Retransmission Scheduling in Layered Video Caches

Michael Zink, Jens Schmitt, Ralf Steinmetz
Darmstadt University of Technology, Darmstadt, Germany
http://www.kom.e-technik.tu-darmstadt.de/

Abstract - **In contrast to classical assumptions in Video on Demand (VoD) research, the main requirements for VoD in the Internet are adaptiveness, support of heterogeneity, and last not least high scalability. Hierarchically layered video encoding is particularly well suited to deal with adaptiveness and heterogeneity support for video streaming. A distributed caching architecture is key to a scalable VoD solution in the Internet. Thus, the combination of caching and layered video streaming is promising for an Internet VoD system, yet, requires thoughts about some new issues and challenges. In this paper, we investigate one particular of these issues: how to deal with retransmissions of missing segments for a cached layered video in order to meet users' demands to watch high quality video with relatively little quality variations. We devise a suite of fairly simple retransmission scheduling algorithms and compare these against existing ones by simulative experiments.**

I. Introduction

The challenges of providing True Video on Demand (TVoD) in the Internet are manifold and require the orchestration of different technologies. Especially, technologies like the distribution and caching of video content and the adaptation of streaming mechanisms to the current network situation and user preferences are still under investigation. Existing work on TVoD has shown caches to be extremely important with respect to scalability, from network as well as from video servers' perspective [1]. Scalability is a premier issue if a TVoD system is considered to be used in the global Internet. Besides scalability, it is very important for an Internet TVoD system to take into account the "social" rules implied by TCP's cooperative resource management model, i.e., to be adaptive in the face of an (incipient) network congestion. Therefore, the streaming mechanisms of an Internet TVoD system need to incorporate end-to-end congestion control to prevent unfairness against TCP-based traffic and increase the overall utilization of the network. Note that traditionally video streaming mechanisms rely on open-loop control mechanisms, i.e., on explicit reservation and allocation of resources. As it is debatable whether such mechanisms will ever be used in the global Internet, e.g., in the form of RSVP/IntServ, we do not assume these but build upon the current best-effort service model of the Internet. An elegant way of introducing adaptiveness into streaming is to use layered video encodings [2] as it allows to drop segments (the transfer units) of the video in a controlled way. However, while the combination of caching and adaptive streaming promises a scalable and "Internet-conform" TVoD system it also creates new challenges for the design of such a system. One particular issue is that video content can only be cached in the form as it has been transmitted. For subsequent requests for that video it must thus be decided if segments from missing layers are retransmitted and if so which ones. The scheduling of these retransmissions affects the perceived quality of the cached video content in a significant way since it is very important that quality variations are minimized as they are disturbing for users [3]. Therefore, we focus in this paper on how to schedule retransmissions in order to minimize quality variations for users that are served from the video cache.

II. Scalable Adaptive Streaming (SAS)

A. Scalability - Caching

As with traditional web caches, caches for TVoD systems allow to store content closer to users, reduce server and network load and increase the system's fault tolerance. Yet, in contrast to web caches the characteristics of the data to be stored is very different. High quality video files are much larger than most web pages and therefore different caching strategies are used in caches for VoD systems. Let us briefly describe our video caching architecture. As caching method we employ so-called write-through caching[*]. With write-through caching a requested stream is either forwarded "through" the proxy cache or it is streamed via multicast and clients and proxy caches join this multicast group if the cache replacement strategy decides to store the requested video on the proxy cache. Subsequent clients can then be served from the proxy cache. This technique reduces the overall network load in a TVoD system compared to a method where the video is transported to the cache in a separate stream using a reliable transmission protocol (e.g., TCP). On the other hand, write-through caching requires a reliable multicast protocol to recover from packet losses. In [4], we present the design and implementation of such a protocol which fits particularly well in a TVoD architecture.

