Cost-optimal Data Retrieval for Video Servers with Variable Bit Rate Video Streams

Ernst W. Biersack
Institut Eurecom
06904 Sophia Antipolis, FRANCE
Email: erbi@eurecom.fr

Maher Hamdi
Institute for computer Communications and Applications (ICA)
Swiss Federal Institute of Technology (EPFL), Switzerland
Email: Maher.Hamdi@epfl.ch

Abstract

We consider the problem of data retrieval from disk storage in a video server where the data is stored in constant size blocks. The two critical resources of a video server are the amount of disk I/O bandwidth and the main memory buffer required to support a video stream. We propose a new algorithm that allows us to determine the cost-optimal trade-off between these two server resources as well as bounds and statistics of the start-up latency seen by the user.

Keywords – Video Server, disk storage, continuous media, VBR video, retrieval.

1 Introduction

Video servers store digitized, compressed, continuous media information on secondary or tertiary storage. The secondary storage devices allow random access 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. 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. The critical resources in a digital video system are the disk bandwidth, storage volume, and main the memory on the server side (see figure 1). Dimensioning and utilizing these resources efficiently is a critical task because they largely determine the cost of the video server and therefore the cost of the service.

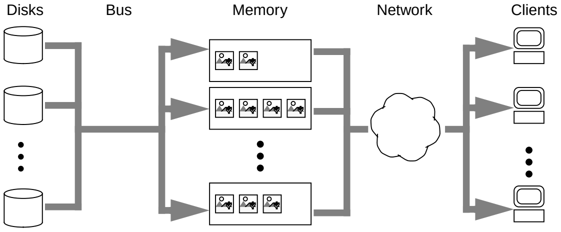


Figure 1. Components of a Video Server System.

The scheduling of the requests to read from disk determines the order in which the requests are served and influences the disk I/O efficiency and the buffer requirement. We assume that the reads from disk are organized in rounds and starvation is avoided by reading ahead an amount of data that lasts in terms of playback duration at least until the end of the next round. The requests that need to be served during one round can be scheduled using SCAN or C-SCAN scheduling [14] to minimize the seek overhead between adjacent retrievals of a single round.

The data retrieval technique determines the way data is read from the disk during a service round. Using VBR as a data model for a video, we can map video data onto data blocks stored on the disk in two ways:

- Variable size blocks of constant playout duration, referred to as **constant time length** (CTL) or

- Fixed size blocks of variable playout duration, referred to as **constant data length** (CDL) [2].

  Throughout this paper we assume CDL retrieval. For a comparison between CDL and CTL see [5].

Constant data length (CDL) retrieval performs non-periodic retrieval of constant amounts of data from the disk. To make CDL compatible with round-based disk retrieval, we introduce the notion of active and idle rounds. During an **active** round, a constant size data block is read from the disk. During an **idle** round, no data at all is retrieved. Whether a round will be active or not, can be decided on-line. The details will be explained below.

Given a fixed amount of these resources, a video server can only deliver a limited number of video streams simultaneously. The quality of service in a video server is determined by the timely delivery of the video information, which is encoded as a *variable bit rate stream* (VBR) of constant quality. If the retrieval rate of data falls behind the transmission rate, the client will experience starvation and the quality of service will be affected negatively. If the arrival of data is ahead of the transmission, the difference between the amount of data that has been retrieved and the amount of data transmitted, which is referred to as backlog, must be buffered until transmission. In this paper we assume that:

- Buffering is done at the server side. Although the buffering can be done on the client side, we prefer the buffer to be at the server side, which allows for sharing the buffer among the different streams.(see figure 1).
- the service is *deterministic*, i.e. there will never be starvation.

