Network Bandwidth Requirements for Scalable On-Demand Streaming[•]

Yanping Zhao Derek L. Eager
Department of Computer Science
University of Saskatchewan
Saskatoon, SK S7N 5A9 Canada
{zhao,eager}@cs.usask.ca

Mary K. Vernon
Computer Sciences Department
University of Wisconsin – Madison
Madison, WI 53706-1685 USA
vernon@cs.wisc.edu

***Abstract*—Recently proposed streaming protocols are able to deliver multimedia files on-demand with required server bandwidth that grows only *logarithmically* with the file request rate. The same efficiencies are achieved for network bandwidth if delivery is over a true broadcast channel. This paper considers the required network bandwidth for on-demand streaming over multicast delivery trees. We consider both simple canonical delivery trees, and more complex cases in which delivery trees are constructed using both existing and new algorithms for various randomly generated network topologies and client site locations. Results in the paper quantify the potential savings from use of multicast trees that are configured to minimize network bandwidth rather than the latency to the content server. Further, we show that it is possible to simultaneously achieve reasonably close to the minimum possible network *and* server, bandwidth usage with a practical on-demand streaming protocol.**

I. Introduction

A key challenge in the current Internet is to provide scalable service and content delivery to potentially vast numbers of heterogeneous clients. Particularly challenging are content delivery applications that involve on-demand streaming of popular multimedia content, such as news clips, distance education content, television shows, or movies. Systems supporting such applications require scalable delivery protocols, effective caching architectures, methods for dealing with packet loss or route failure, and techniques for addressing various security issues.

Scalable delivery protocols for on-demand streaming [1,3,4,6,11-17,19-21,23,24] employ broadcast or multicast to reduce the required server bandwidth from *linear* in the request rate to *logarithmic* in the request rate (in the best protocols). The bandwidth reduction is accomplished by dynamically aggregating clients that make requests closely spaced in time, so that eventually the clients share the same multicast stream(s).

On a satellite channel or in some other true broadcast setting, the same reduction is achieved for network bandwidth. However, in the Internet or any fundamentally point-to-point network, the scalable protocols rely on multicast (either network or application-level) to achieve bandwidth savings. In this case, the achieved reduction in network bandwidth may differ from that for server bandwidth. To our knowledge, previous work has not quantified the network bandwidth savings for scalable streaming protocols in this case.

This paper investigates the reductions in required network bandwidth that are possible with scalable on-demand streaming protocols that use multicast delivery. The principal contributions of the results are:

- A tight bound on the *minimum* network bandwidth required for on-demand streaming for simple canonical multicast delivery trees. Appropriately combining the results for these canonical multicast delivery trees permits analysis for arbitrary delivery trees.
- The bounds, and experimental results, suggest that in practical cases the minimum required network bandwidth for on-demand streaming scales as $O(K/\ln(K))$ where K is the number of client sites and the request rate per site is kept fixed, and as $O(\ln N)$ where N is the total request rate and the number of client sites is kept fixed.
- A shared delivery tree from the server to all clients configured to minimize network bandwidth rather than latency only modestly reduces the minimum required network bandwidth for on-demand streaming (e.g., on average only by 3 – 16 % in the cases examined).
- It is possible to *simultaneously* achieve reasonably close to the minimum possible network, as well as server, bandwidth usage with a practical on-demand streaming protocol.

The remainder of the paper is organized as follows. Section II presents background on scalable on-demand streaming and multicast delivery. Tight bounds on the minimum network bandwidth required for on-demand streaming on simple multicast delivery trees are presented in Section III. Section IV applies these results to more complex cases with randomly generated network topologies and client site locations, and determines the potential bandwidth savings from use of alternative algorithms for constructing multicast delivery trees. Section V considers the question of whether practical on-demand streaming protocols can achieve close to the minimum required network bandwidth, and whether it is possible to achieve close to minimal network and server bandwidth usage simultaneously. Conclusions are given in Section VI.

II. Background

A. Scalable On-Demand Streaming Protocols

Ideally, an on-demand streaming protocol should (1) allow each client to begin playout with minimal delay, (2) support interactive requests such as skip ahead/back and fast forward, and (3) provide *scalable* service such that the bandwidth required to deliver a media file grows only slowly with the file request rate.

[•] This work was partially supported by the Natural Sciences and Engineering Research Council of Canada under Grant OGP-0000264 and by the National Science Foundation under Grants CCR- 9975044 and ANI-0117810.

Periodic broadcast protocols [1,3,13,15,16,19-21,24] provide a partial solution by dividing each media file into segments, each of which is periodically broadcast or multicast on one of several "channels" (e.g., multicast groups or satellite/cable channels). When a client requests a particular file, the client obtains a schedule for tuning in to each channel, and a time to begin playout, such that each segment will be received by the time it is needed (assuming sequential playout by the client from the beginning). The schedule will in general require client receive bandwidth and buffering capacity for receiving multiple segments simultaneously. With clever design of the segment sizes and transmission rates, the required client start-up delay *decreases exponentially* with increasing server bandwidth devoted to multicasting the segments.

Patching [4,6,14,17,23] and *bandwidth skimming* [11,12,20] protocols provide *immediate* service to each client request by initiating a new stream as each request is made. Bandwidth skimming has three main advantages over patching:

- It is more efficient than patching, requiring server bandwidth that scales as the logarithm of the request rate versus the square root of the request rate as in patching.
- Patching requires clients to be able to receive at an aggregate transmission rate equal to *twice* the rate required for real-time playback. With bandwidth skimming, in contrast, the bit rate of the media file can consume most of the available bandwidth on the path to the client, thus allowing the client to receive the highest quality content possible. Only a small "skim" of the available transmission bandwidth is needed for the protocol mechanisms used to provide scalable delivery.
- Bandwidth skimming can easily and efficiently support general interactive requests.

Bandwidth skimming uses *hierarchical multicast stream merging* [12] to dynamically aggregate clients into larger and larger groups that share streams, together with a mechanism for using the bandwidth skim to effect each aggregation [11]. At least one substantive implementation of a media delivery system using bandwidth skimming has been carried out, namely the implementation in the eTeach system at the University of Wisconsin [20].

B. Minimum Required Server Bandwidth

A yardstick for evaluating scalable on-demand streaming protocols is provided by a tight lower bound on the required server bandwidth for *any* such protocol that provides immediate service. Considering delivery of a single media file that clients play sequentially from beginning to end, and using the notation in Table 1, this lower bound is given by [12]:

$$B^{\text{server}}_{\text{minimum}} = \int_0^T \frac{\mathrm{d}x}{x + 1/\lambda} = \ln(T\lambda + 1) = \ln(N + 1)\ . \quad (1)$$

A closely related bound can be derived for periodic broadcast schemes [3, 20].

The bound in expression (1) is derived assuming that request arrivals are Poisson, and is tight when there is no limit on the number or total aggregate rate of transmissions a client can receive concurrently. The Poisson assumption can be relaxed to cover a wide class of arrival processes including those with heavy-tailed interarrival time distributions, yielding a similar result with difference bounded by a constant independent of λ [12]. However, in this paper we restrict attention to Poisson request arrivals, as have been observed for example in [2].

