Multicast Delivery of Web Pages or How to Make Web Servers Pushy (Position Paper) *

*Mostafa H. Ammar** *Kevin C. Almeroth* † *Russ J. Clark* ‡ *Zongmin Fei**

*College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
{ammar,fei}@cc.gatech.edu

†Dept. of Comp. Sci.
University of California
Santa Barbara, CA 93106
almeroth@cs.ucsb.edu

‡Empire Technologies, Inc.
541 Tenth Street, N.W.
Atlanta, GA 30318
rjc@empiretech.com

1 Making the Case

The Web has gained tremendously in popularity over the last several years. Web administrators are struggling to upgrade their servers to handle the huge volumes of requests. One approach to handling the large volume of requests has been to simply buy more hardware. Other solutions have been investigated including improvements in the HTTP protocol, the use of transparent server replication[1] and caching of hot pages[2]. At issue is how Web pages can be delivered to increasing numbers of users given limited server capacity and network bandwidth.

Another technique that can be used in conjunction with caching and replication to improve scalability is *multicast delivery.* With this approach Web pages are delivered to multiple awaiting clients using one server response instance and using underlying network support for point-to-multipoint communication. Although multicast delivery has long been viewed as an approach to provide scalable services [3, 4, 5, 6, 7], its use for multicast Web delivery on an end-to-end basis (multicast from server directly to clients) has received less attention. This has been for mainly three reasons:

1. The Web is largely viewed as a "pull" service where clients' needs for pages are satisfied through individualized point-to-point transmission.
2. There is lingering doubt over the usefulness of multicast Web page delivery. The benefits of the server and network aggregation provided by multicast delivery are clear. What is in doubt is whether these benefits may be more than offset by the overheads within the server and the network to support multicast delivery.
3. End-to-end multicast is not widely deployed over the commercial Internet.

Our work addresses the first two points above explicitly. First, access to Web sites typically follows a skewed pattern with a small number of *hot* pages being accessed very frequently, a larger number (but still small) of *warm* pages being accessed with moderate frequency and a large number of *cold* pages being accessed infrequently or in some cases rarely. Multiple requests received for the hot and warm pages can be aggregated and responded to once by the server. Multicast delivery support within the network can insure a similar form of bandwidth aggregation. An important aspect of our approach is that the system can be built such that clients of a multicast delivery Web server can maintain the illusion that they are interacting with the server according to a pull paradigm.

In addition to defining an architecture and a set of protocols for a multicast Web service, our work also targets an understanding of the basic performance tradeoffs of such an approach. Although simple analysis has clearly shown the advantages of multicast delivery [3, 4], its benefits in a real Internet service context also need to be demonstrated. This is made considerably harder because of the widely

*This work is supported in part by research grants from IBM, Intel, and NSF under contract number NCR-9628379

varying services and applications being grouped under the WWW umbrella.

Finally we mention that multicast deployment over the Global Internet is just a matter of time. Efforts by industry proponents of multicast communication such as the IP Multicast Initiative (www.IPMulticast.com) have resulted in increased awareness of the benefits of multicast by application builders as well an Internet Service Providers. Furthermore, the first commercial offering of a multicast Internet service has been announced by UUnet (one of the largest ISPs) (www.uunet.com/lang.en/products/uucast.shtm).

The remainder of this paper is organized as follows. Section 2 describes the basic design of a multicast delivery Web server. Section 3 gives an overview of the benefits and costs of multicast delivery and how one may judge the cost/benefit tradeoffs. Section 4 describes a set of research issues that need to be tackled in the design and deployment of a multicast delivery web service. Section 5 summarizes our experience with the performance of Web multicast delivery prototypes and simulation [8, 9]. Section 6 concludes the paper.

