I. J. Communications, Network and System Sciences, **2008, 4, 285-385**
Published Online November 2008 in SciRes (http://www.SciRP.org/journal/ijcns/).

Analysis of a Cyclic Multicast Proxy Server Architecture

Olatunde ABIONA[1], Tricha ANJALI[2], Clement ONIME[3], Lawrence KEHINDE[4]
[1]*Department of Computer Information Systems, Indiana University Northwest, Garry, IN 46408, USA*
[2]*Electrical and Computer Engineering Department, Illinois Institute of Technology, Chicago, IL 60616, USA*
[3]*Information and Communication Technology Section, Abdus Salam International Centre for Theoretical Physics, I-34014 Trieste, Italy*
[4]*Department of Engineering Technologies, Texas Southern University, Houston, Texas 77004, USA*
E-mail: [1]oabiona@iun.edu, [2]tricha@ece.iit.edu, [3]onime@ictp.it, [4]kehindelo@tsu.edu

Received September 6, 2008; revised October 7, 2008; accepted October 10, 2008

Abstract

The exponential growths of the World Wide Web (WWW) users have made the deployment of proxy servers popular on a network with limited resources. WWW clients perceive better response time, improved performance and speed when response to requested pages are served from the cache of a proxy server, resulting in faster response times after the first document fetch. This work proposes cyclic multicast as a scalable technique for improving proxy server performance for next generation networks. The proposed system uses a cyclic multicast engine for the delivery of popular web pages from the proxy server cache to increasingly large users under limited server capacity and network resources. The cyclic multicast technique would be more efficient for the delivery of highly requested web pages from the cache to large number of receivers. We describe the operation of the cyclic multicast proxy server and characterized the gains in performance.

Keywords: Caching, Cyclic Multicast, Reliable Unicast, Scalable Data Delivery, Next Generation Network

1. Introduction

A proxy server is a server that sits between a client application, such as a web browser, and a real server. It intercepts all requests to the real server to see if it can fulfill the requests by itself. Otherwise, it forwards the request to the real server. Proxy servers have two main purposes on a network, firstly, to improve network performance through the delivery of previously request objects from the cache and secondly to filter request, i.e. preventing users from accessing some specific sets of website. Proxy caching has been widely used to cache static (text/image) objects on the Internet so that subsequent requests to the same objects can be served directly from the proxy server without contacting the real server.

In order to reduce client perceived access latencies as well as server/network loads, caching of frequently used data at proxies close to the client is an effective technique. This will enhance the availability of objects and reduce packet losses, as local transmission is more reliable than remote transmission.

Proxy caching has become one of the vital components in all web systems. Streaming media, in particular those prestored can have significant performance improvements from proxy caching, due to their static nature in content. Hence proxy servers have found useful applications in media streaming, video on demand and large scale multimedia applications [1–5].

Over the last several years, the WWW has gained tremendous popularity; similarly, the number of WWW users on the Internet has grown exponentially. Making the system administrator to continually battle with ways of improving response times due to large volumes of users' request. Different approaches have been used to solve the problem of scalability; one of such approaches is to simply buy more powerful hardware to upgrade the servers. This is not a cost effective or scalable solution as this approach may fail with increasing WWW users. Another solution is the improvement of the Hyper Text Transport Protocol (HTTP) to reduce the latency associated with HTTP communication by removing overhead of creating a new connection for each HTTP request [6]. Another solution is replicating transparent servers' at the most popular websites [7], caching of hot pages [8] and multicast delivery [9].

The focus of this work is to investigate cyclic multicast architecture for the delivery of WWW pages to increasingly large numbers of user given limited server capacity and network resources for next generation networks. Access pattern to files follows a Zipf-like distribution [10]. Access to website typically follows a skewed pattern, namely; small number of popular pages (hot pages) accessed very frequently, a large number of warm pages accessed with moderate frequency and a large number of cold pages accessed a few times or rarely. We explore the cyclic multicast for the transmission of popular (hot and warm) requested web pages and reliable unicast for other (cold) pages. With this option, web pages are delivered to multiple requesting clients using a single server response based on the network support for point to multipoint communication.

