SDN-Driven Multicast Streams with Adaptive Bitrates for VoIP Conferences

Christelle Al Hasrouty, Vincent Autefage, Cristian Olariu, Damien Magoni, John Murphy

SDN-Driven Multicast Streams with Adaptive Bitrates for VoIP Conferences

Christelle Al Hasrouty[§], Vincent Autefage[§], Cristian Olariu[*], Damien Magoni[§] and John Murphy[*]
[§] University of Bordeaux, LaBRI, Talence, France
[*] University College Dublin, School of Computer Science, Belfield, Dublin 4, Ireland
christelle.alhasrouty@labri.fr, autefage@labri.fr, cristian.olariu@ucd.ie, magoni@labri.fr, j.murphy@ucd.ie

***Abstract*—Achieving high-quality voice conference calls over the Internet is a difficult task. Heterogeneous mobile devices and network dynamics must be properly managed by multiparty VoIP systems to ensure a good quality of experience. In this paper, we propose a multiparty VoIP system based on SDN technology that uses both multicast distribution and dynamic stream adaptation to optimize the conference call quality for each participant. We define a tree construction algorithm that includes two modes: a minimum spanning tree mode and a shortest path tree mode. We have simulated conference calls with various parameters and results show that a trade-off can be achieved between path latency and bandwidth consumption when using multicasted streams with adaptive bitrates. Compared to a typical multiparty VoIP system, up to two-third bandwidth consumption reduction is possible with only a small increase of the average path latency. Our solution could be leveraged by other types of real-time applications such as online gaming.**

I. Introduction

Voice conference calls have always been challenging to achieve due to their specific constraints on network requirements. With the advent of the Internet as a cheaper alternative to the private telecommunication networks, VoIP technologies have gained a wide acceptance among individual users and organizations. VoIP conference calls, also called Multiparty VoIP (MVoIP) calls, were soon developed along the way. However, as the Internet was not initially designed to enforce strict QoS requirements, VoIP and MVoIP applications need specific mechanisms to achieve a proper quality of experience (QoE) for the users. For example, VoIP streams can be heavily degraded in the presence of heavy background traffic or harsh network conditions if no provisioning or priority is given to them. Recent technologies such as real-time adaptive codecs enable the design of new improved VoIP solutions. In particular, the introduction of SDN architectures clearly enables new possibilities for complex application-layer systems such as MVoIP conference calls.

As VoIP does not provide the same uniformity as dedicated telephone networks in terms of performances, the connection characteristics on the client side may vary widely among the participants when a conference call is set up. New audio/voice codecs, such as Opus, are capable of adjusting the bitrate of the audio stream on the fly, allowing a dynamic optimization of the stream to the characteristics of the connection. However, this adaptation is currently occurring inside the users' devices only and there is no possibility of changing the stream's bitrate in the core of the network. In order to leverage multicast, we need to be able to change the bitrates of the streams at the points of duplication inside the network. In order to solve this problem, we propose a new centralized algorithm that builds MVoIP sessions according to the desired reception bitrate of each participant while minimizing the bandwidth usage in the network. The implementation of our algorithm would be done in a SDN controller. Our paper presents the following contributions:

- We discuss previous work related to MVoIP solutions, as well as SDN solutions applied to VoIP communications (Section II).
- We define the overall architecture of our system (Section III).
- We describe an algorithm for building the multicast trees of the voice streams from each participant to all the others (Section IV).
- We perform an evaluation of our proposal by running simulations and we discuss results showing the potential gains on bandwidth usage and the impact on path latency (Section V).

To our knowledge, no research yet has looked at optimizing MVoIP applications by using SDN. In this paper, we define such a MVoIP system which leverages SDN technology to simultaneously perform stream multicasting and dynamic bitrate adaptation.

II. Related Work

MVoIP conference call systems have been the focus of research for many years. An early work by Venkatesha *et al.* in 2003 tackled the issue of noise floor control in MVoIP calls [1]. As mixing several streams increases the floor noise, they proposed a system to set the maximum number of participants to allow in a conference, in order to retain an acceptable audio quality. In 2006, Xu *et al.* proposed a peer-aware silence suppression for MVoIP conferences [2]. They limit the number of concurrent speakers by performing silence suppression and speaker selection by a distributed system running in each client. The design of a complete MVoIP conferencing system was proposed by Sat *et al.* in 2007. The system leverages user-observable metrics as well as network metrics to adapt its transmission topology, loss concealment schemes and play-out scheduling [3]. Elleuch and Houle proposed a large-scale peer-based MVoIP system in 2008. Their solution enables multi-host media process support while the conference control and