Table 1: Notation

Symbol	Definition
λ	Average total request rate for a file
T	File playback duration
N	Average number of requests during period of length T ($N = \lambda T$)
λ_k	Average request rate at client site k
N_k	Average number of requests from site k during period of length T ($N_k = \lambda_k T$)
K	Number of client sites
$B^{\text{server}}_{\text{minimum}}$	Minimum required server bandwidth (in units of the file playback bit rate)
$B^{\text{network}}_{\text{minimum}}$	Minimum required network bandwidth cost (in units of the file playback bit rate times the average cost of a unit of end-to-end bandwidth)

The bound in (1) is derived by considering a small portion of the file at some arbitrary time offset x. For an arbitrary client request that arrives at time t, this portion of the file must be delivered no later than time $t+x$. If it is multicast as late as possible, i.e., at time $t+x$, then (at best) those clients that request the file between time t and $t+x$, can receive the same multicast. Since the average time from $t+x$ until the next request for the file is $1/\lambda$, the minimum frequency of multicasts of the portion at time offset x is $1/(x+1/\lambda)$, which yields the bound.

The analysis approach used for (1) is applied in Section III to tightly bound the minimum required *network* bandwidth using multicast delivery, for any on-demand streaming protocol providing immediate service.

C. Multicast Delivery Network Bandwidth

Issues concerning pricing policies for network bandwidth are orthogonal to and outside the scope of this paper. Here it is assumed simply that the cost of a unit of network bandwidth can be determined for each "link" that a stream traverses. A link may represent a single physical channel of some type, or a sequence of channels such as a route across an Internet Autonomous System or a route between two application-level multicast servers in an overlay network. For clarity and ease of obtaining insight, the formulas and experiments presented in this paper consider cases where the cost of a unit of bandwidth is identical for all links. The analyses can, however, be easily generalized for cases of varying unit bandwidth costs.

For a single multicast stream with a fixed multicast delivery tree (i.e., fixed set of clients), the total network bandwidth cost can be computed by taking the product of the stream rate, the stream duration, and the sum over the links of the delivery tree of the unit bandwidth cost. Since the cost of a unit of bandwidth is identical for all links, the bandwidth cost for a given stream is proportional to the number of links in the delivery tree. If the multicast tree is a shortest path tree, that is, it is composed of the shortest paths between the source and each receiver, previous work [9,22] suggests that the number of links in such a tree scales approximately as $m^{0.8}$ where m denotes the number of receivers. Although shortest path trees minimize the average delay from the source to each receiver, they do not in general minimize the number of links in the tree (i.e., the network bandwidth cost). Wei and Estrin [26] compare shortest path trees, approximate Steiner minimal trees, and center-based trees, with respect to delay, bandwidth cost, and traffic concentration. (The latter is an issue when a multicast group has multiple traffic sources.) Approximate Steiner minimal trees are found to moderately reduce bandwidth cost, with reductions of at most 40% in the cases examined.

For on-demand streaming, the network bandwidth cost is more complex. In particular, the number of streams used in delivery of a given media file varies over time, depending on the pattern of client requests. Furthermore, each stream may be shared by a different, time-varying subset of the clients who are receiving the file.

If the multicast distribution tree associated with each stream is established independently of those for other streams, approximate Steiner minimal trees are difficult to implement due to the time-varying set of clients that join the stream. Alternatively, all streams might share a single delivery tree, with each stream using a subset of the links depending on which clients are currently listening to the stream. This latter approach might be used with an application-level multicast implementation or "overlay network" (e.g., [7,8]). The client sites might in this case correspond to application-level multicast servers, each of which is responsible for forwarding content to its own community of end users. For the most part, this paper focuses on the case where all streams share a single delivery tree, although the results in Section III regarding minimum network bandwidth requirements, and the network bandwidth usage with shortest path trees in Section IV, provide insight that is applicable to the case of independently established distribution trees.

For on-demand streaming with a single shared delivery tree, the network bandwidth cost is *not* directly determined by the number of links (as each stream does not traverse all links), but rather is a function of both tree topology and request rates. The issue of delivery tree topology in this context is therefore somewhat different from in the single stream context. For example, Steiner trees are not necessarily optimal. In fact, for very low request rates, shortest path trees become optimal (assuming that path "length" corresponds to the total cost of a unit of bandwidth on the path) since there is very little stream sharing in this case, and therefore an optimal tree is the one that individually minimizes the cost of delivery to each client site.

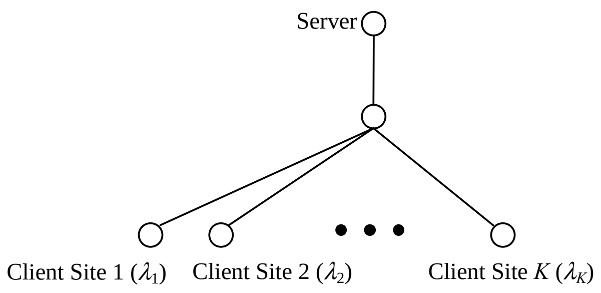

Server
Client Site 1 (λ_1)
Client Site 2 (λ_2)
Client Site K (λ_K)

Figure 1: Shared Link with Fan-Out *K* Topology

III. Analysis For Minimum Network Bandwidth

This section considers the minimum required network bandwidth for on-demand streaming on three simple multicast delivery tree topologies, as a function of the number of client sites and the media file request rate at each. These simple topologies can be combined to form arbitrary topologies, and so the analysis developed in this section can be used to analyze general delivery trees. The particular topologies considered here also capture the extreme points with respect to how network bandwidth requirements scale with the number of client sites, as well as the type of scaling behavior that might be expected in practice.

For each topology, the required network bandwidth is measured in units of the file playback bit rate times the average end-to-end unit bandwidth cost.

A. Shared Link with Fan-Out K Topology

The delivery tree topology shown in Figure 1 consists of a single shared link plus K other links that each serves a distinct client site. This topology (as well as the topology obtained by omitting the shared link) represents a worst case with respect to how network bandwidth scales with the number of client sites. For fixed request rate at each client site, network bandwidth is $O(K)$.

A lower bound on the minimum required network bandwidth for on-demand streaming can be derived by considering each link in isolation, yielding the following result:

$$B_{\text{minimum}}^{\text{network}} > \frac{1}{2}\left(\ln(N+1) + \sum_{k=1}^{K} \ln(N_k + 1) \right). \qquad (2)$$

This bound simply sums the expression in (1), as independently computed for each link using the known rate of requests for content that traverses the link. The cost of a unit of bandwidth on each link is assumed identical, and the required network bandwidth is measured in units of the file playback bit rate times the end-to-end unit bandwidth cost (in this case, equal to twice the unit cost on each link).

