First publ. in: IEEE Globecom (Global Internet Symposium), San Antonio, TX, USA, November 2001

Efficient Media-on-demand over multiple Multicast groups

Marcel Waldvogel
IBM Zurich Research Laboratory
Säumerstrasse 4 / Postfach
8803 Rüschlikon, Switzerland
mwl@zurich.ibm.com

Ramaprabhu Janakiraman
Applied Research Laboratory
Washington University in St. Louis
St. Louis, MO 63130-4899, USA
rama@arl.wustl.edu

***Abstract*— Using multicast for serving popular movies on demand reduces load on the server and the network by eliminating redundant packet transmission. To permit clients to arrive at times of their choosing, periodic rebroadcast is necessary. In addition, splitting the transmission over multiple multicast groups reduces the cost of rebroadcasting by allowing clients to unsubscribe from groups in which they are no longer interested.**

The focus of this paper is to develop techniques for efficient Media-on-Demand delivery to asynchronous clients over multiple multicast groups. We start by describing an existing periodic multicast technique that is near-optimal in terms of server bandwidth. Given a small number of groups α, we then show how to distribute content over these groups in a way that minimizes network impact. We present a theoretical analysis of the performance gains and compare these predictions with simulations over real and generated network topologies. We find that using even a small number of multicast groups provides significant reduction in overall network bandwidth.

I. Introduction

The promise of universal high-bandwidth Internet connectivity has sparked interest in bandwidth intensive Media-on-demand (MoD) systems. However, traditional MoD systems have tended to experience scalability problems, server bandwidth being the primary bottleneck. Given that a significant fraction of requests are for a few popular items [1], multicasting "hot" items help reduce the load on the server considerably. Multicasting, however, introduces complications of its own:

1) Since clients arrive at times of their choice, the server has to rebroadcast data periodically. The rebroadcast rate determines how long each receiver should wait before starting playout.
2) With rebroadcasts, at least some clients will receive the same data more than once.

Recently, *non-reactive* mechanisms have been proposed, in which the server periodically broadcasts the most popular videos without explicit requests from clients. These systems have the attractive property that server bandwidth is essentially independent of client demand, while allowing clients to choose both content and timing. They also do not require clients to announce their actions to the server, and are thus well suited for satellite distribution systems or legacy cable installations without a feedback channel.

In this paper, we describe how to split such a non-reactive movie transmission over multiple multicast groups in a way that minimizes the impact of periodic broadcast on the client and the network. We evaluate the performance of this approach and verify these results by simulation over real and generated network topologies.

A. Background

This work applies to a Media-on-demand (MoD) system, in which clients arrive at times of their choosing, request for movies, watch them from beginning to end and then leave. Movies consist of frames, which, for simplicity, can be assumed to be transmitted atomically in a single network packet. Time is defined in *instants*: it takes one instant to play out a single frame. Bandwidth is defined in terms of *frames per instant*. Initial delay is assumed to be uniformly w instants for all receivers. Thus, each receiver takes $n + w$ instants to consume a movie with n frames.

It is usual but not necessary that our "frames" map to actual video frames. For example, for a pure audio stream, frames are simply conveniently-sized blocks of sequential audio data.

B. Related work

One of the earlier periodic broadcasting schemes was Pyramid Broadcasting (PB) [2], which divides movies into segments of geometrically increasing sizes and broadcasts them on a few high bandwidth streams. A derivative, Skyscraper Broadcasting [3] offers much better performance. *Harmonic Broadcasting* based protocols, [4, 5], split a movie into fixed-size segments and broadcast these segments in streams of harmonically decreasing bandwidth. Sen et. al. [6] derive lower bounds on the performance of non-reactive MoD systems.

To the extent of our knowledge, there has not been any work on quantifying the network impact of these MoD protocols or on optimally distributing content among multiple multicast groups. Bhattacharyya et. al. [7] discuss optimal scheduling of data packets among multiple multicast layers, but their work has greater relevance to layered multicast [8], where the same

Work done at Washington University

Listing 1 Algorithm BASIC

```
for (f ← 1; f ≤ n; f ← f+1) {
   for (t ← f+w; t ≤ t_max; t ← t+f+w) {
      transmit(f, t);
   }
}
```

content at varying qualities is sent in different layers to clients with identical join times.