management are kept simplified and centralized around the administrator [4]. They build two different meshed networks to enable both voice audio distribution between participants and general conference control. Evaluating and comparing audio mixers for MVoIP has been done by Chandra *et al.* in 2009. They proposed their own algorithm which outperforms the others in terms of quality and complexity [5]. In 2009, Mani *et al.* studied MVoIP systems based on the RTP bandwidth used, the quality degradation due to trans-coding and the placement of voice-enhancement modules. They proposed a MVoIP system with adaptive sampling rate scaling up to ten clients [6]. Hoeldtke and Raake defined in 2011 a new state model for MVoIP calls and applied statistical measures to recordings made during a three-party conferencing quality test, which was designed to evaluate the perceived quality under various speech transmission properties [7]. In 2013, Adel *et al.* showed that the G.107 E-model is inaccurate in measuring the QoE of MVoIP calls and thus they proposed an improved E-model specifically designed for MVoIP sessions, which produces results very similar to the PESQ scores [8].

In recent years, researchers have begun investigating the power of the SDN paradigm applied to VoIP applications. Jivorasetkul *et al.* proposed in 2013, an end-to-end header compression mechanism for reducing latency in SDN networks thus improving time-sensitive applications [9]. In the context of home networks, Kumar *et al.* proposed to leverage SDN to enable ISPs to expose some controls to the users to manage service quality for specific devices and applications in their household [10]. In 2014, Saldana *et al.* used SDN to identify flows of small packets (such as VoIP) for removing common header fields and multiplexing packets in the same frame in order to reduce the overhead [11]. A Network Services Abstraction Layer (NSAL) and a unified data model were introduced by Sieber *et al.* for both SDN and legacy devices in order to achieve QoS for time-critical VoIP applications [12]. QoS level guarantee and resource prioritization enforced by SDN has been proposed by Karaman *et al.* in 2015 via limiting bandwidth, assigning flows to different queues and adapting routing decisions based on network conditions. They were able to reduce loss rate by 5% and latency and jitter by more than 50% in VoIP scenarios [13].

III. System Design

The overall architecture of our system is presented in this section. A MVoIP session or call is a multiparty VoIP conference between three or more clients called participants. Each participant is both a sender to, and a receiver from all others. Also we assume that each participant is using a mobile device connected through a 3G access network, hence the access bandwidth is supposed to be scarse. The core network is SDN-enabled. Figure 1 depicts a SDN deployment that transports one VoIP call whose stream is transcoded along the way (i.e., the bitrate of the stream is reduced). The deployment is composed of OpenFlow-enabled switches that are connected in-band to a SDN controller. End users are connected via 3G UMTS base stations to the SDN network, and can place VoIP

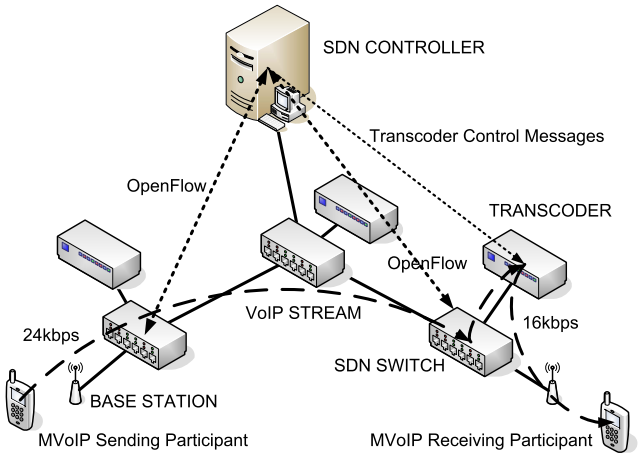

Fig. 1: SDN MVoIP system architecture

calls between them. The end users' devices use voice codecs that can pick from a range of streaming rates. Typically these rates range from a few kbps up to a few tens of kbps. One of the common lowest rate is 8 kbps (e.g., for Opus, AMR, G279), while the upper rate can go up to 64 kbps (e.g., for G711). As changing the bitrate a little has no real effect on the audio quality, the possible bitrates usable by the devices are usually quantified into a few meaningful separate bitrate levels. As depicted in Figure 1, access links are wireless. They are sensitive to signal interference and have variable communication channel conditions. For this reason, this paper considers the access links to have limited capacity, thus sometimes hindering users from either sending or receiving at peak codec bitrate. Higher bitrates are preferred when the network conditions allow, as typically call quality is correlated with codec bitrate.