2 Video Traffic Characterization

To offer a deterministic service for a VBR video we need a deterministic, i.e. a worst case traffic characterization of the video stream. The novelty of our approach consists in using a so called ”burstiness function” to characterize a video. The burstiness function was used by several authors (e.g. [4], [9], [8], [7], [11]) as a deterministic characterization of VBR video streams. It can be seen as a set of leaky buckets to which the traffic conforms in a lossless way with leak rates ranging between the average and peak rate of the traffic of a video stream. More formally, given a VBR video traffic defined by its **instantaneous generation bit rate** $c(t)$ (which defines its consumption rate as well), its burstiness function $b(r)$ is defined as:

$$b(r) = \max_{t_a \leq t_b} \int_{t_a}^{t_b} (c(t) - r)dt$$

where $\lambda \leq r \leq P$ and λ and P denote the **average** and **peak rates** of the video stream. In the context of video servers, we use the burstiness function $b(r)$ to establish a relationship between

- the amount of disk I/O bandwidth, which can vary between the peak and mean bit rates of the video, and
- the amount of buffer required on the server side.

3 System Model

Next we introduce the notations and definitions used to describe the system model:

- N: number of videos
- $[0, T]$: duration of video
- $t_s \in [0, T]$: Point in the video where the playout starts
- $c_i(t)$: transmission (and consumption) rate of video $i \in N$ at time $t \in [t_s, T]$
- r: maximum retrieval rate of video from disk
- $R_i(t)$: current data retrieval rate of video $i \in N$ at time $t \in [t_s, T]$
- $b_i(t_s, r) = \max_{t_a \leq t_b; t_a, t_b \in [t_s, T]} \int_{t_a}^{t_b} (c_i(t) - r)dt$: **burstiness function** of video i
- $p_i(t_s, r) = \max_{t_o \in [t_s, T]} \int_{t_s}^{t_o} (c_i(t) - r)dt$: **prefetch function** of video i
- $d_i(t_s, t_x, r) = \int_{t_s}^{t_x} (c_i(t) - r)dt$: difference between cumulative consumption and production
- $Ts_i(t_s, r) = p_i(t_s, r)/r$: **startup latency**
- $B_i(t_s, t_x, r)$: **buffer level** of video i at time t_x. Any time the buffer level falls below zero is referred to as **starvation**.

3.1 Buffer Preloading

Variable bit rate video is highly bursty with ratios P/λ of peak to mean bit rate often larger than ten. To offer a deterministic service, the cumulative retrieval (production) of the video data from disk must always be *larger* than the cumulative consumption, i.e. $B_i(t_s, t_x, r) > 0, \forall t_x \in [t_s, T]$.

Theorem PL: If we retrieve data at a constant rate r lower than the peak bit rate P , we can avoid starvation by *preloading* the amount $p_i(t_s, r)$ of data into the buffer before the transmission of video starts.

Proof: $p_i(t_s, r) - d_i(t_s, t_x, r) = \max_{t_o \in [t_s, T]} \int_{t_s}^{t_o} (c_i(t) - r)dt - \int_{t_s}^{t_x} (c_i(t) - r)dt \geq 0, \forall t_x \in [t_s, T]$. QED.

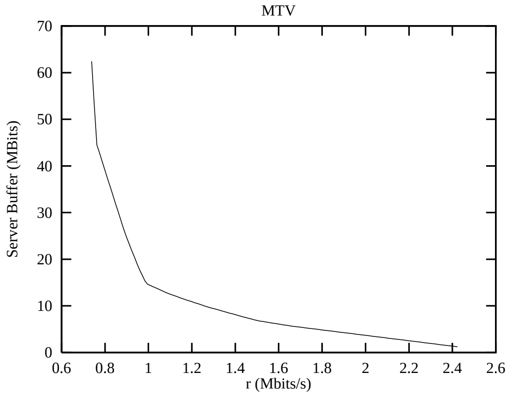
We can easily show that it is sufficient to compute the prefetch function as:
$p_i(t_s, r) = \max_{t_o \in [t_s, t_b]} \int_{t_s}^{t_o} (c_i(t) - r)dt$, where t_b is the right hand side of the interval $[t_a, t_b]$ for which $\int_{t_a}^{t_b} (c_i(t) - r)dt = b_i(t_s, r)$ since due to the definition of $b_i(t_s, r)$ we have:
$\forall t_x \in (t_b, T] : \int_{t_b}^{t_x} (c_i(t) - r)dt \leq 0$.

3.2 Burstiness Function

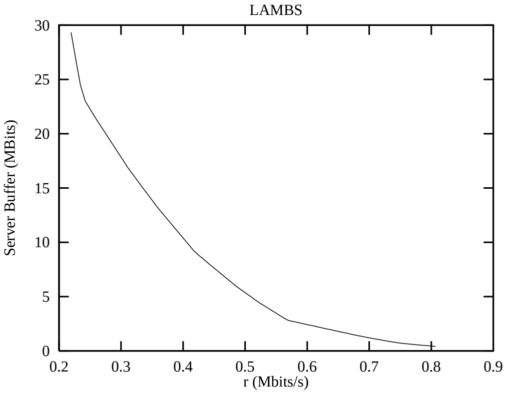
The burstiness function indicates by how much, in the worst case, for any interval $[t_a, t_b] \subset [t_s, T]$ of time, the consumption will be higher than the production, i.e. the amount of video data retrieved from the disk. As we will show later, $b_i(t_s, r)$ is the maximum amount of video data we need to buffer to avoid starvation at any point of time.

If we plot $b_i(t_s, r)$ as a function of the rate r at which video i is retrieved from the disk, we see (figure 2) that the burstiness function is monotonously decreasing and convex. $b_i(t_s, r)$ is defined for the range of rates r, $\lambda \leq r \leq P$ between the videos mean bit rate λ, for which the value of the burstiness function is highest, and the peak bit rate P, for which the value of the burstiness function is zero (figure 2). For different videos the burstiness function takes different absolute values and its shape (steepness) varies. We see, for instance for the MTV trace (figure 2(a)) , that a small increase in the rate r from 0.8 Mbit/s to 1.0 Mbit/s leads to a drastic reduction of the amount of server buffer required. For Lambs, the impact of an increase of r on the buffer requirement is less pronounced.

We use in this paper the MPEG-1 video traces produced by O. Rose [12] that are publicly available. The different names such as MTV, Lambs, or Starwars refer to the traces of films with the same names.



(a) MTV



(b) Lambs

Figure 2. Burstiness Function.

3.3 Retrieval Algorithm

If the data retrieval rate $R_i(t)$ of video i remains constant (equal to r) during the whole session, the server buffer keeps growing at an average rate of $(r - \lambda)$ and its maximum size becomes proportional to the session duration. It is therefore necessary to interrupt the retrieval process to keep the buffer size bounded while still ensuring that no starvation occurs. The retrieval proceeds as follows:

- When a client requests a video i to start at time t_s, the associated buffer (at the server side) will first be filled with the amount $p_i(t_s, r)$ of data. Then the transmission starts at the variable rate $c_i(t)$. Only a small buffer is used at the *client side* to absorb transmission jitter.

- During video transmission, the buffer will be filled at a rate $R_i(t) = r$ while the buffer level is below $b_i(t_s, r)$. If a buffer level of $b_i(t_s, r)$ is reached, the filling of the buffer will be interrupted, i.e. $R_i(t) = 0$, until the buffer level falls below $b_i(t_s, r)$[1].

Theorem SV: The retrieval algorithm will avoid starvation at all times, i.e. $.B_i(t_s, t_x, r) \geq 0, \forall t_x \in [t_s, T]$.

Proof: We distinguish the two possible scenarios:

- if the retrieval rate has never been interrupted between t_s and t_x (equivalently the buffer size has never reached $b_i(t_s, r)$), then we have $B_i(t_s, t_x, r) = p_i(t_s, r) - d_i(t_s, t_x, r) > 0$ (given by theorem PL).
- otherwise, let t_y be the last time in $[t_s, t_x]$ where $B_i(t_s, t_y, r) = b_i(t_s, r)$. Then we have $B_i(t_s, t_x, r) = b_i(t_s, r) - \int_{t_y}^{t_x} (c_i(t) - r)dt$ which is necessarily non-negative (by definition of $b_i(t_s, r)$. QED.