Note that the above bound is not achievable. In other words, the shared transmissions for clients from different sites required to achieve the lower bound on the shared link sometimes conflict with the shared transmissions required of clients within a given site to minimize the bandwidth usage on the unshared (or dedicated) link to that site.

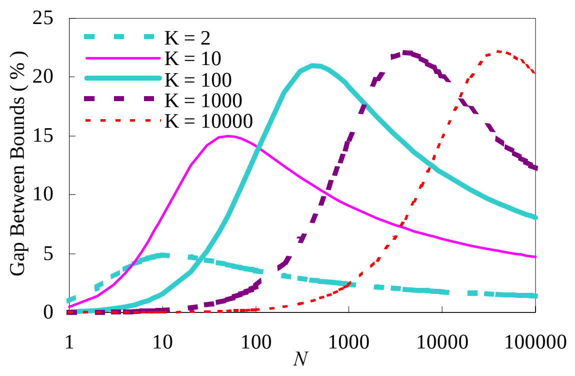


Figure 2: Tightness of Minimum Network Bandwidth Bounds for Shared Link with Fan-Out *K* Topology

An upper bound on the minimum required network bandwidth for on-demand streaming can be derived through analysis of the policy that minimizes the bandwidth usage on the shared link, at the cost of somewhat less efficient bandwidth usage on the dedicated links. Specifically, a multicast on the shared link of the portion at position x is delivered to client site k whenever k is the site whose request triggered that multicast, or if some client at k requests the file during the period of length x from the time of the triggering request, until the time of the multicast. This yields

$$B^{\text{network}}_{\text{minimum}} < \frac{1}{2}\Big(\ln(N+1) + \sum_{k=1}^{K}\int_0^T \frac{1}{x+1/\lambda}\big(\frac{\lambda_k}{\lambda}+\frac{\lambda-\lambda_k}{\lambda}(1-e^{-\lambda_k x})\big)dx\Big)$$

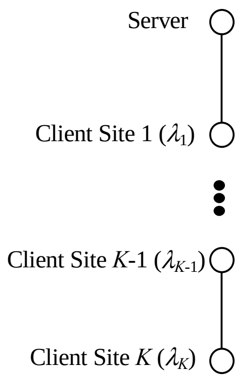
which can be rewritten as

$$B^{\text{network}}_{\text{minimum}} < \frac{1}{2}\Big((K+1)\ln(N+1) - \sum_{k=1}^{K}\frac{\lambda-\lambda_k}{\lambda}\int_0^T \frac{1}{x+1/\lambda}e^{-\lambda_k x}dx\Big). \qquad (3)$$

Figure 2 plots the percent difference between the bounds given in relations (2) and (3), relative to the lower bound, for the case of equal request rates ($\lambda_k = \lambda/K$ for all k).[1] As can be seen, the upper and lower bounds on minimum required network bandwidth are quite tight, particularly for $K < 10$.

The minimum network bandwidth for on-demand streaming using this topology, measured by the lower bound given in relation (2), is shown in Figure 7(a) as a function of K (for fixed request rate per client site) and in Figure 7(b) as a function of N (for fixed K), assuming equal request rate at each client site in both cases. As noted above, for fixed request rate per client site (Figure 7(a)), the minimum network bandwidth is $O(K)$. For fixed K, however, the minimum network bandwidth scales only as ln(N), as seen in Figure 7(b).

[1] In the case of relation (3) and others where a closed form has not been derived, numerical results are obtained using Maple [25].



Server
Client Site 1 (λ_1)
Client Site K-1 (λ_{K-1})
Client Site K (λ_K)

Figure 3: Daisy-Chain Topology

B. *Daisy-Chain Topology*

Figure 3 shows a multicast delivery topology in which K links provide a daisy-chain linking K client sites to the server.

Bounds on the minimum required network bandwidth for on-demand streaming on this topology can be derived in an analogous fashion as for the shared link with fan-out K topology, yielding:

$$B^{\text{network}}_{\text{minimum}} > \frac{2}{K+1}\sum_{k=1}^{K}\ln\left(\sum_{j=k}^{K}N_j+1\right) \qquad (4)$$

and

$$B^{\text{network}}_{\text{minimum}} < \frac{2}{K+1}\Big(K\ln(N+1) - \sum_{k=2}^{K}\frac{\lambda-\sum_{j=k}^{K}\lambda_j}{\lambda}\int_0^T \frac{1}{x+1/\lambda}e^{-\sum_{j=k}^{K}\lambda_j x}dx\Big). \qquad (5)$$

Similarly as for relations (2) and (3), the cost of a unit of bandwidth on each link is assumed identical, and the required network bandwidth is measured in units of the file playback bit rate times the average end-to-end unit bandwidth cost (in this case, equal to (K+1)/2 times the unit cost on each link).

Figure 4 graphs the percent difference between the bounds given in relations (4) and (5) relative to the lower bound, for the case of equal request rates per client site ($\lambda_k = \lambda/K$ for all k). The bounds are very tight with the percent difference under 7% in all cases.

The minimum network bandwidth for on-demand streaming on the daisy-chain topology, as measured by the lower bound given in relation (4), is shown in Figure 7. This topology represents a best case with respect to how network bandwidth scales with the number of client sites. For fixed request rate at each client site and scaling the number of sites K (Figure 7(a)), relations (4) and (5) can be shown to imply that the minimum network bandwidth is O(ln(K)). Note that this is the best possible scaling behavior, since from (1) network bandwidth must be at least O(ln(N)), and N scales linearly with K when the per-site request rate is kept fixed. As illustrated in Figure 7(b), for fixed number of client sites and varying N, the minimum network bandwidth is O(ln(N)).

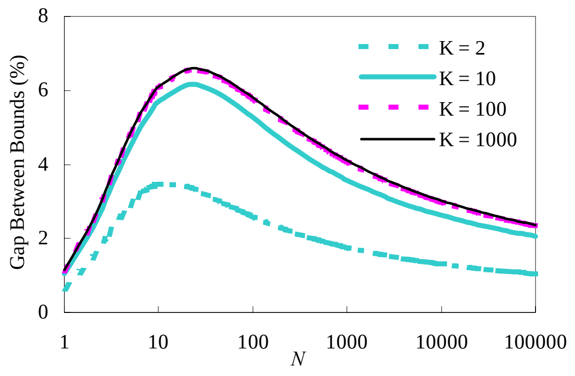


Figure 4: Tightness of Minimum Network Bandwidth Bounds for Daisy-Chain Topology

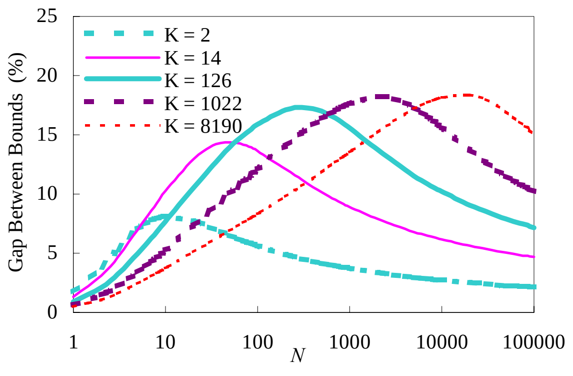


Figure 6: Tightness of Minimum Network Bandwidth Bounds for Balanced Tree Topology

