Minimizing Delivery Cost in Scalable Streaming Content Distribution Systems

Jussara M. Almeida, Derek L. Eager, Mary K. Vernon, and Stephen J. Wright

***Abstract*—Recent scalable multicast streaming protocols for on-demand delivery of media content offer the promise of greatly reduced server and network bandwidth. However, a key unresolved issue is how to design scalable content distribution systems that place replica servers closer to various client populations and route client requests and response streams so as to minimize the total server and network delivery cost. This issue is significantly more complex than the design of distribution systems for traditional Web files or unicast on-demand streaming, for two reasons. First, closest server and shortest path routing does not minimize network bandwidth usage; instead, the optimal routing of client requests and server multicasts is complex and interdependent. Second, the server bandwidth usage increases with the number of replicas. Nevertheless, this paper shows that the complex replica placement and routing optimization problem, in its essential form, can be expressed fairly simply, and can be solved for example client populations and realistic network topologies. The solutions show that the optimal scalable system can differ significantly from the optimal system for conventional delivery. Furthermore, simple canonical networks are analyzed to develop insights into effective heuristics for near-optimal placement and routing. The proposed new heuristics can be used for designing large and heterogeneous systems that are of practical interest. For a number of example networks, the best heuristics produce systems with total delivery cost that is within 16% of optimality.**

***Index Terms*—Content distribution systems, modeling, streaming media.**

I. Introduction

RECENT SCALABLE streaming protocols for on-demand delivery of popular media content promise significant server and network bandwidth savings (e.g., [6], [10], [13], [14]). However, a key unresolved issue is how to design scalable streaming content delivery systems. This problem involves placing replicas of popular objects closer to some of the client sites so as to reduce content delivery cost. The key questions are how many replicas, where each replica should be placed, where to route client requests, and how to route the streams that the clients receive. The goal considered in this paper is to minimize total delivery cost, which in general includes both total network and total server delivery cost. Once the replicas are placed for minimum delivery cost, packet-loss recovery can be provided by using techniques such as those described in [4] or [18], while client latency can be minimized by storing a small prefix closer to the clients.

For conventional unicast delivery of streaming media, total server bandwidth usage (summed over all the servers) is independent of the number and placement of the replicas, and shortest path routing minimizes network bandwidth usage. Despite these simplifying features, optimal replica placement is NP-complete for general network topologies [15]. However, previous work has shown that a greedy algorithm that places one replica at a time, minimizing the total network cost at each step, produces near-optimal solutions efficiently [16], [17], [19], [20].

For scalable streaming protocols, the design of optimal content delivery systems is more complex. As in conventional systems, network delivery cost is proportional to total network bandwidth, and server cost is proportional to total concurrent server bandwidth, since increased server resources (cpus, disks, and network interface speeds) are needed to sustain higher bandwidths. However, the multicast trees that minimize total network bandwidth involve a complex tradeoff between minimizing distance and maximizing the number of clients that share the path segments. Thus, closest server and shortest path routing are not always optimal. Instead, computing the cost of a given placement of replicas requires us to find the set of multicast trees from the replicas to the clients that minimizes total network bandwidth, which greatly complicates the optimal placement problem. Moreover, server bandwidth usage is inversely proportional to the number of clients that are served per multicast, and increasing the number of replicas decreases the average number of clients served per multicast. Thus, as the number of replicas increase, total network bandwidth decreases but total server bandwidth increases. Determining the optimal number of replicas requires an assessment that includes the *relative* cost of server bandwidth and network bandwidth.

Optimal placement and routing for scalable streaming systems have been addressed only partially in previous work. A restricted form of the placement problem, in which the number of replicas is either one or a fixed number P, and the relative cost of a multicast stream from any one of the P servers compared to the cost of a stream from the single server is given as a single fixed input parameter, is studied in [2], [9], [22], and [23]. Other previous work has proposed heuristics for routing of a single live multicast stream [11] or for routing on-demand

Manuscript received January 15, 2003; revised August 1, 2003. This work was supported in part by the National Science Foundation under Grants ACI-0196485, EIA-0127857, and ACI-0082100, by the Natural Sciences and Engineering Research Council of Canada, and by CNPq/Brazil. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Pascal Frossard.

J. M. Almeida is with the Federal University of Minas Gerais, Belo Horizonte, MG 30160-010, Brazil (e-mail: jussara@dcc.ufmg.br).

D. L. Eager is with the University of Saskatchewan, Saskatoon, SK S7N 5A9, Canada (e-mail: eager@cs.usask.ca).

M. K. Vernon and S. J. Wright are with the University of Wisconsin, Madison, WI 53706 USA (e-mail: vernon@cs.wisc.edu; swright@cs.wisc.edu).

Digital Object Identifier 10.1109/TMM.2003.822796

content from a single server [25]. To our knowledge, routing stored content from multiple servers that employ a scalable protocol (e.g., [6], [10], [13], [14]) which requires bandwidth that scales sublinearly in the client load has not been addressed.

For a media object that is delivered on-demand to a given set of client sites, using one of several types of scalable delivery protocols, this paper develops both exact and approximate methods for determining the number and placement of replicas, as well as the routing of requests and multicast streams, so as to minimize the total delivery cost. The key contributions of this work are the following.

- A simple specification of the exact minimum cost replica placement and routing problem for an object, given the number of replicas and the set of client site request rates to the object. The optimal number of replicas is the number for which minimum total cost is achieved.
- Solutions for candidate client sites and measured Internet topologies which show that, in practice, using placement and routing that are optimized for unicast delivery can be substantially suboptimal for scalable system.
- New efficient approximate algorithms that are based on insights from the analysis of simple canonical topologies. In particular, an upper bound is derived for the cost increase of using shortest path routing for scalable delivery. The best approximate algorithms both require significantly less execution time than the exact algorithm and produce solutions with total cost within 16% of the exact minimum cost solution, for all tested configurations.

The remainder of the paper is organized as follows. Section II provides an overview of the design of scalable streaming distribution systems. Section III develops an exact method for determining the minimum cost placement and routing for scalable streaming. Section IV develops insights into efficient placement and routing strategies drawn from the analysis of simple canonical topologies, and Section V defines and evaluates a number of heuristics derived from these insights. Section VI summarizes the paper.

II. Overview of Scalable Content Distribution System Design

The design of a scalable media content distribution system requires as inputs the client site locations, the set of possible access points for replica placement, the network topology that interconnects client sites and replica access points and, for each client site, the request rate for the object to be delivered.

A "client site" is a group of clients that are connected to a given access point. The client site request rates for an object may vary with time. As in conventional delivery system design, the goals are to configure the system efficiently for a given object with a given request rate from each client site, and to reconfigure the system periodically, whenever the request rates for the object change significantly.

Section II-A describes the network topologies that are used in developing and evaluating the methods we propose for design of delivery systems. Section II-B provides an overview of the key assumptions in the new system design methods.

