# Delivering Popular Web Pages Using Cyclic Multicast (Extended Abstract)

Kevin. C. Almeroth[a] and Mostafa. H. Ammar[b]

[a]Department of Computer Science
University of California, Santa Barbara
Santa Barbara, CA 93106-5110

[b]College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280

## ABSTRACT

The World Wide Web has gained tremendously in popularity over the last several years. One solution to the problem of overloaded WWW servers is to use multicast for the delivery of pages. In this work we explore the use of UDP, best-effort multicast as a delivery option. Reliability is achieved through repetitive, cyclic transmission of a requested page. We describe the cyclic multicast technique and consider the various procedures needed for its successful operation. We characterize the gains in performance achieved by our proposal through an extensive performance analysis and reference our ongoing work in simulating and implementing a cyclic multicast server.

**Keywords:** multicast, World Wide Web (WWW), cyclic broadcast, reliable delivery, chunking

## 1. INTRODUCTION

The World Wide Web (WWW) has gained tremendously in popularity over the last several years. WWW administrators are struggling to upgrade their servers to handle the huge volumes of requests. Further adding to the problem is the trend of WWW sites to offer more complex pages including flashy media like sound, graphics, pictures, and video. One straightforward approach to handling the large volume of requests has been to simply buy more hardware. This solution is neither cost effective nor scalable, and will cease to be an option as the WWW continues its growth.

One technique for improving scalability is to use multicast to deliver certain WWW pages. One specific approach that closely addresses the problem of end-to-end scalability has been proposed in our earlier work.[1] The proposed approach uses a reliable multicast protocol[2] to transmit popular pages to groups of users who make the same page request. In this approach, requests reaching the server for the same page can sometimes be aggregated and a single multicast response sent. Using multicast means the server needs to make fewer page retrievals and the network needs to carry fewer copies of the page.

In this work we explore the use of UDP multicast[3] as an alternate delivery option. Reliability is achieved through repetitive, cyclic transmission of a requested page. This solution is expected to be most efficient when used for highly requested pages. Current servers can find that they are always in the process of transmitting such pages. Our approach thus accepts this fact and continuously multicasts the page without needing requests to drive such transmissions. UDP multicast has two important advantages over reliable multicast. First, aside from the initial request, there is no need to communicate with the server. This eliminates the overhead associated with a feedback channel, packet acknowledgments and retransmissions. Heavyweight HTTP over TCP connections can be closed and resources used for receiving additional requests. Second, a significant portion of the Internet already supports UDP multicast. The Multicast Backbone (MBone)[4] has been successfully used for the wide-area, scalable delivery of real-time audio and video.

The concept of cyclic, multicast delivery has been used in other contexts primarily to improve the scalability of a service. Examples of such work are: 1) studies on the performance of cyclic multicast in the context of teletext

---

E-mail: almeroth@cs.ucsb.edu, ammar@cc.gatech.edu; Partially supported by research grants from IBM, Intel, and NSF NCR-9628379

systems (an early technology designed to deliver information over TV channels),[5,6] 2) the work on the Datacycle Architecture as a means to provide scalable access to a large database,[7] and 3) the use of cyclic delivery over a broadcast media to provide a Broadcast Disk.[8]

This paper is organized as follows. We first describe in Section 2 how multiple delivery options can be integrated in a WWW server. Section 3 examines the cyclic UDP multicast technique and considers the various procedures needed for its successful operation. In Section 4 we analyze the basic cyclic multicast protocol and compare its performance to reliable unicast (e.g., HTTP over TCP). Our latest work[9] deals with the further evaluation of cyclic multicast through a simulator and an implementation of a multicast capable WWW server running over the MBone.

## 2. INTEGRATED WWW ARCHITECTURE

Figure 1 shows the architecture of a WWW server capable of delivering pages using reliable unicast, reliable multicast, and cyclic multicast. Requests for TCP connections arrive from 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 pages are served via cyclic multicast. Moderately popular pages are served using reliable multicast. Other 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.