2 Scalable Web Delivery Using Multicast–The Basic Design

Figure 1 shows the architecture of a Web server capable of delivering pages using cyclic multicast, reliable multicast, and reliable unicast. Requests for TCP connections arrive from (perhaps only some) page requesters and are queued until the server can process them. When the server establishes the connection, the user transmits the page request. At this point, the server decides which protocol will be used to serve the request. The decision is based mainly on the popularity of the requested page. Extremely popular (hot) pages are served via cyclic multicast. Moderately popular (warm) pages are served using reliable multicast. Other (cold) pages are served in the traditional way using unicast TCP connections. The decision on which pages to serve via cyclic multicast can be user controlled (through commands sent as part of the request) or can be driven by information that the server maintains in a dynamic or static fashion expressing the popularity of certain pages at the current time.

In the traditional Web architecture an HTTP connection is built on top of a TCP connection. For

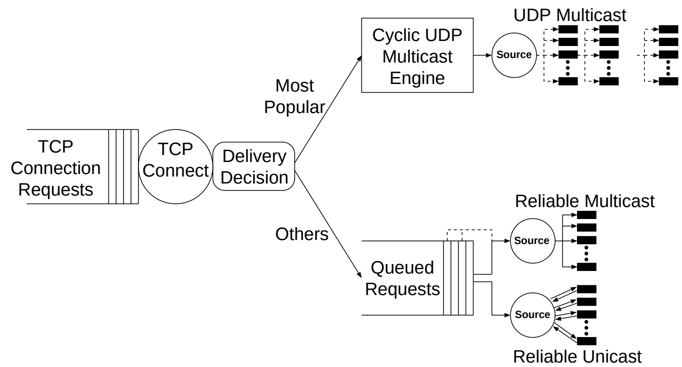

Figure 1: Multicast Web Server Architecture.

multicast delivery we build the HTTP connection on top of multicast delivery facilities (reliable or best-effort). These protocols are built around the IP multicast capabilities. In such an environment, the multicast group formed by the clients requesting the same page needs to be identified by a single multicast IP address and all the clients need to explicitly join that group.

The reliable multicast delivery option requires the use of a reliable multicast transport protocol which insures the delivery of packets to recipients in the multicast group. Such protocols typically will use some form of feedback about reception from the clients which results in the retransmission of missing packets.

The cyclic multicast delivery option in our server does not require such reliability and is most effectively used to deliver a site's most popular and heavily requested pages. The server is capable of delivering multiple pages simultaneously using the cyclic multicast delivery engine. For each page designated for delivery via cyclic multicast:

1. The page to be delivered including all embedded files is divided into a number of *chunks*.
2. All chunks in a page are sequentially transmitted from the server to the group of receivers using an agreed-upon multicast address. A single transmission of each and every chunk constitutes one *cycle*.
3. Receivers join the appropriate multicast group and remain a member until an error-free copy of all chunks has been received. If a chunk is not received the receiver remains part of the group until the missed chunk is re-transmitted

in a subsequent cycle and correctly received.

4. The server continues cyclic transmission as long as it believes there is at least one requester trying to receive the page. This *stopping condition* can be estimated using a formula based on our analysis described in [9].

We now give a brief overview of the operation of the client and server processes in our proposed multicast Web extension.

Client Request Processing and Response Reception

The multicast capable client performs the following steps when requesting a page from a Web server.

1. *Determine group for request:* The client determines the multicast group address a server would use to send this page if it were sent via multicast. We consider various alternatives for how the multicast addresses may be determined in our work.

2. *Join group:* The client begins listening for multicast packets addressed to the previously determined group.

3. *Send query to server:* The client sends the request to the server. This is done using the current connection request model of HTTP. Along with the request message the client sends an indication that it will support a multicast response for this query. In the case of pages delivered using the cyclic multicast engine, the propagation of the client's request to the server is not necessary.

4. *Listen for response:* The response could come as a response over the TCP request connection, over a reliable multicast connection or over of the cyclic channels. If no response arrives within an expected timeout the client should resubmit the request with an indication that multicast should not be used. This will cause the server to reply to this second request over the request HTTP connection. This special retry procedure, which we expect to be a rare occurrence, is needed to overcome problems that may be encountered in multicasting a response.

