**IEEE International Conference on Multimedia Computing & Systems, June 17-23, 1996, in Hiroshima, Japan, p. 151-155**

# Constant Data Length Retrieval for Video Servers with Variable Bit Rate Streams

Ernst Biersack, Frédéric Thiesse, Christoph Bernhardt
Institut Eurécom
2229 Route des Crêtes, 06904 Sophia Antipolis, FRANCE
{erbi,bernhard}@eurecom.fr

## Abstract

*We define a novel retrieval algorithm for variable bit rate video streams called GCDL that reads the video data from the disk as constant size data blocks. We formulate for GCDL a deterministic admission control criterion and evaluate its performance. Compared to existing retrieval algorithms, GCDL decreases the buffer requirement and start-up latency while admitting the same number of clients.*

## 1. Introduction

Video servers store digitized, compressed continuous media information on high-capacity secondary or tertiary storage [6]. The secondary storage devices allow random accessible and provide short seek times compared to tertiary storage. Video server design differs significantly from that of traditional data storage servers due to the large size of the objects stored and the real-time requirements for their retrieval. The critical resources in a video server are disk bandwidth, storage volume, and main memory. Given a fixed amount of these resources, a video server can only deliver a limited number of video streams simultaneously. Before admitting a new client, a video server must use an admission control algorithm to check if there are enough resources for serving the additional client.

We identify and formalize schemes for the **retrieval** of *variable bit rate* video data from magnetic disks. Traditional methods, such as the cyclic retrieval of variable size data segments or the retrieval at the stream's mean bit rate, either cannot profit from smoothing media traffic over larger intervals, or suffer from excessive buffer demand and latency. We derive a novel technique that covers existing methods as special cases. While offering a deterministic service, the novel scheme can drastically decrease the buffer requirement and server latency in an interactive multimedia service.

## 2. Deterministic Retrieval Schemes in Video Servers

A video server must meet the requirements that stem from the continuous nature of audio and video and must guarantee the delivery of continuous media data in a timely fashion. We assume that video information is encoded as a **variable bit rate stream** (**VBR**)[1] of *constant* quality. VBR requires sophisticated resource reservation mechanisms for the server and network to achieve a good utilization of the resources while maintaining a constant quality playback.

### 2.1 Deterministic Constraint Function for VBR Video

To provide deterministic quality of service (QOS) for VBR video, the admission control must employ *worst-case assumptions* about the data rate of the VBR video when computing the number of streams to be admitted. To offer deterministic service, we use a traffic model that is deterministic. The so-called empirical envelope presented in [5] provides a deterministic traffic constraint function for a given video trace. If $A_i[t, t+\tau]$ denotes the amount of video data consumed by a stream $s_i$ in the interval $[t, t+\tau]$, an upper bound on $A_i$ can be given by the **empirical envelope function** $\varepsilon_i(\tau)$ that is defined as:

$$\varepsilon_i(\tau) = \max_t A_i[t, t+\tau], \forall t \in [0, T_{total} - \tau]$$

### 2.2 Round-Based Retrieval Schemes

In the simplest case, continuous playback can be ensured by buffering the entire stream prior to initiating the playback [3]. Such a scheme, however, requires very large buffer space and causes a very large start-up latency. Consequently, the problem of efficiently servicing a single stream becomes one of preventing buffer starvation while

[1] Our model is able to accommodate CBR as a special case.

at the same time minimizing the buffer requirement and the start-up latency. In the most general sense, the buffer requirement in a video server at time $t$ can be stated as the difference between the cumulative arrival function $a(t)$ of the video data read from secondary storage, and the cumulative consumption function $c(t)$ denoting the video data sent to clients.[2] The difference is referred to as **backlog** function [4].

