Osama Abboud, Konstantin Pussep, Aleksandra Kovacevic, Ralf Steinmetz:
Quality Adaptive Peer-to-Peer Streaming using Scalable Video Coding. In: Proceedings of the
12th IFIP/IEEE International Conference on Management of Multimedia and Mobile Networks and
Services, MMNS09, p. 41-54, October 2009. ISSN 978-3-642-04993-4.

Quality Adaptive Peer-to-Peer Streaming Using Scalable Video Coding

Osama Abboud*, Konstantin Pussep, Aleksandra Kovacevic,
and Ralf Steinmetz

Multimedia Communications Lab,
Technische Universität Darmstadt,
Merckstr. 25, 64283 Darmstadt, Germany
{abboud,pussep,sandra,steinmetz}@kom.tu-darmstadt.de
http://www.kom.tu-darmstadt.de/

Abstract. P2P (Peer-to-Peer) video streaming has attracted much attention recently. However, streaming over P2P is still best effort and suffers from lack of adaptation. Therefore, video streaming over P2P either works or not. In this paper, we propose a P2P streaming system with an inherent support for adaptation. By leveraging scalable video coding, our system is able to adapt to different requirements and constraints that heterogeneous peers have in today's Internet. We make a subtle distinction between initial and progressive quality adaptation, which allows for precise adaptation to various parameters of the system and the P2P network. Our decision-taking algorithms for quality adaptation help not only in perfectly matching QoS to resources but also in bringing the P2P network to self organization.

Keywords: P2P, video streaming, SVC, adaptation, QoS.

1 Introduction

Video streaming has recently become the most traffic intensive application in the Internet. Studies [1] show that streaming a video is becoming preferred over video file sharing. One reason behind this, is that a video is usually viewed only once, therefore streaming helps in reducing storage space requirements. In addition, streaming allows for the convenient *watch while you download* experience.

Current technologies for streaming are based on either the client/server or the Peer-to-Peer (P2P) architectures. As an example of client/server streaming, Youtube [2] is a popular medium for viewing user generated content. However, although YouTube provides good performance with high availability rate, it only supports low quality videos. Moreover, YouTube inflicts enormous costs [3]. P2P, on the contrary, allows for a cost efficient solution for video delivery to potentially large audiences. It also provides desirable traits such as self organization

* This work was funded by the Federal Ministry of Education and Research of the Federal Republic of Germany (support code 01 BK 0806, G-Lab).

T. Pfeifer and P. Bellavista (Eds.): MMNS 2009, LNCS 5842, pp. 41–54, 2009.
© IFIP International Federation for Information Processing 2009

The documents distributed by this server have been provided by the contributing authors as a means to ensure timely dissemination of scholarly and technical work on a non-commercial basis. Copyright and all rights therein are maintained by the authors or by other copyright holders, not withstanding that they have offered their works here electronically. It is understood that all persons copying this information will adhere to the terms and constraints invoked by each author's copyright. These works may not be reposted without the explicit permission of the copyright holder.

and resource scalability [4]. However, P2P-based streaming is faced with several challenges, such as churn and the lack of support for peers with weak resources. Moreover, high definition video streaming is still an open challenge. Weaknesses of many P2P streaming systems come from static selection of streaming parameters that are based on average peer resources. This selection might work if all systems in the network would have equal resources, which is not true due to the heterogeneity of the Internet. Internet devices are heterogeneous not only in their resources, but also in the type of connections they have. Therefore, bandwidth, delay and reliability vary drastically, rendering current P2P video streaming techniques best effort, i.e. they either work or not.

A possible solution to the problem of supporting streams with different qualities is achieved by creating a different video file for each quality level and therefore different overlays or swarms. However, this solution is not only inefficient due to data duplication across overlays, but also limited with respect to the level of possible collaboration between strong and weak peers across different overlays. To overcome these challenges, we investigate advanced adaptation mechanisms that are efficient and have potential for an overlay-wide collaboration. In addition, we aim at supporting adaptation for heterogeneous devices of the Internet ranging from handheld mobile devices to high-end computer machines.

Our main contribution is a P2P video streaming system based on Scalable Video Coding (SVC) with an inherent support for adaptation. We make use of a mesh-based streaming architecture that is applicable to both live streaming and video-on-demand. The key feature of our design is that it allows for adaptation to resources with three degrees of freedom. Receivers can have different

- Screen sizes and resolutions,
- Connections with variable downlink bandwidth and delay, and
- Processing capabilities.