C. Optimal Transmission Scheme

Consider the broadcast of a "popular" movie that consists of n frames. Assume the frames are to be broadcast in a way that satisfies the on-demand requirements of clients with different join-times. Now, a client with a join-time of t and a wait-time of w will require frame f at time t_f, no later than time $t+w+f$, i.e., $t \leq t_f \leq t+w+f$. For this to hold true for all t, it is necessary and sufficient that there is at least one transmission of frame f in every stretch of $w+f$ instants. The optimal way to do this is to broadcast frame f every $w+f$ instants.

Listing 1 shows the *transmit* function to implement this optimal scheme: It schedules f for transmission during time t in a transmission queue. The bandwidth usage of algorithm BASIC (1) at both server and client ends is:

$$B = \sum_{f=1}^{n} \frac{1}{w+f} \approx \log \frac{n+w}{w} \tag{1}$$

Earlier work [9] has determined that this is optimal for any broadcast based scheme.

Although Algorithm BASIC is simple, elegant and optimal, things are not so simple in practice. Frame schedules add up combinatorially to give extremely spiky bandwidth usage. This can be circumvented by changing the schedules slightly around bandwidth peaks at the cost of a slight increase in average bandwidth. For brevity, we do not discuss this further here; more details may be found in [10]. For our purposes, it is sufficient to assume that for a movie of n frames, transmitting each frame f_i every $w+i$ instants guarantees a maximum delay of w instants for each client, while using a fixed bandwidth of $\log \frac{n+w}{w}$ for the server.

II. Using multiple groups

A significant problem with the optimal scheme described in Section I-C is that every client continues to receive redundant transmissions, wasting client and network bandwidth. While this is unavoidable in a pure broadcast-based system without a feedback channel (e.g., using satellite transmission), it is wasteful in a multicast situation where there is network support for subscribing to and unsubscribing from a multicast session (e.g., IP multicast in the Internet). It is therefore desirable that each client explicitly deregister its interest in unwanted frames with the multicast infrastructure. The ideal solution of transmitting each frame in its own multicast group, to which a receiver would subscribe until it has received that particular frame, creates unacceptably high network overhead in the form of subscription/unsubscription messages and state information in routers.

In practice, a compromise is effected by splitting content over a manageable number of multicast groups. Each client subscribes to all the groups on startup, shedding groups one by one as it is no longer interested in receiving data it already has received or played out. This is somewhat similar to the layered multicast approach used for non-reactive congestion control [8]. More recently, efficient scheduling in layered multicast is discussed in [7, 11].

Our problem statement is simple: Given a movie of n frames, how do we optimally split these frames over α multicast groups? Increasing the number of frames in a group increases cost in two ways:

1) At any given time, more clients are subscribed to it. Even for network-supported multicast mechanisms, this consumes more network resources, albeit sub-linearly.
2) Those clients are forced to listen to redundant transmissions longer.

On the other hand, since the number of groups, α, is finite, dropping too early and too often would mean running out of groups sooner. As we shall see, there is an optimal solution.

A. Optimal splitting into multicast groups

Our first goal is to find the splitting that minimize the number of redundant frames that each client receives. Consider putting group boundaries such that group number k will be subscribed to for x_k instants and will contain frequencies $\frac{1}{x_{k-1}+1} \cdots \frac{1}{x_k}$ ($x_0 = w, x_\alpha = n+w$). The total number of frames that a client receives on an average is then given by:

$$\begin{aligned} F = {} & x_1\left(\frac{1}{x_0+1} + \frac{1}{x_0+2} + \cdots + \frac{1}{x_1}\right) \\ & + x_2\left(\frac{1}{x_1+1} + \frac{1}{x_1+2} + \cdots + \frac{1}{x_2}\right) + \cdots \\ & + x_\alpha\left(\frac{1}{x_{\alpha-1}+1} + \frac{1}{x_{\alpha-1}+2} + \cdots + \frac{1}{x_\alpha}\right) \end{aligned} \tag{2}$$