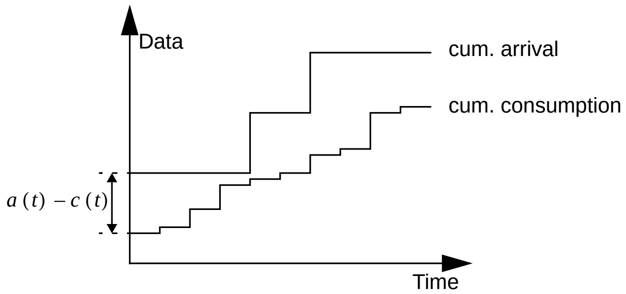


**Figure 1. Backlog function: cumulative arrival – cumulative consumption**

We say that **buffer starvation** at time $t$ occurs if $a(t) - c(t) < 0$. If $b_{total}$ denotes the total amount of available buffer in a video server, then $a(t) - c(t) > b_{total}$ will cause **buffer overflow**.

In order to avoid buffer starvation or buffer overflow, almost all approaches to multi-stream continuous media retrieval have the following characteristics [3]:

1. Processing stream requests in cyclic rounds.
2. Arrival keeps up with consumption.

A video server that operates in rounds generally avoids starvation by *reading ahead* an amount of data that lasts in terms of playback duration through the next round (see figure 2). Data retrieval techniques determine the way data is read from the disk during a service round.

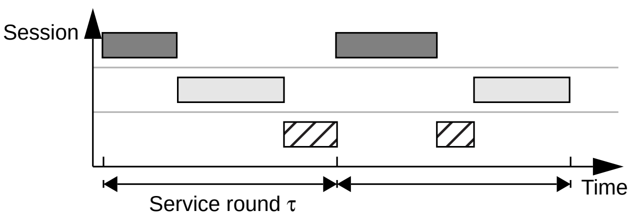


**Figure 2. Sequence of service rounds**

The admission control scheme considered in this paper allows VCR functions such as fast forward, reverse, or pause) under the condition that the data rate required to support these functions is *not higher* than the data rate for normal playback.

Using VBR as data model for a video, one can map video data onto **data blocks (segments)** stored on the disk in two ways: **constant time length** (**CTL**) and **constant data length** (**CDL**) [1]:

- CTL retrieval is characterized as having *variable length* data blocks with *constant playback* duration $\tau$ for stream $s_i$ (see figure 3). During any service round of duration $\tau$, $\tau \cdot r_i$ frames are retrieved from secondary storage, where $r_i$ denotes the constant frame rate of stream $s_i$. Since successive frames of a VBR video differ in size, CTL retrieval results in a periodic but volume-variant retrieval.

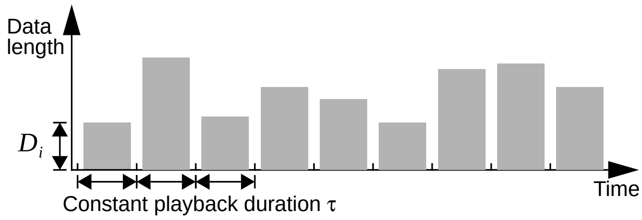


**Figure 3. Constant time length retrieval**

- If we simply inverse the two properties concerning periodicity and data block size, we get constant data length (CDL) retrieval that combines *non-periodic* retrieval with *constant* amounts of data from the disk (see figure 4). At first sight, CDL might seem incompatible with *round-based* disk retrieval, but if we introduce the restriction that the distances between retrieval operations must be multiples of $\tau$, we get sequences of what we call **active** and **idle** rounds. During an active round, a *constant* size data block is read from the disk. Since the data must always be sufficient to supply the client during the following round even in the worst case, the (fixed) size of the data block retrieved is $\varepsilon_i(\tau)$. During an idle round, no data at all is retrieved. The decision, whether a round will be active or not, can be made online. If there is still enough data in the buffer for the current and the next round, the current round is idle, otherwise it must be active. Since an active round is never necessary if the current buffer level is equal or greater than $\varepsilon_i(\tau)$, we can also give a first bound on the buffer requirement by $2 \cdot \varepsilon_i(\tau)$.