Figure 3 summarizes the retrieval and delivery of the video data and how the sender controls the retrieval as function of the buffer level.

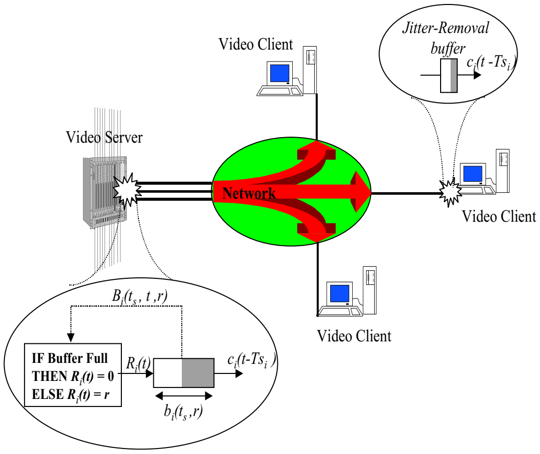


Figure 3. Systems Model.

4 Performance

In this section we present a performance evaluation, in terms of cost and delay, of the proposed scheme.

4.1 Cost Study

Theorem SV allows us to use the burstiness function to optimally trade-off buffer memory for disk retrieval rate. Assume that the price P_{ram} per **Mbit of main memory** and the price $P_{dI/O}$ per **Mbit/s of disk I/O** are known, then, if we ignore for the moment all other cost factors, the **cost** $P_{tot}(i, r)$ **of servicing a single video** i is given as

$$P_{tot}(i, r) = P_{ram} * b_i(t_s, r) + P_{dI/O} * r \qquad (1)$$

Figure 4 depicts the cost function for a given ratio $\frac{P_{dI/O}}{P_{ram}} = 20$, which means that the price for one Mbit/s of disk I/O is 20 times higher than the price for one Mbit of RAM buffer. We see that for both videos there is a **optimal retrieval rate** r^* for which the total cost is minimal. The chosen ratio of 20 is a realistic value given todays prices for disks and RAM buffer.

If we take the first derivative of $P_{tot}(i, r)$ and set it to zero, we obtain the rate r_i^* for which the delivery of the video i is cost-optimal. We get

$$\frac{dP_{tot}(i, r)}{dr} = \frac{db_i(t_s, r)}{dr} * P_{ram} + P_{dI/O} \qquad (2)$$

$$\frac{db_i(t_s, r_i^*)}{dr} = -\frac{P_{dI/O}}{P_{ram}} \qquad (3)$$

Figure 5 depicts the optimal rate r as a function of the cost ratio $\frac{P_{dI/O}}{P_{ram}}$. As the cost ratio increases, i.e. the cost of the disk IO compared to the cost of memory increases, it becomes more cost efficient to decrease the rate r at the expense of an increasing buffer size.

4.2 Start-up latency

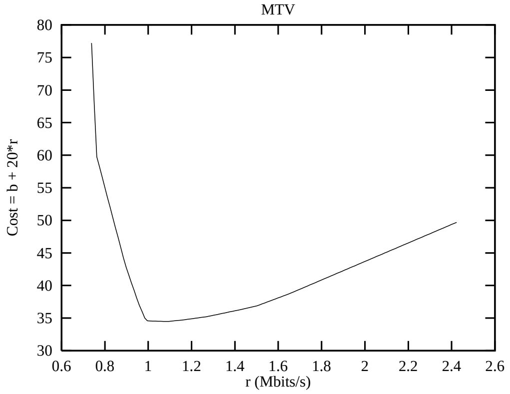
The start-up latency can be defined in different ways. The **worst case** start-up latency $Ts_i(r)$ is defined as

$$Ts_i(r) = \max_{t_s \in [0, T]} Ts_i(t_s, r) \qquad (4)$$

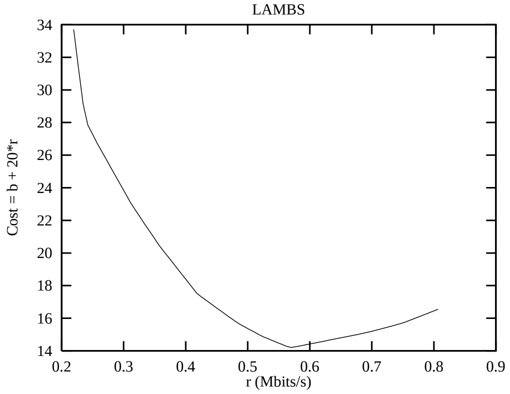
The worst case start-up latency, as function of the rate r depicted in figure 6, has a shape similar to

