A NOVEL TECHNIQUE FOR EFFICIENT PEER-TO-PEER SCALABLE VIDEO TRANSMISSION

Stefano Asioli, Naeem Ramzan, and Ebroul Izquierdo

School of Electronic Engineering and Computer Science, Queen Mary University of London
Mile End Road, E1 4NS, London, United Kingdom
phone: + (44) 20 7882 7880, fax: + (44) 20 7882 7997
email: {stefano.asioli, naeem.ramzan, ebroul.izquierdo}@elec.qmul.ac.uk, web: www.elec.qmul.ac.uk

ABSTRACT

In this paper, we exploit the characteristics of scalable video and Peer-to-peer (P2P) network in order to propose an efficient streaming mechanism for scalable video. The scalable video is divided into chunks and prioritised with respect to its significance in the sliding window by an efficient proposed piece picking policy. Furthermore, a neighbour selective policy is also proposed to receive the most important chunks from the good peers in the neighbourhood to maintain smooth content delivery of certain Quality of Service for the received video. Experimental evaluation of the proposed system clearly demonstrates the superiority of this approach.

1. INTRODUCTION

Multimedia applications over the Internet are becoming popular due to the widespread deployment of broadband access. However, the conventional client/server architecture severely limits the number of simultaneous users, especially for bandwidth intensive applications such as video streaming. On the other hand, P2P networking architectures [1] are receiving a lot of interest, as they facilitate a range of new applications that can take benefit of the distributed storage and increased computing resources offered by such networks. In addition, P2P systems also represent a scalable and cost effective alternative to classic media delivery services. Their advantage resides in their ability for self organization, bandwidth scalability, and network path redundancy, which are all very attractive features for effective delivery of media streams over networks.

In conventional client/server applications, the video server hosts video contents of different fidelities, such as high quality material for storage and future editing and lower bit-rate content for distribution. However, this process requires either on-the-fly transcoding of compressed content or storage of multiple bit-rate video. Both these alternatives require that the content is encoded many times, which is computationally expensive. Moreover, the latter option requires additional storage capacity due to non-exploitation of redundancy among different versions of the video.

Scalable video coding techniques [2, 3] (SVC) promise to partially solve these problems, as they allow to "encode a sequence once and decode it many times, in many different versions". They enable content organisation in a hierarchical manner to allow decoding and interactivity at several granularity levels. That is, scalable coded bit-streams can efficiently adapt to the application requirements. The SVC encoded bit-stream can be truncated at lower resolution, frame rate or quality points. After this operation, the video can be decoded or further truncated.

In this paper, we exploit the features of a scalable video codec and P2P networking to perform an efficient video transmission over a distributed network. The structure of the scalable bit-stream allows to adapt video sequences to available users' resources. As a matter of fact, different viewers will experience different video qualities according to their current download bandwidth. For example, a sudden drop in the download rate might result in a degradation of the video sequence, rather than a pause or a loss of frames.

The problem of scalable video transmission over a P2P network has already been considered in the past [4, 5]. However, the peculiarity of this work is the use of a wavelet-based scalable video codec [3] in a BitTorrent-based [1] P2P system. The main contributions given by this paper are a new piece picking policy and a new neighbour selection policy. The first one associates different gains, which result in different priorities, to different parts of the bit-stream, in order to allow video playback while the sequence is being downloaded. In practice, this policy selects the subset of the video bit-stream with the highest bit-rate that can be currently afforded by a user. On the other hand, there exists a critical part of the stream, which is a very specific subset of the sequence that cannot be removed from the original video. The desired behaviour of the system is that playback should never pause. Pauses occur when a user fails to receive this critical subset before it is needed for decoding. Therefore, a neighbour selection policy is applied, which does not allow a user to request this part of the stream from peers that might fail to deliver it in time.

In this paper, Section 2 explains the proposed framework. The main contributions given by this paper are illustrated in Section 3. Section 4 provides the experimental evaluation of the proposed technique. Finally, Section 5 concludes this paper.

2. PROPOSED FRAMEWORK FOR SCALABLE VIDEO OVER P2P NETWORK

This section will give some details about working principles of the scalable video codec and the P2P client used in this work.