TABLE I
EXAMPLE CLIENT SITES

Symbol	Server Location	Symbol	Server Location
AZ	University of Arizona, US	CMU	Carnegie Mellon University, US
CO-AU	Connect, Australia	CU	Carleton Univ., Canada
EI-CH	Eimer, Switzerland	GU-UK	Glasgow Univ., U. K.
IN-AU	iiNet, Australia	MO-ES	Metrored Online, Spain
MU-AU	Monash Univ., Australia	OC-ES	OCEA, Spain
OI-IT	Officine Informat., Italy	PW-FR	PacWan, France
SDSC	San Diego Super-Computer Center, US	ST-IN	National Centre for Software Technology, India
SE-FR	Univ. St Etienne, France	SU	Stanford Univ., US
TE-AU	Telstra, Australia	TX	Texas A&M Univ., US
UCB	Univ. of California, at Berkeley, US	UG-CH	University of Geneva, Switzerland
UO	Univ. of Oregon, US	UW	Univ. Wisconsin, US
VY	Vineyard.NET, US	XT-CA	XeniTec, Canada

A. Network Topologies

As in previous work that addresses optimal placement and routing for unicast delivery of web content [19], [20], we model the Internet as a graph where each node represents either a router or an autonomous system (AS), and a link represents a hop. (To improve readability in some figures, we have drawn a link that represents a sequence of hops, with a label to indicate the associated hop count.)

We consider Internet topologies for sets of up to 24 client sites selected from the list in Table I. These sites are widely distributed in four continents, allowing us to experiment with realistic wide-area Internet topologies. The insights and conclusions obtained for these systems should be applicable to systems with larger numbers of client sites dispersed over the same geographical area, since each represented client site might approximately model a total client load that is distributed among nodes close to the site.

We consider both router- and AS-level topologies because charging schemes for multicast media streams are still in flux and because the methods for computing optimal placement and routing are the same for both types of topology. For simplicity in presenting the new design methods, the topologies considered are unweighted graphs, but weights could easily be added to represent bandwidth capacity or unequal unit bandwidth cost for different network links.

The router topology for a given set of client sites is created by running 1000 traceroute commands[1] from each client site to every other client site, over an extended period of time (e.g., four weeks). To illustrate the results, Fig. 1(a) shows the router topology for four client sites. To simplify the diagram, each client site is connected to the first router in the traceroute paths that does not have the same domain name as the client site. This simplification can also be made in the topology for the cost model if the cost of delivery within the client site domain is neg-

[1][Online] Available: http://www.traceroute.org/

ligible compared to the cost to deliver to the domain. Also, each sequence of routers that does not intersect with any other sequence of routers is shown and could be specified as a single link labeled with the number of hops in the sequence. All path intersections are shown in the figure and should be specified for the optimal system design.

The AS topology for a given set of client sites is created from the router topology by mapping each router IP address to the corresponding AS, for example, using the information collected by both Route Views[2] and RIPE[3]. This approach, also used in [16], provides a topology for system design that is based on the paths most frequently used, rather than the contractual relationships among the ASes [7]. Fig. 1(b) depicts the AS topology created for the four sites in Fig. 1(a). Note that a more diverse set of client sites from Table I will have significantly more complex router and AS topologies than the client sites shown in Fig. 1.

B. System Assumptions

A number of assumptions are made for designing distribution systems efficiently while retaining the essential features of the system inputs. In particular are the following.

1) Placement and routing for each replica are optimized for a set of client sites, each with a given object request rate, where a site is a group of clients in an AS or in the same domain for router topologies.
2) The delivery tree for each server is assumed to have a fixed topology for all client requests.
3) The total system cost includes network and server costs, which in turn are proportional to the total network and server bandwidth, respectively. Total network bandwidth is a (possibly weighted) sum of the bandwidth required for each hop in the delivery paths; a hop is an AS or a link between two routers, depending on the topology used.
4) Client requests for the object are Poisson, as has been observed for media server workloads [3], [24].
5) Each replica is of the full object, based on previous results that show that full file caching outperforms prefix caching in scalable streaming configurations unless the request rate to each proxy server is very low or the delivery from the proxy server is zero cost [2].

For simplicity in the model presentation and in developing the design heuristics, we also assume that each client receives the entire object. The proposed delivery cost models used in the exact and heuristic solutions can be extended for the case in which only partial objects are delivered in response to interactive client requests, using the results in [21].

The request rate for a client site i, denoted by N_i, is expressed in units of the average number of requests that arrive per time T, where T is the time it takes to stream the entire object to a client. Thus, N_i is equal to the average number of clients who are simultaneously receiving/viewing the object, which is also the average number of concurrent server streams required for unicast delivery of the object.

Server bandwidth is expressed as the average number of concurrent streams delivered by the server, which is sublinear in the

[2][Online] Available: http://www.routeviews.org/

[3][Online] Available: http://www.ripe.net/

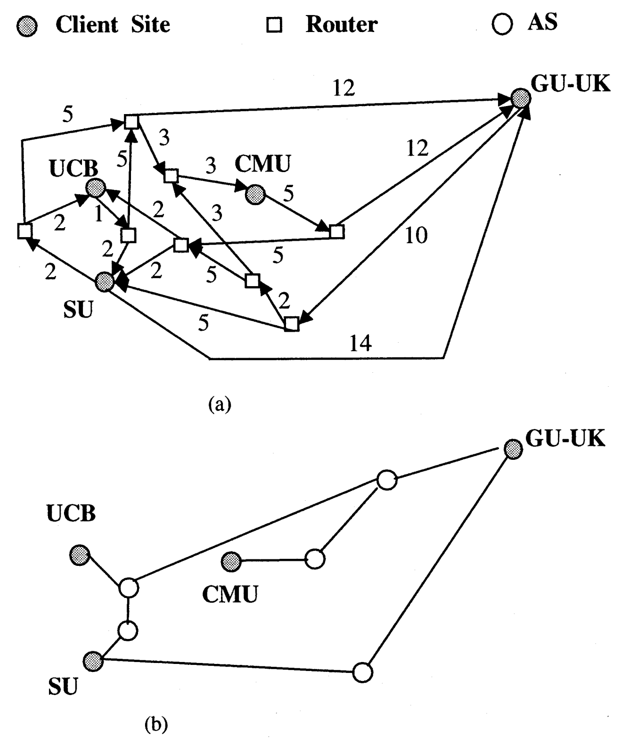


Fig. 1. Example topologies. (a) Router level. (b) AS level (bidirectional links).

request rate for the scalable protocols. Per-hop network bandwidth is also expressed in these units. For total request rate N served by a server or a network link, the average number of concurrent streams for hierarchical merging is approximately $B = 1.63\ln(N/1.63 + 1)$ [10], or $B = \sqrt{2N+1} - 1$ for patching [12], or a constant (e.g., $B = 8$) determined by the client startup latency for a periodic broadcast protocol.

In Sections III–V, we develop methods to compute the optimal and near-optimal delivery configuration for each media object independently, subject to the constraint of the specified possible replica access points. Extensions to constrain different objects to be placed at the same replica access points are beyond the scope of this paper.