---

[1] In practice this is implemented using a threshold, whose value takes into account the discrete way of transferring data blocks

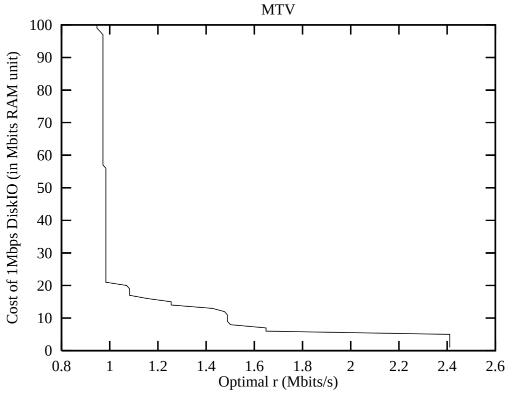


(a) MTV

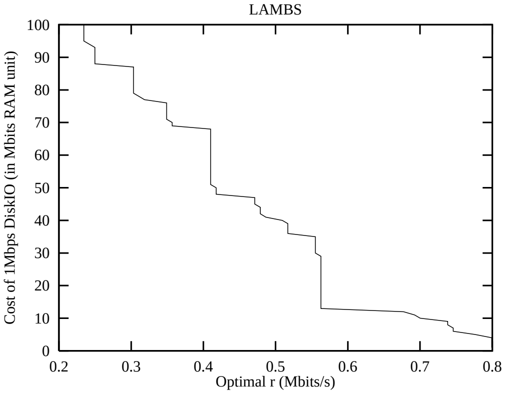


(b) Lambs

Figure 4. Cost function P_{tot} for $\frac{P_{dI/O}}{P_{ram}} = 20$.



(a) MTV



(b) Lambs

Figure 5. Optimal I/O bandwidth vs. cost ratio $\frac{P_{dI/O}}{P_{ram}}$.

the burstiness function (cf. figure 2). Indeed one can easily show that:

$$Ts_i(r) = b_i(t_s, r)/r$$

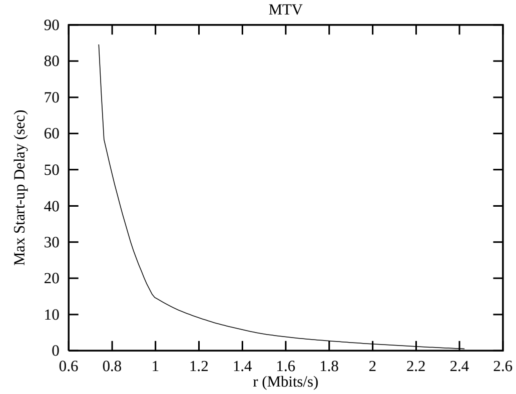
For values of r that are close to the mean rate, the worst case start-up latency is to the order of a minute or higher. Such high values are clearly not acceptable for a user and do not allow any meaningful interactivity during the playout.

However, the worst case start-up latency gives overly pessimistic values as we can see from figure 7 that depicts the **probability distribution function** of the start-up latency for a given rate r. There we see that the probability that the start-up latency will take its maximal value is very low.