In a MVoIP call, users can be affected by the channel limitations discussed above, and a simple solution to this is to limit everybody's voice codec bitrate to match the conditions of the worst receiver. However, there will be an unnecessary loss of quality affecting users that have good channel conditions. Another solution is to establish unicast streams between each participant, where each session will match every peer's receiving capacity. This solution will lead to a wasteful increase of bandwidth usage. Our proposed solution uses multicast trees, thus avoiding the issues from the unicast solution, however with a twist. Each participant will send at a maximized codec rate, in order to satisfy all participants with good receiving conditions, whereas the participants affected by channel conditions will be served by a downgraded version of the same stream. The downgrading will happen at appropriate locations in the network. Setting those locations is the topic of the next section where our algorithm is explained. Thanks to SDN, networks can be observed now globally from a central unit. Once the network topology and channel conditions are known, the MVoIP scenario and its issues described above can be tackled more efficiently. Figure

1 depicts a simple transcoding scenario. The sender's codec operates at 24 kbps, however unaware that the receiver can only receive 16 kbps. As the controller is aware of the channel conditions of the receiver, it will activate a transcoder attached to an OpenFlow switch that will downgrade the stream to 16 kbps. The transcoder is defined as a device which is able to reduce the bitrate of audio/voice streams on the fly. We envision the transcoder as a computing machine that is connected to a nearby OpenFlow switch. The transcoder needs to be instructed about which rate to transcode to, and that can be done in at least two different ways. One option is for the transcoder to be managed by the SDN controller using extended OpenFlow messages, as depicted in Figure 1. This can be done using the Experimenter Field in the OpenFlow specification, while the OpenFlow process inside the transcoder can be run by OpenVSwitch. The coding rates at which streams have to be transcoded are given by the SDN controller directly to the transcoder. Another option is for the transcoder to be a simple machine that will listen to incoming voice packets and transcode them. The desired transcoding rate can be signalled by using different port ranges for different desired codec rates. For example, any packets arriving at port x to port $x+1000$ should be transcoded to 8 kbps, and so on. The reason for having more ports assigned to the same codec rate is that some codecs have packet inter-dependencies that should be factored in when transcoding, making thus transcoding a state-full process. In any of the solutions, the SDN controller sets a flow entry in the corresponding OpenFlow switch with the action to forward the packets of the stream that needs transcoding to the outgoing interface towards the transcoder.

IV. Construction of MVoIP Sessions

A. Example

We start this section by showing a simple example of a scenario where a MVoIP session features calls that need to be transcoded and we will use Figure 2 as reference for our explanations. Each participant needs to build a muticast tree to reach its peers, and for simplicity we depicted one such tree for the participant marked as sender. In our diagram the sender tries to reach all its three receivers. As we use SDN, the controller is the entity where all participants, their channel limitations, and the network topology between them are known. In our example, receiver #1 can receive at 32 kbps, receiver #2 can receive at 24 kbps, receiver #3 can receive at 16 kbps, and all these limits are upper bounds.

The algorithm starts by looking for receivers that don't need transcoding. As receiver #1 can receive at the original codec bitrate, the stream is forwarded there without having to be transcoded. Then the algorithm proceeds by looking for participants that can receive at the next bitrate downwards. In our example, it will find receiver #2. Thus, in the path between switch #2 and #4, a transcoding transformation is necessary. We depicted switch #2 as being such location, however the more general algorithm described in the next subsection will factor in a few extra requirements when making this decision. The algorithm has one more receiver to solve, i.e. receiver #3.

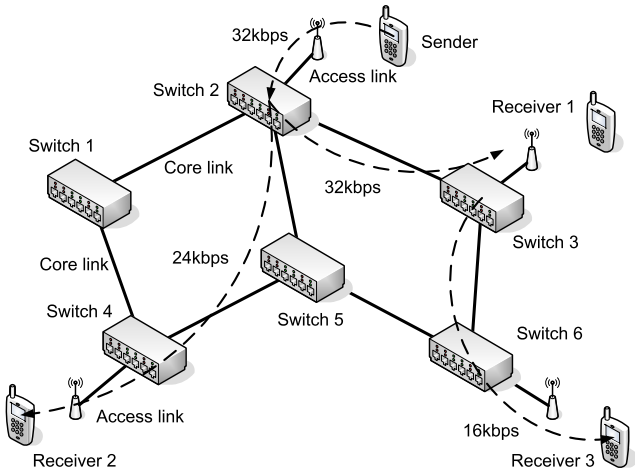

Fig. 2: MVoIP multicast tree for a source

Since there are already two streams in the network that could be transcoded to receiver #3's requirements, the algorithm has to make a decision. Our algorithm is instructed to perform a maximum of one transcoding transformation per stream, i.e. a stream that is obtained from another stream via transcoding, can not be further transcoded. In this way, the algorithm will chose to transcode the stream serving receiver #3 at switch #3 and forward it to switch #6 which finally connects to receiver #3. As mentioned, the algorithm will iteratively solve this topology for each participant while considering the other participants as receivers.

B. Algorithm

The algorithm that builds the MVoIP session containing the multicast trees of all participants (senders) to the others (receivers) relies on several assumptions (that will be changed in future work):

- Access link characteristics for all participants do not vary over the duration of the session.
- The network capacity is considered very high, much above the requirements for a single session, thus the bandwidth of core links is considered to be infinite.
- The SDN controller knows the complete topology of the network and the position of the participants.
- There is no mixing of voice streams inside the network, only transcoding is possible.
- There is one transcoder per SDN switch and it can transcode simultaneously any number of streams.
- There should be at most one transcoding operation per stream over its path from sender to receiver.