The cyclic multicast option is expected to be more efficient for the delivery of highly requested web pages (hot and warm pages) to large number of users. With this option, the web page is broken into chunks, cyclically transmitted and clients can listen at any point in time in the transmission, and continue to listen until all of the data is received.

The rest of the paper is organized as follows. In section 2 we review related work and in section 3 we discuss the architecture of a cyclic multicast proxy server. In section 4 we present the operation of the cyclic multicast proxy server and in section 5 the analysis of cyclic multicast. In section 6 we present the simulation of the server and in section 7 we discuss the result of our performance analysis. The paper concludes in section 8.

2. Related Work

Large popular files can be delivered efficiently from a server to several clients concurrently using multicast or broadcast. Some previous work has shown the use of multicast to provide scalable services [9,11–15]. Some other applications of multicast for the delivery of information, news to large audience and general data base access were described in [14,16,17]. The use of multicast support within the Internet has been largely tied to the delivery of videoconferencing, audio, video and streaming of multimedia applications to large recipients. In this work, we propose the User Datagram Protocol (UDP) best effort multicast for the delivery of popular pages to large numbers of receivers, with repetitive, cyclic transmission of the requested page to ensure reliability. This solution is expected to be scalable and more efficient when used for the delivery of the same content to large numbers of requesters or receivers.

3. The Basic Design of a Cyclic Multicast Proxy Server

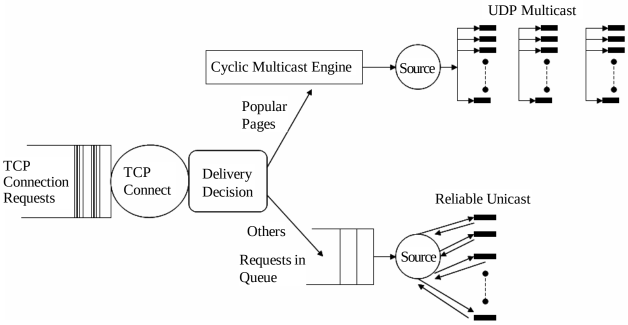

Figure 1. Cyclic multicast proxy server architecture.

Figure 1 shows the basic design of a cyclic multicast proxy server. This server is capable of delivering web pages using cyclic multicast and reliable unicast. When a request for Transmission Control Protocol (TCP) connection arrives from a client for a page, the requests are queued until the server can process them. When the request is about to be processed, the server establishes a TCP connection and the client request is transmitted.

A delivery decision is made based on the popularity of the requested page. There are two possible options for the delivery of a page, cyclic multicast or reliable unicast. The most popular (hot) pages and moderately popular (warm) pages are served using cyclic multicast engine, while other unpopular (cold) pages are served using the traditional TCP unicast connections. The decision for the pages to serve using cyclic multicast is based on the document hit rate of the server. The cyclic multicast operation involves a number of processes ranging from chunking of the page, joining a multicast group to receive all the chunks correctly in one or more cycles and finally leaving the multicast group after receiving all chunks correctly. This architecture is further described in detail in the following section.

4. The Cyclic Multicast Proxy Server Operation

The proposed cyclic multicast engine built in the proxy server is an effective way to deliver most popular and heavily requested pages. The cyclic multicast engine is capable of delivering multiple pages simultaneously using multiple multicast groups; however, in this work we only describe the use of this engine to deliver a single page to several receivers. For the delivery of multiple pages simultaneously, the operation is replicated for every page intended for delivery using cyclic multicast. The cyclic multicast delivery scheme may be summarized in the following steps similar to [18].

- The page including embedded files is divided into a number of chunks.
- A multicast address is used to transmit all chunks in a page sequentially from the server to the group of receivers. A cycle is the transmission of all chunks in a page.

- A receiver joins an appropriate multicast group and remains a member until all chunks are received correctly. If a receiver misses a chunk, the receiver must remain in the group until the missed chunk is re-transmitted and received correctly in subsequent cycle.
- The server (cyclic multicast engine) continues the cyclic transmission if there is at least one receiver in the group waiting to receive the page. The server stops transmission when all members of the group have finished receiving the page.

5. Analysis of a Cyclic Multicast Engine

We use the following analysis to compare the performance of cyclic multicast with reliable unicast. We assume our web page is broken into *C* equal-size chunks and those transmissions out of the server are in packets for both unicast and cyclic multicast with each packet containing one chunk. We assume that *N* receivers make requests for same page at the same time. If the probability that a packet (chunk) is received correctly is *P* and losses are independent of packet and receivers, for reliable unicast, U_c the number of packets (chunks) that will be transmitted such that all *N* receivers get all chunks making up a page is given by:

$$U_C = \frac{N * C}{P} \quad \textbf{(1)}$$

Similarly, for cyclic multicast, the server will continue to cycle through the chunks until all *N* receivers get all chunks. Assume is the number of cycles required, M_c the packets (chunks) transmitted is given by:

$$M_C = C * \alpha \quad \textbf{(2)}$$

Since all *N* receivers make their request at the same time, they will all be waiting to start receiving just before the transmission of the first cycle. We use a discrete time Markov chain to represent the behavior of the system. The chunks represent a state and a page has *K* chunks. Figure 2 shows the Markov chain for one user receiving all *K* chunks correctly.

Similarly, a discrete time Markov chain may be used to represent how a chunk is received by all N receivers. Let the state of the system represent the number of receivers that have received a particular chunk. If *m* receivers have received a particular chunk at the end of cycle *t*, then the probability of *m+n* receivers receiving the chunk at the end of cycle *t+1* will be:

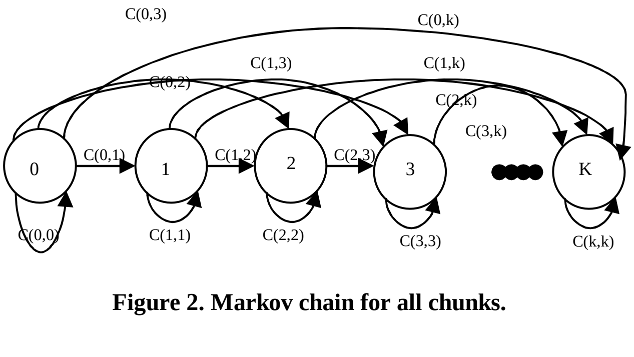

Figure 2. Markov chain for all chunks.

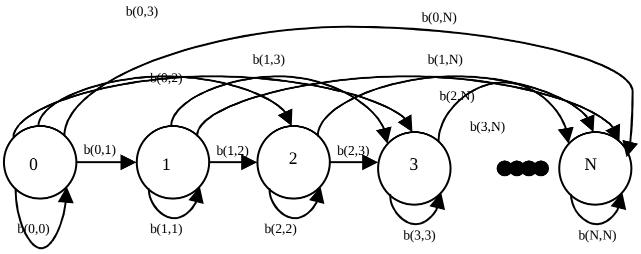

Figure 3. Markov chain for receiving one chunk.

$$b_{(m,m+n)} = \binom{N-m}{n} P^n * (1-P)^{(N-m-n)} \quad \textbf{(3)}$$

The Markov chain for all N receivers receiving one chunk is shown in figure 3. Equation 3 gives the transition between states; the end state is reached when all N receivers have received the chunk correctly.