B. Retransmission Scheduling

With SAS, it is very likely that videos are not cached in their best quality when they are cached for the first time. However, for subsequent requests which shall be served from the

[*]. Adopted terminology from memory hierarchies.

proxy cache it may be unattractive to suffer from the possibly very bad or strongly varying quality experienced by the initial transmission of the video that has been selected for caching. Therefore, missing segments of the cached video should be retransmitted to enable higher quality service from the proxy cache to its clients. The most interesting issue here is how to schedule the retransmissions, i.e., in which order to retransmit missing segments, in order to achieve certain quality goals for the cached video content. A further design issue for retransmission scheduling is when to do it.

1) Retransmission Time

There are two basic alternatives when to do retransmissions: Directly after the initial streaming process: the cache starts requesting missing segments without waiting for further requests for a certain video. During subsequent requests: the proxy cache serves subsequent requests but, simultaneously, also orders missing segments from the origin server. While the first alternative ensures that a cached video's quality is improved as fast as possible, the second alternative inherits the advantage of write-through caching that any bandwidth between proxy cache and origin server is used only if a client request is directly related to it.

2) Scheduling Goals

First of all, it is obvious that any retransmission of missing segments increases the average quality of a cached video. Therefore, all algorithms we investigate use as much bandwidth as available between origin server and proxy cache to retransmit missing segments. That means with respect to average quality they are all the same. However, it is commonly assumed that users react very sensitive to quality variations of a video. Hence, a retransmission scheduling algorithm that tries to avoid or even decrease quality variations for a cached video can be considered superior to others which do not take this into account. The negative effect of quality variations has two dimensions:the frequency of variations, and the amplitude of variations. Hence, the goal of retransmission scheduling should be to minimize, both, the frequency and amplitude of quality variations. To state the scheduling goal more formally, let us define some terms:

h_t - number of layers in time slot t, $t = 1,..., T$

z_t - indication of a step in time slot t, $z_t \in \{0,1\}$, $t = 1,..., T$

Here, we assume without loss of generality a slotted time with slots corresponding to the transmission time of a single (fixed-size) segment and that all layers are of the same size. We can now introduce what we called the *spectrum* of a cached layered video v:

$$s(v) = \sum_{t=1}^{T} z_t \left(h_t - \sum_{j=1}^{T} z_j h_j \right)^2 \quad (1)$$

The spectrum captures the frequency as well as the amplitude of quality variations. The amplitude is captured by the differences between quality levels and average quality levels where larger amplitudes are given higher weight due to squaring these differences. The frequency of variations is captured by the z_t. Only those differences are taken into account that correspond to a step in the cached layered video. While the spectrum as defined in (1) looks very similar to the usual variance of quality levels for the cached video, it is important to note that the introduction of the z_t takes into account the frequency of changes of the quality levels.The retransmission scheduling goal for a video v can now be stated as the minimization of the spectrum $s(v)$.

III. Related Work

[5] were among the first that investigated cache replacement algorithms for multimedia streams. Yet, their work did not take into account transport issues and layered encoded video. [1] put very much emphasis on the examination of a scalable transport infrastructure for cache replacement algorithms. However, the inclusion of adaptiveness as a requirement for Internet VoD was not yet considered. In [6], the authors propose an interesting scheme of caching only the beginning of video streams. While this allows to decrease the setup latency for clients and to accommodate variable bit rate transmission channels it does not address the scalability and adaptiveness issues. Like us, [7] considers the combination of caching and layered video, yet, the latter only for the support of heterogeneous clients but not for congestion control purposes. Furthermore, the emphasis of their work is on optimal cache replacement decisions viewed over all videos stored in a cache. We, however, assume a two-stage decision process where in the first stage a video is selected for storage in a cache and then the retransmissions of missing segments are scheduled independent from the cache status of other videos. While this represents a restricted problem it ensures that the overall problem still remains manageable. [8] present an approach where only server and clients are involved and therefore the client requires sufficient buffer space to allow quality improvement of layer encoded video. Really close to our work and actually inspiring for our work was [9]. However, we extend their work by focussing on the development and comparison of different retransmission scheduling algorithms which are more flexible and performing better than the one presented in [9].

