Sending the Right Signals: How to Deal with the Lack of Signaling with RTP/RTCP for H.264 SVC

Henrik F. Lundin and Patrik Westin

Global IP Solutions,
Magnus Ladulåsgatan 63B, SE-118 27 Stockholm, Sweden
{henrik.lundin,patrik.westin}@gipscorp.com

Abstract. H.264 scalable video coding (SVC) used in real-time video conferencing is considered. A number of conference participants encode SVC streams and send them to a conference router. The router forwards the streams to all receivers, after pruning the stream to accommodate the downlink limitations. As the streams traverse unmanaged IP networks (e.g., the Internet), the available bandwidth will vary over time. The conference router can track the variations and calculate suitable layer operation points, but the signaling means to feed this information back to the sending participants are lacking. This paper proposes a solution to this lack of signaling through an extension of the RTP/AVPF feedback framework.

Keywords: H.264 SVC, RTP/AVPF, TMMBR.

1 Background

The scalable video coding (SVC) extension of the H.264 standard [4] is generally predicted to break through within the next 1–3 years. While SVC may bring some benefit to peer-to-peer video communication, mainly through a moderate increase in error resilience, we will most likely see the largest deployment within the webcast/broadcast and conferencing spaces. The reason is that video conferencing using SVC promises to deliver quality video to a heterogeneous range of receivers with low participant uplink overhead and transcoding-free thin conference routers. That is, a conference router can adapt the outgoing streams to various receiver limits (e.g., bitrate, frame rate, frame size) without having to transcode the media streams; it is simply a matter of discarding the correct set of packets, which is easily done at a fraction of the computational cost of transcoding.

The general concept of video conferencing with SVC using a conference router was described e.g., in [1]. In that paper, the authors use the term scalable video conferencing server (SVCS) for the conference router. They claim that the SVCS provides zero algorithmic delay, avoids transcoding distortion, and reduces the computational complexity by two orders of magnitude compared with a transcoding conference server.

In this paper, we will consider SVC in a real-time conferencing scenario. Each conference participant will encode a layered video stream to be relayed through a conference server to the other participants. The conference server will *not* create a mixed video such as a "Brady bunch", but will forward the individual streams for the receivers to decode and render according to their own preference. The relaying conference server may do "stream thinning" by removing layers; the number of layers removed for each receiving participant will vary depending on the network and receiver limitations. In particular, we consider the situation where each network path between the participants and the server traverses unmanaged Internet Protocol (IP) networks, in most cases the Internet. It is assumed that the available bandwidth on each path may change over the course of the conference session, e.g., due to varying cross-traffic interference and fluctuating link speeds (for wireless local area networks (WLANs) and mobile connections). These assumptions implicate that the SVC encoders will have to adapt their layer bitrates over time to follow the varying needs of the receiver population. However, it is the conference server, and not the sending participants, that has all information necessary to design the layer properties. The problem we face is a lack of standardized protocols to convey this information back to the sender during the conversation.

In Section 2, we will describe the considered conference system setup in more detail, and in Section 3 we point out the lack of signaling support for an adaptive SVC layer rate allocation. In Section 4 an extension of the RTP/AVPF (Real-time Transport Protocol Audio-Visual Profile with Feedback) signaling protocol [9] is proposed to overcome the lack.

2 Conference System Setup

We consider a one-to-many (e.g., webinar) or many-to-many (conference) real-time video telephony system; Fig. 1 depicts the conference scenario. The streams are sent using RTP/UDP (User Datagram Protocol) over an unmanaged IP network, presumably the Internet. An immediate implication of such a scenario is that the available bandwidth for each path between participants and the router can fluctuate quite dramatically during a call, due to e.g., cross-traffic and fading signal strength (primarily for WLAN). There is no dedicated bandwidth reserved for the video traffic on any of the involved network paths. Thus, we envision a conferencing system that must constantly adapt to changing network conditions. This is in stark contrast to systems where the bandwidth and other environment parameters are determined and fixed at call setup, but we consider the adaptive system to become increasingly important as video conferencing applications attract more mobile and home users.