Algorithm 1 is used for building the MVoIP session. In future work, this algorithm will be run periodically to adapt the session to the variations of the access links' characteristics. The variables used in this algorithm are defined in Table I. The algorithm is centralized and runs in the SDN controller. It is composed of three main steps:

TABLE I: Algorithm's variables

Variable	Definition
B_k	Bitrate level $k \in [\![1, m]\!]$
S_i	Sender $i \in [\![1, n]\!]$
U_i	Uplink bandwidth of S_i
u_i	Uplink bitrate of S_i ($u_i = B_l$)
T_i	Tree rooted in S_i
$F_{i,k}$	Subtree(s) of T_i where links have bitrate k
R_j	Receiver $j \in [\![1, n]\!]$
D_j	Downlink bandwidth of R_j
d_j	Downlink bitrate of R_j
N	Network switch and associated transcoder
P_j	Path from R_j to S_i or T_i
SC	SDN controller

1) Collect the measured downlink bandwidth for each participant (line 1).

2) Compute the receiving and sending bitrates for each participant (lines 2-3).

3) Build a multicast tree for each participant to all the others and set transcoding where needed (lines 4-25).

In step 1, the controller retrieves the downlink bandwidth value of each participant by polling its access switch (the one through which the participant is connected). In step 2, the receiving bitrate of each participant is calculated by first dividing its downlink bandwidth by the number of participants minus itself, and then by picking the highest bitrate level lower or equal to this computed value (line 2). This means that any given receiver will ask the same bitrate to all the senders (optimizing reception with different bitrates would increase complexity so we do not consider this option for now). The sending bitrate is then defined as the maximum received bitrate level found among all participants, noted B_l (line 3). As the uplink bandwidth is considered to be always higher than the highest bitrate level B_1, the sending bitrate B_l is thus the same for each participant. It may be lower than B_1.

In step 3, each participant, considered as a sender, builds a multicast tree to all others (line 4). The bitrate levels are sorted from the highest B_1 to the lowest B_m (line 6). The receivers R_j are sorted by decreasing downlink bitrates and, at any given bitrate, by increasing distances (in hops) to the sender (line 7). For the highest bitrate B_l and the first receiver nearest to the sender, a shortest path is built to the sender (line 12). Then, for the other receivers at the same bitrate B_l, two modes are defined for connecting to the sender:

- MST: a shortest path is built up to the closest switch belonging to the tree, thus making a minimum spanning tree which minimizes bandwidth usage in the network (line 15).
- SPT: a shortest path is built up to the sender, potentially stopping at the first switch belonging to the tree, thus making a shortest path tree which minimizes latency between participants (line 17).

The stream is then duplicated at the junction switch found by one of the above methods (line 18). After all the receivers having the highest bitrate B_l are connected to the sender, they form a tree $T_i = F_{i,l}$. The next highest receiving bitrate B_k is then chosen and receivers having this bitrate, again sorted by their closeness to the source, connect to either the $F_{i,l}$ tree (i.e., the subtree with bitrate B_l) or one of the $F_{i,k}$ subtrees by one of the two modes described earlier. Due to the constraint that at most one transcoding is allowed on any path from S_i to R_j, receivers at bitrate k can only connect to switches serving B_k or B_l.

Algorithm 1: Construction of the MVoIP session

```
   BuildMVoIPSession()
1  SC collects all D_i
2  SC computes all d_i = max_{1≤k≤m} (B_k) | B_k ≤ D_i/(n − 1)
3  SC computes all u_i = max_{1≤i≤n} (d_i) = B_l
4  foreach S_i, i ∈ [1, n] do
5      T_i ← {0}
6      foreach B_k, k ∈ [l, m] | ∀ 0 < k < m, B_k > B_{k+1} do
7          foreach R_j, j ∈ [1, n] | R_j ≠ S_i ∧ d_j = B_k
           ∧ dist(R_j, S_i) ≤ dist(R_{j'}, S_i) ∀ j' > j do
8              P_j ← {0}
9              N ← {0}
10             if B_k = B_l then
11                 if j = 1 then
12                     P_j ← BuildShortestPath(R_j, S_i)
13                 else
14                     if mode = MST then
15                         P_j ← BuildShortestPath(R_j,
                           N ∈ T_i)
16                     else if mode = SPT then
17                         P_j ← BuildShortestPath(R_j, S_i)
                           stopping at N ∈ T_i
18                     Set N to duplicate B_l
19             else
20                 if mode = MST then
21                     P_j ← BuildShortestPath(R_j,
                       N ∈ F_{i,l} ∪ F_{i,k})
22                 else if mode = SPT then
23                     P_j ← BuildShortestPath(R_j, S_i) stopping
                       at N ∈ F_{i,l} ∪ F_{i,k}
24                 if N ∈ F_{i,l} then
25                     Set N to transcode B_l → B_k
26                 Set N to duplicate B_k
27             T_i = T_i ∪ P_j
```

V. Evaluation

This section presents an evaluation of our two construction algorithms for setting up a MVoIP session. For this starting work, we consider that the network conditions stay stable for the duration of the session. We have written our own static simulator in Python.