IV. Algorithms for Retransmission Scheduling

Since computation of optimal retransmission schedules is computationally infeasible or at least intensive (see [10]), we discuss some heuristic schemes in this section. One of them has been proposed in [9], whereas the others are devised by us based on shortcomings of the former.

A. Window-Based Lowest Layer First (W-LLF)

The first heuristic we want to look at has been proposed in [9]. It is fairly simple and we call it Window-Based Lowest Layer First (*W*-LLF), because the proxy cache always looks a certain number of time slots ahead of the current playout time and requests retransmissions of missing segments from the server in ascending order of their layer levels. To ensure that the retransmitted segments do not arrive after their playout time (t_p) to the current client, a prefetching offset O_p for the examined time window is introduced. O_p should be chosen sufficiently large such that $O_p >$ RTT for the transmission path between server and cache at all times. Overall, the time window $[t_p + O_p, t_p + O_p + W]$ slides over the video in discrete steps of length W until it is finished (t_{end}). The operation of the algorithm is further illustrated in Figure 1. *W*-LLF bears some

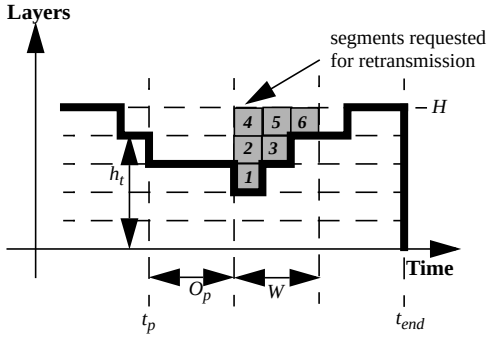

Figure 1. *W*-LLF operation.

obvious disadvantages: If, e.g., an already complete area (all layers are entirely cached) is scanned, no retransmissions are scheduled for this prefetching window, although there might very well be later parts of the video which could benefit from retransmissions. It may be possible that the currently available bandwidth between origin server and proxy cache would allow the transmission of more segments than those that are missing in the current prefetching window and again additional segments could be requested from the server to allow for a faster quality upgrade of the cached video. Although, these obvious drawbacks might be eliminated by extensions of the *W*-LLF algorithm, they exhibit a fundamental weakness of *W*-LLF: the restriction of scheduling missing segments for retransmission only for a certain number of time slots ahead. Therefore, *W*-LLF is likely to be rather shortsighted with respect to the scheduling goal of minimizing the spectrum of videos cached on the proxy. In the following, we introduce a new kind of retransmission scheduling algorithms that eliminates this restricted view.

B. Unrestricted Priority-Based Heuristics

The problems with W-LLF as described above lead us to the idea to avoid the use of a prefetching window for retransmission scheduling. That means we take an unrestricted look at all missing segments ahead of the current playout time (plus the prefetching offset O_p) when making requests for retransmissions from the origin server. Note that our algorithms still send periodic retransmission requests to the server (every W time slots) to ensure on the one hand that retransmissions and playout to the client are kept synchronized and on the other hand that the modified shape of the cached video due to retransmitted segments can be taken into account by the scheduling algorithms. Furthermore, we want to introduce more flexibility into the scheduling decisions by the notion of general priorities for retransmission scheduling decisions instead of rigidly always choosing the segments with the lowest layer level. In the following, we describe three heuristics of the more general class of unrestricted priority-based retransmission scheduling algorithms.

1) Unrestricted Lowest Layer First (U-LLF)

This algorithm is very similar to *W*-LLF because it uses as priority solely the layer level. In contrast to *W*-LLF, however, it always scans the interval $[t_p + O_p, t_{end}]$ in order to request missing segments from the server (every W time slots).