 ISSN 2076-1465

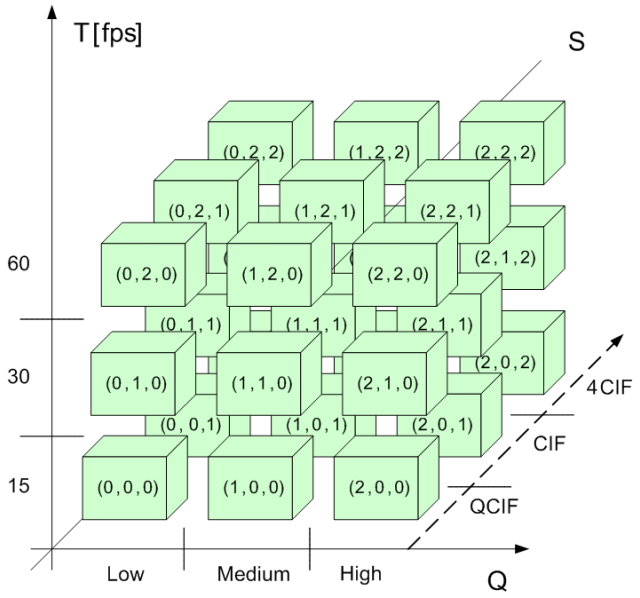

Figure 1 –3D representation of a GOP of a fully scalable video bit-stream.

2.1 Scalable video coding module

A wavelet-based scalable video (aceSVC) [3] is employed in this research work. Architecture of aceSVC features spatial, temporal, quality and combined scalability. Temporal scalability is achieved through repeated steps of motion compensated temporal filtering [7]. To achieve spatial scalability, each frame is decomposed using a 2D wavelet transform. Coefficients obtained through spatio-temporal decomposition are coded through the process of bit-plane coding [8] which provides basis for quality scalability.

2.1.1 aceSVC bit-stream organisation

The input video is initially encoded with the maximum required quality. The resulting bit-stream is divided into Groups of Pictures (GOPs). Each GOP is composed of *atoms* [9]. An atom is the smallest entity that can be added or removed from the bit-stream. Different groups of atoms form different spatial, temporal or quality layers, which can be removed to obtain the desired resolution, frame rate or quality. However, in this work only quality scalability has been considered. Figure 1 illustrates the general structure of a GOP; in this example, atoms are represented by cubes and the three dimensions correspond to the three basic types of scalability. On the other hand, Figure 2 shows that this very structure can be seen as one-dimensional if only quality scalability is considered.

In the main header of the aceSVC bit-stream, the organization of the bit-stream is defined so that the truncation is performed at different user points with low complexity. However, a GOP can only be decoded if its *base layer* has been received. It is shown in Figure 2 and it corresponds to the low quality plane of Figure 1. All the other layers are *enhancement layers*, which can be used to improve the received video quality.

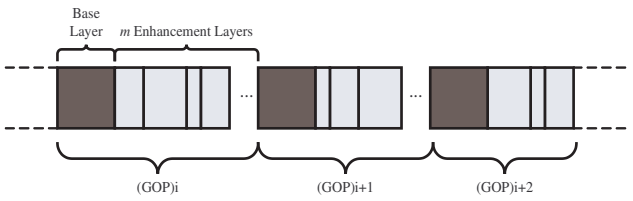

Figure 2 –Structure of scalable video bit-stream when only quality scalability is considered.

2.2 P2P network module

BitTorrent [6] (BT) is a widely used peer-to-peer protocol developed by Bram Cohen in 2003. The main idea behind it is that users' download rates should be proportional to their upload rates, in order to provide a fair mechanism and motivate users to share more. In the original version of the protocol, this is achieved using *tit-for-tat* [6] mechanism, in which peers mainly upload data to peers they are downloading from. Moreover, peers occasionally behave altruistically, in order to discover potentially good neighbours.

In this research work, Tribler [1] is used as a BT client. Its main features include exploitation of social relationships among peers and content discovery through exploration of the network, instead of browsing a torrent repository. Finally, Tribler supports video on demand for non-scalable sequences. In this case *give-to-get* [1] algorithm is used, instead of tit-for-tat, which means that peers will upload to peers that have proven to be 'generous' towards third parties, rather than those which currently have high upload rates.