III. Optimal Server Placement and Routing

A. Motivating Examples

Figs. 2 and 3 illustrate the router-level topology for client sites located close to the Internet 2 backbone.[4] The client sites, indicated by circles, are darkly shaded at replica server locations. The optimal routing is shown as solid directed arcs and unused network links are shown as dotted lines.

Fig. 2(a) and (b) shows the optimal routing from a server at a client site in the upper left to six other client sites that have uniform client request rates, assuming conventional (unicast) streaming in Fig. 2(a) and scalable streaming in Fig. 2(b). Note that the optimal routing for scalable streaming does not use the shortest path to the client site on the far right. If the request rate at each client site is 1000 and the server uses the hierarchical

[4][Online] Available: http://www.geog.ucl.ac.uk/casa/martin/atlas/more_isp_maps.html

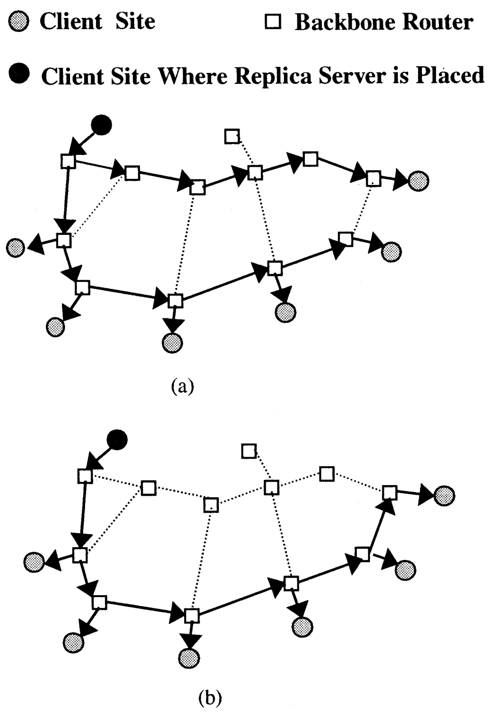


Fig. 2. Optimal routing for fixed replica placement. (a) Conventional. (b) Scalable.

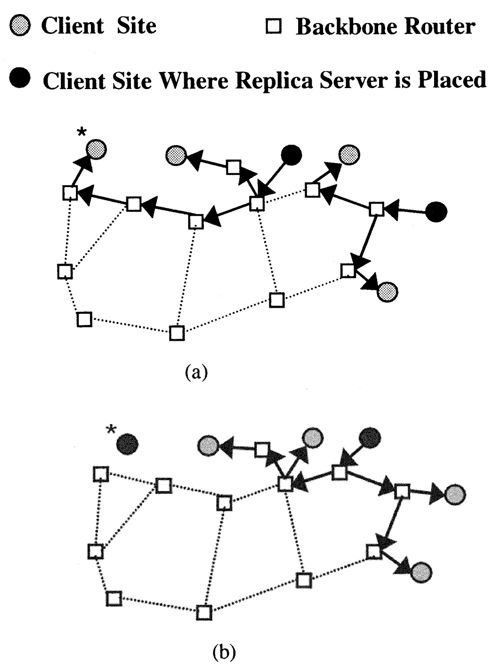


Fig. 3. Optimal placement and routing for two replicas. (a) Conventional. (b) Scalable.

merging scalable delivery protocol, the routing in Fig. 2(a) requires 26% more network bandwidth than the optimal routing in Fig. 2(b) when network bandwidth is measured per-hop and each hop has equal weight. Fig. 3 shows the optimal placement as well as routing for two replicas and six client sites, assuming that the client site marked with "*" has a request rate of 500 and all other client sites have a request rate of 1000. The optimal

$$\underset{S^*, N_{i,j}}{\text{minimize}} \; B_{network} + \gamma \times B_{server}$$

subject to: (1) $|S^*| = m$

(2) $S^* \subseteq S$

(3) $\textit{for all } j \notin S^*: \sum_{(i,j)\in A} N_{i,j} = N_j + \sum_{(j,k)\in A} N_{j,k}$

where:

Hierarchical Merging:

$$B_{server} = \sum_{i \in S^*} 1.63 \ln\left(\sum_{(i,j)\in A} N_{i,j} \Big/ 1.63 + 1 \right) \quad \text{and}$$

$$B_{network} = \sum_{(i,j)\in A} 1.63 w_{i,j} \log\left(N_{i,j} / 1.63 + 1 \right) \quad \forall (i,j) \in A$$

Patching:

$$B_{server} = \sum_{i \in S^*} \left(\sqrt{2 \sum_{(i,j)\in A} N_{i,j} + 1} - 1 \right) \quad \text{and}$$

$$B_{network} = \sum_{(i,j)\in A} w_{i,j} \left(\sqrt{2 N_{i,j} + 1} - 1 \right) \quad \forall (i,j) \in A$$

Periodic Broadcast ($k > 1$) or Scheduled Multicast ($k = 1$):

$$B_{server} = mk \quad \text{and} \quad B_{network} = \sum_{\substack{(i,j)\in A \\ N_{i,j} > 0}} w_{i,j} k \quad \forall (i,j) \in A$$

Unicast:

$$B_{server} = \sum_{i \in C} N_i \quad \text{and} \quad B_{network} = \sum_{(i,j)\in A} w_{i,j} N_{i,j} \quad \forall (i,j) \in A$$

Fig. 4. Basic model for computing placement and routing for m replicas.

unicast solution for scalable delivery is again 26% more expensive than the optimal solution in Fig. 3(b).

As noted in Section I, previous work has not addressed how to compute the optimal configurations for content delivery systems that use scalable delivery protocols.

B. Optimization Model

Fig. 4 provides the basic model that computes, for a fixed number of replicas, the placement and routing that minimizes total delivery cost. Table II defines the eight inputs and four outputs of this model. The optimal placements are given by the set of server nodes, S^*, and the optimal routing is given by the link loads, $N_{i,j}$, for all (i,j) in the arc set A. Total delivery cost is defined as a linear combination of total network bandwidth and total server bandwidth, where γ is the cost of a unit of server bandwidth relative to the cost of a unit of network bandwidth. The constraints guarantee that: 1) the number of replica servers is equal to a given value m; 2) replicas are placed at replica access points; and 3) total flow into any node that is *not* chosen as a replica server equals total flow out of the node plus the flow to clients at that node. Note that constraints that reflect the maximum capacity of each network link or server can easily be added to the model.