5. *Close Connections and Leave Group:* Once the response is received the client can close its connection(s) with the server. The client also stops listening to the multicast address.

Server Request Processing and Response Transmission:

The multicast capable server performs the following steps when processing a request.

1. *Receive request:* The server listens for connection requests and receives the HTTP request once a connection is established.

2. *Determine delivery option:* The server determines, based on the identity of the request, the desired delivery option. If cyclic multicast is the selected option, the request is logged and the incoming TCP connection carrying the request is closed.

3. *Schedule and service reliable-delivery request:* For requests slated for reliable delivery, the server determines when to respond to the request. The scheduling of response transmissions is an issue we investigate in our research. Requests scheduled for transmission are serviced via a reliable multicast connection if there are multiple clients awaiting the page transmission, otherwise pages are transmitted using the normal TCP connection.

3 Multicast Delivery: Benefits and Costs

A system such as the one described above that delivers Web pages using multicast delivery has the following potential benefits:

- Reduced request traffic for the hot pages being multicast cyclically.

- Reduced bandwidth consumption on the link from the server to the network as a result of the aggregation of requests.

- Reduced bandwidth consumption within the network as a result of the bandwidth aggregation inherent in IP multicast routing.

- Reduced server processing since the server will maintain fewer and shorter TCP connections with individual clients.

Multicast delivery also has some overhead associated with its deployment:

- Bandwidth and processing overhead of multicast routing tree maintenance within the network.
- Overhead of additional request processing required to discriminate between request types within the server and other functions not normally performed by a unicast Web server.
- Overhead in the operation of a reliable multicast protocol that is needed for the transport of the warm pages.

Judging whether the benefits outweigh the costs is hard in this case, since these costs and benefits are quantifiable in different units and are "paid for" by a variety of constituents.

It is probably reasonable to make the assumption that large-scale multicast deployment within the Internet will be sufficiently efficient that the costs associated with supporting it will be easily offset by the bandwidth savings due to multicast's bandwidth aggregation. Note that this is essentially an "axiom" for all multicast development. This by no means should "close the book" on this issue which also needs to be examined in detail. Indeed, it may turn out that network bandwidth savings due to multicast more than offset its network support costs. ISP's may then be able to offer pricing incentives for multicast sources, which in turn may make service providers willing to incur a slightly higher cost for their end-systems as long as they are capable of using multicast delivery.

Understanding the cost/benefit tradeoffs within end-systems is perhaps a little easier. In this case one can focus on one performance measure of a web-page delivery system, namely, the client-perceived response time. Assuming that the cost/benefit tradeoffs within the network can be accounted for as discussed above, if a web delivery system can deliver faster response time for the same request load using the same server hardware, then it is reasonable to conclude that the benefits outweigh the costs.

In the next sections we discuss some aspects of the system design that have a direct bearing on this cost/benefit tradeoff. We also report on some of our experience with prototypes and simulations that have looked at various aspects of the performance and the costs of a web delivery system.

4 Multicast Web Service Research Issues

We have identified the following set of research issues in relation to the design, implementation and deployment of a multicast Web service:

1- *Techniques for Multicast Addressing*: The problem we consider is how to get all the clients requesting the same page and the server to agree on the same multicast address. Our initial proposal uses an *address mapping* function that maps a page's URL into a multicast address. All multicast clients and servers use the same function and agreement on the same address is, therefore, possible. Other options include address selection at the server or through other directory mechanisms.