A. Settings

In order to evaluate our solution, we need to model topologies of telco operators. As this information is hard to obtain, we have generated synthetic maps by using one random graph model, the Erdös-Rényi model (ER), as well as two typical network models: the Waxman model (WM) where the probability of creating an edge decreases with the

TABLE II: Topology Models

Topology model	Edge probability $p(u, v)$	Reference
Erdös-Rényi (ER)	C	[14]
Magoni-Pansiot (MP)	$f(d(u,v), deg(u), deg(v))$	[15]
Waxman (WM)	$\beta \exp \frac{-d(u,v)}{\alpha L}$	[16]

TABLE III: Simulation Parameters

Parameter	Values
Modes	Unicast, SPT, MST
Bitrate levels	(8, 16, 24, 32) kbps
Max number of transcoding per path	1
Placement of participants	Random

distance and the Magoni-Pansiot model (MP) where the degree distribution follows a power law (as in the Internet and in large communication networks). These three models thus define different ways of creating edges as shown in Table II. This will enable us to assess the impact of the topology on our solution.

Table III presents the various input parameters used in our simulations. MVoIP sessions are created by the two modes of our algorithm (SPT and MST) presented in Section IV and by a baseline unicast mode (typical of most applications), where each participant has unicast connections to each others. The access downlink capacity between a participant and its access switch is randomly selected from 144 kbps to 384 kbps and the uplink is set to 64 kbps, which are conservative real-life values observed in current 3G UMTS networks [17]. The core links are supposed to have an infinite capacity as we currently evaluate the efficiency of one session at a time only. We leave the study of the usage of the overall network capacity for future work. Access links delays are set to 30 ms (a typical average value as shown in [18] and core link delays are set to 10 ms (as observed in [19]. 100 sessions were created per network and 10 networks were generated per topology type thus leading to 1000 sessions per topology type.

B. Results

This subsection presents some of the results that were obtained by our simulations carried out with the parameters previously described. As stated above, each point is the average of 1000 measured values. Furthermore, 95% confidence intervals are plotted for each point. Figure 3 shows the impact of the number of participants on the total bandwidth used by one session on average (only the audio data streams). As expected, the MST mode is the least bandwidth consuming, closely followed by the SPT mode. They both exhibit a sublinear relationship between the number of participants and the bandwidth consumption. The unicast mode, on the other hand, exhibits a polynomial relationship and consumes much more bandwidth. This is due to the redundancy of the streams on the shared links. With 8 participants, each of the two multicast modes roughly consumes 60% the bandwidth used by the unicast mode and at 16 participants, this ratio drops to 43%. Figure 4 shows the impact of the topology model and network size on the bandwidth. For the ER model random

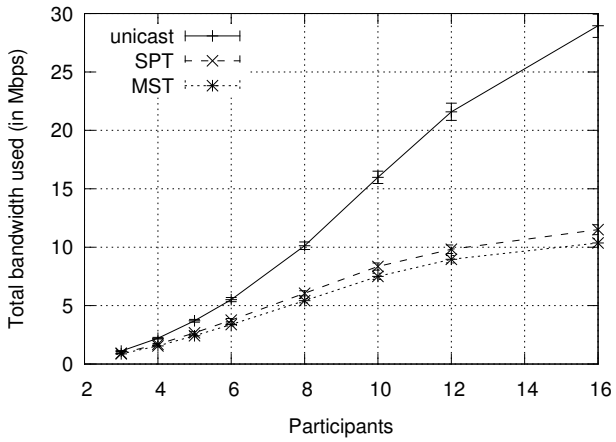

Fig. 3: Total bandwidth consumption *vs* modes used and participants for a 2k-node MP topology