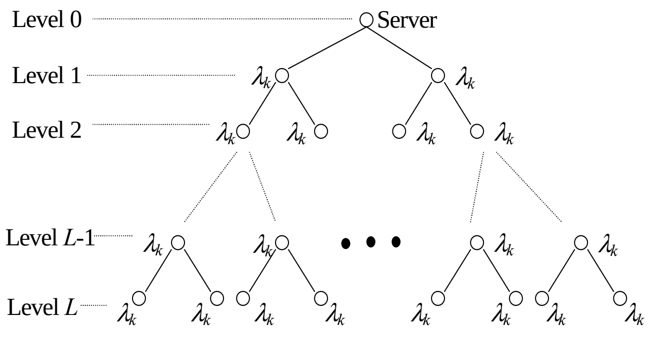


Figure 5: Balanced Tree Topology

C. *Balanced Tree Topology*

Figure 5 depicts a balanced tree topology in which all client sites have the same request rate. In this case the tree is a binary tree of depth L, with the number of client sites K equal to $2^{L+1}-2$. Qualitatively similar results are obtained for balanced m-ary trees, and for trees in which only the leaf nodes correspond to client sites.

Bounds on the minimum required network bandwidth for on-demand streaming on this topology can be derived in a similar fashion as for the previous topologies, yielding:

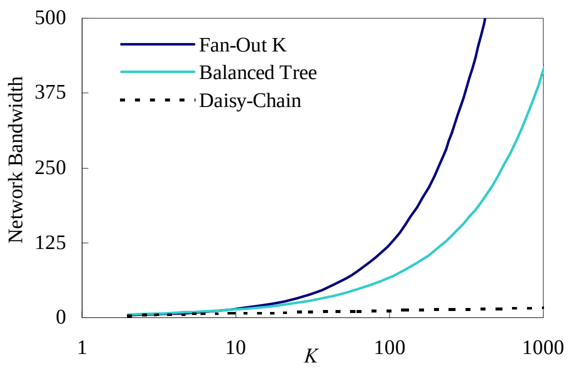
$$B_{\text{minimum}}^{\text{network}} > \frac{2^L-1}{2^L(L-1)+1}\sum_{i=1}^{L} 2^i \ln\left(\frac{2^{L-i+1}-1}{K}N+1\right) \qquad (6)$$

and

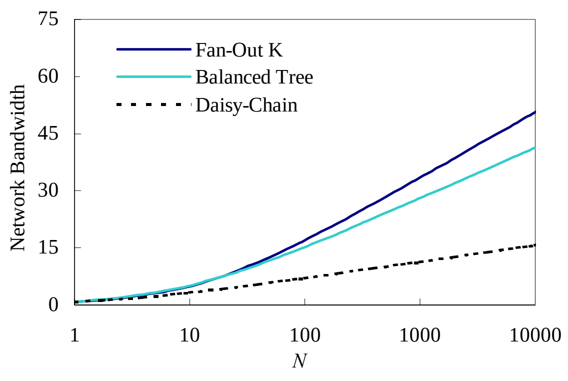
$$B_{\text{minimum}}^{\text{network}} < \frac{2^L-1}{2^L(L-1)+1}\Bigg(K\ln(N+1) -$$

$$\sum_{i=1}^{L} 2^i \frac{K-(2^{L-i+1}-1)}{K}\int_0^T \frac{1}{x+1/\lambda} e^{-\frac{2^{L-i+1}-1}{K}\lambda x}\,\mathrm{d}x\Bigg). \qquad (7)$$

As for the previous bounds, the cost of a unit of bandwidth on each link is assumed identical, and the required network bandwidth is measured in units of the file playback bit rate times the average end-to-end unit bandwidth cost (in this case, equal to $(2^L(L-1)+1)/(2^L-1)$ times the unit cost on each link).



(a) As a Function of *K* (N_k = 10)



(b) As a Function of *N* (K = 14)

Figure 7: Scaling of the Minimum Network Bandwidth Required for On-Demand Streaming

Figure 6 shows that the bounds given in relations (6) and (7) are quite tight, with percent difference relative to the lower bound under 20% for all system configurations.

The minimum network bandwidth for on-demand streaming on the balanced tree topology, as measured by the lower bound given in relation (6), is shown in Figure 7. For fixed request rate at each client site and scaling the number of sites K (Figure 7(a)), relations (6) and (7) can be shown to imply that the minimum network bandwidth is $O(K/\ln(K))$. Results for randomly generated network topologies and client site locations, as presented in the next section, suggest that this scaling behavior is representative of what might be expected in practice. As before, for fixed number of client sites and varying N, the minimum network bandwidth is $O(\ln(N))$.

IV. Multicast Delivery In Large Networks

The results of the previous section show that a wide range of behaviors are possible with respect to how the minimum required network bandwidth for on-demand streaming scales with the number of client sites. A key question is what type of scaling behavior is likely to be seen in practice.

A related issue concerns the algorithms used to build multicast delivery trees. The results of the previous section show that substantial network bandwidth savings are potentially achieved from daisy-chaining client sites, in contrast to using high fan-out delivery trees with few shared links. In practice, however, the disparate locations of client sites may make daisy-chaining inefficient. Achieving an efficient delivery tree requires a compromise between the goals of sharing as many links as possible, and minimizing the average cost of a unit of end-to-end bandwidth (e.g., the average distance in number of links from the server to each client site, within the chosen delivery tree). It is unclear *a priori* as to whether alternative delivery tree structures can yield significantly lower network bandwidth usage, in this context of on-demand streaming, than the shortest path delivery trees commonly used in multicast applications.

This section addresses these questions through experiments on randomly generated network topologies and client site locations. The topologies are generated using GT-ITM (Waxman and Transit-Stub models) [5,27] and Inet-2.1 [18]. In total, experiments were run on more than 100 network topologies.

A. Comparison of Tree Construction Algorithms

Five algorithms for constructing multicast delivery trees are compared. Of interest is the network bandwidth requirement of the resulting delivery trees in the on-demand streaming context. Implementation considerations are largely neglected, as the objective is to determine the *potential* performance benefits of alternative delivery tree structures. Also, in the application-level multicast setting, there may be considerable flexibility with respect to how delivery trees are constructed, and a broad range of feasible algorithms. Note, however, that the algorithms that are considered below do not restrict delivery tree branching points to just the client/server sites, and thus would require a greater number of application-level multicast servers.

As noted in Section II, for clarity it is assumed that the cost of a unit of bandwidth is identical for all links. Generalizations of the algorithms for the case of weighted links, with the weight representing the cost of a unit of bandwidth, are straightforward to formulate.

The five algorithms considered here are as follows:

SP (Shortest Path): A delivery tree is constructed by taking the union over all client sites of a shortest path from the client site to the server. Note that in the topologies considered here, shortest paths are the same as reverse shortest paths (i.e., from client to server). This is a common approach for multicast delivery.

