Bandwidth Reduction for Video-on-Demand Broadcasting Using Secondary Content Insertion

Alexander Golynski   Alejandro López-Ortiz
Guillaume Poirier   Claude-Guy Quimper

Abstract

An optimal broadcasting scheme under the presence of secondary content (i.e. advertisements) is proposed. The proposed scheme works both for movies encoded in a Constant Bit Rate (CBR) or a Variable Bit Rate (VBR) format. It is shown experimentally that secondary content in movies can make Video-on-Demand (VoD) broadcasting systems more efficient. An efficient algorithm is given to compute the optimal broadcasting schedule with secondary content, which in particular significantly improves over the best previously known algorithm for computing the optimal broadcasting schedule without secondary content.

1 Introduction

The traditional way for television broadcasting companies to broadcast their programs is to send through a medium (air or cable) a sequence of TV shows. Viewers take notes of programs they want to watch in a prepublished broadcasting schedule (e.g. "TV Guide") and wait until it is broadcast. If a customer misses the broadcast time of the desired TV show, it is impossible to recover it. The use of a VCR to record the show at the predetermined time addresses this problem but requires the viewer to know in advance which TV show s/he will miss. Alternatively, a Video-on-Demand (VoD) system allows users to select a TV show or a movie from the broadcaster's movie selection and to start watching it at any time.

Multicasting systems create a connection between a client and a server. Two clients downloading the same movie at the same time may partly share the same connection. If the viewers download the movie at different times, the connection cannot be shared and two point-to-point connections are established requiring more bandwidth from the server. In contrast, broadcasting systems broadcast a stream of data on a medium to which the clients tune in. This solution requires a constant amount of bandwidth regardless of the number of clients listening to the media.

Many broadcasting schemes have been proposed. Most divide a movie into several segments with each segment periodically broadcast on a separate channel. When the client connects to the system, it listens on all these channels. After waiting for a period w, the client can start watching the first segment. The scheme ensures that the next segment is fully downloaded before the current segment has been watched. The broadcasting schemes can be divided into two categories. The first category of scheme assign the same bandwidth to all channels. This is the case for pyramid broadcasting [22], permutation-based pyramid broadcasting [1], fast broadcasting [10], client-centric approach [6], and greedy equal bandwidth broadcasting [8] among others. The second category divides the movie in segments of equal size but assign different bandwidth to the channels. Among this category figure harmonic broadcasting [9], cautious harmonic broadcasting [17], quasi harmonic broadcasting [17], and poly-harmonic broadcasting [18].

These schemes presented above can broadcast movies encoded at a constant bit rate (CBR). For variable bit rate (VBR) encoding, the quality remains constant throughout the movie but the consumption rate varies. For the MPEG 4 format, the consumption rate can vary from 5 Kbits/s to 1 Gbits/s. Several broadcasting schemes exist for VBR encoded movies. The Variable Bandwidth Harmonic Broadcasting (VBHB) [19] and VBR Broadcasting (VBR-B) [20] are based on the cautious harmonic and fast broadcasting schemes, respectively. Trace adaptive fragmentation (TAF) [13] improves over VBR-B. The Loss-Less and Bandwidth Efficient scheme (LLBE) guarantees optimal server bandwidth but requires an off-line computation beforehand. The Staircase Broadcasting (SCB) [14] and Simple VBR Staircase Broadcasting (SVSB) [5] also address the problem of optimizing client bandwidth.

Introducing secondary content like advertisements or previews in VoD has been studied as a means to reduce the bandwidth required in multicasting systems [12, 2, 11]. This technique slows down the leading stream by inserting advertisements and let it merge later with the trailing stream. In this paper we show that the bandwidth requirements can be reduced by introducing secondary content in broadcasting systems both under CBR and VBR encoding. We first introduce some concepts used by VoD broadcasting systems and then show how we can reduce the bandwidth in the Harmonic [9] and LLBE [15] broadcasting schemes. The LLBE scheme requires an off-line computation. We give an algorithm with time complexity of $|F|2^{O(\sqrt{\log n \log\log |F|})}$ which outperforms the previously best $O(n|F|^2)$ algorithm described in [15, 7]. To place this improvement in context, the number of frames F in a typical movie is in the order of 200,000. Hence our improved algorithm has a run time several orders of magnitude faster than the previously best known method.