All media streams are routed via a conference server as Fig. 1 illustrates. The main role of the conference server during the call is to provide to each receiving participant a suitable media stream or set of media streams. The conference server will likely play an active part in the call setup process as well, but that is not within the scope of this paper. As an alternative to the centralized server

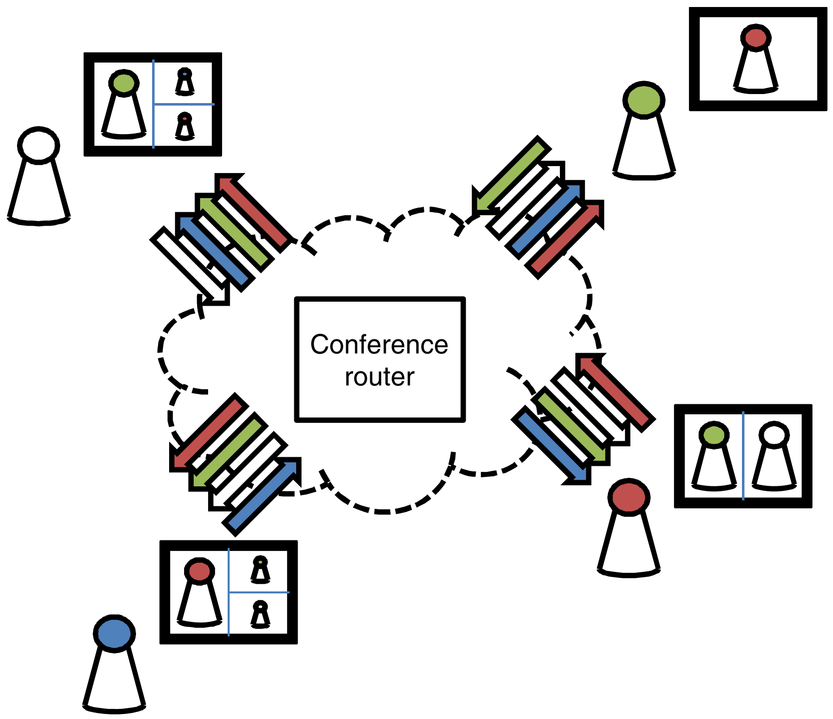


Fig. 1. A video conference system with a centralized conference server/router. Each participant gets all streams from the router, and can select the layout based on local preferences, as illustrated in each participant's "screen".

approach, peer-to-peer based video conferencing has been proposed in the literature, but this is also outside the scope of the present paper. In the many-to-many conferencing scenario, the selection of incoming streams to send out to receiving participants is usually based on voice activity detection; active speakers will be heard and seen, while silent participants may not. There are numerous aspects of selecting the streams – e.g., including handling of VIP users, viewer preferences, participants without video capabilities – but this is not within the scope of the present paper. The one-way webinar (one-to-many) scenario does not have to consider selection of streams, since one and the same source should be distributed to all viewers.

In both scenarios, the conference server must make sure that the outgoing stream to each participant is adapted to the constraints of that particular endpoint and the network path to it. That is, the server must know the current available bandwidth to each receiver, any decoder constraints, and display preferences such as frame size and rate.

