A TCP-friendly Fuzzy Congestion Controller for Transcoded Video over the Internet

E. Jammeh, M. Fleury, M. Ghanbari
Electronic Systems Eng. Dept., University of Essex, Colchester CO4 3SQ, United Kingdom.
Emails: {eajamm, fleum,ghan}@essex.ac.uk

Abstract

Fuzzy logic control of transcoded video streams under UDP offers a flexible congestion response. The paper demonstrates that fuzzy control is compatible with existing TCP-dominated general networks. Simulations across a tight link show that fuzzy control works even when the congestion level feedback signal is not independent of the controlled stream.

Keywords: congestion control, fuzzy inference systems, rate adaptation, transcoding.

1. Introduction

The Internet's growth has encouraged multimedia applications, such as DVD-video streaming, delivery of sports and news clips, and the exchange of personal video clips (peer-to-peer streaming). In a 2003 survey [1], such traffic had risen to 60% of the total in the network core. Network congestion impacts the delivery of encoded video due to an inconsistent and constrained available network bandwidth, increase in delay and jitter, with an increased packet loss rate. Over-provisioning is unlikely to remove this problem [2], because of bandwidth mismatches at the Internet's periphery. Whether the delivery mechanism is unicast, IP multicast, or via a multicast overlay network, then the key problem is reacting to congestion caused by other traffic as the stream passes across a bottleneck.

This paper indicates that a newly developed fuzzy congestion controller for Signal-to-Noise Ratio (SNR) control of a hybrid video transcoder [3] can be engineered to avoid risk of network instability [4]. It does this by respecting existing traffic including TCP flows, which a survey in 1998 [5] indicated were then up to 90% or more of traffic and which use a co-operative form of congestion control. TCP's end-to-end congestion mechanism is principally by the Additive Increase Multiplicative Decrease (AIMD) [6] algorithm, though there are many refinements. However, the 'bursty' transmission behavior, in part caused by AIMD, and the mechanisms employed by TCP to achieve reliability make TCP, as it stands, unsuitable for encoded video transport. Therefore, video transmission commonly takes place by a combination of UDP and application rate control [7]. A conformant non-TCP flow is TCP-friendly "when it does not reduce long-term throughput of any coexistent TCP flow more than another TCP flow" under the same conditions [8][4]. As rapid fluctuations in Quality-of-Service (QoS) unfairly impact multimedia traffic, only average behavior should be TCP-friendly.

As currently framed, fuzzy congestion control is a sender-based system for unicast flows. The receiver returns a feedback message, which indicates time-smoothed changes to congestion level, based on variable packet inter-arrival times. The sender then applies a control signal to the transcoder's quantization level. Variable Bitrate (VBR) video output is smoothed, currently by interleaving the output packet order [9], so that larger and small-sized packets are grouped resulting in a more uniform aggregate size. Turning to multicast, receiver-driven layered multicast (RLM) [10] of pre-encoded video can result in transmission of redundant video layers, with only coarse-grained control by the receiver of which layers are subscribed to. In [11], fine-grained, transcoder architectures are introduced that avoid both of the weaknesses of RLM and to which fuzzy transcoder control can directly be adapted.

In TCP-friendly Rate Control (TFRC) [12] the sending rate is made a function of the measured packet loss rate during a single round-trip time (RTT) duration measured at the receiver. The sender then calculates the sending rate according to the TCP throughput equation [13], using the fed-back receiver measurements. TFRC was designed to produce smooth multimedia flows but because it assumes constant-sized large (or Maximum Transport Unit (MTU)) packet sizes (not the case in the fuzzy-controlled scheme) it introduces a bias against variable-sized packets [14] (and constant-sized small packets). The TCP throughput equation upon which TFRC relies is not designed for conditions in which there are just a few flows (as in our current experiments). This is because changes to the sending rate alter the conditions at the bottleneck link, which affect the feedback, resulting in a non-linear system. Fuzzy control is well suited to control in non-linear systems.

The remainder of this paper is organized as follows. Section 2 explains our control system with emphasis on the method of feedback estimation. Section 3 gives the results of calibration simulations across a tight link. Finally, Section 4 draws some conclusions and indicates further research.