GL-A (Greedy Link – All): Client sites are added incrementally to the delivery tree structure. The client site closest to the server is added first. On each subsequent step, the client site that can be joined to the delivery tree with the fewest number of new links is added, together with those links. This approach attempts to minimize the number of links in the delivery tree.

GL-P (Greedy Link – Participants): Like GL-A, except it considers the number of links on the shortest path to an existing client site (or the server itself). The client site that is closest by this measure to an existing site is added next to the delivery tree structure, together with those links that are part of its shortest path that were not already part of the tree.

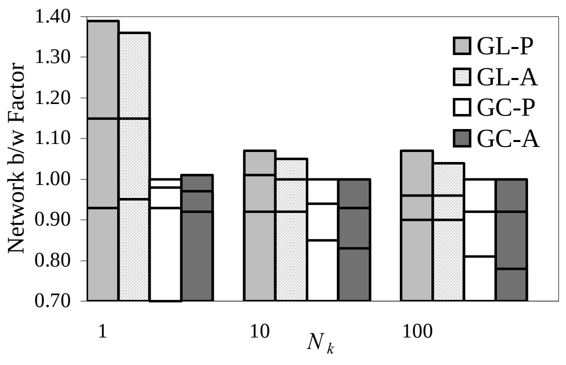
GC-A (Greedy Cost – All): Like GL-A, except that rather than considering the number of new links when choosing which client site to add next (and where to connect that client site), the algorithm considers the incremental network bandwidth cost. The latter is determined using the analysis techniques developed in the previous section (specifically, for lower bounds[2] on the minimum required network bandwidth). Unlike the previous algorithms, GC-A has the disadvantage that it requires knowledge of client site request rates. However, it may be better able to achieve the best compromise between the goals of sharing as many links as possible, and minimizing the average number of delivery tree links on the path from the server to each client site.

GC-P (Greedy Cost – Participants): Like GC-A, except that the algorithm only tries to connect new client sites through a shortest path to an existing client site (or the server itself). At each step, the new client site and shortest path to some existing site, that yields the minimum incremental bandwidth cost, is merged into the delivery tree structure.

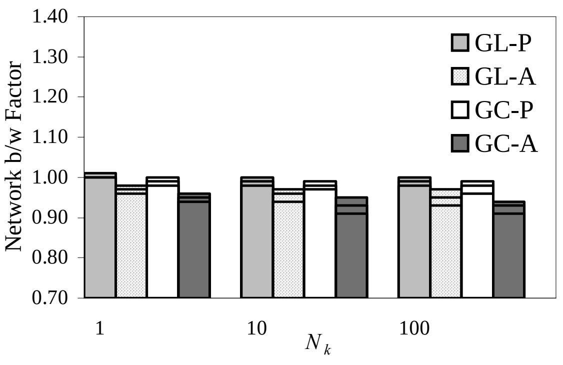
Figure 8 gives experimental results comparing the above five algorithms on network topologies generated using GT-ITM and Inet-2.1. The networks generated are of two sizes: 600 nodes (GT-ITM) and 6000 nodes (Inet-2.1). For each network size, 10 networks are generated using different random number seeds (for GT-ITM, 5 of these use the Waxman model and 5 use the Transit-Stub model). For each of these networks, the number of client sites is chosen as 2%, 10%, or 50% of the total number of nodes. The server location is chosen as one of the nodes with maximum degree, and the appropriate number of client sites are randomly distributed among the remaining nodes. Three per-site request rates are considered (N_k=1, N_k=10, and N_k=100). Each bar in Figure 8 shows the minimum, average, and maximum required network bandwidth for on-demand streaming using the delivery tree constructed with the indicated algorithm, relative to that constructed using SP, over the 10 corresponding networks. Required network bandwidth is measured using the lower bound analysis techniques for minimum network bandwidth from Section III. For the 6000 node networks with 3000 client sites, results for GC-A are not shown owing to the computational expense of this algorithm.

The primary conclusions from this figure are as follows:

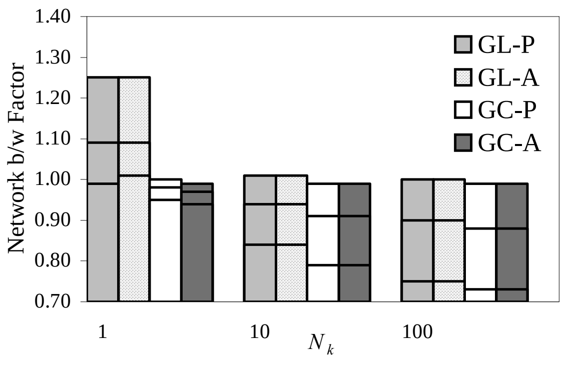
[2] The upper bound analyses could be applied instead, or averages of the upper and lower bounds, but the lower bounds have lower computational cost, and in any case the gap between the bounds is reasonably small.



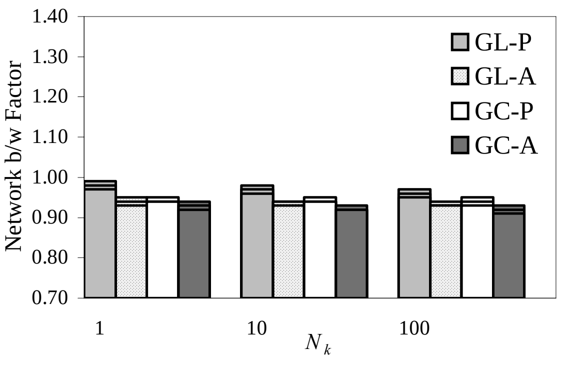
(a) GT-ITM (600 nodes, 2% client sites)



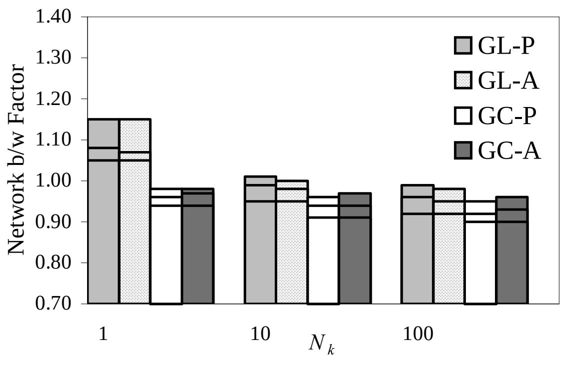
(b) Inet-2.1 (6000 nodes, 2% client sites)



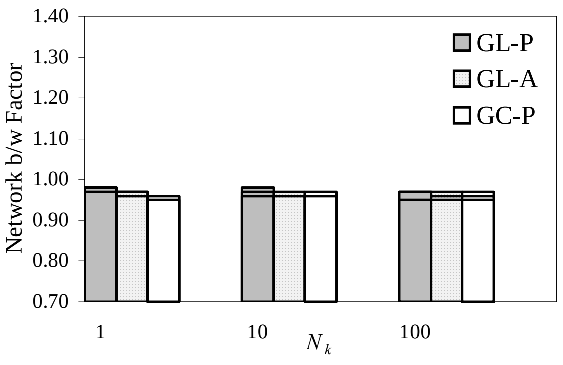
(c) GT-ITM (600 nodes, 10% client sites)



