller than the network size, i.e., $p \ll n$. If we consider the network size as a parameter, Algorithm 4 processes each event in $O(n)$ if the diameter is linear and in $O(\log n)$ in the more realistic case where the diameter is logarithmic. This complexity is much lower than the complexity of MST/SPT that is in $O(n^3)$ if the network is dense and $O(n^2 \log n)$ if the network is sparse [1]. A lower complexity when processing an event implies less consumed resources in the controller and better reactivity for the participants.

IV. Evaluation

A. Parameters

In order to evaluate the efficiency of our solution, we performed simulations that compared it with three other approaches:

- Unicast method with recomputation at each event, called **Uni-recomp.** At each change of access bandwidth, the shortest paths between each pair of participants are recomputed.
- MST method with recomputation at each event, called **MST-recomp.**
- SPT method with recomputation at each event, called **SPT-recomp.**
- Our solution, called **MST-adapt.** MST method is performed to establish the video call then the multicast trees are adapted at each event according to Algorithm 4.

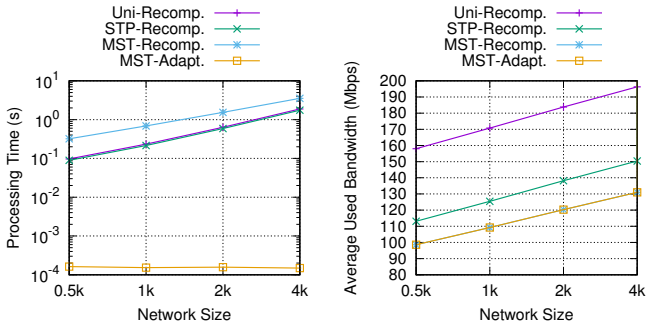
The simulations are performed on two types of random topologies with varying size: Erdős-Rényi (ER) $G(n, \phi)$ with $\phi = 0.5$ and Magoni-Pansiot (MP) [15] which is a scale-free topology derived from real networks. There are 6 participants for each call and 10 events chosen randomly in $\{B(r) \downarrow, B(r) \uparrow, B(s_i) \downarrow, B(s_i) \uparrow\}$. Each result is obtained by averaging 100 runs.

We aim to estimate the trade-off performed by our solution between bandwidth saving and processing time. It is clear that MST and SPT methods should perform equal or more bandwidth saving since they recompute the multicast trees and replace the SVC adaptation rules at each event. However, our solution has a lower complexity, which leads to less processing time and used resources.

B. Results

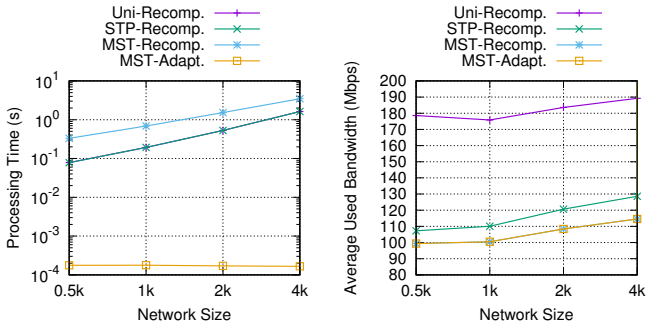
Figures 2(a) and 2(b) show the results obtained on Erdős-Rényi topologies for networks from 500 to 4000 nodes. On Figure 2(a), one can observe the average processing time per event of the different algorithms. Note the logarithmic scale. Our solution is much faster than the other approaches. For instance, on a network of 4000 nodes, our solution takes around 4.10^{-4}s while all the others take around 4s. Thus achieving a ratio of 10^{-4}. The same results are obtained using SPT or unicast instead of MST to establish the video call. Figure 2(b) shows the average used bandwidth in the core network per call. Surprisingly, it is the same for our solution and MST-recomputing, and better than the other algorithms. It appears that the bandwidth variations at an access link are very small compared to the bandwidth scale in the core links. They are too small to induce a change in the multicast tree topologies. Consequently, our solution is thousands of times faster than a total recomputing while providing exactly the same bandwidth saving.

The same results can be observed on MP networks on figures 3(a) and 3(b). All the algorithms are faster because MP networks are sparse while ER ones are dense. However, the ratio between processing time of our solution and the other ones is still around 10^{-4}.



(a) (b)

Fig. 2. Average processing time/bandwidth usage *vs* ER network size.



(a) (b)

Fig. 3. Average processing time/bandwidth usage *vs* MP network size.

The results on both topologies with different sizes show that our approach of replacing the SVC adaptation rules in the multicast trees is well adapted to a dynamic context. While the algorithms proposed in [1] are very efficient to establish a video call, it appears that our adaptation algorithms are more suited for processing the different variations occurring during the call.

V. Conclusion

Videoconferencing is a critical tool in today's connected world. Regrettably, the users still suffer from poor QoS because of management difficulties of the video calls. The emergence of SDN represents a chance to simplify this management and to provide more flexibility. Some works propose to use SDN together with video layering techniques with the aim to adapt the video quality to the users' heterogeneity. In order to minimize the bandwidth consumption in the core network while providing the best possible video quality to the users, multicast trees are computed and video layer adaptations are performed in the nodes of the core network. To be serviceable, these methods should be evaluated in a dynamic context where access channel bandwidth variations occur. In this paper, we proposed to evaluate these methods in such a context and to show their reactivity. We proposed much faster algorithms, based on tree traversal ideas, that adapt the video layering without recomputing the multicast trees. The low complexity of these algorithms allows high reactivity to network changes and low resource consumption at the SDN controller. The simulation results confirm the efficiency of our algorithms in terms or processing time. Moreover, they show that our solution performs as good bandwidth saving as more costly and elaborate methods.

Acknowledgment

This work was supported, in part, by SFI grant 13/RC/2094 to Lero - the Irish Software Research Centre, and in part by ANR grant 10IDEX-03-02 to the University of Bordeaux.

References

[1] C. Al Hasrouty, C. Olariu, V. Autefage, D. Magoni, and J. Murphy, "SVC Videoconferencing Call Adaptation and Bandwidth Usage in SDN Networks," in *IEEE Global Communications Conference*, 2017.

[2] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. on circuits and systems for video technology*, vol. 17, no. 9, pp. 1103–1120, 2007.

[3] M. D. De Amorim, O. Duarte, and G. Pujolle, "Single-loop packet merging for receiver-oriented multicast multi-layered video," in *IEEE International Conference on Communications*, 1999.

[4] K. Katrinis, B. Plattner, B. Brynjúlfsson, and G. Hjálmtýsson, "Dynamic adaptation of source specific distribution trees for multiparty teleconferencing," in *ACM Conference on Emerging Network Experiment and Technology*, 2005.

[5] V. Chandrasekar and K. Baskaran, "Performance of video conferencing in unicast and multicast communication using protocol independent multicast routing," *International Journal of Computer Science and Telecommunications*, vol. 2, 2011.

[6] M. Zhao, B. Jia, M. Wu, H. Yu, and Y. Xu, "Software defined network-enabled multicast for multi-party video conferencing systems," in *IEEE International Conference on Communications*, 2014, pp. 1729–1735.

[7] S. Laga, T. Van Cleemput, F. Van Raemdonck, F. Vanhoutte, N. Bouten, M. Claeys, and F. De Turck, "Optimizing scalable video delivery through OpenFlow layer-based routing," in *IEEE Network Operations and Management Symposium*, 2014.

[8] E.-z. Yang, L.-k. Zhang, Z. Yao, and J. Yang, "A video conferencing system based on sdn-enabled svc multicast," *Frontiers of Information Technology & Electronic Engineering*, vol. 17, no. 7, pp. 672–681, 2016.

[9] D. Wu, Y. T. Hou, W. Zhu, Y.-Q. Zhang, and J. M. Peha, "Streaming video over the internet: approaches and directions," *IEEE Trans. on circuits and systems for video tech.*, vol. 11, no. 3, pp. 282–300, 2001.

[10] L. De Cicco, S. Mascolo, and V. Palmisano, "Skype video responsiveness to bandwidth variations," in *ACM Int'l Work. on Network and Operating Systems Support for Digital Audio and Video*, 2008, pp. 81–86.

[11] M. Vutukuru, H. Balakrishnan, and K. Jamieson, "Cross-layer wireless bit rate adaptation," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 3–14, 2009.

[12] Z. Ding, J. Wu, W. Yu, Y. Han, and X. Chen, "Pseudo analog video transmission based on LTE physical layer," in *IEEE/CIC International Conference on Communications in China*, 2016.

[13] D. He, C. Lan, C. Luo, E. Chen, F. Wu, and W. Zeng, "Progressive pseudo-analog transmission for mobile video streaming," *IEEE Transactions on Multimedia*, 2017.

[14] B. Bollobás and O. Riordan, "The diameter of a scale-free random graph," *Combinatorica*, vol. 24, no. 1, pp. 5–34, 2004.

[15] D. Magoni and J. Pansiot, "Internet topology modeler based on map sampling," in *IEEE Symposium on Computers and Communications*, 2002, pp. 1021–1027.