The model can be applied for systems that employ the hierarchical merging delivery protocol, the patching protocol, a

TABLE II
OPTIMIZATION MODEL INPUT PARAMETERS AND OUTPUTS

Parameter	Description
V	Set of nodes {1, 2, 3, … n}
A	Set of *directed* arcs $\{(i,j)\}$, $i, j \in V$
$w_{i,j}$	Relative cost of one unit of bandwidth on arc $(i,j) \in A$
S	Access points for replica (i.e., server) placement ($S \subseteq V$)
C	Client sites ($C \subseteq V$)
N	Client loads: $N = \{N_i: N_i > 0$ if $i \in C$ and $N_i = 0$ if $i \in V\text{-}C\}$
m	Number of replicas to be placed
γ	Ratio of cost of one unit of server bandwidth to one unit of network bandwidth
S^*	Set of server nodes in the optimal delivery network ($S^* \subseteq S$)
$N_{i,j}$	Sum of loads of the clients that are served via arc $(i,j) \in A$
$B_{network}$	Total network bandwidth summed over all arcs
B_{server}	Total concurrent server bandwidth summed over all servers

periodic broadcast protocol, scheduled multicasts, or unicast streaming, by using the respective bandwidth formulas provided in the figure. Note that the total average server bandwidth for unicast delivery is simply the sum of the average number of concurrent client requests from each client site, independent of the number of replicas. Also note that the square root bandwidth functions for patching assume the servers use an optimal threshold that is based on an accurate estimate of the current client arrival rate, although the arrivals may be bursty and the rate may change over time. Finally, for periodic broadcast systems, the fixed number k of streams in the broadcast must be specified.

The model in Fig. 4 would be a relatively straightforward extension of previous models [16], [17], [19] for optimal placement in conventional unicast systems, except that: 1) the model solves for optimal routing jointly with optimal placement and 2) the network and server bandwidth formulas for scalable delivery protocols such as patching and hierarchical merging introduce nonlinear, nonconvex functions into the model. For the resulting nonlinear integer programs, practical optimization software often fails to find a true (global) optimum. We solve this problem by reformulating these scalable protocol models as *linear* integer programs, thereby enabling the use of mixed-integer-programming codes that are guaranteed to converge to the optimal solution. We then improve the performance of these algorithms by introducing additional constraints into the formulation that reflect knowledge of the nature of the problem and its solution, thereby reducing the search space and allowing more complex design problems to be solved.

We propose an exact technique and a more efficient approximate technique to model the nonlinear functions by means of linear functions. In the exact technique, we create a table with two entries for each possible value of $N_{i,j}$ (i.e., each possible partial sum of client site request rates that might be served over any arc or by any server). The two entries are the possible load (N) and the bandwidth for that load (e.g., $1.63\ln(N/1.63+1)$, for the hierarchical merging protocol). In the second technique, we create a table that specifies a piecewise linear approximation for the logarithmic or square-root function. In both techniques, binary variables $\theta_{i,j,k}$ indicate whether arc (i,j) has load specified in table entry k. Only one of the variables $\theta_{i,j,k}$ is allowed to be 1 for each pair (i,j). The complexity of the optimization problem depends strongly on the number of binary variables. In the case of the piecewise linear approximation, the nature of the logarithmic function is that a good approximation can be obtained with relatively few "pieces". If an accurate solution is desired, we can first solve a model with a few (e.g., three) pieces in the piecewise-linear approximation, and use the solution of this model as a "warm start" for a more accurate model with, e.g., four or five pieces.

Additional constraints that reduce the size of the search space include the following.

- For each client node $j \in C$, there is exactly one arc $(i,j) \in A$ with nonzero $N_{i,j}$.
- Each server is the root of a delivery tree and therefore has no inflow. That is, $N_{i,j} = 0$ for all $j \in S^*$ and all $(i,j) \in A$.
- A node that is neither a client nor a server has at most one parent in the delivery tree; that is, for each j with $j \notin C$, $j \notin S^*$, we have $N_{i,j} \neq 0$ for at most one arc $(i,j) \in A$. Furthermore, such nodes may have out-flowing loads only if they have a nonzero load along one inflowing arc.

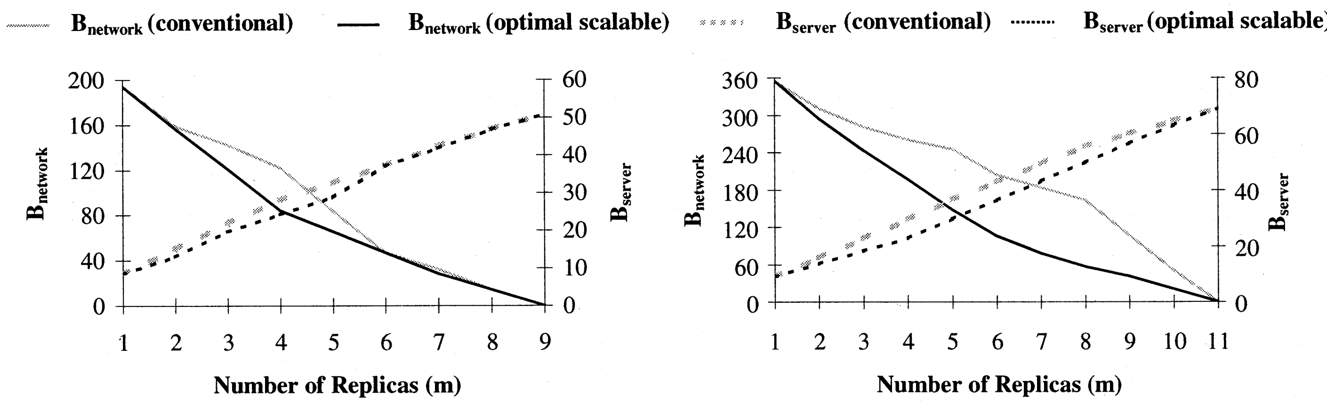
These additional constraints reduce the solution time significantly, allowing the model to be applied to networks with on the order of a dozen widely dispersed client sites with heterogeneous loads interconnected by on the order of 100 additional nodes that represent router-level path intersections, and a replica access point at every node. Larger problems are also tractable if the heuristic solution developed in Section V is used as the starting point in the optimal solution.

C. Results for Example Client Sites

In this section, the optimization models are applied to realistic Internet topologies for example client sites, created as described in Section II. Our goals are to evaluate solution feasibility and to compare the cost of the optimal scalable system to the cost of scalable streaming in a system where placement and routing is optimized for conventional unicast delivery (also computed from the model). The results also illustrate how the optimal number of replicas is determined.

For the results reported in the remainder of the paper, the possible replica server access points include all client nodes and each router or AS that is the intersection of at least two network paths. We show results only for the simple and highly efficient hierarchical stream merging protocol [10]. Results obtained for other sets of replica server access points and other scalable protocols were similar.

For a commercial system design problem, it should be possible to estimate the relative cost of server and network unit bandwidth, γ. Since we are solving hypothetical design problems that do not have realistic server and network bandwidth costs, we set $\gamma = 0$, in which case we obtain the solution that minimizes total network bandwidth. Solutions for positive values of γ can be computed at similar computational cost. We plot both the total server bandwidth and the total network bandwidth for the optimal solution, as a function of the number of replicas, and comment on how the results would differ for $\gamma > 0$



(a)

(b)

Fig. 5. Scalable system delivery costs: conventional versus optimal placement and routing (router-level topology, hierarchical stream merging delivery). (a) Nine heterogeneous clients in the U.S. (four clients: $N_i = 1000$) (UW, UCB, SDSC, SU). Five clients: $N_i = 100$) (UO, TX, CMU, AZ, VY). (b) Eight clients in the U.S., one client in Canada, U.K., India. U.S.: $N_i = 1000$ (AZ, CMU, SDSC, SU, UCB, UW, TX, UO). Others $N_i = 100$ (CU, GU-UK, ST-IN).

and for other scalable protocols. The model is implemented and solved to true optimality using the GAMS system [8].