2) Unrestricted Shortest Gap Lowest Layer First (U-SG-LLF)

Considering the definition of the spectrum in Section 2.3.2 and taking into account our scheduling goal of minimizing the spectrum, we can observe that there are, in principle, two ways to decrease the spectrum of a video: to increase the lowest quality levels (which is taken care of by choosing the lowest layer levels first) and to close gaps in the video, i.e., reduce the number of . The latter is not captured by simply using layer levels as priorities. Therefore, in contrast to W-LLF and U-LLF, we now use a prioritization of the missing segments which also takes closing of gaps into account. We do so by simply sorting the segments according to the length of the gap they belong to and then sort these further by their layer levels. The resulting heuristic we called Unrestricted Shortest Gap Lowest Layer First (U-SG-LLF).

3) Unrestricted Lowest Layer Shortest Gap First (U-LL-SGF)

Since it is by no means clear, which sorting criterion, i.e., gap length or layer level, should go first we also tried the variant where missing segments are first sorted by their layer level and then sorted further by gap lengths, which we called Unrestricted Lowest Layer Shortest Gap First (U-LL-SGF).

V. Simulations

We performed a number of experiments based on a simulation environment implemented (in C++) particularly for that purpose. The simulations are performed in the following manner: For each simulation an instance of a layered video on the proxy cache is randomly generated. Here, we modeled such a layered video instance as a simple finite birth-death process since it is the result of the congestion-controlled video transmission which restricts state transitions to direct neighbor states. $\{0,..., H\}$ is the state space and birth and death rate are chosen equal as $1 - 1/\sqrt{3}$ (for all states) which results in a

mean length of 3 time units for periods with stable quality level . We use a discrete simulation time where one unit of time corresponds to the transmission time of a single segment. Our simulation environment allows then to apply the different algorithms described in Section IV and to vary parameters like, e.g., the bandwidth that is available for retransmissions between origin server and proxy cache. During the simulations, the spectrum (as defined in Section 2) of the cached video instances is continuously calculated to compare the different algorithms and their performance depending on parameters as, e.g., the available bandwidth. In all simulations we assumed a prefetching offset *of* O_p = 5 segments.

A. Comparison of the Heuristics

At first, we performed a series of 1000 simulations with all retransmission scheduling algorithms from Section IV where all parameters were chosen identical (except the windows sizes for *W*-LLF). This large sample ensured that the 95% confidence interval lengths for the spectrum values were less than 0.5% of the absolute spectrum values for all heuristics. The results for the evolution of the spectrum values for the different algorithms are shown in Figure 2. These results indicate that there is a significant gain with respect to the spectrum of the cached video for the unrestricted retransmission algorithms proposed by us in comparison to *W*-LLF. Of course, if window sizes are chosen large enough for *W*-LLF it improves and finally approaches U-LLF. Among the unrestricted algorithms there seems to be little difference such that one may argue for the use of the simplest algorithm, i.e., U-LLF.

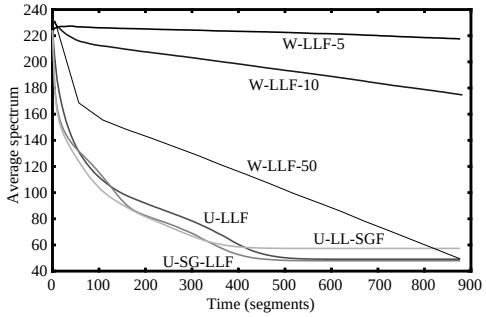

Figure 2. Average spectrum of 1000 simulation runs for each heuristic. (10 layers, retransmission bandwidth = 2, window size = 5 (*W*-LLF-5), window size = 10 (*W*-LLF-10), window size = 50 (*W*-LLF-50)).

B. Parameter Dependency Analysis

In the following, we investigated the heuristics' dependencies on certain parameters. For all of these simulations, we only used the U-SG-LLF heuristic since it showed the best performance of all heuristics in the experiment of the preceding section.