Each of the two retrieval strategies, CTL and CDL, has advantages and disadvantages. Given the real-time requirements of continuous media and the periodic nature of video playback, CTL retrieval appears to be the more natural approach. It can easily be implemented because media quanta are always handled in terms of frames. A sequence of frames that must be sent to a client can therefore easily be mapped to disk I/O requests.

[2] The functions $a(t)$ and $c(t)$ can be alternatively stated in terms of frames or in terms of media data. If stated in terms of frames, the deterministic buffer requirement in terms of data is then determined by the relation between a number of frames and their respective maximum data size.

When using CDL retrieval, the amount of data retrieved at a time does not vary and can correspond in size to a disk block, allowing for efficient disk layout.

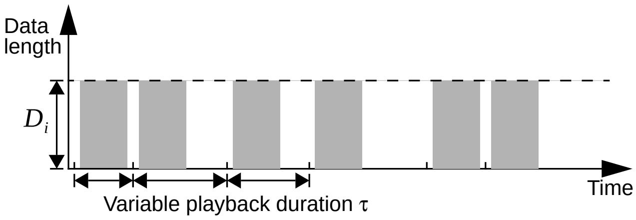


**Figure 4. Constant data length retrieval**

# 3. Generalized CDL

## 3.1 Introduction

Up to now, all papers on periodic retrieval schemes have assumed that

- the **disk service round**, during which data for each stream are read exactly once from disk, and
- the **CDL round,** for which we consider the worst case data consumption given by $\varepsilon_i(\tau)$

have the *same length*.

We will distinguish the two and propose to make the CDL round a *multiple* of a disk service round (The same generalization can be applied to CTL, as shown in [2]):

- The disk scheduling and retrieval still proceeds in rounds of length $\tau$.
- However, we use a set $T = \{\tau_1, \ldots, \tau_n\}$ of CDL rounds with $\tau_i \geq \tau$. To avoid starvation, we require that the amount of data retrieved for stream $s_i$ from the disk during each interval $\tau_i$ must last for a period of $\tau_i$. The CDL round duration $\tau_i$ is an integer multiple $m_i$ of the disk service round duration $\tau$. (see figure 5).When $\varepsilon_i(\tau_i)$ is the deterministic upper bound on the amount of data retrieved for stream $s_i$ during any period $\tau_i$, we require that the amount of data retrieved during any of the $m_i = \tau_i / \tau$ disk service rounds is the same, namely $\varepsilon_i(\tau_i) / m_i$. Note that during any disk service round (of duration $\tau \leq \tau_i$) within the CDL round, *fewer* frames may be read from the disk than are consumed by the client.

In the following, we will refer to the CDL retrieval where CDL rounds and disk rounds have the same length ($\tau = \tau_i$) as **traditional** CDL. When CDL rounds and disk rounds have different length ($\tau \neq \tau_i$), the scheme is referred to as **Generalized CDL (GCDL)** retrieval. The traditional CDL can be regarded as a special case of GCDL with $m_i = 1$.

In GCDL, for a stream $s_i$ there will be active CDL rounds during which a fixed amount $\varepsilon_i(\tau_i)$ of data is read and idle CDL rounds when no data is read.

We will see that the separation of disk service round and CDL round helps smooth the VBR traffic and allows to significantly reduce the buffer demand and the start-up latency while admitting the same number of clients.

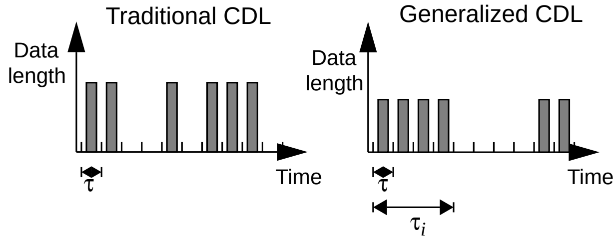


**Figure 5. Traditional and generalized CDL**

## 3.2 Start-up Latency and Buffer Requirement

The **start-up latency** is defined as the delay between user interaction and feedback by the server. Note that the delay introduced by the network and by buffering at the client site, for instance in order to synchronize several streams, is not being considered. Thus, the start-up-latency of a video server is given by the time that passes by from the reception of a playback request until the time the first frame is submitted to network.