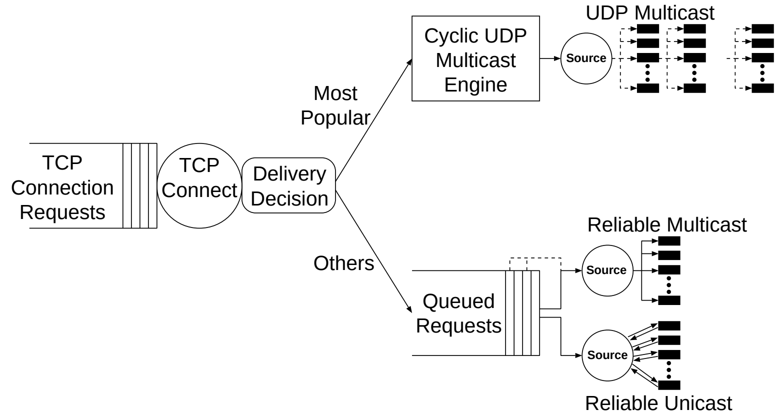


**Figure 1.** Integrated WWW server architecture.

## 3. CYCLIC MULTICAST

The cyclic multicast engine is most effectively used to deliver a site's most popular and heavily requested pages. The engine is capable of delivering multiple pages simultaneously using multiple multicast groups. We only describe here the engine's delivery of a single page. This operation is replicated for every page designated for delivery via cyclic multicast. The cyclic multicast delivery scheme includes the following steps:

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* is estimated using a formula based on our analysis.[9]

**Chunking.** The division of a page into chunks helps the receiver re-construct a WWW page as it is delivered. Chunks are then subdivided into one or more data packets. Chunking is especially useful when data is lost or received out of order due to network conditions or cyclic multicast. 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. An image, table, list, or other HTML object will only be displayed when all chunks in the group have been received.

**Multicast Address Determination.** The multicast group address is chosen using one of two methods. When the first request for a particular page is received, the server chooses a random address. As requests are made, the server informs requesters *explicitly* via the TCP connection used to make the request. The second option uses a hash function to convert the page's Universal Resource Locator (URL) to a multicast group. The same hash function is used by the server and all receivers. For this option, a requester need only make a request if, after joining the multicast group, it discovers the server is not transmitting the page. However, this gives no information to the server about how many requests there are. A compromise solution would have receivers immediately join a multicast group (thereby achieving very fast response if the server is actually transmitting) and then also make the request. In either case, TCP connections are closed.

**Requester Operation.** The initial challenge for requesters is determining how the server will deliver a page. If the server uses the *explicit* mechanism, the task is simple. However, if no such exchange is used, the requester must be prepared to listen on the open TCP connection, on a multicast address used by a reliable protocol, and on another address for cyclic multicast. Listening on three network connections is not a resource intensive task but does add complexity to browser operation. If delivery is via cyclic multicast, requesters will start to receive numbered chunks as soon as the server starts transmitting them. If the server is already transmitting, the receiver will see data immediately after joining the multicast group. If the browser is capable, the page may be displayed in non-contiguous segments. If an error occurs or a chunk is missed, the receiver must wait until the chunk is re-transmitted in the next cycle. After receiving all chunks, each requester will leave the multicast group. In response, routers will prune the receiver from the multicast tree, thereby preventing the delivery of unwanted packets.[10]

**Transmission Duration.** If all receivers have been satisfied and there are no outstanding requests, the cyclic multicast server should stop transmitting. However, without explicit feedback, the server cannot know for certain if there are still any receivers. Our solution is to use the arrival time of all requests and an estimate of the network error probability to estimate, with high certainty, the number of cycles needed to satisfy all receivers.[9]

## 4. CYCLIC MULTICAST ANALYSIS

We now present an analysis that is aimed at understanding and comparing the performance of cyclic multicast with reliable unicast. We consider the transmission of a WWW page that is broken into $C$ equal-size chunks. We assume that, in both the unicast and cyclic multicast cases, transmissions out of the server are in packets with each packet containing one chunk. In our analysis we assume that there are $K$ receivers who make requests for the same page at the same time. The probability that any chunk (or packet) is received correctly is $q_c$ and we assume that the loss is independent from one chunk to the next and among the $K$ receivers. The probability that a chunk is lost in our analysis covers all end-to-end conditions including errors and loss due to congestion.