If a *non-scalable* video codec is used, such as H.264/AVC, adapting the video stream to the varying needs of a heterogeneous range of receivers usually means that the server must do video transcoding, unless the encoded streams include

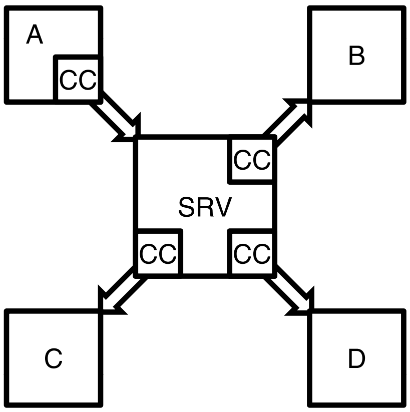


Fig. 2. Simple conference setup with one sending participant (A), one server, and three receiving participants (B–D). The arrows mark the media streams; control messages flow in the opposite direction as well. Note that A could also be receiving media streams, just as B–D could be transmitting, but the figure shows only the media flows starting at A. "CC" indicates that congestion control is used to manage the streams.

B-frames that can be discarded to reduce the rate (which can be argued to be a scalable technique). Transcoding is a resource-intensive task, which severely reduces the channel density of a conference server (i.e., the number of simultaneous channels the server can process). Video conferencing using SVC promises to deliver transcoding-free adaptation of the outgoing video streams using a fraction of the resources needed for transcoding.

While alleviating the conference server processing load and increasing the channel density, scalable video conferencing poses new demands on signaling between the server and the endpoints.

3 Problem

Consider the communication system in Fig. 2. ("CC" denotes congestion control, and will be addressed in Section 3.1.) This could represent a one-to-many webinar scenario, but it could also be a part of a many-to-many conference, as seen from one of the transmitting participants (e.g., considering only the media stream originating from the top–left participant in Fig. 1). A scalable video stream is encoded at the sending participant (participant A in the figure), and the stream is sent to the conference server. The server distributes scaled versions of the stream to its recipients. We consider a system where the rate of each channel is constantly being estimated as the network conditions evolve, and where the bitrates of each SVC layer are adapted to best match the present conditions. This is, however, not adequately covered in the supporting signaling protocols commonly used, RTP/AVPF [9] in particular. We will outline the scaling procedure in the conference server below, and describe in detail what parts of the signaling are missing. A solution is proposed in Section 4.

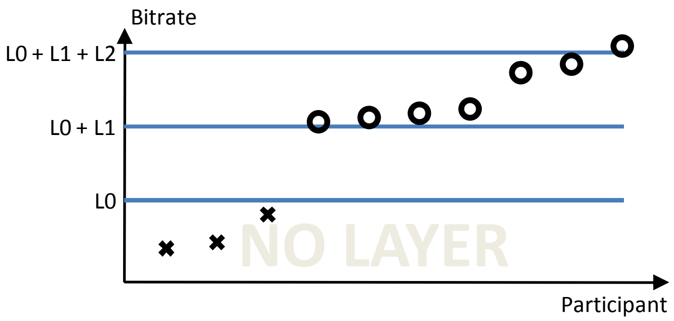
3.1 Channel Estimation

The first step is to determine the current bitrate limits of each outgoing path from the server. The limit is found using a combination of congestion control algorithms, feedback messages from the receiver, and applying any fixed limitations negotiated at call setup (e.g., level limits and maximum bitrate). The congestion control aims at finding the available bandwidth for the path, to be used in the immediate future. Since the UDP protocol does not provide congestion control, the application will have to implement the control. One example of such a control algorithm is the TCP-friendly rate control [3,2], which aims at competing fairly with TCP traffic on medium timescales, while providing a more stable bitrate than TCP on short timescales. Another option is to use the Datagram Congestion Control Protocol (DCCP) [6,5]. Furthermore, many applications apply non-standard congestion controls, that are based on the indicators available in RTCP, e.g., round-trip time and packet loss rate. However, the presentation and proposal made in the sequel does not depend on any specific congestion control algorithm. Finding a suitable bandwidth does not pose any new problems, and is not specifically addressed in the present paper. The key condition is that the conference server has an instantaneous bandwidth estimate for each outgoing path, to be used in the clustering step.