In other words,

$$F \approx \sum_{k=1}^{\alpha} x_k \log \frac{x_k}{x_{k-1}} \tag{3}$$

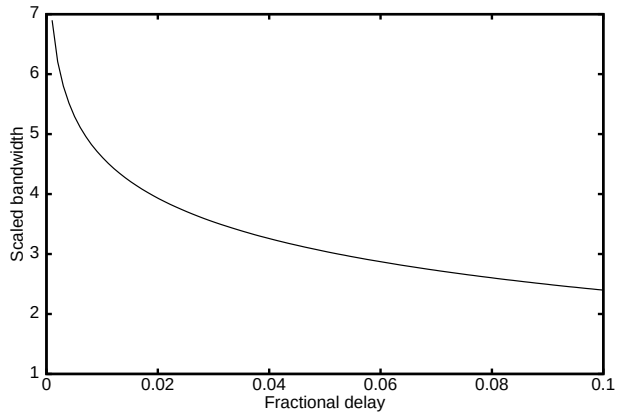
Differentiating partially w.r.t. x_k and equating to 0 for minimum F, we get

$$x_{k+1} = x_k\left(1 + \log \frac{x_k}{x_{k-1}}\right) \tag{4}$$

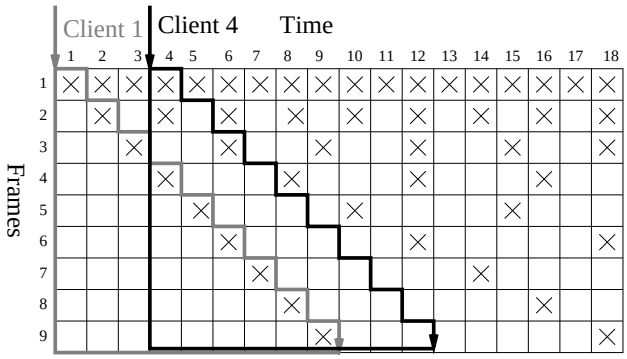
Descending recursively, the first boundary x_1 is determined by,

$$x_\alpha = x_1 \underbrace{\left(1 + \log \frac{x_1}{x_0}\right)\left(1 + \log\left(1 + \log \frac{x_1}{x_0}\right)\right) \cdots}_{\alpha \text{ terms}} \tag{5}$$

where $x_\alpha = n+w$ and $x_0 = w$. This equation can be numerically solved (we use Newton-Raphson iteration) to get the value



(a) Bandwidth vs. Delay



(b) Basic Transmission Pattern

Fig. 1. Optimal Transmission

TABLE I
CLIENT-EFFICIENT DROP TIMES

	Drop time for group # (minutes)				
α	1	2	3	4	5
2	14:30	60:36			
3	7:34	26:46	60:36		
4	5:06	16:01	34:22	60:36	
5	3:52	11:07	22:51	39:17	60:36

of x_1. Once x_1 is found, higher boundaries can be calculated using (4). This set of boundaries is the one that minimizes the number of frames that each client receives.

For example for a one-hour, 25 frames-per-second (fps) movie, $\alpha = 3$ and $w = 36$ seconds (1%), the optimal group boundaries are at 7:34 minutes and 26:46 minutes, reducing average client bandwidth usage to 42 fps from 115 fps without layering. For the same specifications, Table I lists drop times for different values of α.

We define the metric for performance, *Receiver inefficiency* as

$$I_R = \frac{\text{\# of frames received on average}}{\text{\# of frames in movie}} = \frac{1}{n}\sum_{k=1}^{\alpha} x_k \log\frac{x_k}{x_{k-1}}$$

Figure 2(a) shows receiver inefficiency as a function of α for various values of initial delay. In the ideal case when each frame is transmitted in its own group, receiver inefficiency is exactly 1, so it is apparent that using even a small number of groups (around 3) results in significantly fewer redundant frames received by each client.

B. Minimizing Network cost

A connected problem is that of splitting n frames into α groups such that instead of optimizing client bandwidth, the overall *network* bandwidth is minimized. That is, we would like to minimize the total number of frames sent out over the network's links at any time. If the number of links in a delivery tree of m clients is $L(m)$ and the average client arrival rate is λ then the number of clients subscribed to group k at any given time is $\approx \lambda x_k$. Therefore, the sum of frames seen by all links in the tree at any given time is:

$$N \approx \sum_{k=1}^{\alpha} L(\lambda x_k) \log\frac{x_k}{x_{k-1}} \tag{6}$$

In 1998, Chuang and Sirbu [12] made the remarkable discovery that for Internet multicast, $L(m)$ is fairly accurately approximated by a power law of the form, $L(m) \approx \hat{u}m^{\rho}$ where $\rho \approx 0.8$ and $\hat{u}$ is the average unicast path length. This represents the network bandwidth advantage over multiple unicast, which has $L(m) = \hat{u}m$. This was subsequently verified by Phillips et. al. [13]