This paper is structured as follows, Section 2 states the problem statement of this paper. Background of SVC are explained in Section 3. Our proposed architecture for P2P streaming is presented in Section 4. In Section 5, we present some simulation results. Section 6 provides related work. Finally, we conclude this paper in Section 7.

2 Problem Statement

Every device connecting to the Internet has specific resource characteristics, which includes different processing power, screen resolution, and bandwidth. However, video streaming is a demanding application and works only when minimum resource requirements are met. Many devices simply do not meet these requirements. This problem can be overcome if the quality is reduced, hence streaming capacity requirements are lowered, which leads to support for more devices, but at the expense of streaming lower quality. Increasing the quality, on

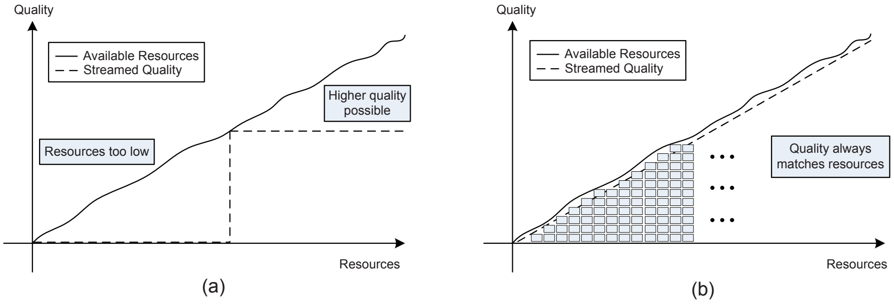


Fig. 1. Comparison of video streaming: (a) without quality adaptation, (b) with quality adaptation

the other hand, increases capacity requirement, but then a wide set of devices are unable to participate. This aspect is depicted in Figure 1.

In this paper we consider the problem of how to stream a video to a device adaptively by identifying the highest quality level supported by its available resources. Here we consider two types of resources:

- Peer resources such as screen resolution, processing power[1], and bandwidth.
- P2P overlay resources such as active neighbors, throughput, and network condition.

We propose an architecture that supports low scale mobile devices and high definition streaming at once. Our quality adaptation mechanisms help in introducing Quality of Service (QoS) into P2P video streaming by taking device resources and network state into account. Hence, we define a set of requirements and constraints with which a device can adapt to different scenarios and therefore achieve best performance.

3 Scalable Video Coding

SVC, which is based on the H.264/MPEG-4 AVC standard [5], allows for scalability by encoding a video stream into multiple layers or sub-streams each with different quality information. The lowest layer, called *base layer*, is always needed for decoding the video. With more enhancement layers received, better video quality is available. SVC is based on three modalities or flavors of scalability: spatial scalability, temporal scalability, and quality scalability. Quality scalability is also called Signal-to-Noise Ratio (SNR) scalability, both terms will be used interchangeably throughout this paper. Now we give a brief overview over SVC, more information can be found in [5] [6].

[1] In mobile devices, processing power directly translates to battery life, hence using processing-power-aware quality adaptation allows for a tradeoff between energy requirements and quality.

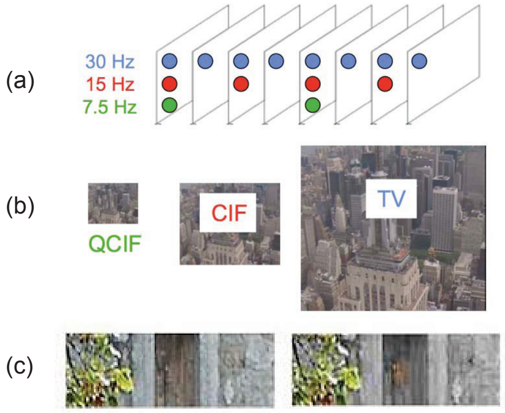


Fig. 2. Scalability variations: (a) temporal scalability, (b) spatial scalability, (c) SNR scalability [6]