(d) Inet-2.1 (6000 nodes, 10% client sites)



(e) GT-ITM (600 nodes, 50% client sites)



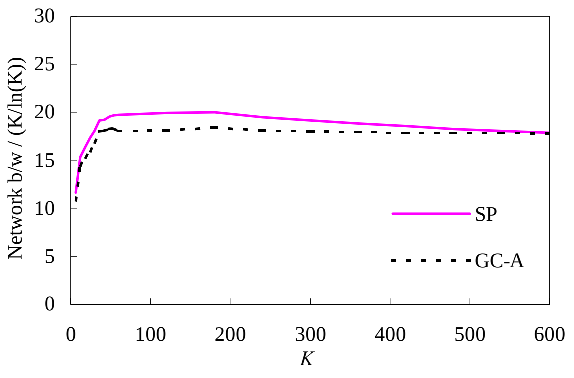
(f) Inet-2.1 (6000 nodes, 50% client sites)

Figure 8: Relative Network Bandwidth Requirements of Alternative Delivery Tree Algorithms
(Each bar shows the minimum, average, and maximum required network bandwidths, relative to the corresponding required bandwidths with SP, over 10 randomly-generated topologies)

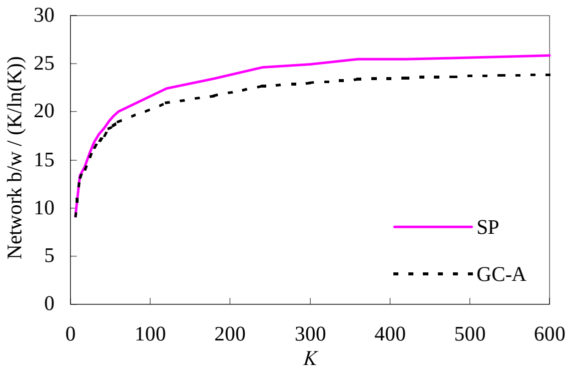
- GC-A performs the best. This is not surprising, as it takes into account the request rates from each client site to determine the incremental network bandwidth cost of each potential addition to the delivery tree.
- Shortest path trees (as constructed by SP) have consistently similar required network bandwidth as those constructed by GC-A for low request rates, and are (typically, but not always) only modestly more expensive even for higher request rates.
- The protocols GL-A, GL-P and GC-P yield only marginally higher required network bandwidth than that with GC-A, in most cases. However, for lower request rates and fewer client sites, GL-A and GL-P can perform poorly.

B. Scaling of Minimum Network Bandwidth

Figure 9 illustrates how the minimum network bandwidth for on-demand streaming scales with the number of client sites. Each data point averages over 10 randomly generated networks and client site assignments. N_k is fixed at 10 in all cases (similar results are obtained with other request rates). Our results indicate that the minimum network bandwidth scales as $O(K/\ln(K))$. To make this scaling behavior clearer, the figure graphs the minimum network bandwidth (as



(a) GT-ITM Topologies (600 nodes)



(b) Inet-2.1 Topologies (6000 nodes)

Figure 9: Scaling of Minimum Required Network Bandwidth (N_k=10 for all k)

estimated using our lower bound analyses from Section III) divided by $K/\ln(K)$. With only a small number of client sites the minimum network bandwidth grows more quickly, since the sites cannot share many links. For larger numbers of sites, however, the minimum network bandwidth appears to be $O(K/\ln(K))$, as indicated by the flattening of the curves.

V. Network Efficiency Of Bandwidth Skimming

This section addresses the question of whether practical on-demand streaming protocols can achieve close to the minimum required network bandwidth, and whether it is possible to achieve close to minimal network and server bandwidth usage simultaneously. We focus on the bandwidth skimming on-demand streaming protocol, and variations thereof, as bandwidth skimming can achieve reasonably close to the minimum required server bandwidth.

The approach taken is two-fold. First, in Section V.A, the network bandwidth used by "network-naïve" bandwidth skimming (i.e., bandwidth skimming as described in [11,12]) is compared against the lower bounds on the minimum required network bandwidth that were developed in Section III. The bounds in Section III do not reflect any limit on the number or total aggregate rate of transmissions a client can receive concurrently, so comparisons are also made with heuristic modifications of these bounds that reflect the finite client bandwidth assumed in the bandwidth skimming protocols.

Second, in Section V.B network-naïve bandwidth skimming is compared against a variant that attempts to further reduce network bandwidth usage, by taking into account client locations when determining which clients to aggregate and serve with a single multicast stream. The idea is that network bandwidth might be saved by preferentially aggregating clients that share more links in the delivery tree. In contrast, network-naïve bandwidth skimming uses an "early merging" policy that simply aggregates first the clients that can be aggregated the most quickly.

The results presented in this section are for bandwidth skimming with an achievable client data rate of twice the rate required for real-time playback. Qualitatively similar results are obtained with lower client data rates.

A. Network-Naïve Bandwidth Skimming vs. Lower Bounds

Results are presented here for the "shared link with fan-out of K" network topology given in Figure 1, which has a lower bound on the minimum required network bandwidth for on-demand streaming given by relation (2). Results for other topologies are qualitatively similar (i.e., they yield the same conclusions). As in Section III, the required network bandwidth is measured in units of the file playback bit rate times the average end-to-end unit bandwidth cost.

Additional useful insight is achieved by considering the minimum network bandwidth required when client data rate is constrained. A conjectured asymptotic lower bound on the required *server* bandwidth for a client data rate of twice the rate required for real-time playback, assuming that the server delivers streams at the real-time playback rate, is derived in [12] as:

$$1.62\ln\left(\frac{N}{1.62}+1\right). \qquad (8)$$

Noting that the bound given in relation (2) simply sums the expression in (1), independently computed for each link, this bound can be heuristically modified for a scenario in which each client can receive at most two real-time playback rate streams concurrently, by using the above expression instead of that in (1). This yields the following approximation for the lower bound on minimum required network bandwidth for on-demand streaming using the topology of Figure 1:

$$\frac{1}{2}\left(1.62\ln(\frac{N}{1.62}+1)+\sum_{k=1}^{K}1.62\ln(\frac{N_k}{1.62}+1)\right). \qquad (9)$$

Figure 10 compares the network bandwidth used by network-naïve bandwidth skimming, as obtained from simulation[3], against the lower bound of relation (2), and expression (9), for K=4 and equal request rates per client site. Note that the gap between the bandwidth skimming results and expression (9) is not large. These results (and qualitatively similar results for other topologies) suggest that there may be only modest room for improvement over network-naïve bandwidth skimming. Further, together with the results in [11,12] for required server bandwidth, they suggest that it is indeed possible to achieve reasonably close to minimal

[3]All simulation results have 95% confidence intervals that are within 5% of the reported values.