The transition probability matrix is given by:

$$B = \begin{bmatrix} 1 & b_{01} & b_{02} & b_{03} & \bullet & \bullet & \bullet & b_{0N} \\ 0 & 1 & b_{12} & b_{13} & \bullet & \bullet & \bullet & b_{1N} \\ 0 & 0 & 1 & b_{23} & \bullet & \bullet & \bullet & b_{2N} \\ 0 & 0 & 0 & 1 & \bullet & \bullet & \bullet & b_{3N} \\ \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ 0 & 0 & 0 & 0 & \bullet & \bullet & \bullet & 1 \end{bmatrix}$$

Let *P*(*k*,*t*) be the probability that *k* receivers have already received a particular chunk at the end of cycle *t*. Then

$$P(k,t) = \sum_{i=0}^{k} b_{(i,k)} * P(i,t-1) \quad \textbf{(4)}$$

The following initial conditions apply to *P*(*k*,*t*):

P(i,j)=1 for i=j, (*i=0,1,2,3...k*)

P(i,0)=0 for (*i=1,2,3…k*)

P(i,j)=0 for i $\geq$ *2, j=i-1*

If $P_{RCVD}(N,C,t)$ represent the probability that all *N* receivers have received all *C* chunks at the end of cycle *t*, and we assume independence of loss then,

$$P_{RCVD}(N,C,t) = [P(N,t)]^C \quad \textbf{(5)}$$

By computing $P_{RCVD}(N,C,t)$ in Equation 5 we obtain the minimum number of cycles, α for the delivery of all *C* chunks to all *N* receivers with increasing values of *t*.

6. Simulation of the Cyclic Multicast Proxy Server Architecture

In this section, we present the results of simulations for the cyclic multicast proxy server which supports cyclic multicast and reliable unicast. Our objective is to provide a comparison in performance based on throughput, response time, end-to- end delay and jitters experienced by clients using a cyclic multicast proxy server and how it compares with clients using a caching or unicast proxy server.

For our simulation we used the ns-2 [19] network simulator. We assume a large number of clients making request which follow a Poisson process and each server will have N pages with all pages of same size. We assume that access pattern follows a Zipf distribution. We use the time to transmit all the chunks that make up a page (time for one cycle) as our time unit. We also assume that there is no propagation delay in making a request and in receiving chunks from the server and that the list of popular pages are know. The reliable unicast server is capable of transmitting streams out of the server using selective reject Automatic Repeat Request (ARQ) protocol, while the first request to the cyclic multicast server for a page is used to start the cyclic multicast engine.

The experiment scenario is shown in figure 4. From Figure 4 the centre node 0 is the server surrounded by client's node 1 to node 8 receiving packets from the server. Each client node (node 1 to node 8) has a link speed of 1Mbps with a delay of 10ms and a drop tail buffer to the central server node 0. There is a TCP/FTP flow from the server node 0 to all the eight clients' nodes. A complete page (all chunks making up the page) can be transmitted in one cycle to all clients receiving transmission from the server using cyclic multicast, but for unicast a cycle is the time to transmit all chunks to a single client. We conducted the experiment for unicast i.e. each node receiving transmission from the server one node at a time and for cyclic multicast which allows several nodes to receive transmission from the server at the same time. For the cyclic multicast, we also vary the time a client joins the multicast group using joining time of (*t=0.2s and t=0.5s*) in our simulation.

7. Performance of a Cyclic Multicast Proxy Server

Throughput is one of the performance parameters studied in our simulation. Throughput is defined as the amount of data (bits) that can be sent in a unit time. For the graph below, our time interval length (TIL) is 0.1s.

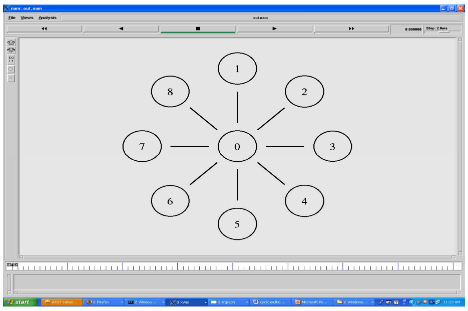

Figure 4. Cyclic multicast proxy server simulation scenario.