2. Fuzzy Control Methodology

2.1 Measuring congestion

The performance of any rate adaptive video streaming application is determined, to a large extent, by the measurement method for the status, usage or availability of the network resources. Several methods of measuring the available network bandwidth use packet dispersion for estimates but mostly differ in the number of packets used and on the method of extracting the estimates from the measured dispersion.

In *Self-Loading Periodic Streams* (SLoPS) [15] the increasing trend of the one-way delays (OWDs) of a number of same-sized packets is used to estimate the available bandwidth. SLoPS calculates the available bandwidth by varying the rate of the probing packets and the rate at which the OWD starts to show an increasing trend is taken as an available network bandwidth estimate. In *Trains of Packet Pairs* (TOPP) [16], several packet pairs are sent from source to destination with an increasing rate and the received packet pair rate is measured at the receiver. The sending packet pair rate is taken as the available network bandwidth when the sending rate equals the receiving rate. *Pathload* [17] use the SLoPS methodology but reports a range for the available network bandwidth rather than giving a single estimate. *pathChirp* [18] is an available network bandwidth measuring tool that sends a number of packets, exponentially separated in transmission time, across a network path. The packet pair dispersions are measured at the receiver and used to estimate the available network bandwidth.

In our methodology, we do not try to explicitly measure the available network bandwidth but rather use a congestion level measure. Consider a train of packets sent from source to destination through a network path. The input rate of the packet train, R_I, is computed as shown in (1) and the packet train exits the network with a dispersion rate of R_D as shown in (2).

$$R_I = \frac{\sum_{i=1}^{N_S} S_i}{\sum_{i=1}^{N_S} \delta_i^S} \tag{1}$$

$$R_D = \frac{\sum_{i=1}^{N_r} S_i}{\sum_{i=1}^{N_r} \delta_i^r}, \tag{2}$$

where S_i is the length of the ith packet and δ_i^S and δ_i^r are the dispersion of the ith packet respectively at the source and destination. Due to packet loss, N_S, the number of packets sent, will not always equal N_r, the number of packets received. Dispersion is the inter-arrival time recorded at the receiver, resulting from inter-departure time, propagation delay, and router queuing time due to cross traffic. In our video streaming system, the inter-packet gap is constant at the source but the inter-arrival time varies at the destination. We measured δ_i^r from the time when the previous packet had arrived to the time when the last data of the current packet had arrived. R_I is not affected by the network but by the sending rate of the packet train. On the other hand, R_D is theoretically determined by the capacity of the tight link and the amount of cross traffic, as formally given by (3) (assuming a fair queuing policy):

$$R_D = \frac{R_I}{R_I + R_C} C_t, \tag{3}$$

where C_t is the link's bandwidth capacity and R_C is the rate of the cross traffic in the tight link. In practice, R_D is measured through the individual timings δ_i^r in (2).

R_D will equal C_t if there is no cross traffic in the network. Otherwise, R_D will depend on the sending rate of the packet train and the available network bandwidth. The expression for the congestion level CON_{lev} is given in (4):

$$CON_{lev} = 1 - \frac{R_D}{R_I}. \tag{4}$$

2.2 Fuzzy Inference System

The fuzzy inference system maps the given inputs to an output using fuzzy logic membership functions to 'fuzzify' the input, *i.e.* if-then rules and fuzzy operators. The main constituents of the fuzzy logic congestion controller are a fuzzy inference system, a rule base and a defuzzifier. Our system uses a

Mamdani model [19] with two inputs: 1) congestion level 2) instantaneous rate of congestion level change, and uses centroid of area defuzzification.

The performance was tuned across congestion levels by changing the number and shape of the membership functions and the if-then rules and by using Matlab's fuzzy toolkit to model the output surface. It appears that it is the level of congestion that is important, implying that a single model will not be appropriate for all network conditions. The model developed is most suitable for high-congestion, whereas two differing models are needed for medium and low congestion.