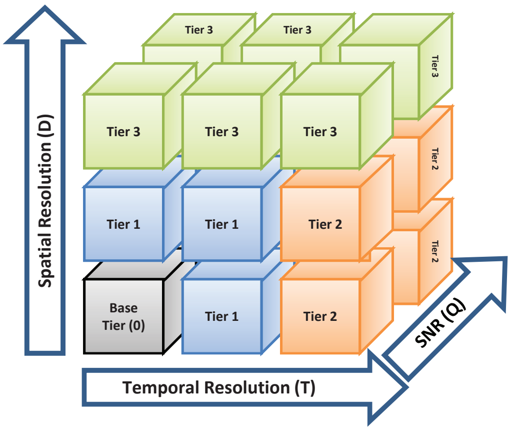


Fig. 3. SVC basic chunk structure: one chunk contains many blocks in the different dimensions of scalability

The different dimensions of scalability offered by SVC are as follows:

- Temporal scalability is based on providing different frame rates for a video stream as shown in Figure 2a. This is achieved through structuring picture and motion estimation dependencies such that complete pictures can be dropped from the bitstream while still providing the possibility of decoding the video stream.
- Spatial scalability is based on providing different resolutions for a video stream as shown in Figure 2b. This is achieved through the usage of lower resolution pictures to predict data of higher resolutions pictures.

- SNR scalability is based on providing different quality levels for a video stream as shown in Figure 2c. This is achieved through hierarchical construction of quantization coefficients for each picture.

To allow for streaming, an SVC stream is divided into chunks. Each chunk contains layers in the three dimensional quality space. The smallest quality unit is called a *block* as shown in Figure 3. A block will be used as basic unit for fetching and distributing video data across the network.

4 Quality Adaptive Streaming

Now we present the core concepts behind our quality adaptive streaming architecture. Quality adaptation based on the SVC design is basically performed during layer selection, which is responsible for making a decision on best matching spatial, temporal and SNR layers. Based on this decision, block selection requestes the blocks needed to stream the selected layers. Our proposed architecture for quality adaptive video streaming is presented in Figure 4.

Quality adaptation mechanisms for layer selection fall into two broad categories: *Initial Quality Adaptation* (IQA) and *Progressive Quality Adaptation* (PQA). When a peer wants to start viewing a video, it first invokes the IQA module, which chooses the quality level best suited to the static resources of the peer. After peer discovery, which locates all peers streaming same or lower quality, peers are selected and put in the active senders set. The peer selection module interacts with the underlay awareness module in such a way to optimize the overlay based on various metrics. After successful connection establishment with provider peers, streaming starts filling the video buffer. During streaming a control loop has to assure proper adaptation to changing conditions. Therefore, the PQA module is responsible for reacting to and overcoming any changes in network condition and available throughput. When necessary, block selection is updated in order to support an increased or decreased quality level.

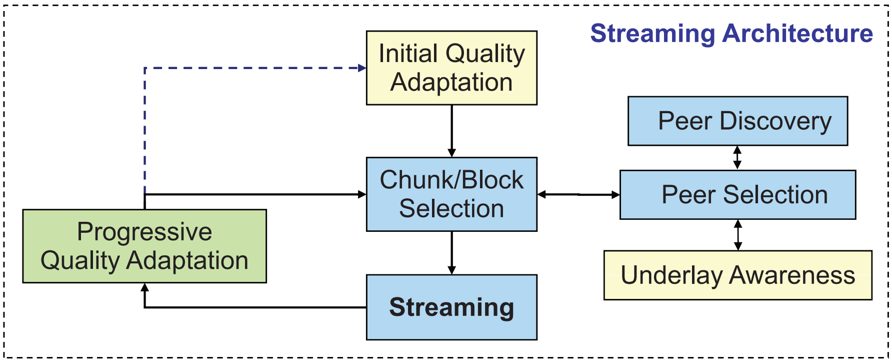


Fig. 4. The quality adaptive P2P streaming architecture

4.1 Quality Adaptation: Layer Selection

We now discuss in more detail the structure of the IQA and the PQA modules that are responsible for layer selection.

Initial Quality Adaptation. The architecture of the IQA module is presented in Figure 5. When the IQA module is invoked, it performs evaluation of current resources and requirements in order to match them with achievable quality. This module mainly handles static parameter, such as screen resolution, bandwidth, and processing power.

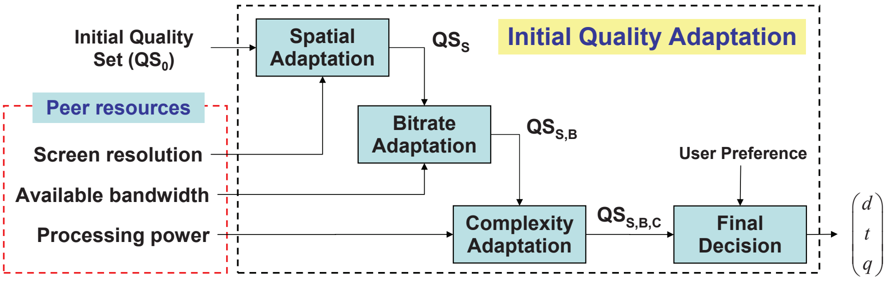


Fig. 5. Initial quality adaptation structure

First an initial quality set QS_0 is populated which contains all possible combinations for spatial, temporal, and SNR scalability. Therefore

$$QS_0 \triangleq \{(d, t, q) : \forall d = 0 \dots D, t = 0 \dots T, q = 0 \dots Q)\}$$

where D, T, and Q are the total number of layers in the spatial, temporal, and SNR dimensions. Each row in this quality set would then represent one quality combination that can be used by the peer.

The IQA works by filtering this initial quality set based on the previously stated peer resources. Therefore, the spatial, birate, and complexity adaptation modules filter out all incompatible rows in QS_0 based on screen resolution, bandwidth, and processing power respectively. Final decision is performed using user preference in case the filtered quality set $QS_{S,B,C}$ has more than one row. Our proposed algorithm for initial quality adaptation is shown in Algorithm 1.

Progressive Quality Adaptation. The PQA architecture is presented in Figure 6. The PQA is executed periodically while streaming as a part of a control loop to ensure smooth adaptation. This module adapts to changes in network conditions in order to maximize available quality at the receiver. Other than using resources information as discussed for the IQA, the PQA relies on real-time overlay status reflected from current throughput and block availability. This allows the peer to quickly react to changes in the P2P network, such as peer churn or a sudden drop in throughput.

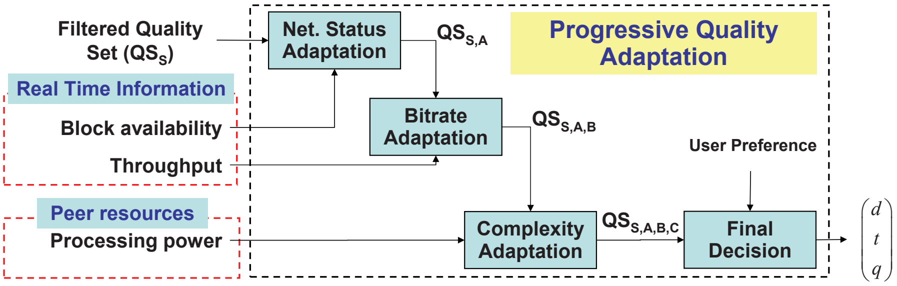


Fig. 6. Progressive quality adaptation structure

Only temporal and SNR adaptation are handled by the PQA because additionally changing the resolution frequently introduces unwanted artifacts. Therefore, the PQA starts from the pre-filtered set QS_S. The network status, birate, and complexity adaptation modules filter out all incompatible rows in QS_S based on block availability, throughput, and processing power respectively. Here, the block availability indicator provides information about layers that are available in the P2P network. User preference here is also used to make the decision in case $QS_{S,A,B,C}$ has more than one row.

Complexity Adaptation. The role of the complexity adaptation module is to take processing requirements for decoding into consideration and to match it with available processing resources. This prevents the video decoder from overloading weak mobile devices. For this end, we need to calculate processing requirements for each layer combination in QS_0. This module uses a complexity

Algorithm 1. Initial quality adaptation algorithm

```
Input: Quality level set $QS_0$
Output: Quality level $t$, $d$, $q$ fulfilling constraints
foreach row $QS_i$ in $QS$ do
    if $QS_i.SpatialLevel \leq Screen\ resolution$ then
        $QS_S.append(QS_i)$;

foreach row $QS_{S,i}$ in $QS_S$ do
    if $Complexity(QS_{S,i}) \leq Processing\ power$ then
        $QS_{S,B}.append(QS_{S,i})$;

foreach row $QS_{S,B,i}$ in $QS_S$ do
    if $Bitrate(QS_{S,B,i}) \leq Bandwidth$ then
        $QS_{S,B,C}.append(QS_{S,B,i})$;

if $QS_{S,B,C}$ is not empty then
    Filter $QS_{S,B,C}$ based on user preference
    return $t$, $d$, $q$ of single row of $QS_{S,B,C}$;
else
    return"Error: resources too low for base stream"
```

Table 1. Symbols for analytical complexity model [7]

Notion	Description
C_I, C_P, C_B	Average macroblock decoding complexity of I-/P-/B-picture
C_S, C_Q	Average macroblock decoding complexity at spatial/quality enhancement layers
$T/D/Q$	Total layer number for temporal-/spatial-/quality-scalability
$t/d/q$	Layer index for temporal-/spatial-/quality-scalability
M	Number of macroblocks per picture
α	Portion of pictures that are I-pictures

estimator that works by mapping every set of quality levels (spacial, temporal, and SNR) into processor cycles required for decoding the video stream.

To estimate the required complexity for each row in QS_0, we use an analytical model following the approach of Zhan *et al* presented in [7]. Based on definitions in Table 1, decoding complexity of an SVC stream can be calculated.

In SVC, an I-frame is a picture which can be decoded independently. This is not the case for P- and B-frames, which are a prediction of other frames in the stream. The smallest predicion unit in a frame is called a *macroblock*. Assume that C_I , C_P and C_B estimate the average complexity for decoding a macroblock in I-, P-, and B-pictures respectively. Moreover, we suppose that α is the portion I-pictures in the video and that M macroblocks exist per picture. Then the complexity for decoding scalable streams having T, D, and Q temporal, spatial, and SNR layers respectively is given by:

$$C_{\text{Decoding}} = M_0\left(\alpha C_I + (1-\alpha)C_P + (2^T-1)C_B\right) + \frac{8^{D+1}-1}{7}2^T M_0 Q C_Q + 4\frac{8^D-1}{7}2^T M_0\left(C_S + C_B\right) . \quad (1)$$

This equation is applied to all rows in the initial quality set QS_0 to calculate the complexity of decoding all possible video streams.

4.2 Block Selection

As stated in Section 3, a video file is divided into chunks. Based on the SVC design, each video chunk is further divided into blocks as depicted in Figure 7. Block selection is an important part of our streaming architecture, since it is behind making a decision on which blocks to request. This module works by assigning a probability for each block and chunk based on its importance.

In this paper, we take a multi-coefficient modeling approach for block selection. Note that here only the blocks of layers selected by the IPQ and PQA are considered. Therefore the calculated priority is always finite. The priority of a block in the video file is given by

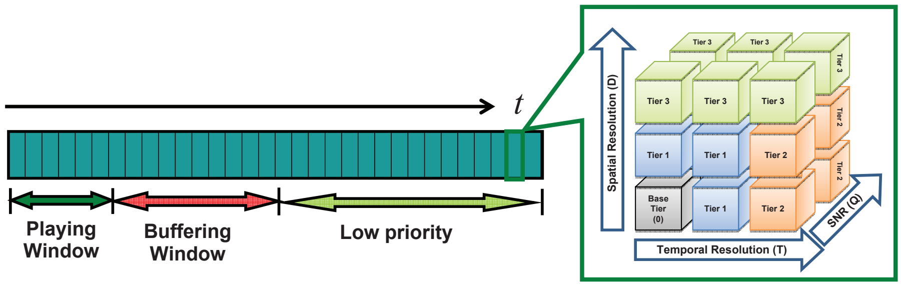


Fig. 7. File and SVC chunk structures

$$\text{Priority}(\text{Block}(t, D, T, Q)) = -A\, t - B(a\, D + b\, T + c\, Q). \tag{2}$$

Chunk priority is separated into two parts, a temporal part and a per-chunk part. This allows for a separation between urgency due to playback and due to importance to the stream. Increasing the temporal and per-chunk parameters decreases the chunk priority factors A and B respectively. Within each chunk, the parameters a, b, c define the different weights for the different blocks within each chunk. Simply put, the priority decreases with time, and is highest for the base layer, and decreases for increasing enhancement layers in any dimension. The interesting thing about such a model, is that the different weights can be different depending on the peers resources and requirements. For example, if a peers is more interested in smoother playback the parameter A is increased. If higher quality is preferred when long buffering times are not an issue, B is increased, and so forth.