2 Background

Broadcasting systems consider movies as an ordered set of frames $F = \{f_0, f_1, \ldots\}$ such that the size of the i^{th} frame is f_i bytes. Constant bit rate encodings produce movies with frames of equal size. Variable bit rate encodings produce movies with frames of different sizes. The secondary content is represented by a multi-set of commercial pauses $C = \{c_1, \ldots, c_m\}$ where each pause c_i contains α frames shown before the movie frame f_{c_i}. Each advertisement frame a_i in $A = \{a_0, a_1, a_2, \ldots\}$ has a_i bytes. An advertisement is encoded by α consecutive frames in A starting with frame $a_{k\alpha}$ for an integer k. These advertisements are shown during commercial pauses but no assumptions can be made about the order in which the advertisements will be presented.

Broadcasting systems like Harmonic Broadcasting (HB) [9], Pyramid Broadcasting (PB) [22] and Loss-Less-Bandwidth-Efficient (LLBE) [15] partition the movie frames F into n segments $S = \{S_1, \ldots, S_n\}$ each contain consecutive movie frames. Each segment S_i is broadcast repeatedly over a logical channel C_i with bandwidth b_i.

When connecting to the system, a client starts downloading all segments over all logical channels for a period of w units of time and then starts watching the movie. After these w units of time, the first segment is fully downloaded and is ready to be shown. After watching the first segment, the second segment is fully downloaded and ready to be shown and so on. In this scheme, each time a segment is fully downloaded, the client stops listening on the corresponding logical channel.

Scheduling a movie consists in segmenting the movie frames in n segments and assigning the bandwidth b_i such that each segment is fully downloaded by the client before watching it. A natural goal to this problem is to minimize the total amount of bandwidth $B = \sum_{i=1}^{n} b_i$ used by the system.

We first present how to modify the Harmonic scheduling scheme to include advertisements in constant bit rate videos and then show how to insert advertisements in variable bit rate videos using LLBE.

3 Harmonic Broadcasting

The harmonic broadcasting scheme [9, 16] is simple and efficient for constant bit rate encoded movies. It considers a partition of the movie frames into n segments of $\frac{|F|}{n}$ frames each of w units of time. We assume the physical media is divided into channels. Each channel has the exact capacity to transmit one movie in real time. Harmonic broadcasting sends the movie in segments with each segment taking a fraction of a physical channel. Let b_i be the fraction

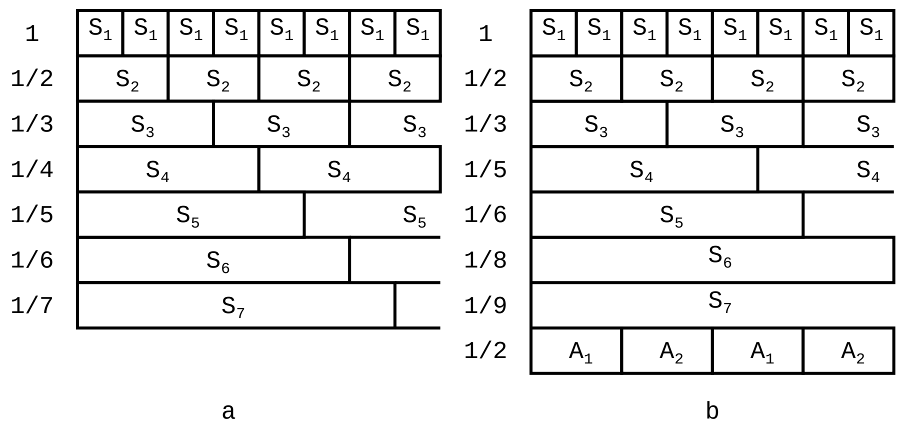

Figure 1: a) Harmonic broadcasting scheme for a movie with 7 segments. b) Modified harmonic broadcasting scheme for 7-segment movie with $p = 2$. There is a commercial pause before S_4 and S_6. This scheme requires more bandwidth than a) for this example but would require less if there were more than 24 movie segments.

of a physical channel used to send segment S_i. We have $b_i = \frac{1}{i}$ for a total of $B = \sum_{i=1}^{n} \frac{1}{i} = H_n$ physical channels where H_i is the i^{th} harmonic number.