For reliable unicast retransmission we assume a best-case selective reject ARQ protocol which re-transmits only lost chunks. We also ignore the overhead of connection and disconnection (this would tend to favor the reliable unicast scheme in our analysis). The number of chunks (or packets) that need to be transmitted in order for all $K$ receivers to receive all chunks of the page is

$$C_u = \frac{K * C}{q_c} \tag{1}$$

For the cyclic multicast system we assume that the server will continue to cycle through the chunks until the probability that all $K$ receivers have received all chunks is greater than or equal to some *certainty threshold*, $\beta$. Let $\gamma$ be the number of cycles required to achieve this threshold. The number of chunks transmitted will be $C_m = C * \gamma$.

Recall that for the purposes of this analysis we assume that all $K$ receivers make their request at the same time and, therefore, they will all be waiting just before the beginning of the transmission of the first cycle. In order to determine $\gamma$ we use an analysis technique similar to one used for probabilistic multicast.[11] A discrete time Markov chain represents the progress of the system. The state of the system is the number of receivers that have received a particular chunk. Given that $i$ receivers have received a particular chunk at the end of cycle $t$, the probability that $i + r$ receivers receive the chunk at the end of cycle $t + 1$ is given by:

$$a_{(i,i+r)} = \left( \begin{array}{c} K-i \\ r \end{array} \right) (q_c)^r * (1 - q_c)^{(K-i-r)} \tag{2}$$

Let $P(n,t)$ denote the probability that $n$ receivers have received a chunk by the end of cycle $t$. $P(n,t)$ is given by

$$P(n,t) \quad = \quad \sum_{i=0}^{n} a_{(i,n)} * P(i,t-1) \tag{3}$$

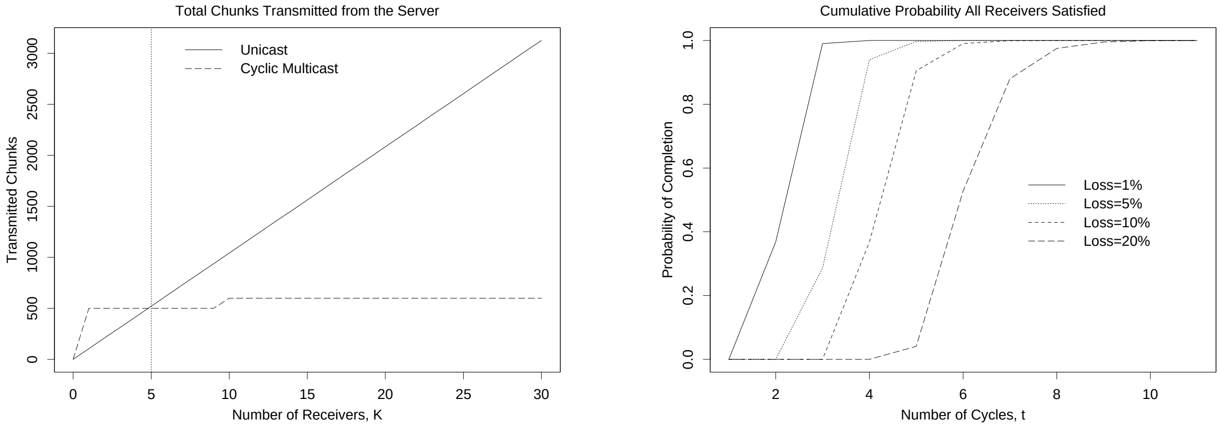
For $P(n,t)$ the following initial conditions apply:

$$P(0,0) = 1 \text{ and } P(i,0) = 0 \text{ for } i > 0 \tag{4}$$

Let $P_{DONE}(K,C,t)$ represent the probability that all $K$ receivers receive all $C$ chunks by the end of cycle $t$. Because we assumed independence of loss among the receivers we have:

$$P_{DONE}(K,C,t) \quad = \quad [P(K,t)]^C \tag{5}$$

The minimum number of cycles, $\gamma$, required to meet the certainty threshold, $\beta$, for delivery to all receivers is determined by computing $P_{DONE}(K,C,t)$ with increasing values of $t$. $\gamma$ is the smallest value of $t$ such that $P_{DONE}(K,C,t)$ is greater than $\beta$. W can then compute the number of chunks transmissions needed as $C_m = C * \gamma$.