3. PROPOSED MODIFICATIONS FOR EFFICIENT SCALABLE VIDEO TRANSMISSION

In this section, we formulate how the scalable layers of aceSVC video are prioritised in our proposed system. First we explain how the video segments or chunks are prioritised in our system and then efficient selection policy of good neighbours for the most important chunks is elucidated. These are new features, which have been implemented on top of Tribler client.

3.1 Piece picking policy

The proposed solution is a variation of the piece picking policy implemented in the give-to-get algorithm [1].

Scalable video bit-streams are split into GOPs and quality layers as explained in section 2.1.1, while in BT files are split into *chunks* or *pieces* [6], which have a piece id. Since there is no correlation between these two divisions, some information is required to map GOPs and layers into pieces and vice versa. This information can be stored inside an index file, which should be transmitted together with the video sequence. Therefore, the first step consists of creating a new torrent that contains both files. It is clear that the index file should have the highest priority and therefore should be downloaded first.

Once the index file is completed, it is opened and information about offsets of different GOPs and layers

in the video sequence is extracted. For the purpose of explanation, we assume for now that all BT pieces are available from at least one peer and that playback has already started. In other words, we assume that there are no *missing pieces*. At this point, it is possible to define the following quantities and functions:

L, total number of GOPs in the sequence.

Q, total number of quality layers.

W, is a window size (in GOPs).

(t, q), represents a GOP index (t) and quality layer index (q), where $(t, q) \in (0, ..., L) \times (0, ..., Q)$. Each of these ordered pairs is associated to a set of BT pieces.

t_p, current playback position (GOP). Since playback has already started, $t_p > 0$.

$v(t, q)$, value function. It measures the value currently associated to BT pieces of (t, q).

$r(t, q)$, request function. It returns the number of BT pieces associated to (t, q) that have never been requested.

S is the set of possible candidates $((t_1, q_1), \ldots, (t_N, q_N))$ which a new picked BT piece belongs to.

$P(S) = (p_1, \ldots, p_k)$ is a function that maps a set of aceSVC GOPs and layers S into a set of BT pieces; $p_1, \ldots, p_k$ are sorted according to their piece id.

This policy uses a sliding window, which consists of W GOPs. The first GOP in the window is the one with index t_p+1, while the last one has index t_p+W. Assuming that $L \gg t_p + W$, the following relations hold:

$$v(t, q) = 0 \quad \text{if} \quad (t \leq t_p,\ q \in (0, \ldots, Q)) \vee r(t, q) = 0 \tag{1}$$

$$v(t, q) = \frac{1}{1 + q}, \quad \text{if} \quad t \in (t_p + 1, \ldots, t_p + W),\ q \in (0, \ldots, Q) \tag{2}$$

$$v(t, q) = \frac{1}{1 + Q + (t - (t_p + W))}, \quad \text{if} \quad t \in (t_p + W + 1, \ldots, L),\ q \in (0, \ldots, Q) \tag{3}$$

The value function $v(t, q)$ is used to determine the benefit given by the receiving of the BT pieces associated to a certain (t, q) pair. In detail, Eq. (1) indicates that no pieces lying before the window will be requested, as no value is associated to the corresponding (t, q) pairs. The same holds for any (t, q) whose pieces have already been requested. Eq. (2) indicates that pieces inside the sliding window that correspond to the same quality layer also have the same value. Finally, Eq. (3) shows that pieces lying after the window have a value that only depends on the GOP they belong to. Moreover, these pieces always have a smaller value with respect to those inside the window.

Therefore, a peer requests pieces from layer $i+1$ only if it has already requested layer i completely. Similarly, a peer only requests pieces lying after the window if all the pieces inside the window are already marked as requested.