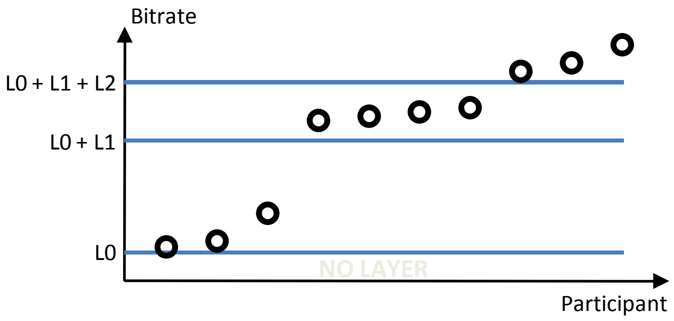
The uplink transmission from each sending participant to the server must also be adapted to the available uplink bandwidth. This is done, as for the downlinks, using a combination of congestion control, feedback messages, and fixed limitations.

3.2 Clustering the Receivers

With the results of the channel estimation procedure at hand, the task of the conference server is to maximize the satisfaction for each receiver. As the number of receiving participants grows, their various requests in terms of bitrates, loss robustness, and other parameters rapidly grow larger than the number of operation points that an encoder can produce. Each receiving participant can no longer be served with a unique stream. The server must cluster the receiving participants into different groups, or classes, as depicted in Fig. 3. Each class will be served with one bitstream, tailored such that all members of the class can use the stream with satisfaction. Clustering methods are usually based on methods for quantizer design, where the class boundaries are optimized with respect to an aggregated user distortion metric; cf. [7,8]. The various streams can differ in bitrate, error resilience, frame size, frame rate, and possibly other parameters – Fig. 3 shows the first-order model including only the bitrate. Moreover, the class boundaries will likely change over the course of the conference, reflecting changes in the receiver demography, such as available bandwidth changes, and participants joining and leaving the conference.



(a) Bad matching between participants and layer bitrates. The three participants to the left do not have enough bandwidth to accept even the first layer (L0), and are thus left without video. Furthermore, no participants are using only L0.



(b) Better matching between participants and layer bitrates. All users receive video streams, with bitrates that are fairly close to their respective available bandwidth.

Fig. 3. Clustering of receivers into groups based on bandwidth limitations. Each marker (circle or cross) represents a receiving participant, and the vertical axis shows the receivers' bandwidth limitations. The horizontal lines represent the cumulative bitrates for the encoded layers. Thus, a participant must be above a layer line to be able to receive that layer. The concept can be expanded to include other parameters, e.g., image size and error resilience preferences. Examples of good and bad matching of clusters and layer bitrates.

3.3 The Missing Feedback Link

The outcome of the clustering is a (typically small) set of media bitrates to produce. When a *non-scalable* codec is used, the server will transcode the source stream into new streams, one for each class.

When a *scalable* codec is used, on the other hand, the server will not do any transcoding. Thus, the available operation points are already determined when the stream is encoded. Two different approaches can be pursued here. Either the server is reactive and simply reduces the scalable video stream (pruning layers) until it fits each channel. There is no guarantee that the available operation

Table 1. Feedback message types (FMT) defined in RFC 4585 [9], RFC 5104 [10], and in the present paper. (The numbers of the new messages are here left undefined to avoid collision with current and future work by the Internet Assigned Numbers Authority.)

FMT	Name
1	Generic Negative Acknowledgement (NACK) [9]
2	Reserved [10]
3	Temporary Maximum Media Stream Bit Rate Request (TMMBR) [10]
4	Temporary Maximum Media Stream Bit Rate Notification (TMMBN) [10]
*****	**Temporary Maximum Media Layer Bit Rate Request (TMLBR)**
*****	**Temporary Maximum Media Layer Bit Rate Notification (TMLBN)**
31	Reserved for future extensions [9]

points fit well with the participants' needs, as is illustrated in Fig. 3. The layer structure may provide operation points that no users really benefit from (such as L0 in the figure, which is not used as target layer for any participant), while some participants may receive no layers at all (exemplified by the 'X' participants whose available bandwidths are too low for L0). A better solution is if the server is proactive, contributing to the design of the layer structure. However, there is insufficient support within RTP Control Protocol (RTP/RTCP) signaling for this design. While the server can cluster the receivers into classes, and design a layer structure to match the clustering, the server cannot convey this information back to the sending participant. This is the missing link in realizing an adaptive video conferencing solution.

4 Possible Solution

The problem presented above is one of the last remaining problems to enable adaptive scalable conferencing. While we recognize that there may be several solutions to this problem, we will suggest one solution in this paper. The solution is based on extending the codec control messages in the AVPF feedback message framework. The AVPF profile is defined in Request for Comments (RFC) 4585 [9], while the currently existing codec control messages are defined in RFC 5104 [10].

We introduce two new types of transport layer feedback messages. These types would be used in the communication between a conference server and a sending participant to request and advertise bitrates for the different layers in the stream produced by the sending participant. RFCs 4585 and 5104 together specify three transport layer feedback messages – NACK, TMMBR, and TMMBN (see Table 1 for explanations of the acronyms) – and reserve another two (types 2 and 31). Table 1 shows the previously allocated types together with the two new types proposed in the current paper.

The common packet format for all AVPF feedback messages is defined in [9], and is shown in Table 2. This is the common structure, while the FCI field varies depending on message type.

Table 2. Common packet format for all AVPF feedback messages, as defined in [9]

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|   FMT   |       PT      |          length               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of packet sender                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  SSRC of media source                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:            Feedback Control Information (FCI)                 :
:                                                               :
```

The Feedback Control Information (FCI) blocks for the two new feedback message types proposed are described in detail in the sequel.

4.1 Temporary Maximum Media Layer Bit Rate Request (TMLBR)

The proposed TMLBR is used to request that the layers obey certain bit rate limitations. The FCI field of a TMLBR message can contain exactly one FCI entry with the syntax as proposed in Table 3.

Table 3. Proposed FCI field of a TMLBR message

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| MxLBR Exp |  MxLBR Mantissa                 |    Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                             :                 :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| MxLBR Exp |  MxLBR Mantissa                 |    Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The syntax is similar to the TMMBR syntax (cf. [10]). However, the FCI field contains several rate requests, one for each layer starting with the lowest. The bitrates in a TMLBR message represent *cumulative layer rates* and must therefore be strictly non-decreasing. The measured average packet overhead value that is included in the TMMBR message is here excluded. In fact, the layer bitrates requested in the TMLBR message are net media bitrates, as defined in [10]. The (implicit) layer index is not necessarily linked to an explicit layer of the encoded stream. Each TMLBR rate describes one possible operation point of the scalable stream. The encoder may choose to encode fewer or more layers than the server requests; the TMLBN message – to defined below in Section 4.3 – will be used to announce what layer rates are actually available.

The SSRC field that is included in the TMMBR FCI field (cf. [10, §4.2.1.1]) is excluded from the proposed TMLBR field. The reason is that we do not foresee the scenario with the reporting entity addressing *several* media senders in one message, which is the motivation for specifying the SSRC in the TMMBR FCI field.

4.2 Sending Participant's Response to a TMLBR Message

Upon receipt of a TMLBR message, a sending participant can choose to obey all or parts of the request. Depending on the configuration of the scalable encoder, its ability to change layer structure on-the-fly, quality constraints, and other parameters governing the sending participant, it may not be suitable or possible to obey all requests. For instance, the scalable encoder may be limited to, say, 3 layers, while the TMLBR message may request more layers. Also, the sending participant may consider a requested bitrate to be too low for that layer.

As a consequence, the sending participant must be allowed to disregard requests that are not feasible.

We suggest the following set of rules for a sending participant:

1. The total bitrate of all layers (i.e., the last cumulative bitrate in the TMLBR message) should not exceed the TMMBR value received from the server, taking the reported average per-packet overhead into account.
