A NEW CONGESTION CONTROL ALGORITHM FOR LAYERED MULTICAST IN HETEROGENEOUS MULTIMEDIA DISSEMINATION

Qiang Liu and Jenq-Neng Hwang

Dept. of Electrical Engineering, Box #352500
University of Washington, Seattle, WA 98195
{liuq, hwang}@ee.washington.edu

ABSTRACT

Layered multicast is a promising technique for disseminating adaptive-quality audio/video to multiple heterogeneous receivers. Congestion control in a layered multicast scheme is very important to support heterogeneity and scalability. Previous works on congestion control use packet loss, delay, and receiving rate to infer whether there is spare capacity along the path, which suffers from slow convergence, lack of inter-session fairness or TCP-fairness, layer oscillations, and loss induced by the join experiments. In this paper, we propose a layered multicast bandwidth inference congestion (BIC) control, which use delay increasing trend detection to infer the spare capacity. The major contribution of this paper is introducing the source probe organization and effective spare capacity inference by delay trend detection algorithm. We evaluate BIC for a large variety of scenarios and show that it converges fast to the optimal link utilization and can adapt to network dynamics effectively. We also show that BIC is stable, inter-session fair and fair to competing TCP traffics.

1. INTRODUCTION

Multimedia applications play an increasing important role in the Internet. If multiple users want to receive the same audio/video at the same time, a multicast dissemination is the most efficient way of transmission. To accommodate heterogeneity, one can use a layered source coding where each layer is sent to a different multicast address and the receivers subscribe to as many layers as their bottleneck bandwidth permits [4]. The multimedia applications can easily be transmitted using cumulative layers: each higher layer contains a refinement of the signal transmitted in the lower layers.

Steven McCanne [6] introduced the first receiver-driven cumulative layered multicast congestion control protocol (RLM) for video transmission over the Internet. RLM assumes that the network provides best effort service and supports IP multicast, and use the term session to denote the set of multicast groups used by the source to send layered video. Receivers in a session drop layers when they detect congestion and add layers when spare capacity is available. RLM detects congestion by measuring packet loss and use join-experiments to detect spare capacity. In a join-experiment, a receiver joins a group and measures the loss rate over a time interval called the decision-time. If the loss is too high, the receiver leaves the group. RLM uses shared learning to scale with the number of receivers. However, RLM has some fundamental weakness. RLM is not fair (neither inter-session fair nor TCP-fair) [5], RLM converges slowly to the optimal rate and tracks this optimal rate slowly, and RLM suffers the losses incurred by the failed join experiments.

Vicisano introduced a TCP-friendly version of a cumulative layered receiver-driven congestion control (RLC) protocol in [8]. RLC is based on the source-generated periodic bursts for bandwidth inference and on synchronization points to scale with the number of receivers. The receivers are only allowed to perform join experiments immediately after receiving a synchronization packet. Synchronization packets are sent periodically as flagged packets in the encoded stream with different period for different layers. This proves to be more scalable than the shared learning algorithm of RLM. Whereas RLC solves some fairness issues, it does not solve issues related to the convergence time, and does not solve the issues related to induced losses.

Thin Streams [9] is another congestion control protocol for receive-driven layered multicast. It tries to avoid the packet losses caused by the failed join experiment by dividing a video layer into multiple thin layers (about 16Kbps per layer). Thin Streams uses the difference between the expected throughput and the actual throughput to detect congestion. It assumes that each multicast layer has a constant bitrate (CBR). Furthermore, the bitrate of one layer is required to be sufficiently small so that the congested network can buffer the excessive data transmitted during a failed join experiment. The actual throughput is calculated as the number of bytes received in an arbitrary measurement interval, whereas the expected throughput is calculated as the number of joined groups multiplied by the well-known constant bitrate. The problem of this protocol is that the thin layers leads to transmission of partial video layers that the receiver cannot be used in the decoding process and hence implies poor bandwidth utilization. Also, the requirement of the layers to be strict CBR poses severe implementation problems. The protocol assumes that the thin layers will not cause router queue overflow during a failed join experiment, however a receiver is far (in the sense of transmission delay) from the source still can cause the bottleneck router queue overflow and so all receivers suffer from packet loss.

Johanson proposed a delay-based congestion control protocol [3], which is quite similar to the Thin Streams. It assume the source directly encode into thin layers (20Kbps as in [3]) so each layer received can be used in decoding and it uses the observed
delay to detect congestion. When a join experiment causes the network congested, the delay will be larger. Using delay doesn’t require each layer to be in strict CBR. However delay is highly dynamic with heterogeneous receivers and it is difficult to assign an appropriate threshold for each layer. Even for a single receiver the observed delay is highly depending on number of congested routers along the path from the source to the receiver. This protocol also has the same problem of packet loss as Thin Streams when the receiver is far from the sender.

Packet-pair layered multicast (PLM) [4] is the first congestion control that effectively infers the spare network capacity without join experiments and it can be used for arbitrary layer bandwidth organization. PLM only works in a network that all routers support Fair Queuing. In a Fair Queuing network, packet-pair technique can be used to infer the available bandwidth directly, so the receiver can join up to appropriate layers without join experiments and adapt to network dynamic fast by knowing the available bandwidth. However, we argue that the efficiency, stability, no-loss, simplicity and fairness of PLM are straightforward results of Fair Queuing. Packet-pair technique alone cannot resolve the problems in the previous algorithms in a FIFO networks. In fact, because the bandwidth estimated from packet-pair technique is multi-modal [1], the available bandwidth estimated in PLM may not be accurate in some specific network topology, even in a Fair Queuing network.

We believe the most important part in the congestion control for a layered multicast is to estimate the spare network capacity. In this paper, we propose a new congestion control scheme (BIC) that use delay trend detection to infer the spare network capacity along the path from the sender to the receiver. To the best of our knowledge, BIC is the first congestion control scheme that can effectively conduct the join/leave without join experiments for arbitrary layer organization in a FIFO network, such as the current Internet.

The remainder of this paper is structured as follows. Section 2 describes the BIC protocol. Section 3 evaluates the performance of BIC by simulations and Section 4 gives the conclusion of this paper.

2. THE BIC PROTOCOL

2.1. Source Layer Data Organization

The BIC can work for arbitrary layered data organization. Suppose \( N \) is the number of total layers. We use the following terms to represent the data bitrate of each layer \( i \):

\[
\begin{align*}
\text{bw}_i & : \text{Bitrate up to layer i} \\
\text{pbw}_i & : \text{Bitrate up to layer i during probes}
\end{align*}
\]

Table 1. Layered bitrate representations.

The sender is required to periodically send probes in each layer so each receiver can use the statistics of the probe packets to estimate the spare capacity along the path. During the probe period, the data bitrate up to each layer \( i \) follows \( \text{pbw}_i \) instead of \( \text{bw}_i \). The relationship between the two rates in Table 1 is:

\[
\begin{align*}
\text{pbw}_i & = \text{bw}_{i+1} (i \neq N) \\
\text{pbw}_N & = \text{bw}_N
\end{align*}
\]

(1)

According to this relationship, a receiver subscribing to layer \( i \) receives data at \( \text{bw}_i \) during normal period while at \( \text{bw}_{i+1} \) during probe period. There are two parameters controlling the behavior of the probe. One is the probe size \( P \), i.e., the number of probe-packets for each layer; the other one is the interval \( T \) of probes. The probe size should be just long enough to be used for delay trend detection and we use 50 for \( P \). The probe interval \( T \) is configured so as the additional data (to keep the probe bitrate) bitrate is less than 10% of the original data bitrate.

We use multi-layer token bucket model to reshape the source-generated raw data and interleave packets from different layers so that the packets in each layer are as even as possible. Fig. 1 shows one example of the output of this multi-layer token bucket model for the first 4 layers of \( \text{bw}_i \) of \((32,64,128,256,512)\) Kbps.

![Figure 1. Output of the multi-layer token bucket model.](image)

Each point represents a packet in Fig. 1. During the probe period for layer 1 (from 3.2s to 4.6s), the packets of layer 2 are sent through multicast group 1 so that the receivers subscribing to layer 1 receive packets up to layer 2, just as they have subscribed to layer 2. The statistics at the receivers during the probe period is used to infer whether the receivers can join up to the next layer. From Fig. 1, we can see that the probe duration for each layer is different and the higher layer has shorter probe duration (e.g., probe period for layer 2 is from 4.6s to 5.4s). This is because the probe packets are counted up to the specific layer. These properties distinguish our source probe infrastructure from the one used in [8].

2.2. Capacity Inference

We use packet loss to detect congestion during normal period in the same way as other protocols [6] [8]. However, we use delay trend detection during probe period to infer the spare capacity. The basic idea is that when the data bitrate is larger or equal to the available bandwidth along the path, the observed packet delay at receivers will have an increase trend. Due to space limitation, we don’t elaborate the details how to effectively detect the delay trend in this paper. Interested readers can refer to [2][10].

From the statistics of the probe packets, an indicator of delay increase trend (a value between 0 and 1; the larger this value, the more obvious of an increase trend) can be calculated. A threshold can be set for each layer so that the receiver can judge whether there is enough spare capacity to subscribe one more layer by comparing the trend detection result with the predefined threshold. If the delay trend result is lower than the
threshold, it is probable that the path can support the probe rate, which is the rate as the receiver subscribes to one more layer according to our source layer organization. If the path doesn’t have enough spare capacity to support one more layer, a delay increase trend will be detected. The router queue can absorb the limited inserted probe packets (normally less than $P/2$ packets), thus it doesn’t cause the router queue overflow.

2.3. Join/Leave Adaptation

The join/leave decision in BIC protocol is straightforward. Each receiver monitors the packet loss and leaves the current layer when the observed packet loss exceeds a predefined loss threshold. Each receiver also detects the delay trend during its probe period and joins one more layer when the delay trend is less than a predefined trend threshold (If loss is detected in the previous normal period or current probe period, the receiver keeps to its current subscription layer). Note the probe also serves as a synchronization point for all receivers in one session for join action, which scales with the number of receivers as the way in [8]. A distinct feature of BIC is that no join experiment is required to be conducted.

BIC has a different approach during the startup phase. The receiver doesn’t wait for the probe period to join the next layer in the startup phase. Instead, just when it receives $P$ packets after receiving the first packet of the current layer, it detects whether there is a delay increase trend for these $P$ packets. If there is no increase trend and no packet loss observed, it joins the next layer; otherwise, it turns to steady phase. A more strict trend threshold is applied for startup phase to avoid over-subscription. This special startup phase approach enables BIC to converge to the optimal subscription level very fast.

Inter-session fairness can be achieved by choosing different loss threshold and delay trend threshold for different layers. More strict thresholds are used for higher layers so that receivers with lower subscription level are easier to go up.

3. SIMULATIONS AND EVALUATIONS

We evaluate the performance of BIC by simulations using ns2 [7]. Figure 2 shows the topology used in the simulations, where $S_i$s are the sources of layered multicast sessions or competing TCP sessions while $D_i$s are the corresponding receivers. The routers use Drop-tail queue and the queue sizes are set to 100.

![Simulation topology](image)

**Figure 2. Simulation topology.**

3.1. Stability, scalability and heterogeneity

In the first simulations, we use only one BIC session $b_{wi}$ of (32,64,128,256,512,1024) Kbps and set the bottleneck capacity to 600Kbps. There are 6 receivers from $D_1$ to $D_6$. $D_1,D_2$ have 10Mbps, $D_3,D_4$ have 300Kbps and $D_5,D_6$ have 200Kbps. $D_1,D_3,D_5$’s link delays are 10ms and $D_2,D_4,D_5$’s link delays are 2 seconds. $D_1$ joins the session at the 5th second and other receivers join the session 14 seconds later one by one. The simulation runs for 600 seconds. The result is shown in Fig. 3.

![Layer subscriptions](image)

**Figure 3. Layer subscriptions of $D_1$, $D_2$, … and $D_6$.**

It is obvious from Fig. 3 that the layer subscription in BIC is very stable. After each layer finds its optimal subscription layer it locks to it without any oscillation. The performance is not affected by the number of receivers (we have tested more simulations using hundreds of receivers, the performance is the same). BIC can support heterogeneity well as in Fig. 3. The receivers with long path delay (e.g., $D_2$) find their optimal subscription layers slower than the ones with short path delay (e.g., $D_1$). BIC also demonstrates very fast convergence in the startup phase, especially for the receivers with short path delay. During the whole simulation, there is no packet loss.

3.2. Inter-Session Fairness and TCP Fairness

In the second simulation, we use two BIC sessions $b_{wi}$ of (32,64,128,256,512,1024) Kbps and set the bottleneck capacity to 1.1Mbps. There are two receivers ($D_1,D_2$) in the session 1 and two receivers ($D_3,D_4$) in the session 2. There are two competing TCP sessions ($S_3$-$D_5$ and $S_4$-$D_6$). All links other than the bottleneck are 10Mbps and the receiver link delays of $D_1,D_3,D_5$ are 10ms while the receiver link delays of $D_2,D_4,D_6$ are 500ms. $D_1,D_2$ join BIC session 1 at 5s; TCP 1($S_3$-$D_5$) starts at 5s; $D_3,D_4$ join BIC session 2 at 65s and TCP 2($S_4$-$D_6$) starts at 65s. The simulation runs for 200 seconds. The results are shown in Figures 4 and 5.

![Layer subscriptions](image)

**Figure 4. Layer subscriptions of $D_1$, $D_2$, $D_3$ and $D_4$.**
Figure 5. Data rates of the two BIC and two TCP sessions.

BIC is demonstrated to be inter-session fair as in Fig. 4. Although BIC session 2 starts later than BIC session 1, the two sessions reach a final equilibrium after the session 2 starts.

TCP’s throughput is very sensitive to the RTT (Round Trip Time). As shown in Fig. 5, TCP session 2 has much less throughput than TCP session 1 due to a larger RTT. Since TCP’s performance depends on RTT while a layered multicast doesn’t (layered multicast is an one-way protocol and the effect of path delay is limited), it is not fair to judge TCP-fairness by comparing the arbitrary throughput. Another issue making the judgment more complex is that the throughput of a layered multicast is discrete (by changing layer subscription level) while TCP’s throughput is continuous. We claim BIC is TCP-fair because it is responsive to competing TCP sessions and all sessions can reach a final equilibrium. Compared to TCP, BIC is less aggressive at increasing data rate (BIC refuses to join one more layer when delay increase trend is detected while TCP keeps increasing rate until packet loss is detected) and more aggressive in decreasing data rate (TCP cuts half the congestion window even for one single packet loss while BIC drops a layer only when the loss rate exceeds a predefined threshold). So TCP-fairness can be achieved by carefully choosing the various thresholds for joining and leaving layers in BIC.

We also conduct simulations using other layer organizations (e.g., bw, of 32, 64, 96, 128, 160, 192, 224, 256 Kbps) and have similar results.

4. CONCLUSION

Large-scale deployment of multipoint multimedia application in a heterogeneous network requires a sophisticated congestion control protocol. Such a congestion control protocol must be scalable to a large number of receivers, efficient in adapting to network dynamic, fair to other receivers in a session, other sessions and other streams such as TCP, stable in received quality, supporting heterogeneity of receivers and easy to deploy (require least support from the network). In this paper we propose a new congestion control (BIC) scheme for layered multicast. BIC uses the delay trend detection for the probe packets generated by the source to infer whether the receiver can join up one more layer. The protocol is end-to-end and only requires a best-effort FIFO network. BIC is loss-free in the available bandwidth inferring process and is shown by simulations to interoperate fairly among the receivers with the same session and with other sessions and TCP traffic. BIC has good performance in both startup phase and steady phase. BIC is scalable to a large number of receivers and perform well with heterogeneous receivers (with different bandwidth capacities and different delays). BIC is also stable in the sense of the minimal layer oscillation.

5. REFERENCES