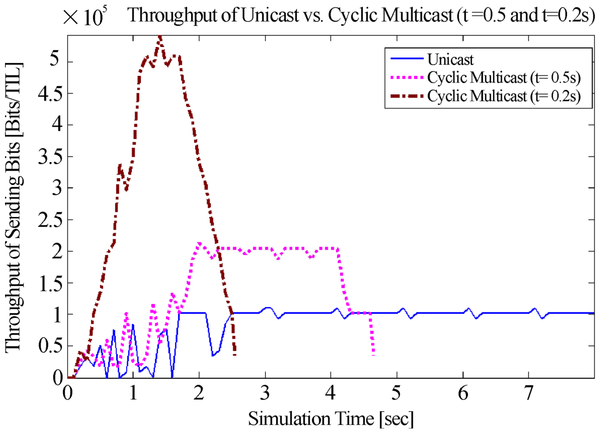

Figure 5. Throughput comparison.

Figure 5 shows the comparison of throughput for unicast and cyclic multicast servers. The throughput achieved by the unicast server was about 10,000Mbps while the throughput of the cyclic multicast server with a joining time *t*=0.5*s* was about 20000Mpbs. Reducing the joining time to *t*=0.2*s* allowed more clients to join the multicast group, further increasing the throughput to about 50,000Mbps.

This results shows a better performance by the proxy server when the number of receivers in a multicast group increases.

Similarly, we studied the response time. Response time is the time it takes to completely receive a page. In Figure 5 we can see a better response time for the cyclic multicast server.

The response time reduces as more clients join the multicast group to receive a page. For the unicast, the response time for all clients to completely receive a page is 8s. For the cyclic multicast with joining time *t*=0.5*s* the response time is 4.5s, while for cyclic multicast with joining time *t*=0.2*s* the response time is about 2.5s. Hence the load on the server is zero for cyclic multicast *(t*=0.2*)* after 2.5s and cyclic multicast *(t*=0.5*)* after 4.5s since there are no more receivers waiting to receive a page.

End-to-End delay is another performance parameter considered. End-to-End delay is defined as the time taken for a packet to be transmitted across a network from source to destination. From Figure 6, the End-to-End delay experienced by the unicast proxy server increases with the simulation time, while the End-to-End delay for the cyclic multicast reduces as the simulation time increases, showing again that the cyclic multicast proxy server out performs the unicast server. Similarly, for cyclic multicast, the server load drops to zero after 4.5s since there are no more requests to serve by the proxy after the last receiver exits the multicast group.

Jitter is another performance parameter considered. Jitter is an unwanted variation of signal characteristics. Jitter may be defined as the variation in the delay. Figure 7 shows the comparison of Jitter for unicast and cyclic multicast. The cyclic multicast experienced less jitters, indicating lower variation in the delay of packets.

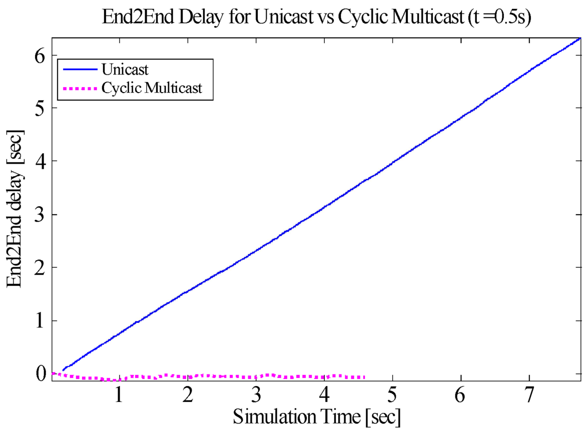

Figure 6. End-to-end delay comparison.

