Overall Considerations for Congestion Control for Interactive Real-Time Communication (IRTC)

M. Zanaty, B. VerSteeg, B. Christensen, D. Benham, A. Romanow
Cisco Systems

In response to the call for papers for the IAB/IRTF Workshop to be held on July 28, 2012 in Vancouver, Canada, the following overall considerations should be discussed in the workshop and included in work toward solution(s).

1. Requirements and Use Case Scope

a. The congestion control algorithm should be widely applicable for:
- all IRTC applications (RTCWEB, CLUE, AVT, etc.),
- across a wide range of media bit rates (few kbps to many Mbps),
- all network types (Wi-Fi, 4G, DSL, HFC, fiber, enterprise, data center), and their corresponding delay, loss and throughput characteristics.

b. The congestion control algorithm should balance the twin goals of maximizing throughput and minimizing delay/loss. It should discover when it is powerless to achieve some goals (for example, due to many other flows dominating the delay, loss or throughput characteristics), and adapt its balance toward the achievable goals accordingly. It should also balance stability and responsiveness.

c. The congestion control algorithm should detect and react to all implicit congestion signals (e.g. delay/loss) as well as explicit congestion signals from the network (e.g. ECN/PCN) if deployed. Ignoring or misinterpreting any signals degrades effectiveness. Implicit signals should be filtered to distinguish between congestion and noise. Explicit signals should not be required for effective operation, but should be required to be utilized by the algorithm when present [Zhu12].

d. Fairness to other flows (self-instances and TCP) should be defined quantitatively. Weighted fairness should be supported [Zan12], with weights specified by the application, and limits on the weights within the algorithm. Equal rates for all applications or flows is not sufficient for fair operation, especially when media streams are multiplexed into a single transport flow.

e. Startup behavior should not result in unfair newcomer advantage or disadvantage. Nor should it be constrained by TCP startup behavior, since even the recent “aggressive” initial congestion window of 10 packets would severely constrain media startup, due to initial video reference frames needed to start the decoding which are often larger than 10 packets.

f. The congestion control algorithm should interoperate with peers that don’t implement the same algorithm, and still provide some basic level of congestion control in both transmit and receive directions, although perhaps not as effectively as when interoperating with itself.

g. The congestion control algorithm should operate effectively in the presence or absence of Active Queue Management.

h. The congestion control algorithm should operate effectively in the presence of multi-point or one-to-many flows, which may require devices or networks to aggregate or summarize feedback from multiple receivers.

2. Design and Testing

a. Statistical methods, such as autoregressive models [Beg12], Kalman filters [Alv12], and other well-known tools, should be considered for computing or estimating key parameters such as queuing delay, expected packet arrival time, round trip time, non-congestion loss or delay, etc. However, complexity may hinder broad adoption and good, consistent implementations. So, complex algorithms must be demonstrated to significantly outperform simpler alternatives such as moving averages, not only analytically but also experimentally.

b. All implicit and explicit signals and key parameters should be cohesively modeled together to determine overall congestion state, harnessing signal fusion concepts. Fusion should yield more accurate estimates of key parameters and overall congestion state, compared to independently processing individual signals and choosing a response based on only one signal.

c. RTP transmission time offsets [RFC 5450] should yield more accurate delay models. Therefore the sender should always include them, and the receiver should utilize them if present, but the receiver model should also operate well in their absence.

d. Feedback from receiver to sender, presumably via RTCP, should be minimized in size and frequency without impacting algorithm effectiveness or responsiveness. This argues for more receiver side computation and aggregation of raw metrics into cooked feedback, or a fully baked rate, rather than frequent raw feedback.

e. Existing RTCP feedback mechanisms (RR, TMMBR, NACK, ECN, etc.) and RTP header extensions (send offsets, etc.) should be used when possible. New feedback and/or extensions should be proposed only when existing standards are insufficient.

f. RTT should only be used when appropriate to model feedback loop delay or TCP rates, and not as a proxy for one-way delay, since asymmetric delay events are common between the forward and feedback directions.

g. Existing guidance on specifying new algorithms [RFC 5033] and metrics for evaluating them [RFC 5166] should be considered.

h. Stability and fairness should be experimentally tested with common competing traffic including TCP variants (Reno, CTCP, BIC, CUBIC) and rate-adaptive applications (DASH, LEDBAT).

3. References

[Alv12] Alvestrand, H., Lundin H. and S. Holmer, “A Google Congestion Control Algorithm for RTCWEB”, IETF Internet Draft, draft-alvestrand-rtcweb-congestion-02, April 2012.

[Beg12] Begen, A., “Timely Detection of Lost Packets in Interactive Media”, IAB/IRTF Workshop on CC IRTC, June 2012.

[Zan12] Zanaty, M., “Fairness Considerations”, IAB/IRTF Workshop on CC IRTC, June 2012.

[Zhu12] Zhu, X. and R. Pan, “Network Assisted Dynamic Adaptation”, IAB/IRTF Workshop on CC IRTC, June 2012.