This scheme ensures that a user arriving at time t has fully downloaded segment S_i by time $t + iw$ and is ready to watch it. Engebretsen and Sudan [4] proved this scheme is optimal assuming there is a maximum waiting time of w and there is a least one client connecting every w units of time.

We generalize the harmonic broadcasting scheme to include advertisements. Consider the same movie of K segments where we insert commercial pauses of w units of time at every p movie segments starting from S_{2p}. We therefore have $\frac{K}{p} - 1$ commercial pauses before the movie segments $S_{2p}, S_{3p}, S_{4p}, \ldots$. Let $N(x)$ be the number of commercial pauses before segment S_x.

$$N(x) = \begin{cases} 0 & \text{if } x < 2p \\ \lfloor \frac{x}{p} \rfloor - 1 & \text{if } x \geq 2p \end{cases} \tag{1}$$

We broadcast each movie segment S_i at speed $b_i = \frac{1}{i+N(i)}$ to ensure they are fully downloaded at time $t + i + N(i)$. Notice that advertisements delay the watching time of following movie segments and therefore reduce the bandwidth of the corresponding channels (see Figure 1). This bandwidth gain would be lost if we were broadcasting each advertisement on its own channel but we consider a unique channel C_a with bandwidth $\frac{1}{p}$ that sequentially broadcasts all advertisements in arbitrary order. A user arriving at time t will download at

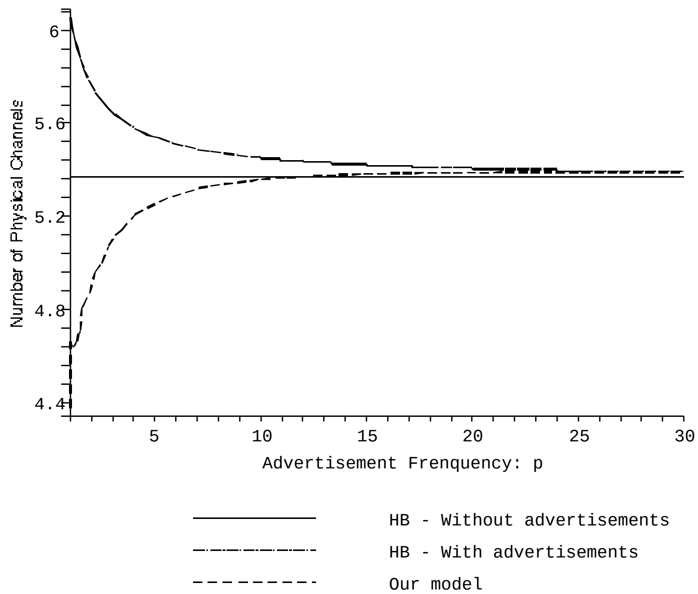

Figure 2: Number of physical channels required to broadcast a movie of 120 segments.

least $i-1$ complete advertisement segments by time $t+iw$. Surprisingly, the gain caused by postponing movie segments can be greater than the bandwidth used for the advertisement channel. It follows that adding advertisements can reduce the amount of bandwidth required for a broadcasting system.

We compare our scheme to the standard harmonic broadcasting scheme by creating two versions of a same movie. The first version is the movie without advertisements. The second version of the movie contains the advertisement frames which we treat as movie frames. We apply the normal broadcasting scheme to the two versions of the movie and compare it with our broadcasting scheme. Figure 2 shows a comparison of the required number of physical channels with respect to the advertisement frequency p.

4 LLBE with Secondary Content

In the previous section, we described how constant length advertisements can be inserted at fixed intervals of time in CBR encoded movies. In this section, we give more flexibility with respect to the appearance time and the duration of each commercial pause. Moreover, we explain how to insert advertisements in VBR encoded movies.

Nikolaidis et al. [15] proposed the Loss-Less and Bandwidth-Efficient (LLBE) broadcasting scheme that can be applied to videos with variable bit rate encodings. We generalize this scheme to introduce commercial pauses at any position in the movie and of different durations.