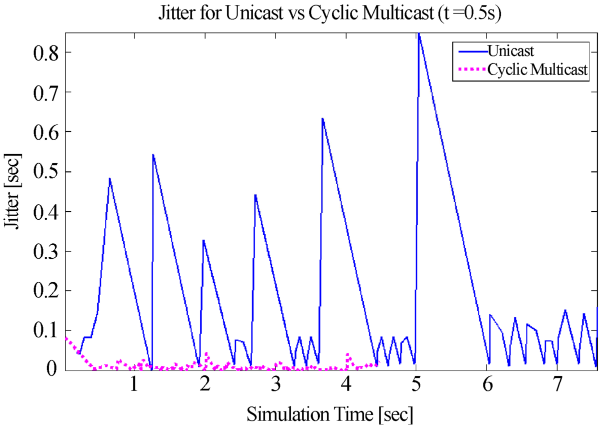

Figure 7. Jitter comparison.

We also plot the Cumulative Distribution Function (CDF) for End-to-End delay and jitter for unicast and cyclic multicast. Figure 8 shows the CDF for delay in unicast and cyclic multicast proxy server.

$$F_X(x) = P(X \le x)$$

For unicast proxy server,

$\Pr(delay \le 3) \approx 0.7$

For cyclic multicast server,

$\Pr(delay \le 0.1) \approx 0.7$

This shows that the cyclic multicast proxy server performs better than the unicast proxy server with respect to end-to-end delay.

Figure 9 shows the CDF for jitter in unicast and cyclic multicast server.

$F_X(x) = P(X \le x)$

For unicast proxy server,

$\Pr(jitter \le 0.1) \approx 0.7$

For cyclic multicast server,

$\Pr(jitter \le 0.01) \approx 0.7$

Again Figure 9 shows that the cyclic multicast proxy server performs better than the unicast proxy server with respect to jitter.

8. Conclusions

In this paper, we propose a proxy server based on the cyclic multicast for next generation networks, as a scalable delivery option for the delivery of web pages to increasingly large number of users under limited server capacity and network resources. Our proposed solution uses a cyclic multicast engine attached to the proxy server to deliver a popular page using UDP multicast with reliability achieved through repetitive, cyclic multicast transmission of a requested page. This solution is expected to be scalable and more efficient when used for the delivery of the same content to large numbers of receivers. Our simulation results show the performance gains achievable with this technique. Our result also shows that the performance of a proxy server can be further enhanced by integrating both delivery options in the proxy server for the next generation networks. A practical implementation of the cyclic multicast proxy server with squid [20], and detailed analysis of the behavior of the cyclic multicast engine using a discrete time Markov chain will be considered for future work.

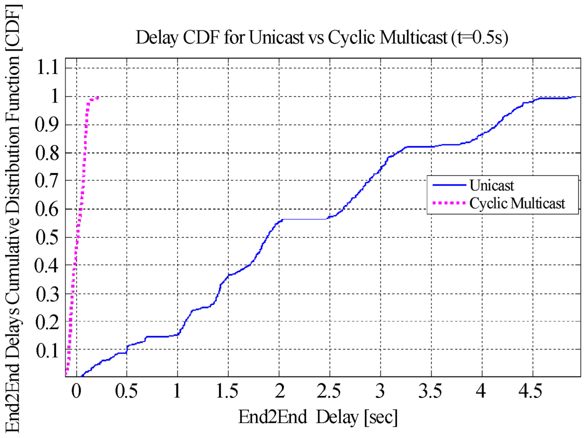

Figure 8. Delay CDF comparison.

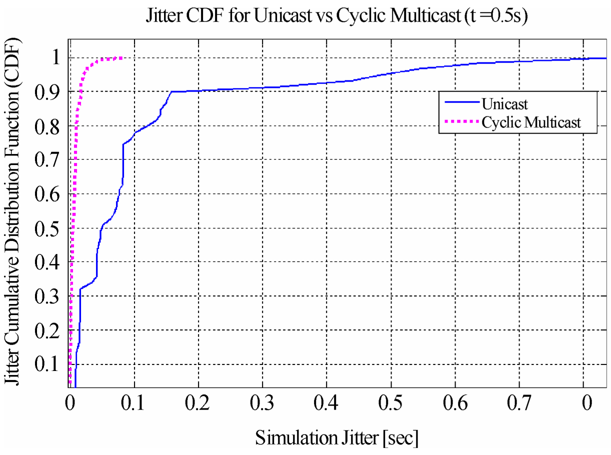

Figure 9. Jitter CDF comparison.