The piece picking rule consists of two steps:

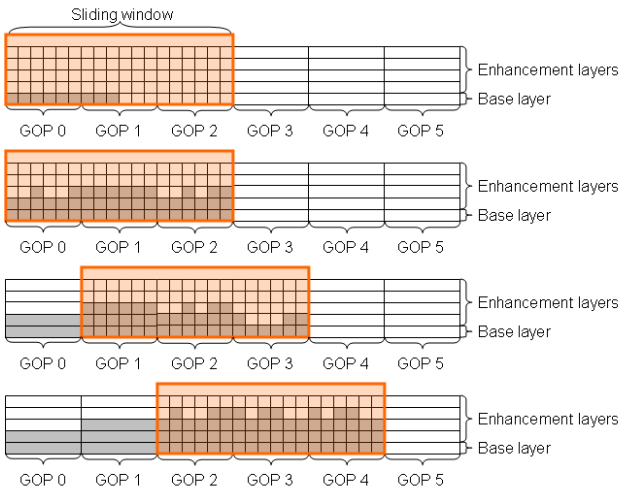

Figure 3 –Sliding window for scalable video bit-stream a) Pre-buffering phase starts, b) Pre-buffering phase ends, c) The window shifts after GOP 0, d) The window shifts after GOP 1.

1. Finding the best set of candidates S, which is given by $S = \arg\max_{(t,q)} \{v(t, q)\}$. It is a set of (t_i, q_i) that corresponds to a set of BT pieces $P(S) = (p_1, \ldots, p_k)$.
2. Choosing a piece from this set. In this step, the decision rule is the following. Given $S = ((t_1, q_1), \ldots, (t_N, q_N))$:
   - If $q_1 = \cdots = q_i = \cdots = q_N = 0$, picked piece p will be the one in $P(S)$ with the smallest id that has not been requested yet. These are pieces that belong to the base layer. This rule gives a higher priority to the base layer of GOPs according to their distance from the current playback position, since the closest ones are the most urgently required for avoiding pauses, as we will see.
   - Otherwise, the rarest piece in $P(S)$ that has not been already requested will be picked. This is the standard BT picking policy applied to a smaller set.

The window shifts every Δ_{GOP} seconds, where Δ_{GOP} represents the duration of a GOP. The only exception is given by the first shift, which is performed after the pre-buffering, which lasts $W \cdot \Delta_{GOP}$ seconds. Pre-buffering only starts when the index file has been received. Every time the window shifts, two operations are made. First, downloaded pieces are checked, in order to evaluate which layers have been completely downloaded. Second, all outstanding requests regarding pieces of a GOP that lie before the window are dropped. An important remark is that the window only shifts if at least the base layer has been received, otherwise the system will auto-pause. As far as missing pieces are concerned, they are treated as chunks that have already been requested. However, this only happens during the piece picking phase. If a missing piece belongs to the base layer, the system is paused until it is received correctly.

Otherwise, the available chunks will be extracted and decoded.

Figure 3 shows the behaviour of the system with $W = 3$. An early stage of the pre-buffering phase is shown in Figure 3a. The peer is downloading pieces from the base layer according to their piece id, while in Figure 3b the first two layers have been downloaded and pieces are being picked from the enhancement layer 2 according to a rarest-first policy. In Figure 3c, the window has shifted. As not all the pieces of enhancement layer 2 of GOP 0 have been received, this layer and higher layers are discarded. In this phase, pieces from the base layer and the enhancement layer 1 of GOP 3 have a higher priority with respect to enhancement layer 2 of GOP 1 and 2. In Figure 3d all the GOPs have the same number of complete layers and pieces are picked from enhancement layer 3. Another issue is the wise choice of the neighbours.

3.2 Neighbour selection policy

It is extremely important that at least the base layer of each GOP is received before the window shifts. Occasionally, slow peers in the swarm (or slow *neighbours*) might delay the receiving of a BT piece, even if the overall download bandwidth is high. This problem is critical if the requested piece belongs to the base layer, as it might force the playback to pause. Therefore, these pieces should be requested from good neighbours. Good neighbours are those peers that own the piece with the highest download rates, which alone could provide the current peer with a transfer rate that is above a certain threshold. During the pre-buffering phase, any piece can be requested from any peer. However, every time the window shifts, the current download rates of all the neighbours are evaluated and the peers are sorted in descending order.