Let the vector C represent the commercial pauses. Component C_i indicates the earliest movie frame that succeeds the i^{th} commercial pause. For instance, if $C = [10, 42, 108]$, there are three commercial pauses respectively presented before the movie frames f_{10}, f_{42}, and f_{108}. All commercial pauses are α frames long. In order to form a longer commercial pause, one can insert two commercial pauses before the same movie frame. For instance, the second commercial pause in $C = [10, 42, 42, 108]$ is twice longer than the others. Let $N(C, x)$ be the number of advertisement frames shown before the movie frame f_x i.e. $N(C, x) = \alpha|\{C_i \in C \mid C_i \leq x\}|$ where $|S|$ denotes the cardinality of set S. Using this function, we deduce that the number of frames (both movie frames and advertisement frames) displayed before frame f_x to be $x + N(C, x)$.

Consider a segmentation S of the movie such that each segment $S_i^j = \{f_i, \ldots, f_{j-1}\}$ in S consists of the consecutive movie frames f_i to f_{j-1}. Summing the size of each movie frames gives the size in bytes of the whole segment, i.e. $\sum_i^{j-1} f_i$. We recall that before watching segment S_i^j, the client needs to watch $i + N(C, i)$ movie and advertisement frames. Given that the movie is played at a consumption rate of R frames per second and that the client must wait for w seconds before starting watching the movie, there are $w + \frac{i+N(C,i)}{R}$ seconds elapsed before the client starts to watch segment S_i^j. Putting all results together, the required bandwidth to broadcast segment S_i^j is:

$$b(i, j) = \frac{R \sum_{k=i}^{j-1} f_k}{Rw + i + N(C, i)} \tag{2}$$

A question remains: how do we segment the movie into n segments $S_{s_i}^{s_i+1}$ such that the total bandwidth $B = \sum_{i=1}^{n} b(s_i, s_{i+1})$ is minimized? Nikolaidis et al [15] give an algorithm for computing the optimal solution for variable bit rate encoding without advertisements. We generalize their solution to include advertisements and also provide a faster algorithm to compute the segmentation.

We construct a graph G of $|F|+1$ nodes such that each frame f_i is represented by a node i plus an extra sink node $|F| + 1$. There is a directed edge (i, j) with weight $b(i, j)$ for each pair of nodes $i < j$. Edge weights can be computed in constant time using an array $A[1..|F|]$ such that $A_1 = 0$ and $A_{i+1} = A_i + f_i$. The summation in Equation 2 can be computed in constant time using $A_j - A_i$. Finding the n-segmentation of a movie with minimum bandwidth consists in finding the shortest path of n links where each link represents a segment. This can be achieved by running the Bellman-Ford algorithm and stopping after the n^{th} iteration as suggested by Nikolaidis et al. [15]. This solution requires $O(n|F|^2)$ steps. We suggest a faster way to compute this segmentation using the concave Monge property of the graph.

Definition 1 (Concave Monge Property [21]) Let $b(i,j)$ be the weight attributed to an edge (i,j) of a weighted directed acyclic graph G. *Graph G satisfies the concave Monge property if the inequality $b(i,j) + b(i+1,j+1) \leq b(i,j+1) + b(i+1,j)$ holds for all $1 < i+1 < j < n$.*

Lemma 1 *The scheduling graph G satisfies the concave Monge property.*

Proof In order to prove that G satisfies the concave Monge property, we have to show that $b(i,j) + b(i+1,j+1) \leq b(i,j+1) + b(i+1,j)$. We know that $i \leq i+1$ and that $N(C,i) \leq N(C,i+1)$ and therefore

$$\frac{Rf_j}{Rw + i + 1 + N(C,i+1)} \leq \frac{Rf_j}{Rw + i + N(C,i)} \tag{3}$$

Adding (4) on each side of Equation 3 results in (5).

$$\frac{R\sum_{k=i}^{j-1} f_k}{Rw + i + N(C,i)} + \frac{R\sum_{k=i+1}^{j-1} f_k}{Rw + i + 1 + N(C,i+1)} \tag{4}$$

$$b(i,j) + b(i+1,j+1) \leq b(i,j+1) + b(i+1,j) \tag{5}$$

Consequently, graph G satisfies the concave Monge property. □

The concave Monge property suggests that altering the size of the segments at the end of the movie will not affect the bandwidth as much as altering the size of the segments at the beginning of the movie.