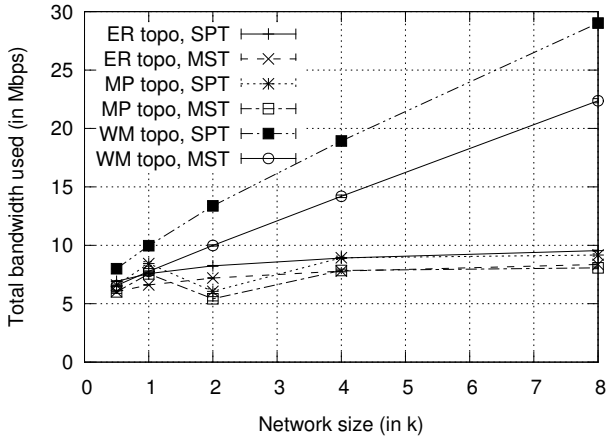

Fig. 4: Total bandwidth consumption *vs* modes used, network type, and size in nb of nodes for 8 participants

graphs and the MP model power-law graphs, an increase in the size of the network only very moderately increases the bandwidth consumption. For the WM model, however, there is a nearly linear relationship between network size and bandwidth usage. This is expected because, in the WM model, the presence of edges is inversely proportional to the size of the graph. Thus, when the network size increases, the distances inside it linearly increase thus leading to increased bandwidth usage. The network topology characteristics of the SDN network will have to be thoroughly studied in order to control its impact on bandwidth usage.

The impact of the number of participants on the average path latency is shown in Figure 5. The latency takes into account access and core link delays but not codec and de-jitter delays. The SPT and unicast modes yield the same results whatever the number of participants, which is expected as they both use shortest paths. The latency does not depend on the number of participants as they are randomly selected. We also observe that the values of the MST mode are

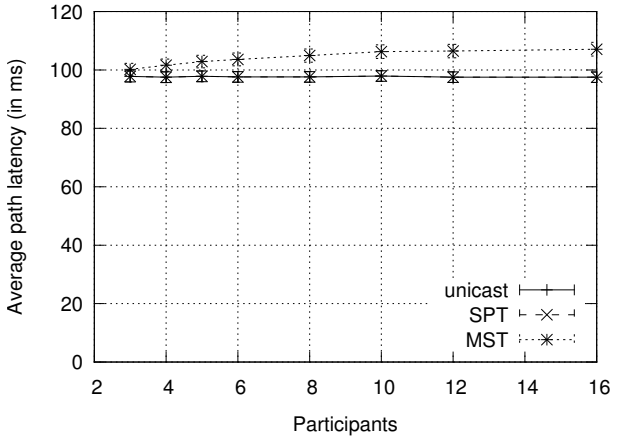

Fig. 5: Average path latency *vs* modes used and participants for a 2k-node MP topology

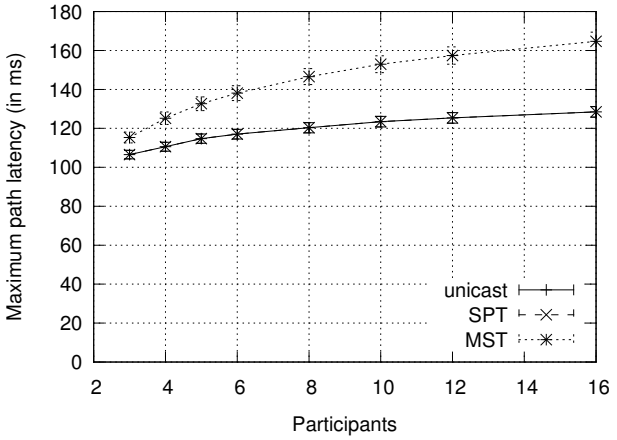

Fig. 7: Maximum path latency *vs* modes used and participants for a 2k-node MP topology

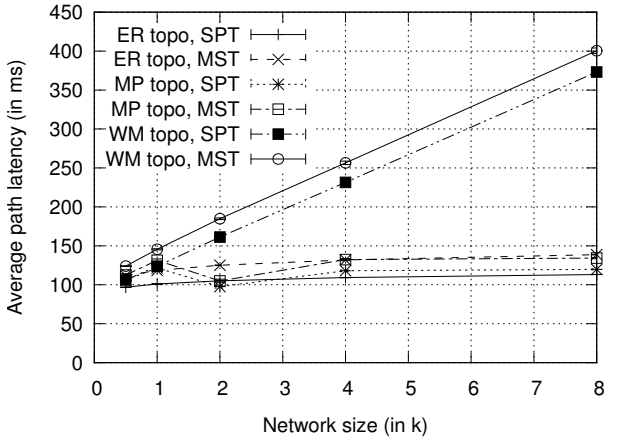

Fig. 6: Average path latency *vs* modes used, network type, and size in nb of nodes for 8 participants