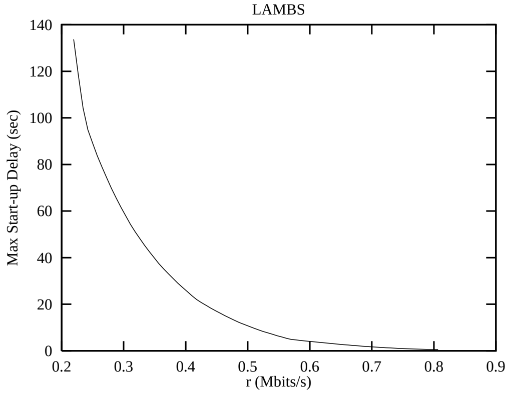
Clearly, the start-up latency depends on the starting time t_s of the playback. We could think about the average value of $Ts_i(t_s, r)$ (over all t_s values) as a good measure of the start-up latency. However this assumes that users start watching movies at a time that is uniformly distributed over the movie duration. A more reasonable assumption would be that the probability of starting at time t_s decreases exponentially in time which reflects that most of the users start watching the sequence from the beginning. We therefore suggest another measure for the start-up latency that we call **weighted mean start-up latency** and this is defined as

$$\hat{T_i}(r) = \int_0^T k.e^{-kt} Ts_i(t, r) dt \qquad (5)$$

where k is a tunable parameter that is fixed (for our

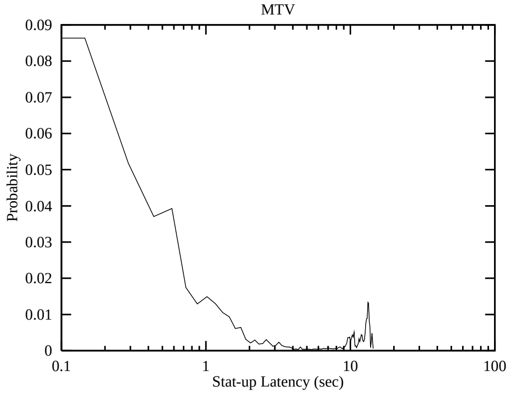


(a) MTV

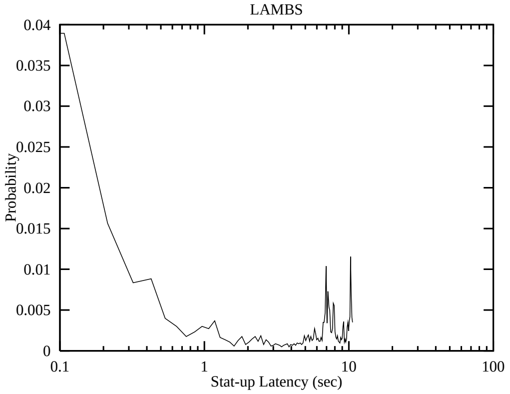


(b) Lambs

Figure 6. Worst case start-up latency.



(a) MTV



(b) Lambs

Figure 7. Distribution of the start-up latency.

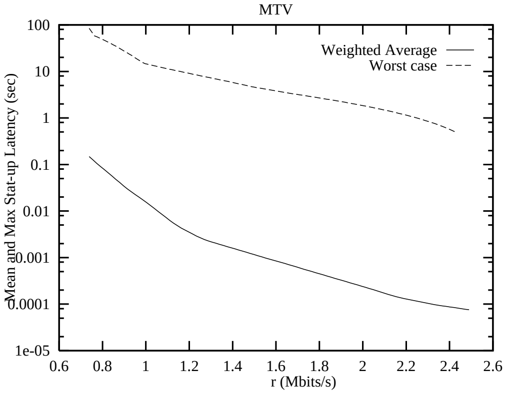
experiments) so that 90% of sessions start before $t_s = 60$ sec.

The weighted mean start-up latency, together with the worst case start-up latency, as function of the rate r, is depicted in figure 8. We see that the weighted mean start-up latency is about two orders of magnitude lower than the worst case start-up latency.