Generally, a request of a new client must wait until the beginning of the next disk service round before it can be processed. In the worst case this takes a time period of $\tau$. Data consumption must be delayed another period $\hat{t}_i$ until enough data has arrived at the client to guarantee that buffer starvation is avoided during the playback of the video. Therefore, the total start-up-latency is given by $\tau_i + \hat{t}_i$.

For the general case $\tau_i = \tau \cdot m_i$ with $m_i > 1$, data will arrive at the client $m_i$ times during the first CDL round, namely at the end of each disk service round. Therefore, it is possible to start the playback earlier than for traditional CDL.

To avoid buffer starvation, the backlog function $b_i(t) = a_i(t) - c_i(t - \hat{t}_i)$ must be equal or greater than 0 at all times. The determination of the optimal start-up latency $\hat{t}_i$ can therefore be described as a linear minimization problem:

$$\hat{t}_i \rightarrow min \qquad , \hat{t}_i \in [\tau, \tau_i]$$

$$b_i(t) \geq 0 \qquad , t \in [0, T_{total}]$$

This optimization problem can be solved by a simple search algorithm that simulates all possible CDL rounds for different delays $\hat{t}_i$. The details and possible optimizations of the algorithm that determines $\hat{t}_i$ are omitted here.

The deterministic **buffer requirement** is closely related to the playout delay $\hat{t}_i$. We assume that the server under consideration retrieves data for multiple streams in a round-robin schedule. The order of the streams within a round is determined by a SCAN algorithm [7], so the data can arrive at the buffer at any time during a service round. The worst case with regard to the start-up latency is given by assuming the latest possible arrival of media data while the worst case concerning the buffer assumes that all media data to arrive as early as possible.

Because of the delay $\hat{t}_i$ between the arrival and the consumption of frames, at the end of any CDL round at least $\lceil r_i \hat{t}_i \rceil$ frames, which have not been consumed yet, remain in the buffer. In the worst case, a CDL round becomes active although there are almost enough frames in the buffer at the beginning to mark it idle. The maximum buffer level at the beginning of an active round is therefore given by:

$$\varepsilon_i\left(\frac{\lceil r_i(\tau_i + \hat{t}_i)\rceil}{r_i}\right)$$

Suppose that throughout this active round the consumed frames are very small, for instance they all equal the minimum frame size, while the arriving frames are very large. This clearly marks the worst case because the maximum amount of data arrives while the number of buffered frames decreases the least. Throughout the CDL round no media data are consumed while $\varepsilon_i(\tau_i)$ media data arrive from the storage device. An upper bound to the deterministic buffer requirement can be given by:

$$\varepsilon_i(\tau_i) + \varepsilon_i\left(\frac{\lceil r_i(\tau_i + \hat{t}_i)\rceil}{r_i}\right)$$

This estimation can be improved, since the pessimistic assumption that no frames are consumed during a round is no longer acceptable for higher values for $\tau_i$. We can use the function

$$\xi_i(\tau) = \min_t A_i[t, t+\tau], \forall t \in [0, T_{total} - \tau]$$

that characterizes the *minimum* data consumption in an interval $\tau$ to determine a better estimation that is valid for all $\tau_i$. Suppose that $\xi_i(k\tau)$ states the minimum amount of media data that are consumed during any $k$ subsequent disk service rounds. A better bound for the deterministic buffer requirement that also takes into account the consumed media data is given by:

$$\max_{k \in \{0, \ldots, m_i - 1\}} \left\{ (k+1)\,\varepsilon_i\left(\frac{\tau_i}{m_i}\right) - \xi_i(k\tau) + \varepsilon_i\left(\frac{\lceil r_i(\tau_i + \hat{t}_i)\rceil}{r_i}\right) \right\}$$