very close to the other two modes and only get moderately worse when the number of participants increases. It must be noted that this latency represents communications between participants located inside the same network operator as stated in our hypotheses. Most of the path latency comes from the uplink and downlink access latency (i.e., 2×30 ms), thus the distances covered in the core of the network are only moderately impacting the overall latency. This result would of course change if the network were a worldwide one with large distances or if several network operators were involved in the session. The influence of the topology model and the network size on the average path latency is shown in Figure 6. Similarly to the results shown in Figure 4, only the WM model has a significant influence on the latency, because of the linear increase of the distances with respect to the size of the network. With a number of participants set to 8, latency in WM networks is barely acceptable in networks up to 8k nodes with values slightly above 400 ms. The maximum path latency, which measures the biggest latency between any two pairs of participants in a session, is shown in Figure 7. The number of participants increases the latency sublinearly. As expected, unicast and SPT modes have the same results as they both use shortest paths. The MST mode exhibits larger latencies, with an increasing gap when the session grows up to 32% more at 16 participants. This contrasts with the average path latency results shown in Figure 5. Given our hypotheses and simulation parameters, the worst case path latency still remains below 180 ms with 16 participants in a typical 2k-node MP modeled operator network.

Figure 8 shows the average number of transcoded streams per box (counting boxes transcoding at least 1 stream, duplicating a stream is not counted) *vs* the number of participants. This value logically increases with the number of participants whatever the mode used. There is no transcoding at 3 to 5 participants as the downlink bandwidth is sufficient to handle all 32kbps streams. However there is a threshold at 12, were this value no longer increases when using multicast modes. This is most probably due to having all strategically placed boxes transcoding all needed streams already. Adding more participants thus only implies duplicating at these boxes instead of transcoding in other boxes. As the MST mode aggregates more paths, we see in the figure that it has more streams per box than the SPT mode. The influence of the number of participants on the total number of boxes carrying out at least one transcoding operation is shown in Figure 9. The SPT mode is the most sensitive to the session size as it exhibits the biggest number of transcoding boxes whatever the session size. As the SPT mode does not minimize the size of the multicast tree (unlike the MST mode), it uses boxes where needed inside the network. This also explains why the SPT mode has the lowest number of streams per box shown previously in Figure 8. The MST mode uses much less boxes than the SPT mode because, by construction, it minimizes the size of the multicast tree.

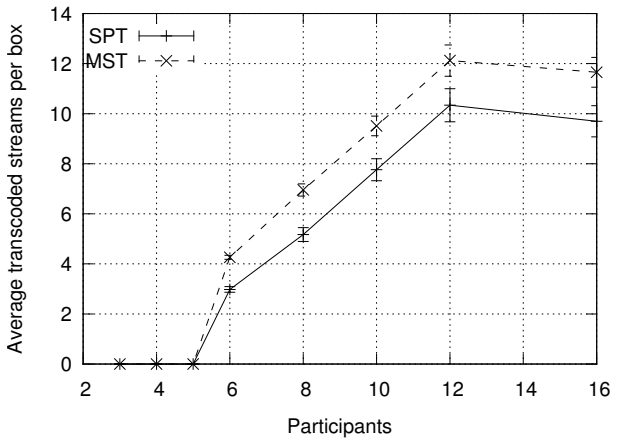

Fig. 8: Average transcoded streams *vs* modes used and participants for a 2k-node MP topology

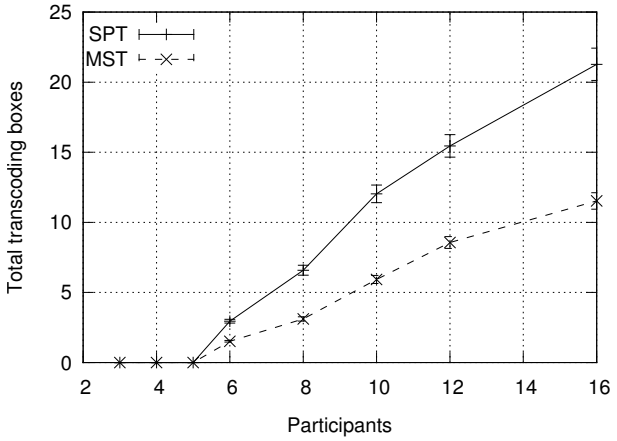

Fig. 9: Total transcoding boxes *vs* modes used and participants for a 2k-node MP topology

VI. Conclusion

Designing efficient VoIP conference systems is challenging. In this paper, we have defined a MVoIP system leveraging the SDN paradigm in order to provide adaptive bitrates to each participant. The combined use of multicast distribution and stream transcoding placement enables important bandwidth savings. Our results have shown that a MVoIP session based on our MST mode uses only a fraction of the bandwidth of a unicast-based MVoIP system (60% at 8 participants and 43% at 16 on a 2k-node MP topology network) while increasing the average path latency by only a few percents and the maximum path latency by 32% at 16 participants, with acceptable values remaining under 200 ms. The results also show that the number of boxes required to transcode streams is roughly halved when using the MST mode over the SPT one. Our system is designed to dynamically adapt to the possibly time evolving access bandwidth of the participants, but we haven't validated this aspect yet. In our future work, we will analyze our system over the duration of a session, and maximize the use of the total network capacity by optimizing the construction of the multicast trees over many simultaneous sessions.