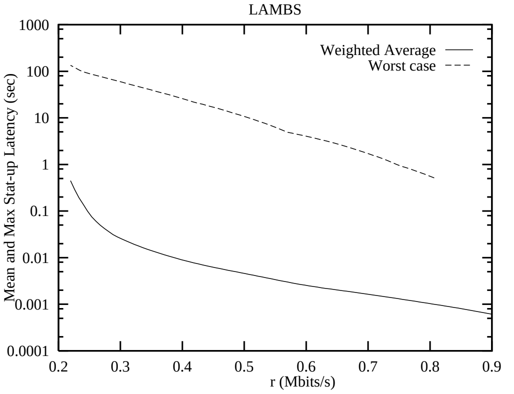
In figure 9 we see the start-up latency as a function of the starting time t_s. This figure confirms that in most of the cases the start-up latency is much lower than the worst-case start-up latency. To reduce the worst-case start-up latency, one can think of a modification of the scheduling algorithm: When a client starts a movie at a point of time where the start-up latency perceived would be higher than a few seconds, data can be sent with a rate higher than r until the buffer is full. This ”fast load” [10] is possible since the video server has, for each client, idle rounds where no data are retrieved. The percentage of idle rounds for stream i with mean bit rate λ_i that is allocated a rate r_i, with $r_i > \lambda_i$, is given as $(r_i - \lambda_i)/r_i$. Note that these idle rounds occur even when the maximum number of clients that can be served is admitted.

5 Related Work

Previous work has been done in the area of data retrieval in video servers for VBR streams. Chang and Zhakor were one of the first to discuss CTL and CDL-based retrieval for VBR streams [2, 3]. For CDL, there are no idle rounds and the bit rate allocated is equal to the mean bit rate of the video, which

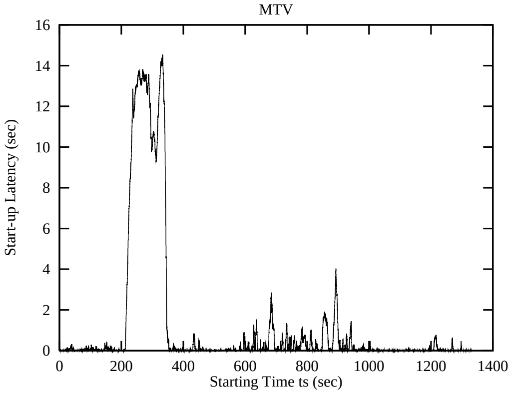


(a) MTV

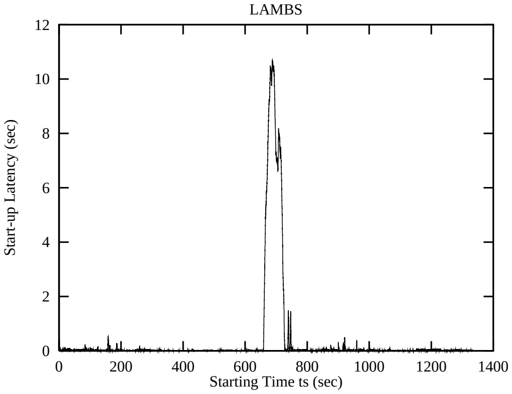


(b) Lambs

Figure 8. Worst-case and weighted mean start-up latency.



(a) MTV



(b) Lambs

Figure 9. Start-up latency as function of time t_s where playout starts.

corresponds to one point on the burstiness curve. As we know, choosing the mean bit rate results in a large start-up latency and a large buffer requirement.

In [5, 1] the authors introduced the idea of idle rounds for CDL. They also distinguish the disk service round, during which data for each stream are read exactly once from the disk, and the smoothing interval, which is used to compute the amount of I/O bandwidth that must be reserved for a stream: the smoothing interval is a *multiple* of a disk service round. The separation of disk service round and smoothing interval reduced the amount of I/O bandwidth required, which in turn improved the overall efficiency of the video server. However, smoothing was still limited to an interval of several seconds of video.

Recently, Sahu [13] looked at *deadline-based* scheduling as opposed to round-based scheduling, which was assumed by the papers cited above. Sahu showed that deadline-based scheduling allows smoothing over much larger intervals than round-based scheduling, which reduces the rate variability and improves overall disk utilization. However, deadline-based scheduling does not allow for seek optimizations as does SCAN that is used in combination with round-based scheduling.