Lemma 2 *There exists an algorithm that computes the n-link shortest path of the scheduling graph G in $|F|2^{O(\sqrt{\log n \log\log |F|})}$ steps.*

Proof By Lemma 1, graph G satisfies the concave Monge property. Schieber's algorithm [21] finds the k-link shortest path for such a graph in $|F|2^{O(\sqrt{\log k \log\log |F|})}$ steps. This algorithm is best known so far in the case $k = \Omega(logn)$, there are faster algorithms if $k = o(\log n)$, see [21] for more information. This algorithm verifies the lemma when $k = n$. □

From the two previous lemmas, we obtain the following theorem.

Theorem 1 *There exits an algorithm that computes the optimal segmentation of a movie in* $|F|2^{O(\sqrt{\log n \log\log |F|})}$ *steps.*

Proof The proof follows from Lemma 1 and Lemma 2. $\square$

This is an important improvement over the previous $O(n|F|^2)$ algorithm suggested in [15, 7] since most movies have more than 200,000 frames ($|F|$). Indeed our implementation of LLBE resulted in run times in the order of several hours in a personal computer running Linux.

Our bandwidth minimization problem also has the following geometric interpretation. Consider the following set of points on the plane: for each frame j we have the point with coordinates

$$P_j = \left(R \sum_{k=1}^{j-1} f_k, \frac{1}{Rw + j + N(C,j)} \right) \tag{6}$$

The task is to find a staircase-like polygonal chain that connects the first point P_1 to the point $(P^x_{|F|}, 0)$ (where P^x_i denotes the x-coordinate of P_i) using $k-1$ intermediate vertices and has minimal area under it, e.g. see Figure 3.

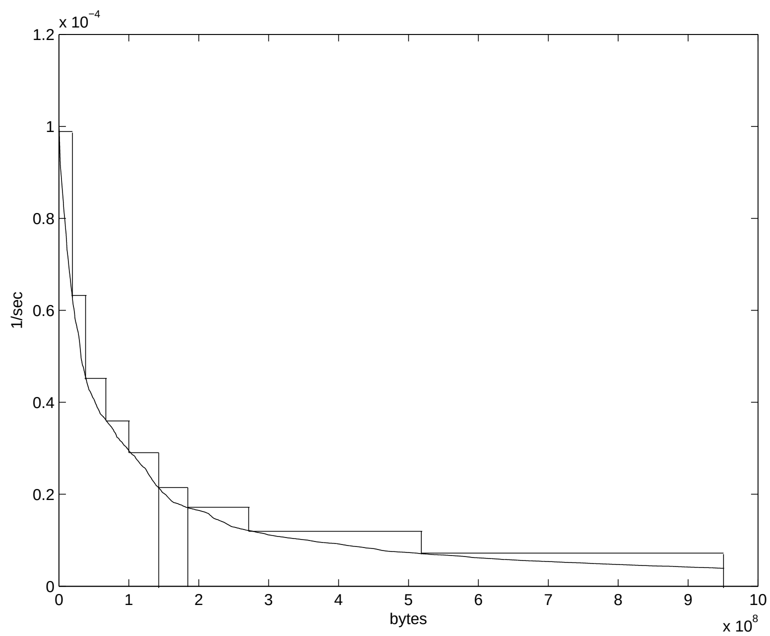

Figure 3: An example of staircase for the case $|F| \approx 250000$, k = 9; points are generated according to frame sizes of a real movie, waiting time is set to 10000 frames (about 5 min). Sixth rectangle is shown for explanation purposes.

The edge from i to j ($i < j$) in the graph corresponds to a rectangle defined by points P_i and P_j: P_i being the upper-left vertex and $(P_j^x, 0)$ lower-right vertex. The area of this rectangle is exactly $b(i, j)$. It is not hard to see that the problem of finding a staircase-like polygonal chain (to be precise, the problem of k-link shortest path in the corresponding graph) has the Monge property, however, not all problems with the Monge property can be expressed in this form. Therefore, potentially, there might be a faster algorithm for minimizing bandwidth than it is proposed by the Theorem 1. We leave it an open problem.