2. Congestion control should be used to limit the total outgoing bitrate as for any sender.
3. Only one TMLBR message can be active for an outgoing stream. All received TMLBR messages but the last one are ignored.
4. If a TMLBR message has been received, the encoded base layer should obey the first rate request in the message, if possible.
5. The encoder may ignore any operation point requests in the TMLBR message.
6. The encoder may insert additional operation points between the layers requested in the TMLBR.

4.3 Temporary Maximum Media Layer Bit Rate Notification (TMLBN)

When an SVC sending participant has updated its layer bitrates, it will emit a TMLBN message. A proposed FCI field of a TMLBN message is shown in Table 4. Again, the SSRC field found in the TMMBN FCI entry is here omitted, with the same motivation as in Section 4.1.

The message contains the actual operation points of the encoded SVC stream. Each line corresponds to one layer, and provides the cumulative bitrate of this layer and all lower layers together with the `priority_id` value [4] for that layer. The `priority_id` is set by the encoder and serves as a unique identifier of an operation point. The encoder will label the network abstraction layer (NAL) units accordingly with matching `priority_id` values.

Table 4. Proposed FCI field of a TMLBN message

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| MxLBR Exp |  MxLBR Mantissa                 |   priority_id   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                             :                 :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| MxLBR Exp |  MxLBR Mantissa                 |   priority_id   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The TMLBN message makes the scaling and routing process in the conference server much easier, since the server can readily access the target bitrate for each operation point. For each outgoing scaled stream, the server will only have to match the available bandwidth for that path with the rates provided in the TMLBN message, and sift out the appropriate layers using the `priority_id` values.

Contrary to the rules for TMMBN, there is no need to delay the effective rate change in relation to the transmission of the TMLBN message.

4.4 System Example

In this section, we will provide a tangible example of how the new TMLBR and TMLBN messages can be used in a conference scenario.

Consider once again the conference setup in Fig. 1. Each endpoint encodes a video stream and sends it to the conference server. The server relays all incoming streams to all participants, except back to the source of each stream. Let us for simplicity, and without loss of generality, consider participant A to be a sender and the remaining participants to be receivers. The static properties of the call – e.g., payload types, profile and level limitations, and layer configuration – are settled in the call setup or possibly fixed by design. We consider the available bandwidth and other network parameters on each network path to be variable during the course of the call. The goal is to continuously adapt each encoded stream such that the overall user satisfaction is maximized.

Each transmitting entity must have a congestion control to adapt the outgoing stream to the path limitations. This also extends to that each outgoing stream from the server must be congestion controlled. The congestion controllers are indicated with "CC" in the figure. This congestion control monitors the available bandwidth of each path as it evolves over time, and forms an integral constraint on the bitrate for each stream. Furthermore, each receiving participant (B–D in our simple example) can signal further bandwidth limitations using TMMBR back to the conference server. TMMBR messages are terminated in the server, and each receiver will get a TMMBN message indicating that they are all "owners" of a rate limitation. This is simply to guarantee that all receivers continue to feedback their limitations through TMMBR as they evolve.

The collected network knowledge from congestion controllers and feedback messages are aggregated in the server. In particular, the server will form a set of the bandwidth constraints, one for each outgoing stream. The constraints are compensated for overhead to obtain the net media bitrates. The server then applies a clustering algorithm to the collected rates to form a few target layer rates – all participants that fall into the same cluster will get the same scaled video stream.

Assume now that the three outgoing paths from the server to participants B, C, and D, have been probed to convey net media bitrates 200 kbps, 650 kbps, and 700 kbps, respectively. The server may consider three rates to be feasible, and send a TMLBR message back to participant A, requesting the three layer rates 200 kbps, 650 kbps, and 700 kbps.