1) Number of Layers

For this simulation, we varied the number of layers per cached video from 5, 10 to 20 layers. To isolate the effect of encoding videos with different number of layers, the available retransmission bandwidth was scaled in proportion to the amount of layers, i.e., for 5 layers we assumed 2 segments of retransmission bandwidth per time unit, for 10 layers we used 4, and for 20 layers 8. For each of these 3 alternatives we ran 1000 simulations and calculated again the average of the spectra over time. As Figure 3 shows, the spectrum converges for each of the three alternatives. Yet, the higher the number of layers the higher the average spectrum. This is intuitive because the more fine-grained the layered encoding the more variations may be introduced during a congestion-controlled transmission and the harder it is for the retransmission scheduling to smooth these variations.

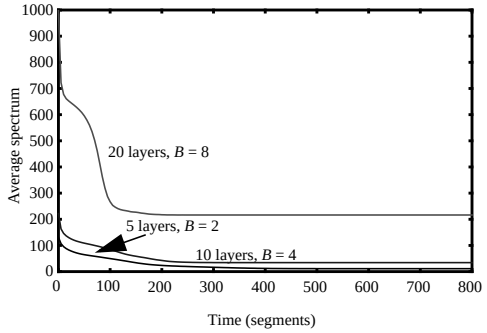

Figure 3. Different number of layers.

2) Available Retransmission Bandwidth

In the next set of simulations, the effect of different amounts of available retransmission bandwidth on the performance of U-SG-LLF was investigated. Not surprisingly, the spectrum converges faster with a higher available retransmission bandwidth. The reason for the very similar spectrum curves for B = 6 and 10 is due to sufficient retransmission bandwidth for both cases which allows to retransmit all missing parts of the rear 3/4 of the cached video. Due to the

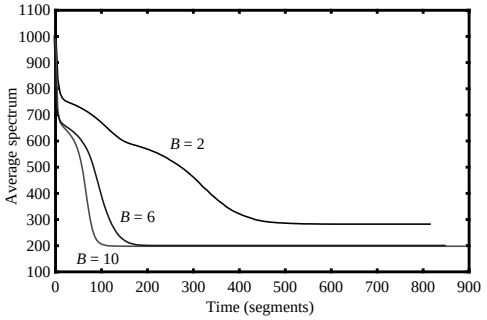

Figure 4. Different amounts of available retransmission bandwidth.

· We have to admit that the parameter choice is rather arbitrary. However, simulations with other values showed no significant impact on our results.

prefetching offset, missing segments from the beginning cannot be retransmitted and therefore a spectrum of 0 is not achieved.

3) Initial Transmission Quality

We performed a series of simulations where different initial transmission qualities were assumed resulting in cached videos where the maximum number of cached layers is lower than the maximum number of layers for the original video. In contrast to the preceding experiments, we did not sample the spectrum values but used a single simulation since the striking effects can be shown in more detail. For each simulation, the ratio between the maximum of cached layers (MCL) and the maximum of original layers (MOL) was modified.

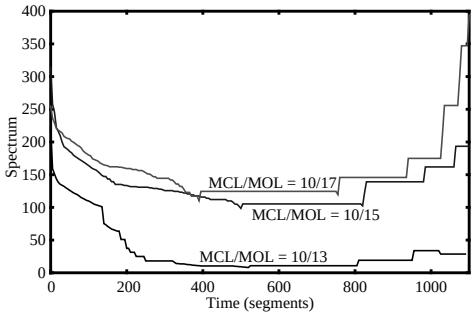

Figure 5. Influence of initial transmission quality.

As Figure 5 shows, spectrum values start to rise again for the last third of the video. This effect is especially pronounced for the worse initial transmission qualities. Looking at the cached video that results from the retransmission scheduling heuristic (U-SG-LLF) in Figure 6 sheds light on the reason for this effect. We observe that the retransmission scheduling "built a staircase" at the end of the cached video which, of course, is not good with respect to the minimization of the spectrum. The reason for this behavior is that the algorithm only considers missing segments ahead of the playout time (t_p + O_p). Thus, if all gaps are closed the algorithm starts to request segments from the next layer starting from t_p + O_p. This happens several times leading to the staircase shape exhibited in Figure 6. Further simulations can be found in [10]

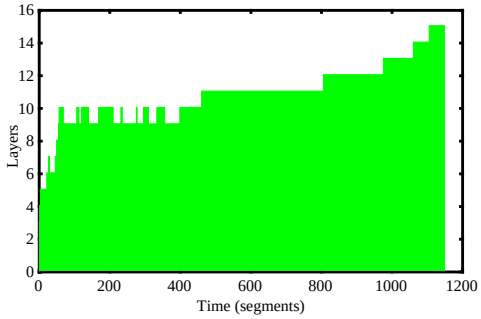

Figure 6. Cached video after retransmission phase.

VI. Conclusions and Future Work

Recent work has shown that layered encoded video is a technique that supports adaptive streaming well. High scalability for VoD in the Internet can be achieved by a distributed caching architecture. Our SAS (Scalable Adaptive Streaming) approach combines both caching and adaptive streaming and promises a scalable "Internet-conform" TVoD system. Our work, in this paper, is focused on the problem how to deal with retransmissions of missing segments for a cached layered video in order to meet users' demands to watch high quality video with relatively little quality variations. Inspired by [9], we developed and compared different retransmission scheduling algorithms from the general class of unrestricted priority-based heuristics to tackle this problem. Our simulation results indicate that this class has the potential to improve existing algorithms significantly.

VII. References

[1] C. Griwodz. *Wide-area True Video-on-Demand by a Decentralized Cache-based Distribution Infrastructure*. PhD thesis, Darmstadt University of Technology, Darmstadt, Germany, April 2000.

[2] J.-Y. Lee, T.-H. Kim, and S.-J. Ko. Motion Prediction Based on Temporal Layering for Layered Video Coding. In *Proceedings ITC-CSCC'98*, pages 245–248, July 1998.

[3] R. Gopalakrishnan, J. Griffioen, G. Hjalmtysson, C. Sreenan, and S. Wen. A Simple Loss Differentiation Approach to Layered Multicast. In *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies 2000, Tel-Aviv, Israel*, pages 461–469, March 2000.

[4] M. Zink, C. Griwodz, A. Jonas, and R. Steinmetz. LC-RTP (Loss Collection RTP): Reliability for Video Caching in the Internet. In *Proceedings of the Seventh International Conference on Parallel and Distributed Systems: Workshops*, pages 281–286, July 2000.

[5] R. Tewari, H. Vin, A. Dan, and D. Sitaram. Resource-Based Caching For Web Servers. In *Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking (MMCN), San Jose, USA*, January 1998.

[6] S. Sen, J. Rexford, and D. Towsley. Proxy Prefix Caxching for Multimedia Streams. In *Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies 1999, New York, NY, USA*, pages 1310–1319, March 1999.

[7] J. Kangasharju, F. Hartanto, M. Reisslein, and K. W. Ross. Distributing Layered Encoded Video through Caches. In *Proceedings of the Tenth Annual Joint Conference of the IEEE Computer and Communications Societies 2001, Anchorage, NY, USA*, April 2001.

[8] S. Nelakuditi, R. R. Harinath, E. Kusmierek, and Z.-L. Zhang. Providing Smoother Quality Layered Video Stream. In *Proceedings of the 10th International Workshop on Network and Operating System Support for Digital Audio and Video, Raleigh, NC, USA*, June 2000.

[9] R. Rejaie, H. Yu, M. Handley, and D. Estrin. Multimedia Proxy Caching for Quality Adaptive Streaming Applications in the Internet. In *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies 2000, Tel-Aviv, Israel*, pages 980–989, March 2000.

[10] M. Zink, J. Schmitt, and R. Steinmetz. Retransmission Scheduling in Layered Video Caches. Technical Report TR-KOM-2001-07, Darmstadt University of Technology, February 2001.