Let's suppose that p is a piece belonging to the base layer that a peer wants to download and $N_1, \ldots, N_k$ are the peer's neighbours that are currently uploading to (or *unchoking* [6]) it that own this piece. Since neighbours are sorted, $R(N_1) > \cdots > R(N_i) > \cdots > R(N_k)$, where $R(N_i)$ indicates the current download rate from neighbour i and k is the number of neighbours. The threshold value is calculated as:

$$R_T = \frac{n_0 \cdot l(p)}{W \cdot \Delta_{GOP}} \tag{4}$$

where n_0 represents the number of pieces in the base layer that are currently inside the window, $l(p)$ is the size of the BT piece for this file and the other quantities have been already defined. In other words, R_T is the minimum rate that allows these pieces from the base layer to be received in time. Assuming that

$$T^* \hat{=} \min_T \sum_{i=1}^{T} R(N_i) > R_T,\ T \leq k \tag{5}$$

T^* is the minimum number of neighbours which own this piece, whose sum of download rates exceeds the threshold and $N_1, \ldots, N_{T^*}$ is the set of neighbours from which base layer pieces can be requested.

4. EXPERIMENTAL EVALUATION

The performance of the proposed framework has been extensively evaluated to transmit aceSVC encoded video over P2P network, where seeders and leechers [6] exist. Tribler client is used for P2P communication. No restrictions are applied to download bandwidth of the leechers, while, as far as seeders are concerned, constraints are applied. The proposed scheme can be implemented for any SVC encoded bit-stream. For the experiments, Crew and Soccer CIF sequences at 30 fps were encoded in aceSVC format. Crew consists of 12 quality layers (192 kbps to 1536 kbps), while Soccer is made of 10 (96 kbps to 768 kbps). In both cases the window size is $W = 6$.

In the first experiment, a comparison between different behaviours of the system with scalable and non-scalable sequences is given. Both the piece picking policy and the neighbour selection policy are active. For the non-scalable case, Crew scalable sequence is treated as a non-scalable one, which means that the window is only shifted if all the quality layers have been correctly received. On the other hand, for the scalable case the behaviour of the system is the one described in Section 3. There are only two peers that are currently unchoking the considered leecher. Upload limit is set to 400 kbps and 200 kbps for them. The seeder with an upload rate of 200 kbps periodically disconnects and reconnects to the network and pre-buffering lasts 12.8 s in both scalable and non-scalable cases.

Figure 5 shows that the proposed system allows the received video bit-rate to follow the download rate of the leecher in the scalable case, since the upper layers are discarded. This is not possible with a non-scalable video, which is paused several times, as it is shown in

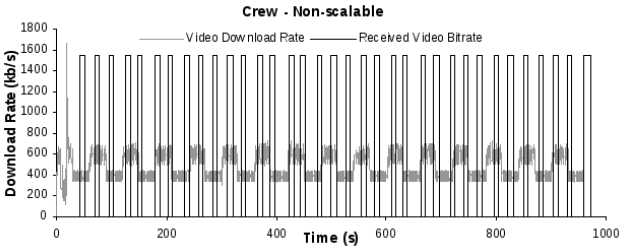

Figure 4 –Behaviour of the system with a non-scalable version of Crew sequence.

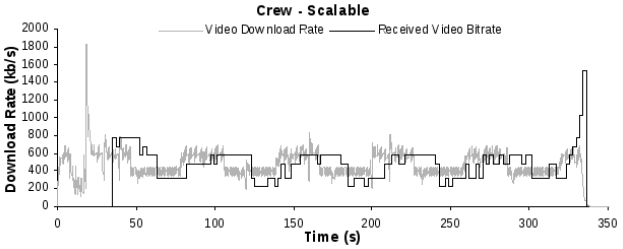

Figure 5 –Behaviour of the proposed system with a scalable version of Crew sequence.

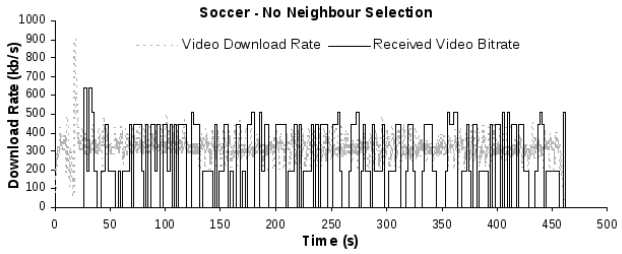

Figure 6 –Behaviour of the system when neighbour selection is not active, Soccer sequence.

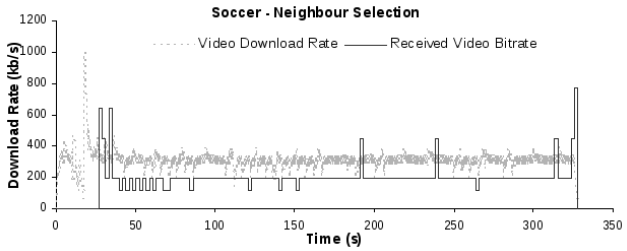

Figure 7 –Behaviour of the system when neighbour selection is active, Soccer sequence.

Figure 4. Under the same circumstances, the only way to avoid pauses in the latter case is to increase prebuffering time, which is the behaviour of the original Tribler.

As far as the second experiment is concerned, Figures 6 and 7 show the impact of the proposed neighbour selection policy on the performance of the system. In this case, upload limit for the seeders unchoking the considered leecher is set to 320 kbps and 40 kbps. The piece picking policy is active in both experiments. However, when the neighbour selection policy is not active, as Figure 6 shows, the slowest peer is requested pieces from the base layer which are not delivered in time. This results in a large number of pauses. On the other hand, Figure 7 illustrates that this behaviour is not observed when this policy is in force.

5. CONCLUSIONS

In this paper, we have presented a novel technique which enabled us to stream scalable video sequences over P2P network. We proposed a new piece picking policy and a new neighbour selection policy. The results show that this piece picking policy allows to adapt the received video bit-rate to the current bandwidth availability of a certain user. Moreover, they also show that this neighbour selection policy actually prevents a user from requesting base layer pieces from slow peers, which results in a playback without pauses. The concepts presented in this paper can be also applied to H.264/SVC, however in this work we focused on a wavelet-based scalable video codec. Our future work will focus on the importance of social-based features in these types of networks.

6. ACKNOWLEDGEMENT

This research was partially supported by the European Commission under contract FP7-216444 PetaMedia and FP7-248474 SARACEN.

REFERENCES

[1] J. A. Pouwelse, P. Garbacki, J.Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, M. R. van Steen, and H. J. Sips, "Tribler: A social-based based peer to peer system," in *5th Int'l Workshop on Peer-to-Peer Systems (IPTPS)*, Feb. 2006

[2] T. Wiegand, G. Sullivan, J. Reichel, H. Schwartz, and M. Wien. "Joint Draft 11 of SVC Amendment, Joint Video Team (JVC)," *Doc. JVT-X201*, Geneva, Switzerland, Jun. 2007

[3] M. Mrak, N. Sprljan, T. Zgaljic, N. Ramzan, S. Wan, and E. Izquierdo, "Performance Evidence of Software Proposal for Wavelet Video Coding Exploration Group," Technical Report, ISO/IEC JTC1/SC29/WG11/MPEG2006/M13146, 2006

[4] P. Baccichet , T. Schierl , T. Wiegand , and B. Girod, "Low-delay Peer-to-Peer Streaming using Scalable Video Coding," in *Proc. International Packet Video Workshop*, Lausanne, Switzerland, Nov. 2007

[5] Z. Liu, Y. Shen, S. S. Panwar, K. W. Ross, and Y. Wang, "Using Layered Video to Provide Incentives in P2P Live Streaming," in *Proc. of the 2007 workshop on Peer-to-peer streaming and IP-TV*, Kyoto, Japan, 2007

[6] B. Cohen, "Incentives build robustness in BitTorrent," in *Proc. of First Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, Jun. 2003.

[7] M. Mrak, and E. Izquierdo, "Spatially Adaptive Wavelet Transform for Video Coding with Multi-Scale Motion Compensation," in *IEEE International Conference on Image Processing*, vol. 2, pp. 317–3320, Sep. 2007

[8] T. Zgaljic, N. Sprljan, and E. Izquierdo, "Bit-Stream Allocation Methods for Scalable Video Coding Supporting Wireless Communications," *Signal Processing: Image Communications* vol. 22, pp. 298–316, Mar. 2007

[9] N. Ramzan, S. Wan and E. Izquierdo, "Error Robustness Scheme For Scalable Video Based On The Concatenation Of LDPC and Turbo Codes," in *Proc. 14th IEEE International Conference on Image Processing (ICIP)*, San Antonio, USA, Sep. 2007