The sending participant A will upon receipt of the TMLBR message match the request to what is feasible and reasonable given the current layer configuration in terms of what types of layers are being encoded (which likely cannot be changed). The sender will decide on a layer rate allocation, which may or may not be identical to what was requested in the TMLBR, and send a TMLBN message with the new layer rates. The sender will also instruct its encoder to change the rate controller's targets accordingly as soon as possible. In our example above, the sender could for instance encode layers at the requested rates (200, 650, and 700 kbps). It could also consider the two top layers to be too close, and cluster them into one layer at 650 kbps.

Finally, the server can use the layer rate information provided in the TMLBN message when selecting which layers to send to each receiver. This is a clear advantage over use of Supplemental Enhancement Information (SEI) [4] messages from the H.264 encoder, in that extracting the layer bitrate information from the Scalability information SEI message is a quite involved task.

4.5 Extension to Requesting Frame Sizes and Frame Rates

We have restricted the proposed signaling schemes in this paper to only requesting bitrate limitations. A useful extension would be if the server could also request frame size and frame rate limitations in the TMLBR message. This has not been considered in the current work, but could likely be realized by using the reserved 8 bits at the end of each line in Table 3. However, since an encoder may have to re-initialize in order to change, the conference server will probably have to ask for a layer configuration identical to the one already being sent. It is nevertheless probably worth pursuing an extension of the TMLBR format to include frame size and rates, and it is our opinion that this should be considered in future work.

5 Conclusions

In this paper, we have presented a typical use case for a scalable video codec such as H.264 SVC. We pictured a scenario where several participants communicate in a real-time video conference, using a conference server. The server would relay encoded streams to the participants, and would also make use of the scalability to adapt each outgoing stream to the limits and demands of each receiver. In particular, we targeted the case where the transport is done over unmanaged IP networks, usually the Internet. This puts higher demands on stream adaptation over the course of the call.

It was identified that today's signaling protocol in RTP/AVPF does not provide sufficient tools for solving the rate adaptation task at hand. The conference server cannot request specific bitrates for the different layers (operation points) in the stream. An extension to RTP/AVPF was proposed to solve this problem. The extension consists of a new feedback message, used by the server to request a number of layer bitrates, and a matching notification message used by the sending participant to notify the server what rates it is providing.

The suggested signaling could be used with other scalable code cs as well, as long as the `priority_id` values in the TMLBN message (Table 4) can be adapted to something similar for that codec.

References

1. Eleftheriadis, A., Civanlar, M.R., Shapiro, O.: Multipoint videoconferencing with scalable video coding. Journal of Zhejiang University-Science A 7(5), 696–705 (2006)
2. Floyd, S., Handley, M., Padhye, J., Widmer, J.: TCP Friendly Rate Control (TFRC): Protocol Specification. IETF. RFC 5348 (September 2008)
3. Floyd, S., Handley, M., Padhye, J., Widmer, J.: Equation-based congestion control for unicast applications. In: SIGCOMM 2000: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, pp. 43–56. ACM (2000)
4. ITU-T. Advanced video coding for generic audiovisual services. H.264 (March 2009)
5. Kohler, E., Handley, M., Floyd, S.: Datagram Congestion Control Protocol (DCCP). IETF, RFC 4340 (March 2006)
6. Kohler, E., Handley, M., Floyd, S.: Designing DCCP: Congestion control without reliability. SIGCOMM Comput. Commun. Rev. 36(4), 27–38 (2006)
7. Kozica, E., Bastiaan Kleijn, W.: Analytical rate optimization for multicast. In: Proceedings European Signal Processing Conference (August 2008)
8. Kozicaand, E., Bastiaan Kleijn, W.: Joint optimization of the redundancy of multiple-description coders for multicast. In: Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing (April 2009)
9. Ott, J., Wenger, S., Sato, N., Burmeister, C., Rey, J.: Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF). IETF, RFC 4585 (July 2006)
10. Wenger, S., Chandra, U., Westerlund, M., Burman, B.: Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF). IETF, RFC 5104 (February 2008)