2-*Cyclic Multicast Server Operation*

Realization of the cyclic multicast service option requires the investigation of the following:

- *Server Request State*: With the availability of cyclic multicast it is no longer necessary for all clients to transmit their requests to the server. Whereas this has clear advantages, it is not necessarily desirable from some content provider's perspective. An investigation is needed for how to best accommodate such provider's needs and yet reap the full benefits of the cyclic multicast delivery option.
- *Page Chunking:* The division of a page into chunks helps the receiver re-construct the Web page as it is delivered. Chunks can then be further subdivided into one or more data packets. This is especially useful when chunks are received out of order due to network conditions including loss or congestion. Chunking can range in complexity from simple page segmentation to options that include page structure information in each chunk. The simplest form of chunking, page segmentation, creates chunks of uniform length independent of page

content. An alternate form of chunking uses HTML tags (where they exist) and embedded file markers to break a page into displayable groups of chunks. An image, table, list, or other Web page object will only be displayed when all chunks in the group have been received.

- *Congestion Control:* The cyclic multicast engine will need to incorporate some mechanism for congestion control in order not to overburden the network with its transmissions. The mechanisms are needed to insure good network "citizenship" and to insure reasonable loss rates.

3- *Request Scheduling at the Server*: For the reliable delivery option it is necessary to use a central *request* queue. This queue provides a rendezvous point for matching common requests. In the context of our earlier work on broadcast delivery we have analyzed a number of possible queuing disciplines [10]. Our research considers these scheduling disciplines in the context of the Web service. More recent work in this area [11] which builds on our own work has revealed some interesting new scheduling ideas.

4- *Interaction with Multicast Routing*: By multicasting responses we are able to satisfy multiple client requests with a single transmission from the server. This clearly saves on server resources and allows it to handle a heavier request load. Multicasting can also potentially save on the network bandwidth required to satisfy client requests. Whether bandwidth savings are possible is very much dependent on the characteristics of the multicast routing protocols being used.

Another issue to consider with multicast routing protocols is the time lag between the point where a host requests to join a group and the point where it is able to receive messages sent to the group. Clients join multicast groups by issuing requests to network routers. Propagation of this information across the network eventually results in the particular client's host machine receiving multicast messages destined to the group. How this information propagates and how long it takes is a function of the particular routing protocol in use.

5- *Reliable Multicast Consideration*: Multicast delivery of Web pages requires the use of a reliable multicast protocol (in the same way that HTTP uses TCP as the reliable unicast protocol). Our initial prototype used a connection-oriented multicast transport protocol that we developed [12]. Our work is not dependent on this protocol and it is possible to use one of the many recently proposed protocols [14, 15, 16, 17]. A protocol newly developed in our group that targets a mobile database application [13] seems to be a good candidate for our multicast Web environment.

6- *Interaction with Multicast Deployment Model:* The benefits derived from the use of our multicast Web service proposal will be dependent to a large extent on how multicast will be deployed in the Global Internet. UUCast's model charges the cost of multicast to the source with no additional charge for receivers. If this model persists in other offerings, it will have implications on which type of multicast transport and application protocols make economic sense. For example protocols requiring receivers to multicast packet retransmission requests will not be feasible. Also the actual differential between the unicast and multicast traffic charges will affect the feasibility of options such as our proposed cyclic multicast transmission. Uncertainty in this area dictates extremely flexible designs that can adapt well to the final multicast deployment scenario.

5 Performance of Multicast-Delivery Web Service

Our research to date [8, 9] on the provision of multicast Web service has included performance evaluation of the two multicast delivery options. We summarize these performance results below.

5.1 Reliable Multicast for Warm Pages

The earliest analysis for on-demand multicast delivery (such as the one we propose for the transmission of warm Web pages) can be found in [3, 4]. This analysis confirms the intuitive expectation that multicast delivery can dramatically reduce response time under heavy load and also increase the server's capacity.

In [8] we built a prototype reliable-multicast Web delivery system using our connection-oriented reliable multicast protocol [12]. We performed experiments using the prototype the details of which can be found in [8]. The results of the experiments are shown in Figure 2 where the re-

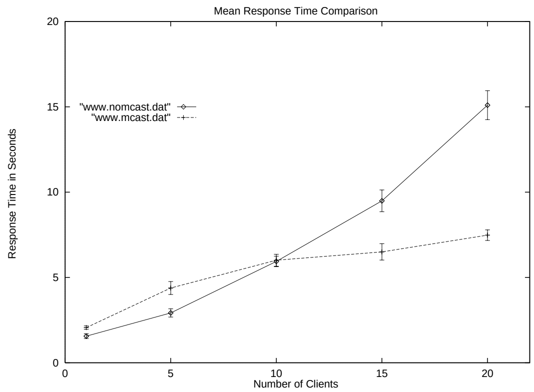

Figure 2: WWW Mean Response Time Comparison

sponse time experienced by clients is plotted against the number of clients accessing the server. The line labeled "www.nomcast.data" indicates the response times using individual delivery with a dedicated response for every request. The line labeled "www.mcast.data" indicates the response times using multicast delivery. The error-bars around each data point indicate the 95% confidence interval for the sample mean.

The figure demonstrates the benefits of multicast delivery and also shows the overhead of using multicast and its slight negative effect when the load is light.

5.2 Cyclic Delivery for Hot Pages

We next show a sample result from our extensive simulation of a Web server that includes cyclic multicast and reliable unicast. The complete set of results can be found in [9] which includes simulations of systems with all delivery options.

Simulation Model

The server in our simulation has a maximum number of simultaneous outgoing streams that it can support. We assume that from the point of view of a server, streams carrying cyclic multicast and reliable unicast are equivalent and interchangeable.

We assume a large number of potential clients that make requests according to a Poisson process and that all requested pages are of the same size. Each server has L pages that it can serve. We assume that the probability that a request is for a particular page follows a Zipf distribution[18]. That is, if we label the pages in decreasing order of popularity, the probability that a particular request is for page i is given by κ/i where κ is the normalization constant $[\sum_{i=1}^{L} 1/i]^{-1}$.

We use the time to transmit a page in chunks (i.e., the time for transmitting one cycle) as our time unit. We further assume there is no propagation delay in making a request, or in sending chunks from the server. Requests are queued and wait if the server does not have an available stream.

Reliable unicast transmissions out of the server use a selective reject ARQ protocol.

In our evaluation we focus primarily on *response time* defined as the time it takes to completely receive a page, measured in cycles. A cycle in the multicast case is simply the time to transmit all chunks in a page. For unicast, a cycle is the time to transmit all chunks to a single user assuming no packet loss.

Figure 3 shows the response time for each of 10000 simulated and completed requests. The systems being compared are cyclic multicast and reliable unicast. Both servers can deliver up to five streams simultaneously and both offer five different WWW pages. The probability of packet loss is 5%. The request arrival rate for this graph is 4.76 requests per second which is just below the maximum rate that the unicast server can handle. (This maximum rate can be computed using Little's Theorem.) A request rate any greater than the maximum rate would cause the unicast server to be unstable, i.e., it would receive requests faster than it could service them.

More results from our simulation can be found in [9].

5.3 Cyclic Multicast Prototype Evaluation

We have implemented a prototype cyclic multicast engine in order to judge the practical feasibility of our proposed approach. Here we report on some preliminary measurements of performance from ex-

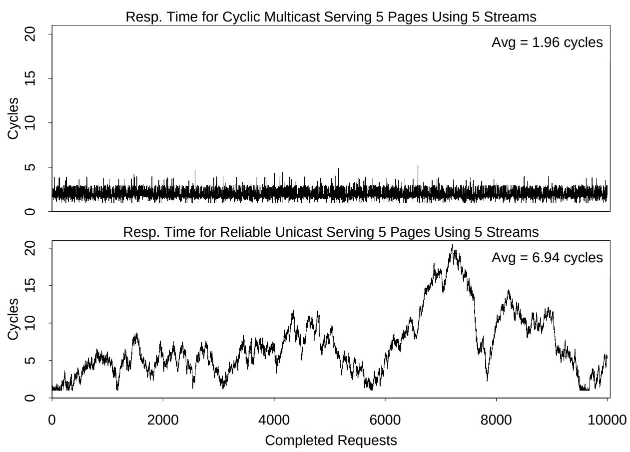

Figure 3: Comparison of response times.