## 3.3 Deterministic Admission Control

The number of streams admitted is limited by the length of a disk service round, the available buffer space and the disk bandwidth. The admission control criterion for GCDL (and also for GCTL) is given by:

$$\sum_{i=1}^{n} \left\lceil \frac{\varepsilon_i(\tau_i)}{m_i} \right\rceil \cdot r_{disk}^{-1} + \sum_{i=1}^{n} \left\lceil \left\lceil \frac{\varepsilon_i(\tau_i)}{m_i} \right\rceil \cdot c_{cyl}^{-1} \right\rceil \cdot t_{track} + n \cdot (t_{track} + t_{rot}) + t_{seek} \le \tau$$

In this formula $r_{disk}$ denotes the disk bandwidth, $c_{cyl}$ equals the capacity of a single cylinder and $t_{track}$, $t_{rot}$ and $t_{seek}$ denote the track-to-track seek time, the rotational latency and the maximum seek time for a complete scan over the entire disk. $m_i$ denotes the number of disk service rounds within a single CDL or CTL round.

The effect of the choice $\tau_i$ and $\tau$ on the number of admitted streams is demonstrated in figure 6. The disk service round duration is varied between 0 s and 3 s while the parameter $m_i$ takes integer values between 1 and 10. A CTL/CDL round can therefore be up to 30 s long. It is important to note, that we get the same results for CTL as for CDL in the deterministic case because both use the same function $\varepsilon_i(\tau)$ to characterize the maximum amount of data that must be retrieved in one round. Obviously, the smallest number of streams can be admitted if both parameters are chosen small.

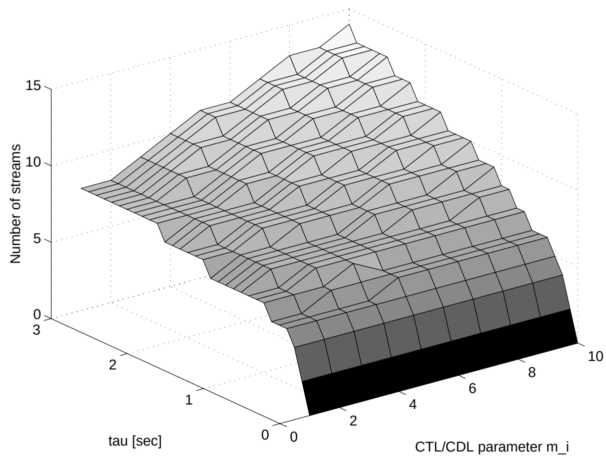


**Figure 6. Effect of $\tau$ and $\tau_i$ on the number of streams (for MTV trace)**

There are two noteworthy effects:

- The left edge of the graph indicates the curve for traditional CTL/CDL with $\tau = \tau_i$. The maximum number of admitted streams rises with growing $\tau$ for two reasons: As $\tau$ grows toward the duration of a video $T_{total}$, the amount of data $\varepsilon_i(\tau)$ that must be retrieved in the worst case tends to get characterized by the mean bit rate instead of the peak bit rate. The disk I/O efficiency improves due to a decrease in seek overhead when the media data are retrieved in larger blocks.
- All other values depict results for generalized CTL/CDL. As $\tau_i$ increases, for a fixed $\tau$, more streams can be admitted since $\varepsilon_i(\tau_i)/\tau_i$ decreases.

Table 1 gives a summary of the performance of CDL with various combinations of $\tau$ and $\tau_i$ and illustrates the advantages of generalized CDL over traditional CDL. If we take, for instance, the case where 9 clients can be admitted we see that GCDL reduces the buffer requirement per stream by more than 1 MByte and the start-up latency by more than 50%.