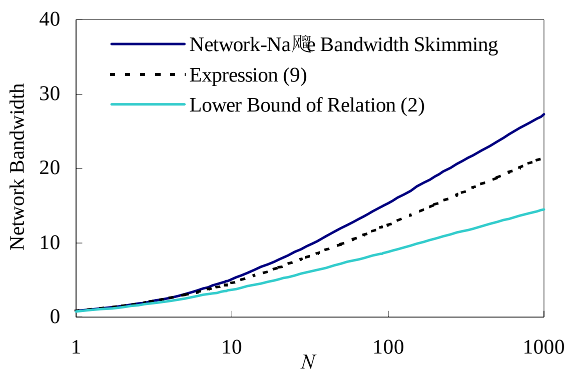


Figure 10: Bandwidth Skimming Network Efficiency
(Figure 1 shared link with fan-out of K topology, K=4)

network and server bandwidth usage simultaneously, with a practical on-demand streaming protocol.

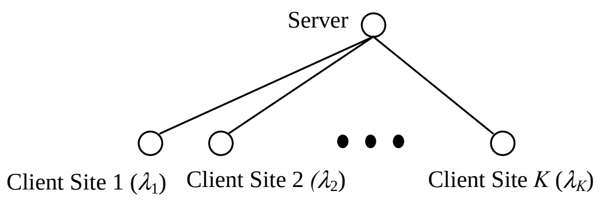
B. Network-Aware Bandwidth Skimming

Although the gap in Figure 10 between network bandwidth usage for network-naïve bandwidth skimming and expression (9) is not large, it is greater than the gap shown in [12] between server bandwidth usage for bandwidth skimming and the corresponding conjectured asymptotic lower bound on required server bandwidth in expression (8). Thus, this section explores the potential for practical protocols with reduced network bandwidth usage. We first consider a topology in which clients from different sites do not share any portion of the delivery tree (shown in Figure 11), and a "network-aware" bandwidth skimming policy that aggregates clients only if they are from the same site. Since there are no shared links, there is no network bandwidth benefit to merging clients from different sites, and thus this policy should minimize network bandwidth usage in comparison to any other possible bandwidth skimming policy, on this topology. Further, the improvement in network bandwidth usage through use of a network-aware policy should be maximized.

For the topology in Figure 11, an approximation for the minimum required network bandwidth for on-demand streaming when each client can receive at most two real-time playback rate streams concurrently, can be derived in a similar fashion as expression (9), yielding:

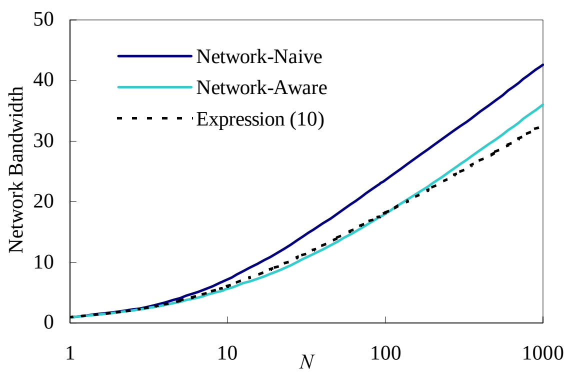
$$\left(\sum_{k=1}^{K} 1.62\ln\left(\frac{N_k}{1.62}+1\right)\right) . \qquad (10)$$

Note that if the average "distance" (i.e., end-to-end unit bandwidth cost) between the server and each client site is the same in the Figure 1 and Figure 11 topologies, the topology in Figure 1 has lower minimum network bandwidth requirements (i.e., expression (9) is less than expression (10)). This makes sense, since in the Figure 1 topology half the end-to-end distance is traversed using a shared link. Facilitating bandwidth comparisons among different topologies that have the same average end-to-end distance is the motivation behind expressing network bandwidth requirements in units of the average end-to-end unit bandwidth cost.

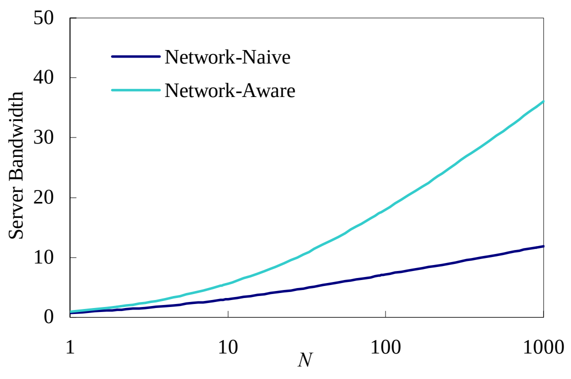


Server
Client Site 1 (λ_1)
Client Site 2 (λ_2)
Client Site K (λ_K)

Figure 11: Fan-out Topology with No Shared Links



(a) Required Network Bandwidth



(b) Required Server Bandwidth

Figure 12: Network-Aware/Naïve Bandwidth Skimming
(Figure 11 fan-out topology, K=4)

Figure 12(a) compares the required network bandwidth for the network-aware and network-naïve bandwidth skimming policies, and expression (10), for the topology shown in Figure 11 with K=4 and equal request rates per client site. For this topology, the network-aware policy reduces the required network bandwidth to very close to the approximate lower bound given by expression (10). However, this improvement in network bandwidth usage is achieved at a high price in server bandwidth, as shown in Figure 12(b).

Even if the network bandwidth improvements seen in Figure 12(a) were deemed worth the cost of the associated increase in required server bandwidth, it is not clear that improvements of this magnitude could be realized in practical topologies. While preferentially aggregating clients from the same client site has a modest positive impact on the bandwidth usage of non-shared links, it can have a negative impact on the bandwidth usage of shared links. This is illustrated in Figure 13, which shows the required network bandwidth for the

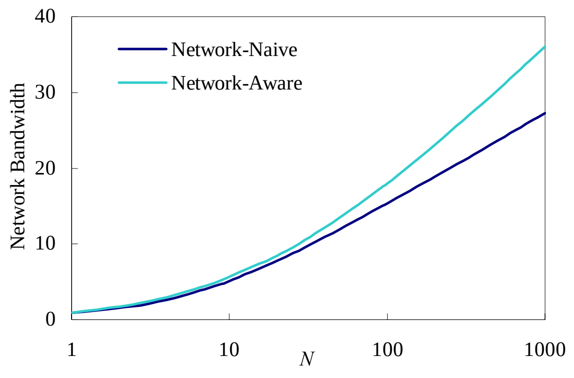


Figure 13: Network-Aware/Naïve Bandwidth Skimming with Shared Link (Figure 1 topology, K=4)

simple network-aware and the network-naïve bandwidth skimming policies on the topology of Figure 1, with K=4 and equal client site request rates. This is the same scenario as in Figure 12, except with a single additional shared link.

As seen in the figure, the presence of the shared link reverses the relative performance of the network-aware and network-naïve policies. We have examined more complex policies that are more clever (and careful) in taking client location into account, on topologies with shared links, but such policies were found to not substantively reduce network bandwidth usage (yet increased server bandwidth usage) in comparison to network-naïve bandwidth skimming. It appears that substantive reductions in network bandwidth usage are not possible with network-aware policies in realistic settings.

VI. Conclusions

This paper has studied the network bandwidth requirements of scalable on-demand streaming protocols. Previous work has considered only the server (disk I/O and network interface) bandwidth, or has assumed a greatly simplified network bandwidth model.

Our results show that in the multicast setting, practical on-demand streaming protocols can achieve substantial reductions in required network bandwidth as well as server bandwidth, with network bandwidth scaling as $O(K/\ln(K))$ where K is the number of client sites and the request rate per site is kept fixed, and as $O(\ln N)$ where N is the total request rate and the number of client sites is kept fixed. In contrast, with unicast delivery, network bandwidth scaling is linear in K (for fixed request rate per site), and in N.

In an application-level multicast setting, there may be considerable flexibility with respect to the design of the multicast delivery trees. For some applications it may be desirable to configure multicast trees from a server to a set of client sites so as to minimize network bandwidth rather than the latencies between the clients and the content server. However, in most cases, achievable reductions in the network bandwidth usage in delivery, in comparison to that with shortest path trees, were found to be quite small (i.e., 3-16%).

Current research includes consideration of implementation and other issues in application-level multicast and delivery tree construction, in the context of a prototype on-demand streaming media system.

References

[1] C. C. Aggarwal, J. L. Wolf, and P. S. Yu, “A Permutation Based Pyramid Broadcasting Scheme for Video-On-Demand Systems”, Proc. IEEE ICMCS’96, Hiroshima, Japan, June 1996.

[2] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, “Analysis of Educational Media Server Workloads”, *Proc. NOSSDAV 2001*, June 2001.

[3] Y. Birk and R. Mondri, “Tailored Transmissions for Efficient Near-Video-On-Demand Service”, *Proc. IEEE ICMCS’99,* Florence, Italy, June 1999.

[4] Y. Cai, K. A. Hua, and K. Vu, “Optimizing Patching Performance”, *Proc. MMCN’99,* San Jose, CA, Jan. 1999.

[5] K. Calvert and E. Zegura, “GT Internetwork Topology Models (GT-ITM)”, http://www.cc.gatech.edu/fac/Ellen.Zegura/gt-itm/gt-itm.tar.gz.

[6] S. W. Carter and D. D. E. Long, “Improving Video-on-Demand Server Efficiency Through Stream Tapping”, *Proc. ICCCN’97,* Las Vegas, NV, Sept. 1997.

[7] Y. Chawathe, S. McCanne, and E. A. Brewer, “An Architecture for Internet Content Distribution as an Infrastructure Service”, http://yatin.chawathe.com/~yatin/papers/scattercast.ps, Feb. 2000.

[8] Y.-H. Chu, S. G. Rao, and H. Zhang, “A Case for End System Multicast”, *Proc. ACM SIGMETRICS 2000*, Santa Clara, CA, June 2000.

[9] J. Chuang and M. Sirbu, “Pricing Multicast Communications: A Cost-Based Approach”, *Proc. INET’98*, Geneva, Switzerland, July 1998.

[10] A. Dan, D. Sitaram, and P. Shahabuddin, “Scheduling Policies for an On-demand Video Server with Batching”, *Proc. ACM Multimedia’94,* San Francisco, CA, Oct. 1994.

[11] D. L. Eager, M. K. Vernon, and J. Zahorjan, “Bandwidth Skimming: A Technique for Cost-Effective Video-on-Demand”, *Proc. MMCN 2000,* San Jose, CA, Jan. 2000.

[12] D. L. Eager, M. K. Vernon, and J. Zahorjan, “Minimizing Bandwidth Requirements for On-Demand Data Delivery”, *IEEE Trans. On Knowledge and Data Engineering*, Special Section of invited papers from MIS’99, Vol. 13, No. 5, Sept./Oct. 2001, pp. 742-757.

[13] L. Gao, J. Kurose, and D. Towsley, “Efficient Schemes for Broadcasting Popular Videos”, *Proc. NOSSDAV’98,* Cambridge, UK, July 1998.

[14] L. Gao and D. Towsley, “Supplying Instantaneous Video-on-Demand Services Using Controlled Multicast”, *Proc. IEEE ICMCS’99,* Florence, Italy, June 1999.

[15] A. Hu, “Video-on-Demand Broadcasting Protocols: A Comprehensive Study”, *Proc. IEEE Infocom 2001*, Anchorage, AK, April 2001.

[16] K. A. Hua and S. Sheu, “Skyscraper Broadcasting: A New Broadcasting Scheme for Metropolitan Video-on-Demand Systems”, *Proc. ACM SIGCOMM’97 Conf.,* Cannes, France, Sept. 1997.

[17] K. A. Hua, Y. Cai, and S. Sheu, “Patching: A Multicast Technique for True Video-On-Demand Services”, *Proc. ACM Multimedia’98,* Bristol, U.K., Sept. 1998.

[18] C. Jin, Q. Chen, and S. Jamin, “Inet Topology Generator”, http://topology.eecs.umich.edu/inet.

[19] L. Juhn and L. Tseng, "Fast Data Broadcasting and Receiving Scheme for Popular Video Service", *IEEE Trans. On Broadcasting 44*, 1 (March 1998).

[20] A. Mahanti, D. L. Eager, M. K. Vernon, and D. Sundaram-Stukel, “Scalable On-Demand Media Streaming with Packet Loss Recovery”, *Proc. ACM SIGCOMM 2001 Conf.*, San Diego, CA, Aug. 2001.

[21] J.-F. Paris, S. W. Carter, and D. D. E. Long, “A Hybrid Broadcasting Protocol for Video On Demand”, *Proc. MMCN’99,* San Jose, CA, Jan. 1999.

[22] G. Phillips and S. Shenker, “Scaling of Multicast Trees: Comments on the Chuang-Sirbu Scaling Law”, *Proc. ACM SIGCOMM’99 Conf.*, Cambridge, MA, Aug./Sept. 1999.

[23] S. Sen, L. Gao, J. Rexford, and D. Towsley, “Optimal Patching Schemes for Efficient Multimedia Streaming”, *Proc. NOSSDAV’99,* Basking Ridge, NJ, June 1999.

[24] S. Viswanathan and T. Imielinski, “Metropolitan Area Video-on-Demand Service using Pyramid Broadcasting”, *Multimedia Systems 4,* 4 (Aug. 1996).

[25] Waterloo Maple: http://www.maplesoft.com/

[26] L. Wei and D. Estrin, “The Trade-offs of Multicast Trees and Algorithms”, *Proc. ICCCN’94*, San Francisco, CA, Sept. 1994.

[27] E. W. Zegura, K. Calvert, and S. Bhattacharjee, “How to Model an Internetwork”, *Proc. IEEE Infocom’96*, San Francisco, CA, April 1996.