Multicast Protocols for Scalable On-Demand Download

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

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

Categories and Subject Descriptors
C.4 [**Computer Systems Organization**]: Performance of Systems; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms
Performance, Design

Keywords
Scalable Download Protocols, Cyclic Multicast, Batching

1. INTRODUCTION
This paper considers the problem of efficiently delivering large files on-demand from a server to potentially large numbers of requesting clients, using scalable download protocols that employ either IP or application-level multicast. Previously proposed and new scalable download protocols are evaluated against bounds we develop that quantify the best achievable performance.

We consider the problem of how to minimize the average server bandwidth required to achieve a given average or maximum client delay (download time), or equivalently how the average bandwidth allocated to deliver a given file should be used to minimize the average or maximum client delay. Due to space limitations, results are described here for maximum client delay only. Policies and bounds for average delay are provided in the full version of the paper.

It is assumed that each requesting client downloads the entire file; i.e., clients never depart while waiting for delivery to commence or after having received only a portion of the file. We do not model packet loss recovery, although our analyses and protocols are compatible with erasure coded data. We assume that each client has successfully received the file once it has listened to server transmission of an amount of data equal to the file size. Poisson request arrivals are assumed, although generalizations are straightforward in several cases. In Sections 2, 3, and 4, it is assumed that all clients have the same maximum sustainable reception rate. Section 5 relaxes this assumption.

2. BASELINE PROTOCOLS
Previous scalable protocols for downloading large, popular files from a single server include batching and cyclic multicast. Consider first batching protocols in which the server multicasts the entire file to those clients that have requested it since it was last multicast. Any client whose request arrives while a multicast is in progress, simply waits until the next multicast begins. Here, we consider the batching protocol that achieves the minimum value of maximum client delay for a given average server bandwidth. Letting T denote the time at which some file multicast begins, the server will begin the next multicast at time $T+a+\Delta$, where a denotes the duration of the time interval from T until the next request arrival, and Δ is a parameter of the protocol. Using the notation defined in Table 1, it is straightforward to derive the server bandwidth and maximum client delay for this *batching with constant batch delay* (*bcd*) protocol, as follows:

$$B_{bcd} = \frac{L}{\Delta + 1/\lambda}; \qquad D_{bcd} = \Delta + L/r .$$

Note that the optimal value of the multicast transmission rate r is equal to the maximum sustainable client reception rate b.

In contrast to batching protocols, cyclic multicast allows clients to join an on-going multicast and begin receiving file data immediately (e.g., [1][2][3]). Perhaps the simplest protocol of this type is to continually multicast file data at a fixed rate r on a single multicast channel, regardless of whether or not there are any clients listening. Here we consider a more efficient cyclic multicast protocol that turns off the multicast when no clients are listening. The performance metrics for this protocol, *cyclic/turn off idle* (*c/ti*), are given as follows:

$$B_{c/ti} = r\left(1 - e^{-\lambda L/r}\right); \qquad D_{c/ti} = L/r .$$

The rate r is the only protocol parameter, and determines the tradeoff between server bandwidth usage and client delay.

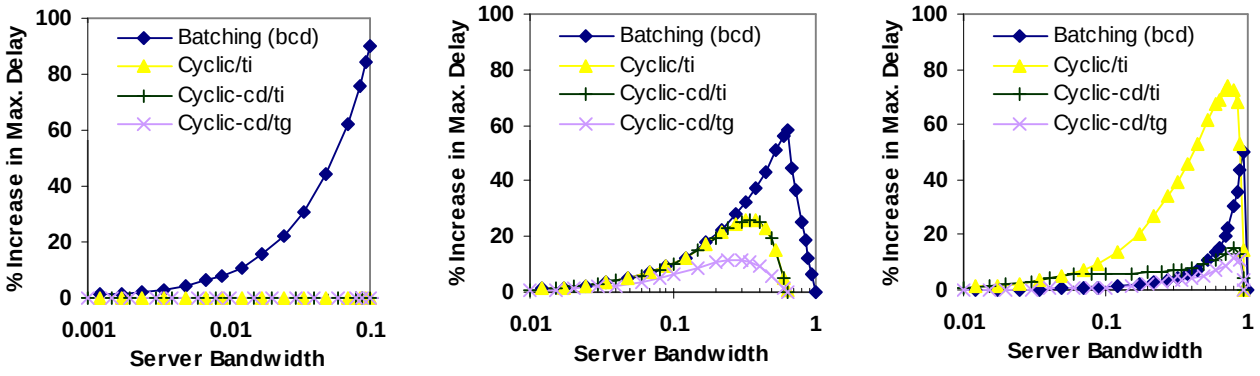
Table 1: Notation

Symbol	Definition
λ	File request rate
L	File size
b	Maximum sustainable client reception rate
r	Transmission rate on a multicast channel ($r \le b$)
B	Required server bandwidth
D	Maximum client delay
Δ	Batching delay parameter

3. LOWER BOUNDS
For a specified client reception rate b and maximum delay D, the average server bandwidth is minimized by a cyclic *send as late as possible* (*slp*) protocol. The *slp* protocol cyclically multicasts file data at rate b whenever there is at least one active client that has no "slack" with respect to the delay bound D.