9. References

[1] Z. Miao and A. Ortega, “Scalable proxy caching of video under storage constraints,” IEEE Journal on Selected Areas in Communications, Vol. 20, pp. 1315–1327, September 2002.

[2] Z. L. Zhang, Y. Wang, D. Du, and D. Su, “Video staging: a proxy-server-based approach to end-to-end video delivery over Wide-Area Networks,” IEEE/ACM Transactions on Networks, Vol. 8, No. 4, pp. 429–442, August 2000.

[3] H. Fahmi, M. Latif, S. Sedigh-Ali, A. Ghafoor, P. Liu, and L. Hsu, “Proxy servers for scalable interactive video support,” IEEE Computer, Vol. 43, No. 9, pp. 54–60, September 2001.

[4] L. Guo, S. Chen, S. Ren, X. Chen, and S. Jiang, “PROP: A scalable and reliable P2P assisted proxy streaming system,” Proceedings IEEE International Conference on Distributed Computing Systems, Tokyo, Japan, March 2004.

[5] J. Liu and J. Xu, “Proxy caching for media streaming over the Internet,” IEEE Communications, pp 88–94 August 2004.

[6] V. N. Padmanabhan and J. C. Mogul, “Improving http latency”, 2nd World Wide Web Conference ’94, available: http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings, October 1994.

[7] E. Katz, M. Butler, and R. McGrath, “A scalable http server: The ncsa prototype,” Computer Networks and ISDN Systems, Vol. 27, pp. 155–164, 1994.

[8] 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, December 1995.

[9] R. J. Clark and M. H. Ammar, “Providing scalable web services using multicast communication,” Computer Networks and ISDN Systems, Vol. 29, No. 7, pp. 841–858, 1997.

[10] L. Breslau, P. Cao, L. Fan, G. Philips, and S. Shenker, “Web caching and zipf-like distributions: Evidence and implications,” Infocom’99, March 1999.

[11] J. W. Wong and M. H. Ammar, “Analysis of broadcast delivery in a videotext system,” IEEE Transactions on Computers , Vol. 34, pp. 863–866, September 1985.

[12] J. W. Wong and M. H. Ammar, “Response time performance of videotext systems,” IEEE Journal on Selected Areas in Communication, Vol. 4, pp. 1174–1180, October 1986.

[13] 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.

[14] G. Herman, G. Gopal, K. Lee, and A. Weinrib, “The datacycle architecture for very high throughput database systems,” Proceedings ACM SIGMOD, 1987.

[15] S. Acharya, M. Franklin, and S. Zdonik, “Dissemination based data delivery using broadcast disks,” IEEE Personal Communications, Vol. 2, pp. 50–60, December 1995.

[16] K. Lidl, “Drinking from the firehose: Multicast USENET news,” in USENIX Winter ’94, USENIX Association Press, pp. 33–45, January 1994.

[17] D. K. Gifford, “Polychannel systems for mass digital communications,” Communications of the ACM, 33(2), pp. 141–151, February 1990.

[18] K. C. Almeroth, M. H. Ammar, and Z. Fei, “Scalable delivery of web pages using cyclic best effort multicast,” Proceedings INFOCOM, pp. 1214–1221, 1998.

[19] UCB/LBNL/VINT Network Simulator - ns (version 2), 1997, available: http://www.isi.edu/nsnam/ns.

[20] D. Wessels, “Squid internet object cache,” 1996, available: http://www.nlanr.net/squid.