Improving the performance of single-rate reliable multicast congestion control

Jeremiah Scholl, Luleå University of Technology

Abstract

This paper presents some ideas we are currently investigating in single-rate reliable multicast congestion control. We discuss the advantages and disadvantages of current schemes and speculate on how they could be improved. At this time we only present our general ideas and have not tested them with simulations or refined them in any way. For this reason the paper should be viewed as a work in progress.

1 Introduction

Reliable multicast congestion control has become a “hot” research topic. This is primarily due to the push for standardized reliable multicast protocols and comments by the IETF stating that congestion control is an absolute necessity before they standardization can take place [2]. The main challenge researchers face in designing an effective congestion control scheme is how to deal with varying congestion conditions in different parts of the multicast tree. Single-rate schemes guarantee TCP-friendliness for all receivers in a multicast session by identifying the worst-case receiver and adjusting the send rate accordingly.

An alternative to single-rate schemes is to have the congestion control scheme send out redundant data at different rates on several multicast channels. Each receiver can then sign up for the channel with the send rate that is most desirable. There are advantages and disadvantages to both multi-rate and single-rate schemes. However, a comparison of multi-rate vs. single rate schemes is beyond the scope of this paper.

There are two recently proposed single-rate schemes that have been shown to provide TCP-friendliness in preliminary simulations. PGMCC is the most widely accepted single-rate scheme at the moment and is the scheme most likely to be standardized by the IETF. LE-SBCC is the first purely sender-based scheme and although it can provide TCP-friendliness in some situations it has been shown to act aggressively when used in conjunction with total NACK aggregation. We provide a quick overview of the two schemes below.
2 Overview of PGMCC and LE-SBCC

PGMCC

PGMCC is the single-rate scheme that is most likely to be standardized for use with NORM, a NACK based protocol that is currently being worked on by the IETF reliable multicast transport work group. PGMCC identifies the worst-case receiver by having receivers add loss event and round trip time information to NACKs. The sender can then use this information to estimate each receiver’s throughput rate using the formula given in [1]. Once the sender has identified the worst-case receiver it adjusts its send rate by setting up TCP-like AMID congestion window with this receiver (referred to as “the acker”).

Sometimes the worst-case receiver changes over time. PGMCC detects these changes by analyzing the loss and round trip time information reported by receivers in NACKs. When a NACK is received by the sender that reports a send rate that is lower than the current send rate multiplied by a constant C, (usually between .6 and .8) the receiver that sent the NACK becomes the new acker. The scheme does not view a change in the acker as a congestion signal so it does not halve the sending window when a new acker is selected. Instead it keeps the window at the current size and only halves the sending window when the new acker experiences loss.

LE-SBCC

LE-SBCC is a purely sender based scheme that uses loss indications to estimate the available bandwidth from the worst-case receiver. The scheme analyzes incoming NACKs and converts them into loss events. The scheme uses this information along with round-trip time information to select the worst-case receiver. It then adjusts its send rate by plugging this round-trip time and loss-event rate information into an AMID simulator.

A quick comparison

The primary difference between the two schemes is how they process loss events in order to drive their AMID module. PGMCC drives its AMID module by adding loss event information to NACKs and setting up a unicast ACK connection with the worst-case receiver. LE-SBCC processes loss events at the sender and inputs these loss events into an AMID simulator rather than setting up a unicast connection for ACKs with the worst-case receiver.
This ack connection allows PGMCC to react faster to congestion signals but gives LE-SBCC a clear advantage in overhead. This could be particularly evident in multi-sender environments because multiple senders are likely to designate the same receiver as their acker. This situation would result in several ACK connections being set up with the receiver that has the least available bandwidth and could significantly reduce the rate of “useful” data available for the session.

It should also be noted that LE-SBCC does not share the general acceptance of the research community that PGMCC does. At this time it has only been demonstrated to work effectively in simple configurations with protocols that use partial NACK aggregation. Initial tests showed it to act aggressively towards TCP when used in conjunction with full aggregation and no tests have been run to see how it will perform with NACK suppression.

**What could be improved:**

Reducing the effect of feedback suppression:

Congestion control schemes require feedback from receivers or network elements so that the sender can detect its acceptable send rate. Feedback is generally available in the form of NACKs or ACKs generated by the recovery mechanism of the underlying protocol. However, reliable multicast protocols generally suppress feedback in order to achieve scalability.

For PGMCC, this can cause a delay in identifying the acker. Because the worst-case receiver will generally send out more NACKs over time than other receivers the correct acker should be chosen eventually. Distance estimation that weights suppression timers in favor of receivers with low rtts can further this problem when the worst-case receiver has a relatively large rtt from the sender compared to other hosts in the session. At this time it is still unclear if LE-SBCC will handle NACK suppression well or if it will perform aggressively because of the reduced number of loss indications.

Feedback suppression is even greater when FEC or local recovery is used. At this time there is not an available scheme that has demonstrated TCP-friendliness when used in conjunction with local recovery or FEC. One possible avenue for dealing with local recovery and FEC is to find a way for a host to detect when it is being treated unfairly and send some kind of report to the sender when it realizes that the sender has not backed off in a
reasonable amount of time. This reporting mechanism would also need to include its own suppression technique to keep it from impacting scalability when many receivers feel that they are being treated unfairly.

Frequent acker switches and PGMCC:

The effect of frequent acker switches on the send rate was given a lot of attention by the authors of PGMCC. Two different methods were investigated for adjusting the sending window when an acker switch occurs. Previously an acker switch was treated as a congestion signal and the sending window was halved every time a switch occurred. This caused the scheme to act too passively in some situations. So, the current implementation of PGMCC waits to see if the new acker faces further packet loss before reducing the window size.

However, this can create the opposite problem because constant acker switches can lead to a situation where the window size is not reduced enough for TCP-friendliness to occur. To keep frequent acker switches from creating a TCP-unfriendly state the scheme multiplies all reports by a decimal value C so that the selection process favors the current acker.

It is not clear how this will affect the scheme in heterogeneous environments. Different hosts (for example wireless hosts) might need different values of C in order for the scheme to perform well. It is also not clear how quickly the scheme will achieve TCP-friendliness when a new host joins with an available bandwidth that is lower than that of the current acker by a percentage that is less than 1-C.

One possibility might be to adjust the sending window to the size that produces a send rate that is closest to the rate reported by the NACK triggering the switch. This will allow the send rate to be adjusted during acker switches but will not lead to the constant halving of the sending window that causes the scheme to become too passive. This might handle frequent switches well enough so that the selection process will no longer need to be weighted in favor of the current acker.

Future Work:

Real world tests:

Although PGMCC has shown promise in initial simulations it has not been implemented and tested in the real world
over a variety of topologies. Our first priority is to run PGMCC over the Internet in a multi-sender environment and how it handles a variety of hosts and network conditions. For example, how the scheme performs with mixed mobile and wired members of the session is of particular interest. The results of these simulations should give us an idea about what areas of the protocol could be improved and also where it performs better than expected.

We will also run tests on LE-SBCC to see how it performs in conjunction with SRM in a multi-sender environment. At this time we think that it will perform too aggressively as the session grows. If this is the case then we will look for ways to improve source-based schemes because of the reduced overhead that they could provide.

References:


