DEADLINE-AWARE SCHEDULING FOR WIRELESS VIDEO STREAMING

Günther Liebl

Institute for Communications Engineering
Munich University of Technology
liebl@tum.de

Mark Kalman, Bernd Girod

Information Systems Laboratory
Stanford University
{mkalman, bgirod}@stanford.edu

ABSTRACT

We present a new algorithm for deadline-aware scheduling of video streams over a wireless shared channel, which only requires the computation of a single metric per user and transmission slot. By incorporating side information about the video stream structure and the future channel behavior in the scheduling algorithm, our approach outperforms existing solutions by slowing down the transmission of streams to users with favorite channel conditions until their deadline is approaching. Hence, in overload situations, the quality of the bad users is significantly increased, while good users are almost unaffected. As a consequence, this leads to a fairer distribution of the achievable video quality among all users.

1. INTRODUCTION

In this paper we address the issue of streaming pre-encoded video from a server to multiple users in a wireless network. We assume that transcoding of the streams, as discussed in [1], is not possible. Hence, the media server is only able to adapt the rate for each user by *pruning* the video sequence (*ie* drop less important frames). At the media client, a decoder buffer compensates small variations in the received rate by adding a fixed *initial delay* to the playout process.

The reconstruction quality for each video sequence depends on the channel state of each user and the scheduling strategy at the air interface: Purely channel-dependent schedulers lead to unacceptable performance of users with low reception quality, since good users occupy the channel most of the time although they are well-ahead of their video decoding deadlines. Queue-dependent schedulers, however, tend to waste resources for data units which are already expired at the client. Finally, hybrid policies only perform well for larger values of the initial delay [2].

Significant performance improvements are expected from incorporating additional side information about the video stream structure and the future channel behavior in the scheduling algorithm. The idea is to slow down the transmission to users with favorite channel states until the deadline of their Head-of-Line (HOL) data units in the radio link buffer is approaching. An R-D optimal solution for this problem has been presented in [1], however, it amounts to comparing more than one metric per user and transmission slot. Therefore, we propose an alternative solution for systems that require scheduling decisions for a large number of users every few milliseconds (like HSDPA). Our approach relies on one single metric per user, and, though suboptimal with respect to full deadline-and-distortion–aware scheduling, already yields large gains vs. standard schedulers.

The remainder of the paper is organized as follows: Sec. 2 contains some preliminaries of a wireless video streaming scenario. In Sec. 3 we present our new algorithm for deadline-aware scheduling, followed by performance results in Sec. 4. Sec. 5 concludes the paper.

2. PRELIMINARIES

We consider the same end-to-end wireless video streaming scenario as in [2]: It consists of K users in the serving area of a base station in a mobile system, which have requested to stream multimedia data from a server on the backbone.

2.1. Video and Streaming Model

We assume that the server stores pre-encoded video streams of length N. The latter result from encoding *source units* s_n (*ie* video frames) with a video coder and mapping them one-to-one on *data units* $\mathcal{P}_n$ (*ie* packets). Note that other mappings are possible (*eg* by increasing the granularity of source units from complete frames to slices or by combining multiple frames within one packet). However, since our proposal can be modified in a straightforward way to include these aspects, we will not consider them here to keep our analysis as simple as possible. Each data unit bears a Decoding Time Stamp (DTS) $\mathrm{t}_{\mathrm{DTS},n}$, and the server starts transmitting the first one at time $\mathrm{t}_{\mathrm{s},1}$ and continues with the rest as follows: Compared to *Timestamp–Based Streaming* (**TBS**), it forwards data units even before their nominal sending time $\mathrm{t}_{\mathrm{s},n}$. This so-called *Ahead-of-Time Streaming* (**ATS**) assures that radio link buffers never underrun (note: overrun is still possible!) and all users are always considered by the scheduler (thus maximizing the statistical multiplexing gain from channel-aware scheduling).

A data unit $\mathcal{P}_n$ is completely received at the media client at $\mathrm{t}_{\mathrm{r},n}$, and the interval $\delta_n \triangleq \mathrm{t}_{\mathrm{r},n} - \mathrm{t}_{\mathrm{s},n}$ is called the channel delay (note: loss on the transmission path can be modeled as $\delta_n = \infty$). $\mathcal{P}_n$ is then kept in the client buffer until it is forwarded to the video decoder at decoding time $\mathrm{t}_{\mathrm{d},n}$. We assume that $\mathrm{t}_{\mathrm{DTS},1} = \mathrm{t}_{\mathrm{d},1}$ and define the *initial delay* as

0-7803-9332-5/05/$20.00 ©2005 IEEE

$\delta_{\text{init}} \triangleq t_{d,1} - t_{s,1}$. Hence, data units with $\delta_n > \delta_{\text{init}}$ are no more useful to the decoder (so-called *late-loss*). In addition, the temporal dependencies among them (which result from the use of hybrid video codecs and can be represented via directed acyclic graphs [3]) have to be satisfied: Only those data units whose ancestors have all been received in time can be decoded. For any lost data unit, the corresponding source unit is *concealed* by the timely–nearest reconstructed source unit. Now let the individual quality (*eg* Peak Signal-to-Noise–Ratio (PSNR)) of s_n be Q_n and the concealment quality, if s_n is represented with s_i, be $\tilde{Q}_n(i)$. The *importance* of data unit $\mathcal{P}_n$ reflects the increase in quality at the receiver if s_n is correctly decoded [2], *ie*

$$I_n \triangleq \frac{1}{N}\left(Q_n - \tilde{Q}_n(c(n)) + \sum_{\substack{i=n+1 \\ n \vdash i}}^{N} \left[\tilde{Q}_i(n) - \tilde{Q}_i(c(n))\right]\right). \tag{1}$$

Here, $c(n)$ is the number of the concealing source unit for s_n, and $n \vdash i$ indicates that i depends on n. The single-user quality for a set of channel delays $\delta = \{\delta_1, \ldots, \delta_N\}$ is then ($\mathbf{1}\{A\}$ equals 1 if A is true and 0 otherwise)

$$Q(\delta, \delta_{\text{init}}) = Q_0 + \sum_{n=1}^{N} I_n \mathbf{1}\{\delta_n \le \delta_{\text{init}}\} \prod_{\substack{m=1 \\ m \prec n}}^{n-1} \mathbf{1}\{\delta_m \le \delta_{\text{init}}\}, \tag{2}$$

with Q_0 the quality if all frames are presented as grey.

2.2. Wireless System Model

We assume that the core-network is over-provisioned such that congestion on the backbone is not an issue. The streaming server forwards the packets directly into the *radio link buffers* of the base station, where packets are kept until they are transmitted over the shared wireless link to the media clients. For each *transmission slot* τ of fixed duration T_{slot} a scheduler assigns channel access to one single user. We further assume that the current channel state of user k (*ie* the average *signal-to-noise-and-interference–ratio* $\text{SNIR}_k(\tau)$) is known to the base station at the start of each slot and that the modulation and coding format can be adjusted such that transmission at channel capacity is employed. Hence, user k may transmit a transport block of at most

$$l_{\text{TB},k}(\tau) = \lfloor B_{\text{sys}} \cdot \log_2(1 + \text{SNIR}_k(\tau)) \rfloor \tag{3}$$

bytes reliably, where B_{sys} is a system-dependent constant. In addition, the base station keeps a history of previous channel states and total transmitted data volume of each user over a certain time window, which can be used for computing relative scheduling metrics or dynamic estimation of channel state distributions.

If the radio link buffers are not emptied fast enough because the channel quality of some users is currently poor, the system is in *overload*. In [2] it has been found that for streaming users it is beneficial to keep the buffer size N_{RL} finite and drop data units already at the radio link buffers to reduce the excess load and convert late-loss at the media client into controlled packet removals, thus achieving an in-time delivery of a temporally scaled version of the video stream. The best drop strategy which requires only a minimum amount of side information (*ie* the Group-of-Pictures (GOP) structure and its relation to a simple packet priority scheme) is to remove data units from the HOL group in the buffer such that the temporal dependencies present in the video streams are not violated (**Drop Dependency Based – DDB**) [2]. The average reconstruction quality over all users,

$$Q(K, \delta_{\text{init}}) = \frac{1}{K}\sum_{k=1}^{K} Q(\delta_k, \delta_{\text{init}}), \tag{4}$$

strongly depends on the ability of the scheduling strategy to cope with both varying media and channel characteristics.

2.3. Media-Aware System Improvements

Performance can be improved by reducing the set of data units transmitted to each user such that a long-term supportable target bit rate can be maintained. This procedure, called *pruning*, means removing those data units already at the server which contribute least to Eq. 2. The R-D optimal transmit pattern for a sufficiently large number of rate-distortion-quality triples (R_u, D_u, Q_u) can be determined offline via a Markov decision process and used for finding a convex closed-form description $\tilde{D}(r)$ via simple curve-fitting. Based on long-term averaging of the supported net data rate $\overline{r}_{S,k}$ of each user, the optimal rate share is

$$\underline{\chi}^* = (\chi_1^*, \ldots, \chi_K^*) = \arg\min_{\underline{\chi}} \frac{1}{K}\sum_{k=1}^{K} \tilde{D}(\chi_k \cdot \overline{r}_{S,k}), \tag{5}$$

where $\chi_k \in [0, 1]$, $\sum_{k=1}^{K} \chi_k \le 1$. The transmit pattern for user k is then chosen with respect to the rate point R_u closest to $\overline{r}_{\text{opt},k} = \chi_k^* \cdot \overline{r}_{S,k}$. Since this optimization is based on long-term average channel quality, a more sophisticated scheduler at the slot level is still required to compensate for short-term variations. An R-D optimal solution, as described in [1], would operate as follows: For each transmission slot, the combined buffer management and scheduling algorithm computes the approximate expected distortion (or quality) for each user and a set of possible options. These include whether the user is scheduled or not and all possible dependency-aware omit patterns for each radio link buffer. The computation is based on the users' channel state distribution, as well as on assuming that each user will only have access to part of the future slots according to Eq. 5. After having determined the best possible omit pattern for being scheduled or not, the scheduler selects the user which yields minimum average distortion (or maximum average quality) over all users, if scheduled. Besides the required additional side information (*ie* channel state distribution, distortion/importance impact and deadline of each data unit) the main drawback of this solution is the relatively high

complexity, if applied in systems with slot intervals of few milliseconds. Hence, we have developed a suboptimal algorithm, which still relies on the above side information, but does not include the search over all possible omit patterns.

3. PROPOSED SCHEDULER

The main idea behind our proposed scheduling algorithm is to consider not only how much it costs a user on average (*eg* in throughput), if he does not get the current slot, but how likely it is then to violate future deadlines (and thus dependencies) of data units.

3.1. Cost Function

Let $\underline{\psi} = (\psi_1, \ldots, \psi_K)$ be a transmission schedule for a specific slot, where $\psi_k \in \{0, 1\}$ and $\sum_{k=1}^{K} \psi_k = 1$ (*ie* only one of the K users is scheduled at the same time). The optimal schedule with respect to average expected cost (increase in distortion or decrease in quality) is

$$\begin{aligned} \underline{\psi}^* &= \arg\min_{\underline{\psi}} C(\underline{\psi}) = \arg\min_{\underline{\psi}} \frac{1}{K} \sum_{k=1}^{K} C_k(\psi_k) \\ &= \arg\min_{\underline{\psi}} \frac{1}{K} \sum_{k=1}^{K} [C_{k,1} \cdot P_k(\psi_k) \\ &\quad + C_{k,2} \cdot (1 - \psi_k) \cdot (1 - P_k(\psi_k))] , \quad (6) \end{aligned}$$

where $C_{k,1}$ denotes the cost, if the deadline of the HOL packet of user k is violated (*immediate constraint*), and $C_{k,2}$ the cost, if no deadline violation happens, but the user is not scheduled and thus an *average constraint* with respect to the whole video sequence will not be met. Finally, $P_k(\psi_k)$ is the probability of future deadline violation of the HOL packet under ψ_k and reflects the *elasticity budget* still left for a given expected channel behavior and future rate share of user k. Letting $P_k(\psi_k) = P_k(0) \cdot (1 - \psi_k) + P_k(1) \cdot \psi_k$, Eq. 6 can be rewritten as

$$\underline{\psi}^* = \arg\min_{\underline{\psi}} \left[- \sum_{k=1}^{K} a_k \cdot \psi_k \right] = \arg\max_{k} a_k , \quad (7)$$

where

$$a_k = C_{k,1} \cdot [P_k(0) - P_k(1)] + C_{k,2} \cdot [1 - P_k(0)] . \quad (8)$$

Hence, our new *Minimum Deadline Related Cost* (**MinDRC**) scheduler amounts to a simple metric computation for each user and then selecting the best one for transmission in each slot. In addition, DDB management of the radio link buffers with a straightforward enhancement is employed: All pending data units with already expired deadlines are removed from the radio link buffer, as well as all of their pending dependants. Thus, instead of combined buffer management and scheduling as mentioned above, our suboptimal approach relies on light coupling between these two entities via the structure of the media stream.

3.2. Metric Computation

Computation of the metric contributions at the beginning of a slot (*ie* at time t_{now}) can be done as follows: The immediate constraint related cost of user k is the sum of the importance of the HOL packet n_k and its dependants, *ie*

$$C_{k,1} = I_{k,n_k} + \sum_{\substack{i=n_k+1 \\ i \succ n_k}}^{n_k+N_{RL}-1} I_{k,i} . \quad (9)$$

In order to determine the average constraint related cost, the average net data rates $\hat{r}_{S,k}(\psi_k)$ up to time $t_{now} + T_{slot}$ are evaluated for the case of scheduling the user in the slot or not (using the stored record of the transmitted data volume in the past and the current channel state). For both rate values, the closest $(R_{u_k}, D_{u_k}, Q_{u_k})$ triple is found and $C_{k,2}$ results from linear interpolation as

$$\begin{aligned} C_{k,2} &= \frac{Q_{u_k}(1) - Q_{u_k}(0)}{R_{u_k}(1) - R_{u_k}(0)} \cdot [R_{u_k}(1) - \hat{r}_{S,k}(0)] \\ &\quad \cdot \mathbf{1}\{\hat{r}_{S,k}(0) < \overline{r}_{opt,k}\} . \quad (10) \end{aligned}$$

For the elasticity budget we need the number of future slots until the deadline t_{DL,k,n_k} of the HOL data unit, *ie*

$$Z_{DL,k} = \lfloor \chi_k^* \cdot \frac{t_{DL,k,n_k} - t_{now} - T_{slot}}{T_{slot}} \rfloor , \quad (11)$$

and the distribution of the supported transport block size over multiple slots (which can be dynamically re-estimated from the recorded channel state history). The deadline violation probability for the remaining load $l_{rem,k}(\psi_k)$ of the HOL data unit is then

$$P_k(\psi_k) = \Pr\left(\sum_{z=1}^{Z_{DL,k}} L_{TB,k,z} \le l_{rem,k}(\psi_k) \right) . \quad (12)$$

4. PERFORMANCE RESULTS

We have used a downlink scenario similar to the HSDPA setup in [2] with $K = 10$ users attached to a base station, but with a simplified segmentation procedure at the data link layer and the channel model according to Eq. 3. However, the slot-wise SNIR traces of the users are the same and have been further processed to yield the statistics needed in our algorithm. Each of the users has requested a streaming service for the same H.264/AVC–coded QCIF sequence of length $N = 2698$ frames as in [2], with $QP = 28$, 30 fps, and no rate control. The GOP structure is IBBPBBP..., with an I–frame distance of 1 s. The overall Y-PSNR is 36.98 dB, and the average bit–rate is 178.5 kbit/s. The simulation length is 89.934 s, *ie* equivalent to the duration of the video sequence. Although the latter is looped endlessly, only the first loop contributes to Eq. 2 and 4 (to penalize too fast retrieval). We have evaluated the performance for varying initial delay, a fixed radio link buffer size of $N_{RL} = 30$