We have applied the model both for AS and for router topologies in which the client sites have different degrees of dispersion (e.g., all client sites in one continent, and client sites in four continents). For each set of client sites, we varied the client-site load distribution from the homogeneous case (all client loads identical) to the case in which 25%–50% of the client sites (depending on the complexity of the topology) have 0.1–0.01 times the demand of the other sites. We experiment with client loads N_i of 10, 100, and 1000.

Fig. 5 shows representative results for two sets of client sites with different degrees of dispersion. Note that for either the conventional placement and routing or the optimal scalable placement and routing, the total network bandwidth (solid curves) decreases with the number of replicas, but the total server bandwidth (dotted curves) increases as a function of the number of replicas. For values of $\gamma > 0$, the curves for the conventional solution and the endpoints of the optimal scalable solution are the same, but the network bandwidth for the optimal scalable system might initially decrease somewhat less rapidly as each new replica is added, since greater emphasis would be placed on minimizing server bandwidth. Thus, for $\gamma > 0$, the difference between the conventional and optimal scalable content distribution designs may be somewhat greater for server bandwidth and somewhat less for network bandwidth than is the case for $\gamma = 0$.

The optimal value for the number of replicas is the value that minimizes the weighted sum of the network and server bandwidth costs. Thus, for $\gamma > 0$, the optimal number of replicas typically will be smaller than the number of client sites. Similar results are obtained for other scalable streaming protocols because for each such protocol (unlike unicast delivery): 1) total server bandwidth increases as the number of replicas increases and 2) optimal placement and routing differ from the conventional placement and routing since shared servers and path segments reduce bandwidth costs.

For the example topologies, the additional cost of using a conventional unicast optimal placement and routing solution for a system with a scalable delivery protocol can be quite significant, for both homogeneous and heterogeneous client loads. We observed increases in total network bandwidth and total server bandwidth as high as 50%–150% and 30%, respectively, for m between 1 and the number of client sites. As discussed below, the increases are due mainly to different placement decisions in the optimal scalable solution.

Fig. 6(a) and (b) shows the optimal placement and routing configuration for the conventional unicast and the hierarchical merging scalable delivery protocol, respectively, for five replicas in the topology that has the results plotted in Fig. 5(b). The optimal solution for conventional delivery protocols [Fig. 6(a)] consists of placing all replicas at high demand client sites, which results in delivery trees with long paths to the client sites that have low demand. The optimal solution for the scalable delivery protocol [Fig. 6(b)] has servers placed at client sites that are farther away (in the U.K., Canada, India), even though they have lower demand. In this case, some of the high-demand sites are served by somewhat longer but mostly shared paths, since the incremental bandwidth cost over these shared paths is lower than the incremental cost of using long paths to the low-demand sites.

If the solution in Fig. 6(a) is used for the scalable delivery protocol system, the total network bandwidth is 66% higher than for the optimal configuration of Fig. 6(b). We note that packet-loss considerations may also favor the overall shorter paths in the optimal scalable solution. Bandwidth formulas for scalable protocols that are optimized for packet-loss protection are very similar to the formulas used to obtain the results in this paper [18]. Somewhat greater protection may be required for longer paths, although the incremental bandwidth cost for this added protection is likely to be small. Detailed consideration of the impact of replica placement, path length, and path segment bandwidth on packet-loss recovery cost is beyond the scope of this paper.

Key conclusions from the experiments are

1) the optimal solutions for unicast delivery may be significantly suboptimal for scalable delivery;
2) efficient heuristics are needed to obtain near-optimal scalable systems for large numbers of client sites;
3) the optimal number of replicas involves a tradeoff between total network and total server bandwidth, which is accounted for in the model, to the first order and more precisely than in previous work [2], [9], [22], [23].

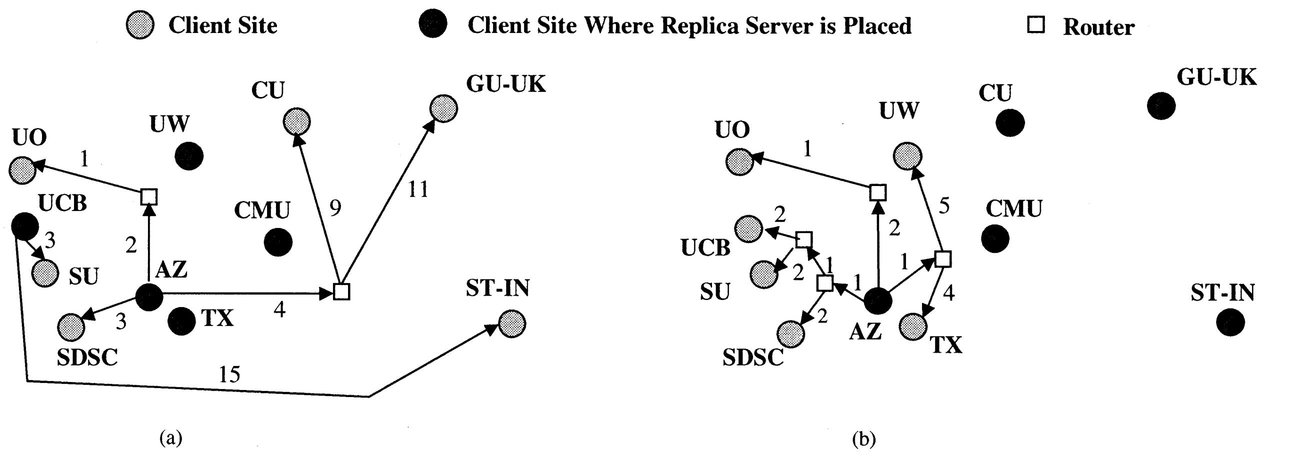


Fig. 6. Example distribution systems (11 clients, five replicas, $N_{\text{GU-UK}} = N_{\text{CU}} = N_{\text{ST-IN}} = 100$, $N_{\text{others}} = 1000$). (a) Optimal conventional system. (b) Optimal scalable system.

IV. Insights for Replica Placement and Routing

We have analyzed simple canonical topologies with two or three client sites, one or two additional possible replica placement sites, and variable hop counts between the sites, to compute the relative cost of alternative placement and routing heuristics for scalable delivery. The main results are summarized below. The details of the analysis are in [1].

Simple placement heuristics, such as placing replicas at the highest demand client site or at the most highly connected node, have been shown to perform well for conventional unicast delivery systems [16], [19], [20]. However, we found topologies in which each such heuristic is significantly suboptimal for a scalable delivery system. Furthermore, a heuristic motivated by the result presented in Fig. 6, which places a replica at the client site that would otherwise have the longest delivery path, also performs poorly for some topologies. These observations imply that a good placement heuristic for scalable systems must consider *both* the distances and the demands of the client sites.

Regarding routing heuristics for scalable delivery, we note that, for protocols such as hierarchical merging or patching, the per-hop network bandwidth is sublinear in the request rate of the clients served by the hop. Thus, for these protocols, the optimal routing involves a tradeoff between minimizing the distance from a replica to each client and maximizing path sharing among the clients. Furthermore, the tree with fewest links (t_{fl}) from the replica to the clients provides the maximum sharing, although some of the clients may be served through very long paths. On the other hand, the tree of shortest (weighted) paths (t_{sp}) provides the minimum distance to each client but may have low sharing. For periodic broadcasts and scheduled multicasts, t_{fl} is optimal, while t_{sp} is optimal for unicast delivery. Each of these trees represents one extreme in the tradeoff for the scalable protocols that have network bandwidth sublinear in the client request rate. For these sublinear protocols, we have derived the maximum cost increase (compared to optimal) of using each of t_{fl} and t_{sp} for scalable delivery in a simple topology with one replica site, two separate client sites, and variable hop counts in the shared and nonshared path segments in each delivery tree [1]. The main results are as follows.

- The maximum cost increase associated with t_{fl} is a high factor, limited only by the maximum path length in the Internet. Thus, routing based solely on maximizing sharing is not reliable for scalable systems that use protocols such as hierarchical merging or patching.
- The cost increase of using t_{sp} is bounded by $1/(f+1) \leq 100\%$, where f is the shortest distance between the clients divided by the sum of the lengths of the unshared segments in the tree of shortest paths. The upper bound can be much lower than 100% depending on the value of f. Generalizing this bound to $n > 2$ client sites yields a maximum cost increase of $(n-1) \times 100\%$ in t_{sp}. This bound is achieved if each client has a separate shortest path to the server of length H, and the distance between each pair of clients is negligible relative to H. For Internet topologies, the increase will typically be much lower.
- The cost increase associated with shortest path routing can be more significant (i.e., up to 100% for two client sites) than previously observed [25]. Thus, it may be worthwhile to use more sophisticated heuristics, such as connecting clients to the trees one at a time, based on the minimum incremental cost [25].

V. Designing Near-Optimal Distribution Systems

This section develops and evaluates a set of heuristic placement and routing algorithms using the insights in Section IV. These heuristics require significantly less execution time than solving the exact model and thus can be applied to designing large and heterogeneous near-optimal scalable systems.

We propose two heuristics for placing the replicas in the network, and three routing heuristics. The placement heuristics are "greedy"; they place one replica at a time, keeping the previous placement fixed at each step. The routing heuristics are applied for all client sites after each new replica is tentatively placed. After the routing is computed, the new replica is permanently placed, as described in Section V-A.

Both placement heuristics assume that the trees of shortest paths from each possible replica server access point to all client sites, and from each client site to all other nodes in the network, are pre-computed. Using Dijkstra's algorithm, this computation has complexity $O(|A|+|V|\times\ln(|V|))$ [5] for each of the $|S\cup C|$ trees, where $|V|$, $|S|$, $|C|$ and $|A|$ are the numbers of nodes, replica access points, client sites, and arcs, respectively, in the network.

Sections V-A and B describe the placement and routing heuristics, respectively, including their cost complexity. The heuristics are evaluated against the optimal design for a number of example Internet topologies in Section V-C.

A. Heuristics for Computing Replica Placement

For tentative placement of the first replica, we select the access point that has the lowest total cost in its tree of shortest paths to all client sites. This heuristic has complexity $O(|S|)$, can be computed efficiently and has been shown to perform reasonably well for scalable routing from one server [25].

For tentative placement of the i^{th} replica in the network, where $i > 1$, we propose two heuristics: *min-cost tsp* placement and *maximum savings* placement. The *min-cost tsp* heuristic selects, among all unused access points for placement, the one that, together with the previously placed $i-1$ replicas, has the lowest delivery cost when each client receives content from the closest replica via the shortest path. This heuristic has complexity $O(|S| \times |V|)$ and is motivated by the bound on the cost increase of using shortest path routing as compared with optimal routing for scalable delivery, discussed in Section IV. Note that this heuristic, as well as the heuristic for placement of the first replica, specifies only replica placement and not the routing algorithm that will be used.

The *maximum savings* heuristic selects, among all unused access points that are in the delivery tree(s) for the $i-1$ replicas, the one such that has maximum bandwidth savings for severing it (and all its descendents) from its tree, and delivering to those descendents from a new replica at the severed access point, using the delivery paths computed for the $i-1$ replicas. If the $i-1$ delivery trees contain no unused access points, the *min-cost tsp* placement heuristic is used. This algorithm has complexity $O(|S| \times |V|)$.

An improvement is added to each placement heuristic, after computing the routing using one of the heuristics discussed below, by moving the newly placed replica from the root to one of the internal nodes of its tree, if such a move is feasible and results in lower total cost. This is determined by using the reverse links in the path from the root to the internal node. It can be shown this algorithm has complexity $O(|V|)$. Note that the overall complexity of either placement heuristic is the complexity of pre-computing the shortest path trees.

B. Heuristics for Computing Delivery Trees

Heuristics for building delivery trees for a scheduled multicast or a periodic broadcast protocol (which have constant required bandwidth) were proposed in [11]. The heuristics derived here are for scalable protocols that have a required bandwidth that is sublinear (but not constant) in the client load, such as hierarchical merging or patching.

For each placement heuristic, we investigate three possible routing heuristics: *shortest path, min-inc-cost*, and *ordered min-cost*. Shortest path routing from the closest server is motivated in the same way as its use in replica placement.

The *min-inc-cost* heuristic, investigated in [25] for one replica, is generalized here for multiple replicas. This heuristic builds the delivery tree(s) for the replica(s) by adding one client node at a time, selecting at each step the client that can be connected to *any* of the already partially built trees with the minimum incremental bandwidth cost. It can be shown that the complexity of this heuristic is $O(|C|^2 \times |V|)$.

The o*rdered min-cost* heuristic adds client nodes in order of decreasing load and, in case of tie, in order of increasing distance to the client's closest replica. As in the *min-inc-cost* heuristic, each client is connected to the tree by the path that has minimum incremental bandwidth cost. By adding clients in a pre-defined order, the complexity of the heuristic is reduced to $O(|C| \times |V|)$. Note that this and the shortest path routing have lower complexity than the placement heuristics.

C. Performance of the Heuristics

We consider the six heuristic algorithms created by pairing one placement heuristic with one routing heuristic. This section evaluates the performance of each algorithm using AS and router topologies for six different sets of client sites with varying client heterogeneity and dispersion—the same data sets as in Section III-D. We use the hierarchical merging protocol as the basis for evaluation. We expect similar results for patching, since the same factors influence optimal design for both protocols. Experiments with near-optimal placement for periodic broadcast are left for future work.

We found that the difference among the heuristics is larger for networks with greater client dispersion. Three representative results for the total network bandwidth cost obtained with the heuristic solutions and the exact model are shown in Fig. 7. Similar results were obtained for the total server bandwidth. For comparison purposes, we also show the bandwidth cost if the optimal solution for conventional unicast delivery is used for scalable delivery.

The key observations from the results are as follows.

- All heuristics perform very well for all homogeneous and heterogeneous client loads, router topologies, and AS topologies examined.
- For all configurations examined, the heuristics that use *min-cost tsp* placement performed as well or better than the *maximum savings* placement heuristics [e.g., Fig. 7(c)] and, whenever possible, produced results that are significantly better than the conventional system design. In many cases, the *min-cost tsp* heuristics produced the optimal designs; in all other cases they produced systems that have cost no more than 16% higher than optimal.
- For all configurations analyzed using *min-cost tsp* placement, *shortest path* routing performed as well as o*rdered min-cost* routing, and each of these heuristics performs as well as the more complex *min-inc-cost* routing heuristic. However, with *maximum savings* placement, we observed cases where shortest path routing produced designs with higher cost than the other two routing heuristics (e.g., Fig. 7(b), $m = 4$ to 6).
- The maximum observed cost increase for shortest path routing with *maximum saving* placement was 28%. This is a higher discrepancy than previously observed [25], but is lower than the 100% upper bound.

We conclude that the *min-cost tsp* placement combined with either *ordered min-cost* or *shortest path* routing are the most

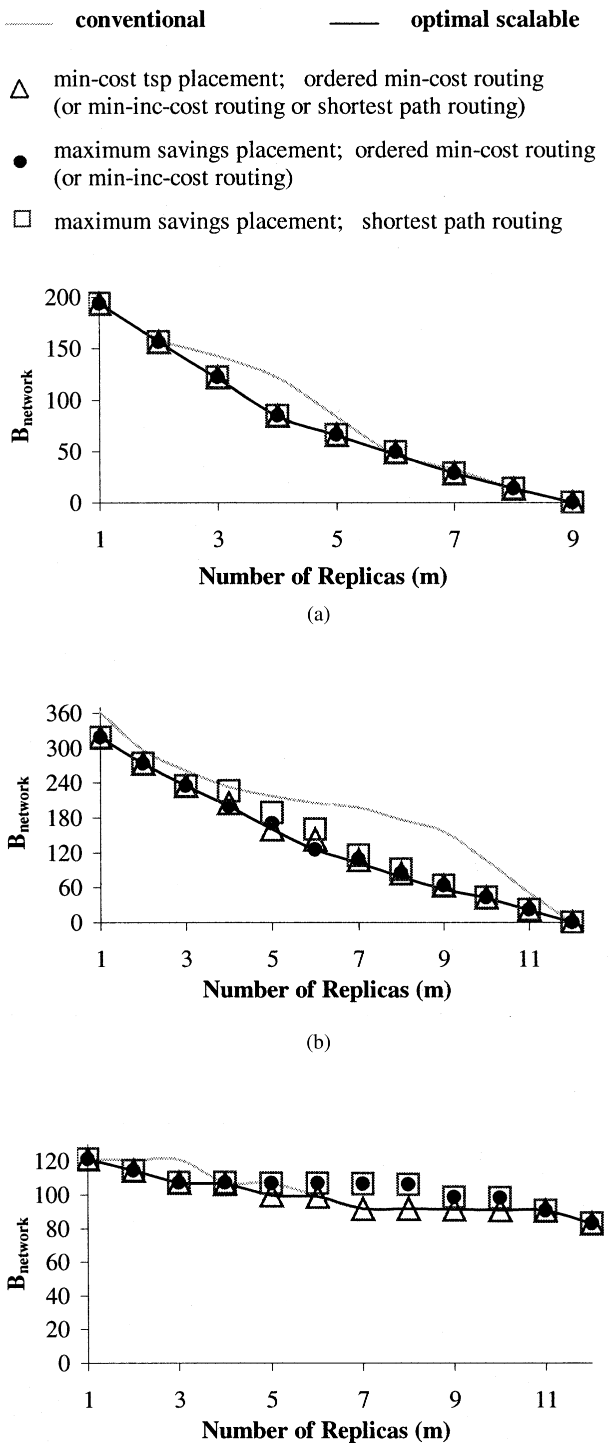


Fig. 7. Example network delivery cost comparison of different heuristic algorithms for replica placement and routing. (a) Nine heteregoneous clients in the U.S. Four clients: $N_i = 1000$ (UW, UCB, SDSC, SU). Five clients: $N_i = 100$ (UO, TX, CMU, AZ, VY). Router-level topology. (b) Twelve heterogeneous clients in the U.S., Canada, and Europe. Eight clients in the U.S. and one in Canada: $N_i = 1000$ (AZ, CMU, SDSC, SU, UCB, UW, TX, UO, CU). Three clients in Europe: $N_i = 100$ (GU-UK, SE-FR, UC-GH). Router-level topology. (c) Twelve homogenous clients in the U.S., Canada, and Europe: $N_i = 1000$ (AZ, CMU, SDSC, SU, UCB, UW, TX, UO, CU, GU-UK, SE-FR, UC-GH). AS-level topology.

promising heuristics, based on observed performance and solution efficiency. The choice between these two algorithms depends on the tradeoff between the possible implementation advantages of shortest path routing (such as load insensitivity) and the potential for shortest path routing to be suboptimal. These heuristics, which produced systems with total cost within 16% of optimality for all configurations tested, are valuable tools for designing larger and more complex scalable systems with near-optimal delivery cost.

VI. Conclusion

This paper has addressed the problem of replica placement, client request routing, and multicast stream routing in media content distribution systems employing scalable streaming protocols. Although the design of such systems is significantly more complex than the design of conventional unicast delivery systems, we formulated fairly simple optimization models for a variety of scalable protocols including hierarchical merging, patching, periodic broadcasts, and scheduled broadcasts. With the aid of additional constraints that must hold in the scalable delivery system solution, we showed that a variety of realistic scenarios can be solved exactly using available optimization software. We also showed that using the optimal conventional unicast content distribution system for scalable delivery results in network costs that can be as much as 50%–150% higher than optimal. Finally, we developed six possible heuristics that find near-optimal solutions at much lower computational cost than the exact algorithm, but which yield results of comparable quality. The best near-optimal algorithms use a greedy *min-cost tsp placement* heuristic, and either shortest path routing or a greedy *ordered min-cost* heuristic for routing client requests and multicast streams. These algorithms have a total solution complexity equal to the complexity of applying Dijkstra's shortest path algorithm to each client site and each possible server access node in the network. Furthermore, the solutions were within 16% of optimal for all scenarios evaluated.

Future work includes evaluating the performance of our heuristic solutions on larger and more diverse system design scenarios (including periodic broadcast protocols), extending the formal model to specify delivery costs for interactive client requests, investigating the impact of interactive client loads on the optimal and near-optimal content delivery systems, and investigating the problem of optimal joint placement and routing for multiple objects.

References

[1] J. M. Almeida, "Streaming content distribution networks with minimum delivery cost," Ph.D. dissertation, Univ. Wisconsin, Madison, 2003.