This work was partially supported by the Natural Sciences and Engineering Research Council of Canada, and by the National Science Foundation under grants ANI-0117810 and EIA-0127857.

SIGMETRICS'04, June 12-16, 2004, New York, NY, USA.
ACM 1-58113-664-1/04/0006.



(a) b = 0.1 **(b) b = 1.0** **(c) b = 10.0**

Figure 1. Policy Performance Comparisons: Maximum Delay Relative to Lower Bound (L = 1, λ = 1)

A very accurate approximation for the average server bandwidth with the *slp* protocol is given by

$$B_{slp} \approx \left(\frac{\left(e^{\lambda L/b} - 1\right)/\lambda + D - L/b}{e^{\lambda L/b}/\lambda + D - L/b} \right) \frac{L}{D}.$$

Exhaustive comparisons against simulation results indicate that the above approximation has relative errors under 4% in absolute value. It can also be shown that the approximation is exact for all boundary cases (i.e., $\lambda \to 0$, $\lambda \to \infty$, $D \to \infty$, $L \to 0$, $b \to \infty$, and $D = L/b$, holding other parameters fixed in each case).

4. NEAR-OPTIMAL PROTOCOLS

The cyclic *slp* protocol can result in fragmented transmission schedules. Thus, also of interest are simpler, yet near-optimal, protocols. Specifically, we consider protocols that have clients begin listening to an on-going multicast at the times of their requests, but also limit server transmissions to time periods in which (probabilistically) more clients are listening. We consider first a hybrid of batching and cyclic multicast, in which a cyclic multicast is initiated only after a batching delay, and terminates when there are no remaining clients with outstanding requests. The average server bandwidth and maximum client delay achieved with this *cyclic constant delay turn on/turn off idle* (*ccd/ti*) protocol, with constant batching delay parameter Δ and transmission rate r ($r \le b$), are given by

$$B_{ccd/ti} = r \frac{O}{O + 1/\lambda + \Delta}; \qquad D_{ccd/ti} = \Delta + L/r.$$

Here O denotes the expected duration of a channel on time, $(e^{\lambda L/r} - 1)/\lambda$. Optimal settings of r and Δ can be obtained numerically. Interestingly, the optimal r is not necessarily $r = b$.

A higher performance protocol, called here *cyclic constant delay turn on/ turn off gated (ccd/tg)*, can be devised by using a better policy for when to stop transmitting. The key observation is that although clients that make requests while a multicast is already in progress should listen to this multicast, they have some slack for receiving the full file, given that the performance metric of interest is maximum (rather than average) client delay. Thus, the multicast can be allowed to terminate after the clients whose requests arrived before commencement of the multicast have received the full file. Any clients remaining at this point will have experienced no channel idle time, and thus the cyclic multicast should be resumed after a delay of duration Δ. If there are no such clients, the cyclic multicast can be restarted after a delay of Δ following the next request arrival.

Note that for the *ccd/tg* protocol, each client obtains the entire file in either one busy period, or in two busy periods separated by an idle period of length Δ. Observing that the optimal value for r is the maximum possible (b), a key advantage of *ccd/tg* is that it has just one parameter (Δ), which is chosen based on the desired trade-off between maximum delay and bandwidth usage rather than by numerical optimization. As illustrated in Figure 1, simulation results suggest that the performance of this protocol is within 15% of the lower bound provided by the *slp* protocol. Note that the choices of L and λ for this figure serve only to fix the units for data volume and time, without loss of generality.

5. HETEROGENEOUS CLIENTS

Consider now different classes of clients that have different achievable reception rates. In our full paper, comparison against a conjectured lower bound suggests that the best of the new protocols that we develop for this case leaves only modest room for further improvement. In this best new protocol, the server transmits on multiple channels, using on each a modification of the near-optimal *cyclic constant delay turn on/ turn off gated* protocol. In general, it is important to achieve a high performance trade-off between the use of higher aggregate server transmission rates, which enable improved batching opportunities for the clients that can receive at those rates, and low aggregate rates that maximize the sharing of server transmissions among clients of different classes. Our protocol balances these two factors when determining how much data a client receives on each channel.

6. REFERENCES

[1] K. V. Almeroth, M. H. Ammar, and Z. Fei, "Scalable delivery of web pages using cyclic best-effort (UDP) multicast", *Proc. IEEE INFOCOM '98*, San Francisco, CA, Mar. 1998, pp. 1214--1221.

[2] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data", *Proc. ACM SIGCOMM '98*, Vancouver, BC, Canada, Sept. 1998, pp. 56--67.

[3] S. Rost, J. Byers, and A. Bestavros, "The cyclone server architecture: Streamlining delivery of popular content", *Proc. WCW '01*, Boston, MA, June 2001, pp. 147--163.