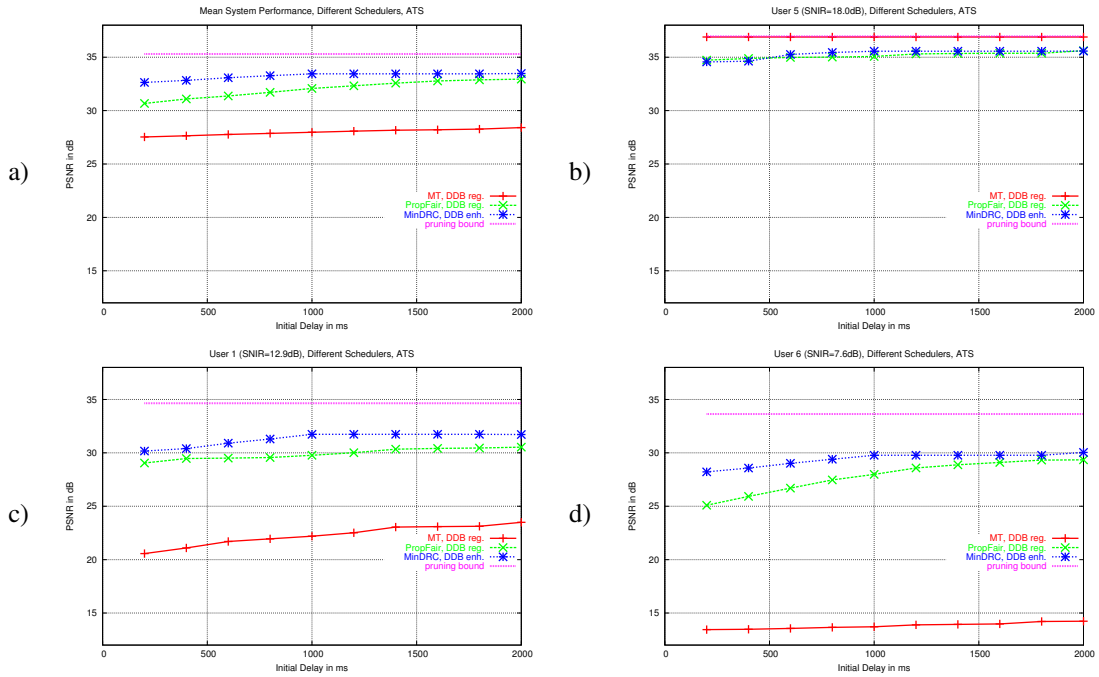

Fig. 1. Average (a) and user-specific PSNR (b,c,d) vs. initial delay for different scheduler strategies.

data units, and $B_{sys} = 0.85 * 900$ bytes. In detail, we have compared our new MinDRC scheduler with the enhanced DDB buffer management to a *Maximum Throughput* (**MT**) and a *Proportional–Fair* (**PropFair**) scheduler, both with regular DDB. While MT always selects the user with the currently highest throughput per slot, PropFair selects the one with the currently highest ratio of slot to average throughput [4]. From Fig. 1a we can observe that in terms of average PSNR over all users PropFair clearly outperforms MT, which yields unacceptable quality over the whole range of initial delay values. The reason for this behavior can be found in Fig. 1b-d, which contain the PSNR for the user with the best, intermediate, and worst average channel quality over the simulation duration: While MT strictly favors the best users and suppresses all others by a large amount, the PropFair algorithm achieves a fairer distribution of the achievable PSNR among users at the expense of only slightly deteriorating the best ones. The preferred choice, however, is our new MinDRC scheduler, which yields a substantial gain over PropFair for both the intermediate and worst user in Fig. 1c,d, while the performance of the best user is about the same. Hence, slowing down the best users until their deadline is approaching increases fairness even more and also leads to better average PSNR as depicted in Fig. 1a. Finally, the horizontal straight lines in all four figures represent the overall PSNR of the input streams after pruning according to Eq. 5. Since MinDRC is still a suboptimal solution for deadline-aware scheduling, it does not achieve this bound. However, note that Eq. 5 does not consider the actual distribution of the supported net data rate and is certainly too optimistic especially for bad users.

5. CONCLUSIONS

We have presented a new algorithm for deadline-aware scheduling of video streams over a wireless shared channel, which only requires the computation of a single metric per user and transmission time slot. Hence, we consider it suitable for implementation in future wireless systems that require scheduling on a fast timescale of few milliseconds. Besides the information about the video stream structure, which needs some implicit or explicit signaling by a media-aware gateway, the only other required information is the channel state distribution, which is available in most of today's systems. Future research concerns extension of the metric computation to practical channel models (including retransmission) and improvements on the pruning part.

6. REFERENCES

[1] M. Kalman and B. Girod, "Optimized transcoding rate selection and packet scheduling for transmitting multiple video streams over a shared channel," in *Proc. ICIP 2005*, Genua, Italy, Sept. 2005.

[2] G. Liebl, H. Jenkac, T. Stockhammer, and C. Buchner, "Joint buffer management and scheduling for wireless video streaming," in *Proc. ICN 2005*, La Reunion, Apr. 2005, pp. 882–891.

[3] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," Tech. Rep. MSR-TR-2001-35, Microsoft Research, Feb. 2001.

[4] S. Shakkottai and A. L. Stolyar, "Scheduling algorithms for a mixture of real-time and non-real-time data in HDR," in *Proc. ITC-17*, Salvador, Brazil, Sept. 2001.