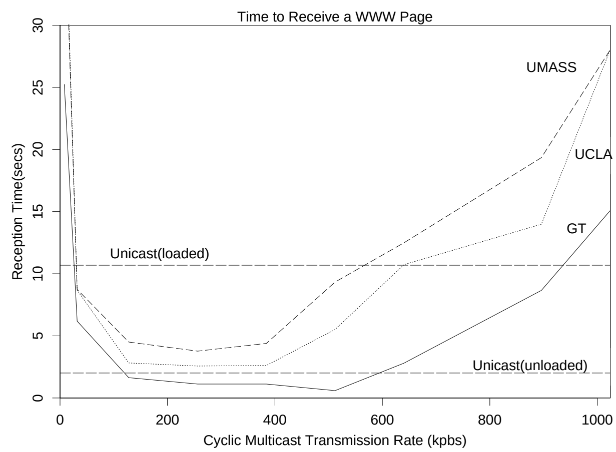

Figure 4: Observed time to receive WWW pages.

periments using our prototype. We measure two components of user-perceived response time. The first is *reception time*, the time between when a receiver gets the first chunk of a page until all chunks have been received. The second component is the *first-packet time* defined as the time from when the user issues a request until the first chunk is received. For these two measures we are interested in the performance of the cyclic multicast prototype and how it compares with a typical unicast Web server.

Our experiments were conducted over the MBone with a server at Georgia Tech and clients located at Georgia Tech (GT), the University of California at Los Angeles (UCLA), and the University of Massachusetts at Amherst (UMASS). The clients all request the same page which requires 50 chunks, each of which is 512 bytes. Figure 4 shows the reception time at the three clients for different transmission rates. The reception time decreases quickly as the sending rate increases up to 128 Kbps. Between 128 Kbps and 480 Kbps the time to receive remains relatively flat. This version of cyclic multicast does not implement congestion control, and so as the transmission rate increases, chunk loss starts to increase. At 512 Kbps and above, the transmission rate increases to the point where congestion negatively affect reception time. As a performance benchmark, Figure 4 also includes the reception time for a page delivered via reliable unicast (actually HTTP over TCP) for an unloaded and loaded Web server. Both the client and the server were at Georgia Tech in the unicast experiments. It should be noted that the cyclic multicast server reception times will be unaffected for differently loaded servers.

In the second set of experiments we measure the *first-packet time*. One client was placed at UCLA with the server at GT. For reliable unicast this time includes setting up the TCP connection, making the page request, and then waiting to receive the first chunk. The first line in Table 1 shows that on average (over 100 experiments) it takes 264 msecs to receive the first chunk.

For cyclic multicast, join time depends on whether the page is already being transmitted and whether there are any branches of the multicast tree close to the new receiver. In the worst case, the page is not being transmitted and there are no other receivers. In this scenario, the user will request the page via a TCP connection while at the same time joining the appropriate multicast group. The time to receive the first chunk will be the maximum of (1) the time it takes for the server to receive the request and begin sending the page (comparable with the TCP join time value above), and (2) the time for the receiver to join the multicast group. The two cyclic multicast entries in Table 1 assume the page is being transmitted and reports the time to join the multicast group. The first entry shows the case where there are no receivers and the multicast tree must be constructed between UCLA and GT. The second entry considers the case where there is another group member close by. We conclude here that the multicast join time will tend to be, in the worst case, on the same order as the TCP first-packet time. Our experiments with the prototype confirm that cyclic multicast of a page can drastically improve response time, especially for loaded servers.

	Response Time	
	Avg	Std. Dev.
TCP (reliable unicast)	264 ms	135 ms
Multicast: one receiver	241 ms	57 ms
Multicast: recvr at UCLA	149 ms	68 ms

Table 1: Time from request to first packet.

6 Concluding Remarks

In this paper we argue the case for the deployment of multicast Web delivery. We analyze the various costs and benefits of such deployment and discuss the research and practical issues that need to be investigated before it becomes a reality.