Acknowledgment

This work was supported, in part, by SFI grant 10/CE/I1855 to Lero - the Irish Software Research Centre, and in part by ANR grant 10IDEX-03-02 to the University of Bordeaux.

References

[1] R. Venkatesha Prasad, H. Jamadagni, and H. Shankar, "On the problem of specifying the number of floors for a voice-only Conf. on packet networks," in *Int'l Conf. on Information Technology: Research and Education*, Aug 2003, pp. 22–26.

[2] X. Xu, L.-W. He, D. Florencio, and Y. Rui, "PASS: Peer-Aware Silence Suppression for Internet Voice Conferences," in *IEEE Int'l Conf. on Multimedia and Expo*, July 2006, pp. 2149–2152.

[3] B. Sat, Z. Huang, and B. Wah, "The Design of a Multi-party VoIP Conferencing System over the Internet," in *9th IEEE Int'l Symp. on Multimedia*, 2007, pp. 3–10.

[4] W. Elleuch and A. Houle, "Multiparty Voice over IP (MVoIP) Peer-Based System for Large-Scale Conference Support," in *IEEE Int'l Conf. on Wireless and Mobile Computing*, Oct 2008, pp. 415–416.

[5] S. Chandra, K. Senthil, and M. Bala, "Audio mixer for multi-party conferencing in VoIP," in *IEEE Int'l Conf. on Internet Multimedia Services Architecture and Applications*, Dec 2009, pp. 1–6.

[6] S. K. Mani, B. M. Prasad, Kamesh, and N. Mani, "DSP subsystem for multiparty conferencing in VoIP," in *IEEE Int'l Conf. on Internet Multimedia Services Architecture and Applications*, 2009, pp. 1–6.

[7] K. Hoeldtke and A. Raake, "Conversation Analysis of Multi-Party Conferencing and Its Relation to Perceived Quality," in *IEEE Int'l Conf. on Communications*, June 2011, pp. 1–5.

[8] M. Adel, H. Assem, B. Jennings, D. Malone, J. Dunne, and P. O'Sullivan, "Improved E-model for monitoring quality of multi-party VoIP communications," in *IEEE Globecom Workshops*, Dec 2013, pp. 1180–1185.

[9] S. Jivorasetkul, M. Shimamura, and K. Iida, "Better network latency with end-to-end header compression in SDN architecture," in *IEEE Pacific Rim Conf. on Communications, Computers and Signal Processing*, Aug 2013, pp. 183–188.

[10] H. Kumar, H. Gharakheili, and V. Sivaraman, "User Control of Quality of Experience in Home Networks using SDN," in *IEEE Int'l Conf. on Advanced Networks and Telecommuncation Systems*, 2013, pp. 1–6.

[11] J. Saldana, F. Pascual, D. de Hoz, J. Fernandez-Navajas, J. Ruiz-Mas, D. Lopez, D. Florez, J. Castell, and M. Nunez, "Optimization of Low-efficiency Traffic in OpenFlow Software Defined Networks," in *Int'l Symp. on Performance Evaluation of Computer and Telecommunication Systems*, July 2014, pp. 550–555.

[12] C. Sieber, A. Blenk, D. Hock, M. Scheib, T. Hohn, S. Kohler, and W. Kellerer, "Network configuration with quality of service abstractions for SDN and legacy networks," in *IFIP/IEEE Int'l Symp. on Integrated Network Management*, May 2015, pp. 1135–1136.

[13] M. Karaman, B. Gorkemli, S. Tatlicioglu, M. Komurcuoglu, and O. Karakaya, "Quality of Service Control and Resource Prioritization with Software Defined Networking," in *1st IEEE Conf. on Network Softwarization*, April 2015, pp. 1–6.

[14] P. Erdös and A. Rényi, "On Random Graphs I." *Publicationes Mathematicae*, vol. 6, p. 290297, 1959.

[15] D. Magoni and J.-J. Pansiot, "Internet Topology Modeler Based on Map Sampling," in *IEEE Symp. on Computers and Communications*, 2002, pp. 1021–1027.

[16] B. Waxman, "Routing of Multipoint Connections," *IEEE J. on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.

[17] M. Glabowski, S. Hanczewski, and M. Stasiak, "Calculation of Available Bandwidth for UMTS-HSDPA/HSUPA Users," in *Int'l Conf. on Computer as a Tool*, 2007, pp. 1145–1152.

[18] M. Laner, P. Svoboda, and M. Rupp, "Latency analysis of 3G network components," in *18th European Wireless Conf.*, 2012, pp. 1–8.

[19] M. Hoerdt and D. Magoni, "Cartographie distribuée du coeur de l'Internet," *Annales des Télécomm.*, vol. 60, no. 5-6, pp. 558–587, 2005.