The function to be minimized, then, is:

$$N \approx K \times \sum_{k=1}^{\alpha} x_k^{\rho} \log\frac{x_k}{x_{k-1}} \tag{7}$$

where $K = \hat{u}\lambda^{\rho}$ is a constant for a given transmission.[1] Proceeding as in Section II-A, the analogous result of (4) is:

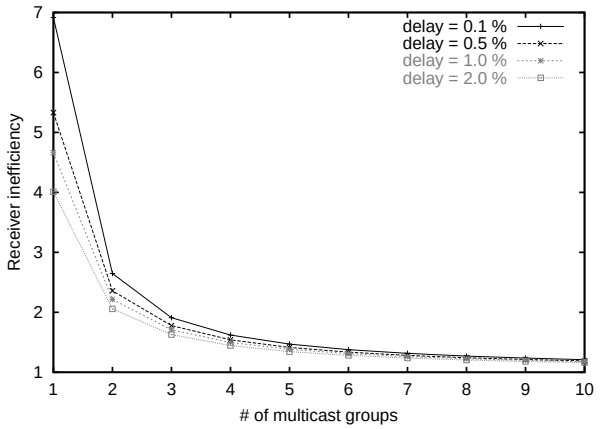
$$x_{k+1} = x_k(1 + \rho \log\frac{x_k}{x_{k-1}})^{\frac{1}{\rho}} \tag{8}$$

Thus, for the network-optimal case,

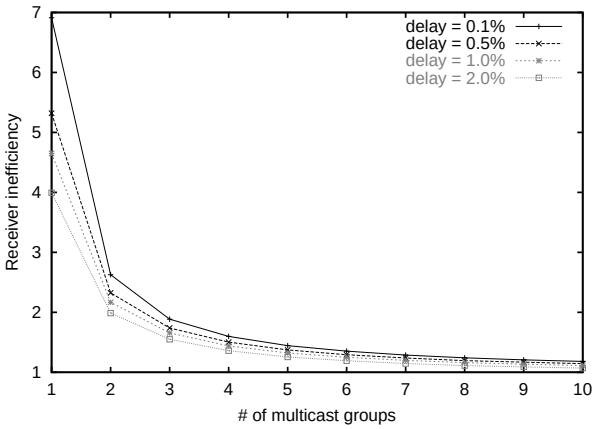
$$x_{\alpha} = x_1 \underbrace{(1 + \rho\log\frac{x_1}{x_0})^{\frac{1}{\rho}}\,(1 + \log(1 + \rho\log\frac{x_1}{x_0}))^{\frac{1}{\rho}}\,\cdots}_{\alpha \text{ terms}} \tag{9}$$

For the same parameters as in Section II-A, optimizing network bandwidth results in group boundaries at 6:27 minutes

[1] For now, we assume constant λ. Variable λ will be discussed in the next section.



(a) Predicted: I_R vs. α



(b) Simulated: I_R vs. α

Fig. 2. Client Inefficiency with multiple groups

TABLE II
NETWORK EFFICIENT DROP TIMES

	Drop time for group # (minutes)				
α	1	2	3	4	5
2	12:52	60:36			
3	6:27	24:28	60:36		
4	4:16	13:56	32:02	60:36	
5	3:14	9:24	20:20	37:06	60:36

and 24:28 minutes for a network bandwidth usage $\approx 42\%$ of that without layering. Table II lists drop times for various values of α.

Now, consider the ideal situation in which each frame is multicast in its own group. In this case, the expected number of receivers waiting for frame f_k would be $\lambda \times (w+k)$. The network bandwidth in this case is simply $\sum_{k=1}^{n} \hat{u}(\lambda(w+k))^{\rho} \frac{1}{w+k}$ or simply $K \sum_{k=1}^{n} (w+k)^{\rho-1}$. Accordingly, let us define network inefficiency as

$$I_N = \frac{\text{\# of frames in network at any time}}{\text{minimum \# of frames in network}} = \frac{\sum_{k=1}^{\alpha} x_k^{\rho} \log \frac{x_k}{x_{k-1}}}{\sum_{k=1}^{n} (w+k)^{\rho-1}}$$

Figure 3(a) shows network inefficiency versus α for various w.