We now have to determine the bandwidth of the advertisement channel. Recall that an advertisement is a sequence of α frames $a_{k\alpha} \dots a_{(k+1)\alpha} \subset A$, such a sequence must be downloaded by the client before it is played. If a client connects while the system has already broadcasted the beginning of an advertisement, the whole advertisement is lost since there is no guaranty that the beginning can be downloaded later. In order to download x complete advertisements, a client has to download $\alpha(x+1)$ frames. Notice that the size in bytes of these frames might be different depending on which advertisement is broadcast when the client connects to the system. Let $W(x)$ denote the size in bytes of the largest window (in terms of bytes) of αx consecutive advertisement frames.

$$W(x) = \max_{i=1\dots|A|} \sum_{j=i}^{i+\alpha x-1} a_{j \bmod |A|} \tag{7}$$

The bandwidth b_c of the advertisement channel must be high enough to ensure that $W(i+1)$ bytes are broadcast before playing the commercial pause C_i. We therefore have.

$$b_c \quad = \quad \max_{i=1\dots|C|} \frac{W(i+1)F}{wF + c_i + \alpha(i-1)} \tag{8}$$

Equation 7 can be computed in linear time. We first compute the size of a window for $i = 1$. We obtain the size of the window for $i+1$ by subtracting a_i and adding $a_{i+\alpha x \bmod |A|}$. After performing this operation for all values of i, we can compute the value $W(x)$ in $O(|A|)$ steps. It follows that Equation 8 can be computed in $O(|C||A|)$ steps.

A system broadcasting a set of M different movies can use a single advertisement channel for all movies. We simply have to compute the value b_c for each movie and consider the highest one. Notice that as the number of movies increases, b_c becomes negligible when compared to the total system bandwidth.

5 Experiments

We have recorded a movie broadcast on a public TV channel and encoded it with the MPEG 4 format. The movie is 2 hour 41 minutes long and contains 37 minutes of advertisements divided into 12 commercial pauses. Each commercial pause is either 180 or 210 seconds long. We represent these commercial pauses with our model by a series of 6 or 7 consecutive commercial pauses of $\alpha = 30$ seconds each. The movie was preceded by 1 minute of advertisements for the TV station, the distributor, and the producer of the movie. Since there are few producers and distributors for main stream movies, we assume their advertisements are pre-downloaded on the client side. We therefore treat the first minute of advertisements as the latency time of $w = 60$ seconds. We compare our broadcasting scheme with LLBE. The latter has been compared with other broadcasting schemes in [15].

Table 1 shows that broadcasting a movie with LLBE without advertisements requires a bandwidth of 579.49 Kb/s. Inserting advertisements increases the duration of a movie which increases the required bandwidth in VoD systems. With LLBE, inserting advertisements in the movie increases the bandwidth to 658.19 Kb/s which represents an increase of 13.6 %. If we insert advertisements in a movie as described in Section 4, we need a bandwidth of 559.14 Kb/s for broadcasting the movie and 48.12 Kb/s for the movie channel. The total bandwidth is therefore 607.27 Kb/s with savings of 50.92 Kb/s with respect to the LLBE+ads solution.

The strength of our solution is revealed when we consider a system broadcasting many movies all sharing the same advertisements channel. Since only one advertisement channel is required, its bandwidth can be amortized over the number of movies. Amortizing increases the total bandwidth of the system but reduces the bandwidth per movie. Suppose 5 movies share the same advertisement channel, the amortized bandwidth for one movie is the bandwidth required to broadcast the movie plus a fifth of the advertisement channel. Table 1 shows a saving of 1.9 % of the bandwidth compared to the bandwidth used

Scheme		Bandwidth (Kb/s)	Comparison
LLBE without advertisements		579.49	0.0 %
LLBE with advertisements as content		658.19	+13.6 %
Our System	Movie	559.14	
	Advertisement channel	48.12	
	Total	607.27	+4.4 %
	Amortized over 5 movies	568.76	-1.9 %

Table 1: Bandwidth required by different schemes to broadcast a movie. The comparison column shows for each scheme the increase of bandwidth

to broadcast a movie *without advertisements!*

6 Conclusion

Adding advertisements in Video-on-Demand broadcasting systems could increase by 13.6% the server bandwidth if the advertisements are not handled properly. By using the scheme suggested in this paper, it is even possible to reduce the server bandwidth as compared to the ad-less solution; especially when many movies share the same advertisement channel. We also proposed a faster algorithm to compute the optimal broadcasting schedule with or without advertisements.

7 Acknowledgments

We thank Timothy Chan for pointing out that the problem of finding staircases with the minimum area has the concave Monge property.

References

[1] C.C. Aggarwal, J.L. Wolf, and P.S. Yu A Permutation-Based Pyramid Broadcasting Scheme for Video-on-Demand Systems *Proceedings of the IEEE Int'l Conf. on Multimedia Computing and Systems*, 1996

[2] P. Basu, A. Narayanan, W. Ke, T.D.C. Little and A. Bestavros. Scheduling of Secondary Content for Aggregation in Commercial Video-on-Demand Systems. *Proceedings of the 8th International Conference on Computer Communications and Networks* (Oct 1999).

[3] T. Biedl, E. D. Demaine, A. Golynski, J. D. Horton, A. Lopez-Ortiz. G. Poirier and C-G. Quimper. Optimal Dynamic Video-On-Demand using Adaptive Broadcasting. *Proceedings of European Symposium on Algorithms*, 2003.

[4] L. Engebretsen and M. Sudan. Harmonic broadcasting for video-on-demand service. *Proc. 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp 431–432, 2002.

[5] Hsiang-Fu Yu, Hung-Chang Yang, Li-Ming Tseng, Yi-Ming Chen. Simple VBR staircase broadcasting (SVSB). *Consumer Communications and Networking Conference*, CCNC 2004. First IEEE, 2004.

[6] K Hua, S Sheu Exploiting Client Bandwidth for More Efficient Video Broadcast *Proceedings of IEEE ICCCN* '98, 1998.

[7] A. Hu "Video-on-Demand Broadcasting Protocols: A Comprehensive Study" *Proceedings of INFOCOM* 2001.

[8] A Hu, I. Nikolaidis, P. van Beek On the design of Efficient Video-on-Demand Broadcast Schedules *Proceedings of MASCOTS* 1999.

[9] L. Juhn and L. Tseng. Harmonic broadcasting for video-on-demand service. *IEEE Transactions on Broadcasting*, 43(3):268–271, 1997.

[10] L.-S. Juhn, L.-M. Tseng, Fast Data Broadcasting and Receiving Scheme for Popular Video Service *IEEE Transactions on Broadcasting*, 44(1), 1998.

[11] R. Krishnan Timeshared Video-on-Demand: A Workable Solution to VOD, *IEEE Multimedia* vol 6 no 1, 1999.

[12] R. Krishnan, D. Venkatesh and, T.D.C. Little, A failure and overload tolerance mechanism for continuous media servers. *Proceedings of the fifth ACM international conference on Multimedia,* pp 131–142, 1997.

[13] F. Li, I. Nikoalidis Trace-Adaptive Fragmentation for Periodic Broadcast of VBR Video *Proceedings of NOSSDAV* '99, 1999.

[14] F. Li, I. Nikoalidis Staircase Broadcast for Media-on-Demand Systems *Proceedings of MoMuC* 2000.

[15] I. Nikolaidis, F. Li, and A. Hu. An inherently loss-less and bandwidth-efficient periodic broadcast scheme for VBR video (poster session). *ACM SIGMETRICS Performance Evaluation Review vol 28, #1*, 2000.

[16] J.-F. Paris, S.W. Carter, and D.D.E. Long A Low Bandwidth Broadcasting Protocol for Video on Demand *Proceedings of the 7th International Conference on Computer Communications and Networks* 1998, pp. 690–697.

[17] J.-F. Paris, S.W. Carter, D.D.E. Long Efficient Broadcasting Protocols for Video on Demand *Proceedings of MASCOTS* '98, pp 127–132, 1998.

[18] J.-F. Paris, S.W. Carter, D.D.E. Long A Low Bandwidth Broadcasting Protocol for Video on Demand *Proceedings of IC3N* '98, pp 690–697, 1998.

[19] J.-F. Paris A Broadcasting Protocol for Compressed Video *Proceedings of the Euromedia 99 Conference*, pp 78-84, 1999.

[20] D. Saparilla, K. Ross, M. Reisslein Periodic Broadcasting with VBR-Encoded Video *Proceedings of IEEE Infocom* '99, pp. 464–471.

[21] B. Schieber Computing a minimum-weight k-link path in graphs with the concave Monge property, *Journal of Algorithms*, 29(2), pp 204–222, 1998.

[22] S. Viswanathan and T. Imielinski. Metropolitan area video-on-demand service using pyramid broadcasting. *Multimedia Systems*, 4(4), 1996.