We also sample some of the performance results obtained in our research that demonstrate the benefits of end-to-end Web multicast. Our results also show that ideally, servers should integrate all three delivery options (cyclic and reliable multicast and reliable unicast) for best performance and flexibility.

References

[1] E. D. Katz, M. Butler, and R. McGrath, "A scalable HTTP server: the NCSA prototype," *Computer Networks and ISDN Systems*, vol. 27, pp. 155–164, 1994.

[2] H. Braun and K. Claffy, "Web traffic characterization: an assessment of the impact of caching documents from NCSA's web server," *Computer Networks and ISDN Systems*, vol. 28, pp. 37–51, Dec 1995.

[3] J. W. Wong and M. H. Ammar, "Analysis of broadcast delivery in a videotex system," *IEEE Transactions on Computers*, vol. 34, pp. 863–866, September 1985.

[4] J. W. Wong and M. H. Ammar, "Response time performance of videotex systems," *IEEE Journal on Slected Areas in Communication*, vol. 4, pp. 1174–1180, October 1986.

[5] M. Ammar and J. Wong, "On the optimality of cyclic transmission in teletext systems," *IEEE Transactions on Communications*, vol. 35, pp. 68–73, January 1987.

[6] G. Herman, G. Gopal, K. Lee, and A. Weinrib, "The datacycle architecture for very high throughput database systems," in *ACM SIGMOD*, 1987.

[7] S. Acharya, M. Franklin, and S. Zdonik, "Dissemination based data delivery using broadcast disks," *IEEE Personal Communications*, vol. 2, Dec 1995.

[8] R. J. Clark and M. H. Ammar, "Providing a scalable web service using multicast delivery," in *Proceedings of the IEEE Workshop on Services in Distributed and Networked Environments*, pp. 19–26, June 1995. (also in *Computer Networks and ISDN Systems*, vol. 29, pp. 841-858, 1997.)

[9] Almeroth, K. C., Ammar, M. H., Fei, Z., "Scalable Delivery of Web Pages Using Cyclic Best-Effort (UDP) Multicast," Proceedings of IEEE INFOCOM 98, March 1998.

[10] H. Dykeman, J. Wong, and M. Ammar, "Scheduling algorithms for videotex systems under broadcast delivery," in *IEEE International Conference on Communications '86*, pp. 1847–1851, IEEE, June 1986.

[11] D. Aksoy, M. Franklin, "Scheduling for Large-Scale On-Demand Data Broadcasting," Proceedings of IEEE INFOCOM 98, March 1998

[12] R. Talpade and M. H. Ammar, "Single connection emulation (SCE): An architecture for providing a reliable multicast transport service," in *15th IEEE Intl Conference on Distributed Computing Systems*, June 1995.

[13] S. Mahajan, M. Donahoo, S. Navathe, M. Ammar, S. Malik, "Grouping techniques for update propagation in intermittently connected databases," Proceedings of the IEEE International Conference on Data Engineering, February 1998, Orlando, Florida.

[14] B. Whetten, T. Montgomery, and S. Kaplan, "A high performance totally ordered multicast protocol," tech. rep., University of California, Berkeley, 1995.

[15] J. C. Lin and S. Paul, "RMTP: A reliable multicast transport protocol," in Proceedings of IEEE INFOCOM 96 , March 1996.

[16] S. Floyd, V. Jacobson, S. McCanne, L. Zhang, and C.-G. Liu, "A reliable multicast framework for lightweight sessions and application level framing," in *Proceedings of ACM SIGCOMM*, pp. 328–341, 1995.

[17] H. Holbrook, S. Singhal, and D. Cheriton, "Log-based receiver-reliable multicast for distributed interactive simulation," in *Proceedings of ACM SIGCOMM*, pp. 342–357, 1995.

[18] G. Zipf, *Human Behavior and the Principle of Least Effort.* Reading, MA: Addison-Wesley, 1949.