C. Variable arrival rate

In Sections II-A and II-B, we have assumed that clients arrivals are distributed uniformly. Specifically, we assumed that with an arrival rate λ, the number of clients in time t would be $\lambda \times t$. But to be more realistic, client arrivals follow a distribution centered around a mean λ. For Poisson-distributed arrivals, the equivalent of Equation 7 would be:

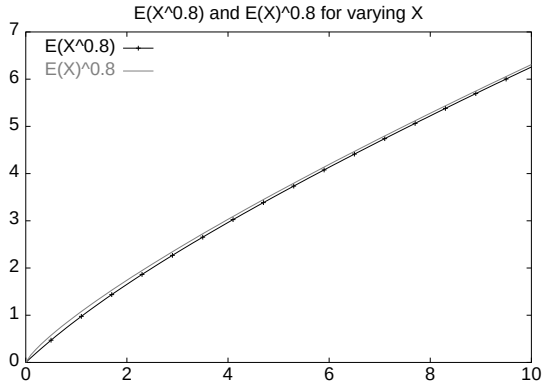


Fig. 4. Comparing $E(X^{\rho})$ with $E(X)^{\rho}$

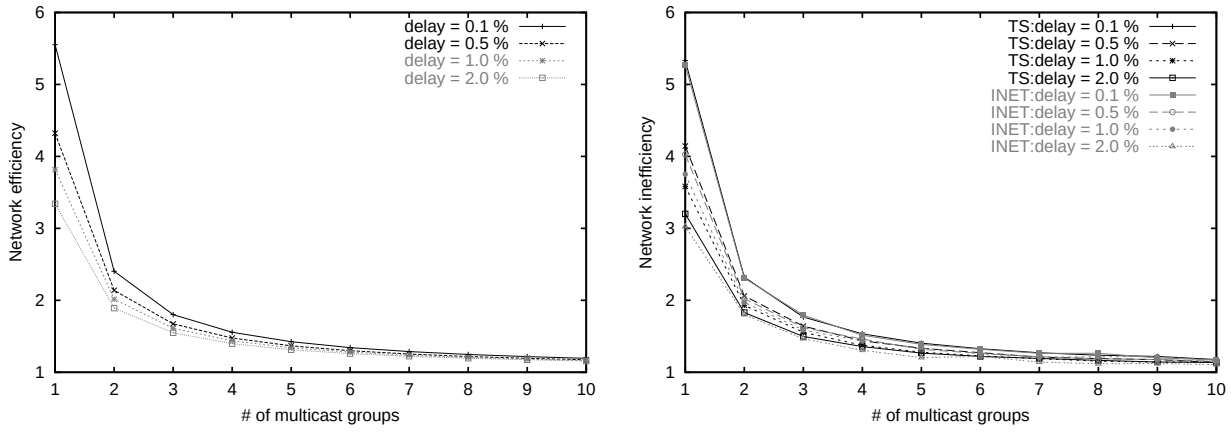
$$N \approx K' \times \sum_{k=1}^{\alpha} E(X^{\rho}, \lambda x_k) \log \frac{x_k}{x_{k-1}} \tag{10}$$

where $E(f(X), \lambda)$ is $\sum_{k=0}^{\infty} \frac{e^{-\lambda} \lambda^k f(k)}{k!}$.

That is, in (7), we have implicitly assumed $E(X^{\rho}, \lambda x_k) \equiv E(X, \lambda x_k)^{\rho}$, which is not true in general. However as Figure 4 shows that as X exceeds 1, these two expressions converge rapidly. Since x_k is of the order of a few minutes and the content is popular, there is good justification for the approximation, at least for the Poisson case.

III. PERFORMANCE ANALYSIS

We now study the performance of these techniques in realistic situations. Our simulation setup is as follows: For the synthetic topology, we generate a transit-stub graph containing $\approx$ 10,000 nodes and 36,000 edges using the GT-ITM topology generator [14]. For the real network topology, we used the merged traces of the SCAN project [15] and the Internet mapping project [16] at Bell Labs. Rather than work with this huge database in its entirety, we have chosen to construct a subgraph



(a) Predicted: I_N vs. α

(b) Simulated: I_N vs. α

Fig. 3. Network Inefficiency with multiple groups

by doing a traversal with maximum depth 8 starting from an arbitrary node.[2]

Having generated a graph from this data, we pick a random source S (In the GT-ITM model, this is a stub node). Now for random nodes n_i and start times s_i, we choose m unique receivers $R_i = (n_i, s_i)$ with an average arrival rate of λ and construct the distribution tree. At random times t_j, we use the rule that for each multicast group k, R_i is subscribed at time t_j iff $t_j \geq s_i$ and $t_j \leq s_i + x_k$ to calculate L_{kj}, the distinct links involved in group k at time t_j. We average this to get an estimate $\hat{L}_k$ of $L_k = L(\lambda x_k)$. Now, overall network bandwidth can be estimated as $\sum_{k=1}^{\alpha} \hat{L}_k \log \frac{x_k}{x_{k-1}}$.

In Figure 3(b), we plot the performance predicted by (7) compared to values obtained by graph simulations. *ts10000* refers to the GT-ITM generated graph and *Inet* refers to the Internet trace. As the figure shows, there is good agreement between estimated and empirical values.

IV. Summary and Conclusions

Non-reactive periodic broadcast protocols are attractive from a server-bandwidth point of view, but can cause clients to receive redundant data. Splitting content over multiple groups reduces redundant transmissions by restricting multicast scope.

In this paper, we developed techniques to optimally transmit continuous media over multiple groups and quantified their effect from a receiver and network point of view. We found that using a small number of groups (< 10) resulted in performance within 10–20% of optimal. Subsequent increases in α do not have as much effect.

The Chuang-Sirbu law of multicast scaling was originally developed as a pricing mechanism for multicast on the Internet. We have used it to quantify and thereby minimize overall network bandwidth usage in a Media-on-demand system. From the close correspondence with graph based simulations, we are convinced this approach will find broad application.

[2]These traces, along with programs for their manipulation, may be found at `http://www.arl.wustl.edu/~rama/traces/`.

References

[1] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in *Proceedings ACM Multimedia '94*, Oct. 1994, pp. 391–398.

[2] S. Viswanathan and T. Imielinski, "Metropolitan area video-on-demand service using pyramid broadcasting," in *Multimedia Systems, Vol. 4*, 1996, pp. 197–208.

[3] K. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems," in *ACM SIGCOMM '97*, Sept. 1997.

[4] L. Juhn and L. Tseng, "Harmonic broadcasting for video-on-demand service," *IEEE Transactions on Broadcasting*, vol. 43, no. 3, pp. 268–271, Sept. 1997.

[5] J.-F. Pâris, S. W. Carter, and D. D. E. Long, "Efficient broadcasting protocols for video on demand," in *Proceedings 6th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, July 1998, pp. 127–132.

[6] S. Sen, L. Gao, and D. Towsley, "Frame-based periodic broadcast and fundamental resource tradeoffs," Tech. Rep. 99-78, University of Massachusetts, Amherst, 1999.

[7] S. Bhattacharyya, J. Kurose, D. Towsley, and R. Nagarajan, "Efficient rate-controlled bulk data transfer using multiple multicast groups," in *Proceedings of IEEE Infocom '98*, June 1998, pp. 1172–1179.

[8] V. Jacobson, S. McCanne, and M. Vetterli, "Receiver-driven layered multicast," in *Proceedings ACM SIGCOMM '96*, Aug. 1996, pp. 117–130.

[9] D. Eager, M. Vernon, and J. Zahorjan, "Minimizing bandwidth requirements for on-demand data delivery," in *Proceedings MIS '99*, Oct. 1999.

[10] Marcel Waldvogel, Wei Deng, and Ramaprabhu Janakiraman, "Media broadcasting for multiple asynchronous receivers," Tech. Rep. WUCS-01-02, Washington University in St. Louis, 2001.

[11] Jörg Nonnenmacher, Ernst W. Biersack, and Don Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Transactions on Networking*, vol. 6, no. 4, pp. 349–361, Aug. 1998.

[12] J. Chuang and M. Sirbu, "Pricing multicast communications: A cost based approach," in *Proceedings INET '98*, 1998.

[13] G. Phillips, H. Tangmunarunkit, and S. Shenker, "Scaling of multicast trees: Comments on the Chuang-Sirbu scaling law," in *ACM SIGCOMM '99*, 1999.

[14] Ellen W. Zegura, Kenneth L. Calvert, and Michael J. Donahoo, "A quantitative comparison of graph-based models for Internet topology," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 770–783, 1997.

[15] "The Mercator Internet mapping project," http://www.isi.edu/scan/mercator/maps.html.

[16] "The Internet Mapping project," http://cm.bell-labs.com/who/ches/map/index.html.