Our work also increases the smoothing interval while still allowing the use of round-based scheduling. To our knowledge, we are also the first to use explicit relationship between the amount of I/O bandwidth and the amount of main memory. Since the burstiness function is decreasing and convex, there exits an optimal bandwidth for each VBR video, in the sense that the overall cost for the resources required to serve the video will be minimal.

There is a large body of work on smoothing techniques for the *transmission* of prerecorded VBR video [6] that aims at reducing the variability of the video stream that is transmitted over the network: A prefetch buffer at the *client* is used to smooth the video stream transmitted over the network. This work is complementary to ours since it does not at all consider the aspect of video retrieval from a video server but focuses on the video transmission aspect. It would be interesting to combine both aspects to conceive an end-to-end solution for efficient video retrieval and transport.

6 Conclusion

An appropriate characterization of pre-coded VBR video is required to efficiently dimension a video server's resources. The two main resources usually considered are the disk I/O bandwidth and the memory used to buffer data before transmission. The benefits of using the burstiness function as a deterministic video traffic characterization is that it provides an explicit relationship between the retrieval bandwidth and the required memory. In this way, it can be considered the most suitable traffic model for a deterministic service. Although the proposed mechanism is described at the fluid scale its implementation is straight forward, in particular in the constant data length version.

References

[1] E. W. Biersack and F. Thiesse. A new class of constant data length retrieval algorithms for video servers with vbr streams. In *Multimedia Storage and Archiving Systems, SPIE Photonics East '96 Symposium*, number 2916, pages 316–329, Boston, MA, Nov. 1996.

[2] E. Chang and A. Zakhor. Admission control and data placement for vbr video servers. In *Proceedings of the 1st International Conference on Image Processing*, pages 278–282, Austin, Texas, November 1994.

[3] E. Chang and A. Zakhor. Cost Analyses for VBR Video Servers. In *IS and T/SPIE International Symposium on Electronic Imaging: Science and Technology, Volume 2667: Multimedia Computing and Networking*, volume 2667, pages 381–397, San Jose, Jan. 1996.

[4] R. Cruz. A calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Transactions On Information Theory*, 37(1):114–131, Jan. 1991.

[5] J. Dengler, C. Bernhardt, and E. W. 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.

[6] W. Feng and J. Rexford. A Comparision of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Video. In *Proc. IEEE INFOCOM'97*, Kobe, Japan, Apr. 1997.

[7] M. Hamdi, J. Roberts, and P. Rolin. Rate Control for VBR Video Coders in Broad-band Networks. *IEEE Journal on Selected Areas on Communications*, 15(6):1040–1051, Aug. 1997.

[8] S. Low and P. Varaiya. Burstiness bounds for Some Burst Reducing Servers. In *INFOCOM'93*, pages 2–7, Mar. 1993.

[9] D. Lucantoni, M. Neuts, and A. Reibman. Methods for Performance Evaluation of VBR Video Traffic Models. *IEEE/ACM Transactions On Networking*, 2(2):176–180, Apr. 1994.

[10] G. Petit, D. Deloddere, and W. Verbiest. Bandwidth resource optimization in video-on-demand network architectures. In *Proceedings of the 1994 1st International Workshop on Community Networking*, pages 91–97, San Francisco, CA, July 1994.

[11] J. Roberts and M. Hamdi. Video Transport in ATM Networks. *Interoperable Communication Networks Journal, Baltzer Science*, 1:121–143, 1998.

[12] O. Rose. Statistical Properties of MPEG Video Traffic and Their Impact on Traffic Modelling in ATM Systems. Technical Report 101, Institute of Computer Science, University of Wuerzburg, FTP address of video traces: ftp://ftp-info3.informatik.uni-wuerzburg.de/pub/MPEG/, University of Wuerzburg, February 1995.

[13] S. Sahu, Z. Zhang, J. Kurose, and T. D. On the efficient retrieval of vbr video in a multimedia server. In *Proc. 1997 Int'l Conf. on Multimedia Computing and Systems (ICMCS97)*, Ottawa, Canada, June 1997.

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