[2] J. M. Almeida, D. L. Eager, M. C. Ferris, and M. K. Vernon, "Provisioning content distribution networks for streaming media," in *Proc. INFOCOM*, New York, June 2002.

[3] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon, "Analysis of educational media server workloads," in *Proc. NOSSDAV*, Port Jefferson, NY, June 2001.

[4] J. Apostolopoulos, T. Wong, W.-T. Tan, and S. Wee, "On multiple description streaming with content delivery networks," in *Proc. INFOCOM*, New York, June 2002.

[5] M. Barbehenn, "A note on the complexity of Dijkstra's algorithm for graphs with weighted vertices," *IEEE Trans. Comput.*, vol. 47, p. 263, Feb. 1998.

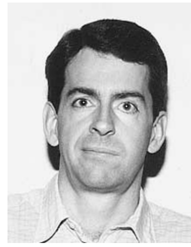
[6] A. Bar-Noy and R. E. Ladner, "Competitive on-line stream merging algorithms for media-on-demand," in *Proc. ACM-SIAM Symp. Discrete Algorithms*, Washington, DC, Jan. 2001.

[7] A. Broido and K. Claffy, "Internet topology: Connectivity of IP graphs," in *Proc SPIE Int. Symp. Convergence of IT and Communication*, Denver, CO, Aug. 2001.

[8] A. Brooke, D. Kendrick, and A. Meeraus, *GAMS: A User's Guide*. San Francisco, CA: Scientific, 1988.
[9] D. L. Eager, M. C. Ferris, and M. K. Vernon, "Optimized caching in systems with heterogeneous client populations," *Perf. Eval., Special Issue on Internet Performance Modeling*, vol. 42, no. 2/3, Sept. 2000.
[10] D. Eager, M. Vernon, and J. Zahorjan, "Minimizing bandwidth requirements for on-demand data delivery," *IEEE Trans. Knowl. Data Eng.*, vol. 13, pp. 742–757, Sep./Oct. 2001.
[11] Z. Fei, M. Ammar, and E. Zegura, "Multicast server selection: Problems, complexity and solutions," *IEEE J. Select. Areas Commun.*, vol. 20, pp. 1399–1413, Sept. 2002.
[12] L. Gao and D. Towsley, "Supplying instantaneous video-on-demand services using controlled multicast," in *Proc. IEEE Int. Conf. Multimedia Computing and Systems (ICMCS)*, Florence, Italy, June 1999.
[13] K. Hua and S. Sheu, "Skyscraper broadcasting: A new broadcasting scheme for metropolitan video on demand systems," in *Proc. ACM SIGCOMM*, Cannes, France, Sept. 1997.
[14] K. Hua, Y. Cai, and S. Sheu, "Patching: A multicast technique for true video-on-demand services," in *Proc. ACM Multimedia*, Bristol, U.K., Sept. 1998.
[15] M. R. Garey and D. S. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.
[16] S. Jamin, C. Jiu, A. Kurc, D. Raz, and Y. Shavitt, "Constrained mirror placement on the Internet," in *Proc. INFOCOM*, Anchorage, AK, Apr. 2001.
[17] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," *IEEE/ACM Trans. Networking*, vol. 8, pp. 568–582, Oct. 2000.
[18] A. Mahanti, D. L. Eager, M. K. Vernon, and D. Sundaram-Stukel, "Scalable on-demand media streaming with packet loss recovery," in *Proc. ACM SIGCOMM*, San Diego, CA, Aug. 2001.
[19] L. Qiu, V. Padmanabhan, and G. Voelker, "On the placement of web server replicas," in *Proc. INFOCOM*, Anchorage, AK, Apr. 2001.
[20] P. Radoslavov, R. Govindam, and D. Estrin, "Topology-informed Internet replica placement," *Comput. Commun.*, vol. 25, no. 4, pp. 384–392, Mar. 2002.
[21] H. Tan, D. L. Eager, and M. K. Vernon, "Delimiting the effectiveness of scalable streaming protocols," in *Proc. Performance*, Rome, Italy, Sept. 2002.
[22] O. Verscheure, C. Venkatramani, P. Frossard, and L. Amini, "Joint server scheduling and proxy caching for video delivery," *Comput. Commun.*, vol. 25, no. 4, pp. 413–423, Mar. 2002.
[23] B. Wang, S. Sen, M. Adler, and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," in *Proc. INFOCOM*, New York, June 2002.
[24] S. Zhang, "Further analysis of the eTeach client arrivals," unpublished.
[25] Y. Zhao, D. L. Eager, and M. K. Vernon, "Network bandwidth requirements for scalable on-demand streaming," in *Proc. INFOCOM*, New York, NY, June 2002.

Jussara M. Almeida received the B.Sc. and M.Sc degrees in computer science from the Federal University of Minas Gerais (UFMG), Brazil, in 1994 and 1997, respectively, and the M.Sc. and Ph.D. degrees in computer science from the University of Wisconsin, Madison, in 1999 and 2003, respectively.

Currently, she is a Research Associate in the Department of Computer Science, UFMG. Her research interests are in the areas of performance analysis, modeling, and large distributed systems, with focus on web and streaming media systems.



Derek L. Eager received the B.Sc. degree in computer science from the University of Regina, Regina, SK, Canada, in 1979, and the M.Sc. and Ph.D. degrees in computer science from the University of Toronto, Toronto, ON, Canada, in 1981 and 1984, respectively.

He is currently a Professor in the Department of Computer Science, University of Saskatchewan, Saskatoon, SK. His research interests are in the areas of performance evaluation, streaming media and bulk data content distribution, and distributed systems.

Mary K. Vernon received the B.S. degree (Hons.) in chemistry and the Ph.D. degree in computer science from the University of California at Los Angeles.

In 1983, she joined the Computer Science Department, University of Wisconsin, Madison, where she is currently a Professor of computer science and industrial engineering. Her research interests include performance analysis techniques for evaluating computer/communication system design tradeoffs, parallel/distributed architectures and software systems, computer system security, network protocols, and scalable content distribution.

Dr. Vernon has served on the Editorial Board of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the Board of Directors of the Computing Research Association, and as Chair of the ACM SIGMETRICS. She received a National Science Foundation Presidential Young Investigator Award in 1985, the ACM Fellow Award in 1996, and a University of Wisconsin Vilas Associate Award in 2000.

Stephen J. Wright received the B.Sc. (Hons.) and Ph.D. degrees from the University of Queensland, Qld., Australia, in 1981 and 1984, respectively.

After holding positions at North Carolina State University, Argonne National Laboratory, and the University of Chicago, he joined the University of Wisconsin, Madison, in 2001 as a Professor of computer sciences. His research interests lie in theory and algorithms for optimization problems involving real variables, and in applications of optimization.

Dr. Wright serves as the Editor-in-Chief of *Mathematical Programming, Series B*, and currently serves on the Editorial Boards of the *SIAM Journal of Scientific Computing* and *Mathematical Programming, Series A*.