| $\tau$ | $\tau_i$ | Start-up latency | | Buffer requirement for one stream | | Number streams admitted | |
|---|---|---|---|---|---|---|---|
| | | Sec | % | Bytes | % | | % |
| 1 | 1 | 2.0 | 100% | 1,073,856 | 100% | 6 | 100% |
| 1 | 2 | 2.2 | 110% | 1,425,901 | 133% | 7 | 116% |
| 1 | 3 | 2.2 | 110% | 1,757,614 | 164% | 7 | 116% |
| 1 | 4 | 2.2 | 110% | 2,081,280 | 194% | 7 | 116% |
| 1 | 6 | 2.2 | 110% | 2,701,920 | 252% | 8 | 133% |
| 2 | 4 | 4.1 | 205% | 2,669,929 | 248% | 8 | 133% |
| 2 | 6 | 4.2 | 210% | 3,312,919 | 308% | 8 | 133% |
| 2 | 8 | 4.6 | 230% | 3,833,602 | 357% | 9 | 150% |
| 1 | 1 | 2.0 | 100% | 1,073,856 | 100% | 6 | 100% |
| 2 | 2 | 4.0 | 200% | 2,001,711 | 186% | 8 | 133% |
| 3 | 3 | 6.0 | 300% | 2,959,519 | 276% | 8 | 133% |
| 4 | 4 | 8.0 | 400% | 3,664,247 | 341% | 8 | 133% |
| 5 | 5 | 10.0 | 500% | 4,340,219 | 404% | 8 | 133% |
| 6 | 6 | 12.0 | 600% | 5,022,548 | 468% | 9 | 150% |

**Table 1. Comparison of traditional and general CDL (for MTV trace)**

## 4. Conclusion

We have proposed and evaluated a novel CDL scheme where the length of the CDL round is *decoupled* from length of the disk round. This permits to individually adapt the duration of the CDL round *for each stream* to the buffer and start-up latency constraints at the client, while maintaining a *common* disk round length.

The separation of the disk service round from the CDL round pays off in terms of buffer space savings and reduced start-up latency.

Comparing GCDL with traditional CDL, we observe that

- GCDL drastically reduces the buffer requirements and start-up latencies
- GCDL admits the same number of streams.

By choosing the right values for the disk service round $\tau$ and the CDL $\tau_i$, one can trade-off between disk utilization, start-up latencies, and buffer space.

We also saw that CDL and CTL use the same admission control criteria and therefore admit the same number of clients. This is also true for traditional and GCDL and GCTL.

**Acknowledgment.** The work described in this paper was supported by the Siemens Nixdorf AG, Munich. We thank Johannes Dengler for his contribution to the definition of GCDL.

## 5. References

[1] E. Chang and A. Zakhor. Admission control and data placement for VBR video servers. In *Submitted to Proceedings of the 1st International Conference on Image Processing*, Austin, Texas, November 1994.

[2] J. Dengler, C. Bernhardt, and E. Biersack. Deterministic admission control strategies in video servers with variable bit rate streams. In B. Butscher, E. Moeller, and H. Pusch, editors, *Interactive Distributed Multimedia Systems and Services, European Workshop IDMS'96, Berlin, Germany*, volume 1045 of *LNCS*, pages 245–264. Springer Verlag, Heidelberg, Germany, Mar. 1996.

[3] D. J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe. Multimedia storage servers: A tutorial and survey. *IEEE Computer*, 28(5):40–49, May 1995.

[4] E. W. Knightly, R. F. Mines, and H. Zhang. Deterministic characterization and network utilizations for several distributed real-time applications. In *Proc. of IEEE WORDS'94*, Dana Point, CA, Oct. 1994.

[5] E. W. Knightly, D. E. Wrege, J. Liebeherr, and H. Zhang. Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic. In *Proceedings Sigmetrics '95 / Performance '95*, volume 23 of *Performance Evalution Review*, pages 98–107, Ottawa, Canada, May 15-19 1995.

[6] L. A. Rowe and R. R. Larson. A video-on-demand system. The University of California at Berkeley and International Computer Science Institute, 1993.

[7] R. Steinmetz and K. Nahrstedt. *Multimedia: Computing, Communications and Applications*. Innovative Technology Series. Prentice Hall, Englewood Cliffs, NJ, 1995.