(a) (b)

**Figure 2.** Number of packets and cycles needed to complete a transmission.

These equations are used to produce three results. Figure 2(a) shows the number of packets needed to satisfy different numbers of receivers (x-axis) with an error probability of 5%. In this case the slope of the reliable unicast line is $\frac{C}{1-p}$. The number of packets needed for cyclic multicast is relatively flat with more than one receiver. Furthermore, 5 receivers is the “breakeven” point where cyclic multicast transmits fewer packets. Figure 2(b) shows the cumulative probability that all receivers ($K = 100$) are satisfied by the end of each cycle for three different error probabilities. The lower the probability of error, the fewer cycles required to achieve a high probability of all receivers satisfied.

Figure 3 shows the minimum number of multicast group members necessary for cyclic multicast to transmit fewer packets than reliable multicast given the error probabilities represented on the x-axis. At lower error rates, cyclic multicast requires significantly fewer cycles and packets to satisfy receivers. This makes cyclic multicast much better at low error rates. As the probability of error increases so does the number of cycles and the minimum number of multicast receivers to achieve the breakeven point.

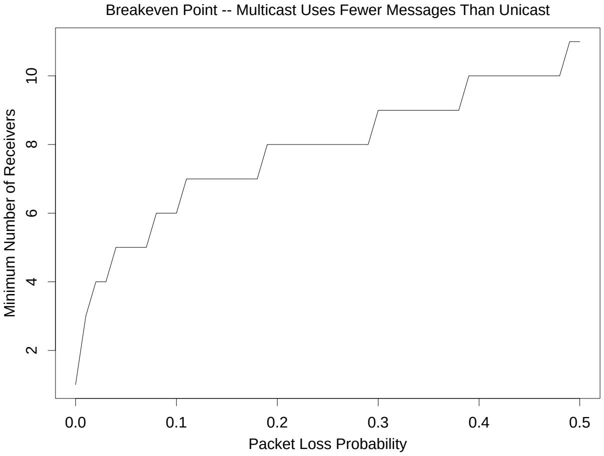


**Figure 3.** Breakeven point for multicast.

## REFERENCES

1. R. Clark and M. Ammar, "Providing scalable web service using multicast delivery," in *IEEE Workshop on Services in Distributed and Networked Environments*, (Whistler, Canada), June 1995. (to appear in *Computer Networks and ISDN Systems*).
2. R. Talpade and M. Ammar, "Single connection emulation: An architecture for providing a reliable multicast transport service," in *Proceedings of the Int. Conference on Distributed Computing Systems*, IEEE, 1995.
3. S. Deering and D. Cheriton, "Multicast routing in datagram internetworks and extended LANs," *ACM Transactions on Computer Systems*, pp. 85–111, May 1990.
4. S. Casner, *Frequently Asked Questions(FAQ) on the Multicast Backbone(MBone).* USC/ISI, December 1994. Available from ftp://ftp.isi.edu/mbone/faq.txt.
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. M. Ammar and J. Wong, "Design of teletext broadcast cycles," *Performance Evaluation*, vol. 5, pp. 235–242, November 1985.
7. G. Herman, G. Gopal, K. Lee, and A. Weinrib, "The datacycle architecture for very high throughput database systems," in *Proceedings of SIGMOD*, ACM, 1987.
8. S. Acharya, M. Franklin, and S. Zdonik, "Dissemination-based data delivery using broadcast disks," *IEEE Personal Communications*, vol. 2, pp. 50–60, December 1995.
9. K. Almeroth and M. Ammar, "Using UDP multicast for the cyclic delivery of WWW pages," Tech. Rep. file://ftp.cc.gatech.edu/pub/coc/tech_reports/1997/GIT-CC-97-20.ps.Z, Georgia Tech College of Computing, July 1997. (submitted to conference).
10. W. Fenner, "Internet group management protocol (IGMP), version 2," Tech. Rep. draft-ietf-idmr-igmp-v2-*.txt, Internet Engineering Task Force (IETF), January 1997.
11. M. Ammar, "Probabilistic multicast: Generalizing the multicast paradigm to improve scalability," in *Proceedings of Infocom '94*, (Toronto, Canada), pp. 848–855, June 1994.