4.3 Peer Discovery: Neighbor Set Management

This section describes the required support for scalable video coding in the signaling component. Our architecture utilizes a tracker-based approach known from BitTorrent and utilized by many IPTV systems. Here, the tracker manages the information about all peers participating in the swarm. Each new peer registers itself at the tracker and receives a list of potential neighbors. Peers renew their registration at the tracker periodically (to show that they are still alive) and obtain updated lists of participants.

In order to support the proposed layer adaptation mechanisms, the tracker protocol is realized as follows: the tracker manages the list of active peers together with the layers they are currently streaming. The clients send an announce request to the tracker when they: join the overlay (only base layer supported by default), perform the initial adaptation (shortly after joining), and perform the progressive adaptation. In the latter case, a minimum re-announce interval assures that the tracker is not contacted too often. Since the layer adaptation is not done at a timescale smaller than a few seconds, the impact is uncritical.

Information about the currently streamed layers is crucial since clients streaming an additional layer cannot do so if they don't have neighbors possessing blocks from the same layer. Additionally, the clients advertise the currently supported layers to their neighbors. They do so by two means:

- The common *bitfield* message is extended to support the higher granularity, which means that the message contains availability information of SVC blocks instead of whole chunks (that are different depending on the number of downloaded layers). Hence, the message contains a list of bitfields, one per supported layer, tagged with the layer ID. This way each peer can calculate useful blocks owned by its neighbors.
- Peers announce the supported layers to their neighbors. This is done during the connection establishment phase (handshaking) and later after the successful PQA (support of new layers, or discontinued support of some layers).

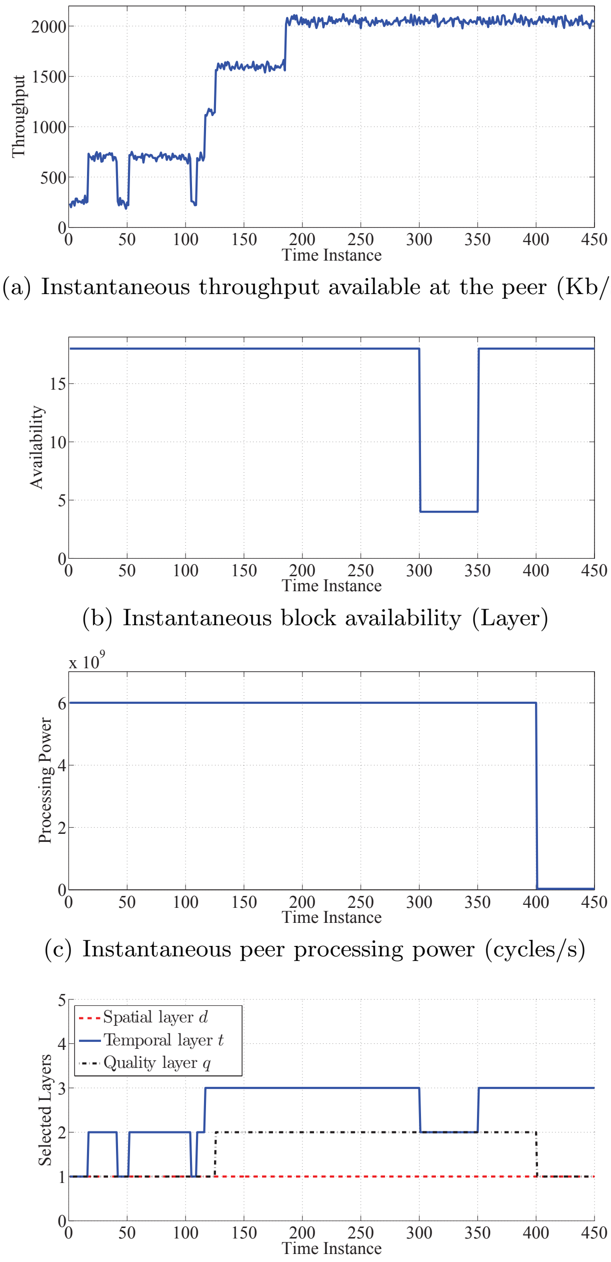
This way a peer can detect whether the required layers are supported by the current neighbor set. If the number of peers supporting a given layer falls below the threshold of, e.g. 4 peers, the peer contacts the tracker for additional new neighbors. The request contains the list of layers currently not properly supported by the neighbors. The tracker responds with a list of peers supporting these layers. Since the mechanism is bi-directional, the peers are eventually clustered according to their capabilities, while the seeders support both weak and strong peers.

4.4 Peer Selection

After assigning priorities to the different blocks in the chunk selection module, the peer selection module chooses peers to request needed chunks. In this paper, we make use of an underlay-aware peer selection strategy. Peer selection with underlay awareness can be defined as using underlay information to optimize some aspects of communication and therefore enhancing the performance of the streaming system. Different underlay metrics affect the overlay in various ways. For video streaming, it has been identified that bandwidth is the underlay metric with the greatest impact [8]. Therefore, whenever a peer has a choice between more than one provider peer, the one that can offer more bandwidth is selected. This also helps in creating a healthy clustering of the network based on streamed quality as discussed in Section 4.3. Therefore, fast peers will tend to get their blocks from similarly fast peers. Here again we extend the BitTorrent protocol to support information about bandwidth offered by every peer. This is achieved by requiring all registering peers to provide information about offered bandwidth.

5 Evaluation

Here we present a preliminary evaluation of our proposed quality adaptation mechanisms. We simulate changing parameters and see how the PQA reacts to them. We consider having 3 layers for both spatial and temporal scalability and 2 layers for SNR scalability. This leads to the total of 18 possible layer



(a) Instantaneous throughput available at the peer (Kb/s)

(b) Instantaneous block availability (Layer)

(c) Instantaneous peer processing power (cycles/s)

(d) Decision output of the PQA module

Fig. 8. Evaluation of the PQA module

combinations. We also suppose that the IQA has already decided on a basic spatial level, i.e. $d = 1$.

The PQA adapts to three parameters: block availability, throughput, and processing power. To evaluate how fast the PQA module reacts to the different parameters, we simulated a changing throughput using a Markov chain with transition matrix

$$P = \begin{bmatrix} 0.9 & 0.1 & 0 & 0 & 0 \\ 0.05 & 0.9 & 0.05 & 0 & 0 \\ 0 & 0.05 & 0.9 & 0.05 & 0 \\ 0 & 0 & 0.05 & 0.9 & 0.05 \\ 0 & 0 & 0 & 0.1 & 0.9 \end{bmatrix} \tag{3}$$

This model is used for the first 250 time instances, then the throughput is fixed at 2 Mb/sec. The next step was to change the chunk availability to simulate peer churn. In this case, blocks of layer 5 to 18 are no longer available from time instant 300 to 350. Then we simulate a sudden drop in processing power at time instant 400. These test scenarios and the results represented by the instantaneous decision on d, t, and q are shown in Figure 8. The results show that our mechanisms are able to quickly react to different changes in the system to provide a persistent availability despite an unpredictable network and changes in peer resources.

6 Related Work

Many current video streaming systems are based on the client/server architecture with servers providing the video content. However, such systems either provide low quality content, or introduce costs high enough to prevent deployment when trying to provide high quality. To shift load from servers and to allow for reduced costs, streaming solutions based on P2P architectures have been considered, like BiToS [9] and Octoshape [10]. In such systems, users act as both producers and consumers of video content.

There exists two types of overlay topologies for video streaming, push-based multicast trees and pull-based mesh topology. In push-based solutions [11], peers are structured in a tree topology with content providers positioned at the top of the tree pushing video data down the tree towards the leaves. In pull-based systems [9], a peer actively requests parts of the video from peers that have already downloaded it thus forming a mesh topology. Mesh-based streaming solutions are characterized by lower overlay maintenance costs and higher flexibility in block selection.

There is a plethora of research on P2P systems with support for adaptation such as [12] [13] [14]. PALS [14] is a receiver driven P2P video streaming system with quality adaptation. However, PALS only considers single dimensional scalability (as the case for many layered streaming systems) and therefore cannot adapt to heterogeneous characteristics of peers. Baccichet *et al.* [13] use a prioritization mechanism and multicast trees to distribute SVC streams. Lee et al.

discuss in [15] issues related to deploying a P2P video streaming system based on SVC. These approaches differ from ours since we focus on using three dimensional scalability to adapt to different peer resources and network conditions using QoS-aware decision-taking algorithms.

7 Conclusion

In this paper, we considered quality adaptive video streaming to heterogeneous devices using a collaborative P2P architecture. By leveraging quality adaptation through SVC, our system can support heterogeneous peers while providing the highest supported quality to every device. Using SVC, which allows to combine video layers from various sources, flexible received quality is possible. Therefore, weak peers receive and process only lower layers, and strong peers are able to process and share more layers, and thus can consume better video quality.

It is crucial to estimate the achievable QoS from system resources to provide highest achievable quality without overloading the devices. We have considered adapting to both peer resources as well as network state in such a way to fully utilize system capabilities.

The distinction between initial and progressive quality adaptation is crucial in separating adaptation stages of a streaming session. The initial stage assures that static resources of a peer are considered and matched to prevent overloading. Progressive quality adaptation handles what is considered as a limiting factor for P2P video streaming systems, fluctuating throughput and churn. Moreover, using an underlay aware peer selection, our solution is able to better optimize the streaming performance. Our preliminary simulation results show that our mechanisms react quickly to various system changes while providing best quality that matches current resources and network state.

As future work, we plan to do more extensive evaluations of the proposed mechanisms. Moreover, we will investigate the dynamics of both chunk and peer selection. We also want to implement a prototype to validate our findings in realistic application scenarios.

Quality adaptation is the key for next generation multimedia distribution where more heterogenous devices are joining the Internet. Client/server systems' scalability issues and high costs require the switch to P2P. But without support for heterogeneity, P2P video streaming will not be widely deployed. Nonetheless, the possibility of supporting such a wide range of collaborating peers makes us believe that our vision is not so far off.

References

1. Dissecting the Gap Between Downloading and Streaming Video, `http://www.ipsos-ideas.com/article.cfm?id=3804`
2. YouTube - Broadcast Yourself, `http://www.youtube.com`
3. Huang, C., Li, J., Ross, K.W.: Can Internet Video-on-demand be Profitable? In: SIGCOMM 2007, pp. 133–144. ACM, New York (2007)

4. Jurca, D., Chakareski, J., Wagner, J.P., Frossard, P.: Enabling Adaptive Video Streaming in P2P Systems. IEEE Communications Magazine 45(6), 108–114 (2007)
5. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. IEEE Transactions on Circuits and Systems for Video Technology 17(9), 1103–1120 (2007)
6. The Scalable Video Coding Amendment of the H.264/AVC Standard, `http://ip.hhi.de/imagecom_G1/savce/`
7. Ma, Z., Wang, Y.: Complexity Modeling of Scalable Video Decoding. In: ICASSP 2008, USA, pp. 1125–1128. IEEE, Los Alamitos (2008)
8. Spoto, S., Gaeta, R., Grangetto, M., Sereno, M.: Analysis of PPLive through Active and Passive Measurements. In: HotP2P 2009 (2009)
9. Vlavianos, A., Iliofotou, M., Faloutsos, M.: BiToS: Enhancing BitTorrent for Supporting Streaming Applications. In: 9th IEEE Global Internet Symposium 2006 (2006)
10. Octoshap, `http://www.octoshape.com`
11. Liu, Y., Guo, Y., Liang, C.: A Survey on Peer-to-peer Video Streaming Systems. Peer-to-Peer Networking and Applications 1(1), 18–28 (2008)
12. Padmanabhan, V.N., Wang, H.J., Chou, P.A.: Supporting Heterogeneity and Congestion Control in Peer-to-Peer Multicast Streaming. In: Voelker, G.M., Shenker, S. (eds.) IPTPS 2004. LNCS, vol. 3279, pp. 54–63. Springer, Heidelberg (2005)
13. Baccichet, P., Schierl, T., Wiegand, T., Girod, B.: Low-delay Peer-to-peer Streaming Using Scalable Video Coding. In: Packet Video 2007, pp. 173–181 (2007)
14. Rejaie, R., Ortega, A.: PALS: Peer-to-Peer Adaptive Layered Streaming. In: ACM NOSSDAV 2003 (2003)
15. Lee, T.C., Liu, P.C., Shyu, W.L., Wu, C.Y.: Live Video Streaming Using P2P and SVC. In: Pavlou, G., Ahmed, T., Dagiuklas, T. (eds.) MMNS 2008. LNCS, vol. 5274, pp. 104–113. Springer, Heidelberg (2008)