In tests, a tight link along the end-to-end path was represented by a “dumbbell” topology in the NS-2 simulator, with cross-traffic forming congestion. The delay across the tight link was 1 ms. Drop-tail router queues were used with queue size set at 20, as the low link bandwidth-delay product means average queuing delay is not significant. Notice that some window-based TCP-friendly protocols become “unfriendly” [20] under high-congestion with drop-tail queues, requiring Random Early Detection (RED) queues to be present.

3. Results

Fig. 1 is a sample from a set of calibration experiments, showing the result of streaming a 2.7 Mbit/s non-video trace (Constant Bit Rate (CBR) traffic) under fuzzy control of the transcoder, with a CBR background cross traffic. The bottleneck rate was set to 2 Mbit/s, which is typical of constrictions at the core network boundary before entering an edge LAN. The CBR cross traffic changes its rate in time alternating between 0.5 Mbit/s and 1.5 Mbit/s. Fig. 2 shows the result for a 3 Mbit/s bottleneck rate. Both results demonstrate that the fuzzy control scheme could detect and adapt to the available network bandwidth. The controlled rate exhibits a low frequency fluctuation, due to the control algorithm continuously detecting and adapting to available network bandwidth. The current fuzzy models do not include a zero change response, whereas its inclusion would result in smoothing.

A fairness index shows how computed rate compares to the spare capacity in the network. This is taken as a ratio of the fuzzy system rate to the available bandwidth and should ideally be one. Fig. 3 and Fig. 4 show the result of controlling CBR traffic across varying bandwidth bottlenecks with respectively CBR background traffic and FTP background traffic (carried on TCP New Reno).

The results demonstrate that the system was able to detect the spare capacity and to adapt to it in a ‘friendly’ way. There is a diminishing trend towards regions of low congestion, illustrating current tuning for high congestion regimes. For example, in Fig. 4 the fuzzy-controlled traffic is disadvantaged for bottleneck rates beyond about 2.5 Mbit/s.

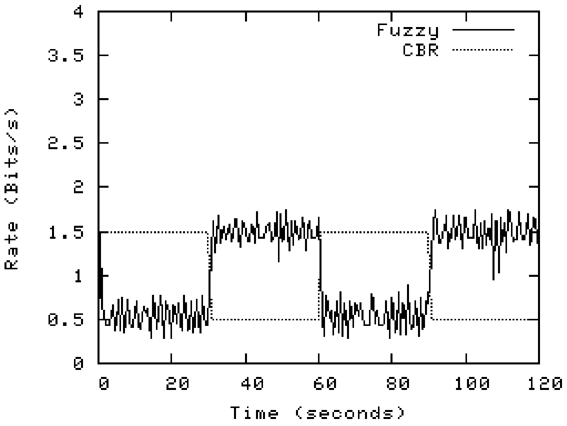

Fig. 1 Instantaneous rate for a 2.0 Mbit/s bottleneck with CBR traffic and CBR cross traffic.

Figs. 5 and 6 illustrate a second set of calibration experiments; in this case, streaming a 3 Mbit/s MPEG-2 ‘news clip’ video trace at 25 frame/s against a CBR background cross-traffic. The bottleneck link was again set respectively to 2 Mbit/s and 3 Mbit/s. The results show that the fuzzy system clearly detects and adapts to the available network bandwidth. The fuzzy-controlled rate is more ‘bursty’ than the previous CBR stream, as a consequence of the inherent burstiness in the instantaneous rate of the news clip, especially when a scene contains motion. In practice, one would also apply a smoothing scheme [9] to the video, which further improves rate control. The average rate of the controlled video reacts to the available network bandwidth, Fig. 7, which shows the average sending rate for a bottleneck rate varied between 2 Mbit/s and 3 Mbit/s, i.e. each data point is the resulting of (replicated) applications of a simulation like that of Fig. 5. Fuzzy control closely tracks an ideal bandwidth-sharing source; as the bottleneck capacity increases, the fuzzy-controlled stream acquires more of that bandwidth, while the CBR share remains constant.

The transcoder architecture was tested directly for TCP-friendliness against an FTP flow (compare [21]) and a summary of the results is shown in Fig. 8. The Figure is a comparison of the time-averaged rates of the fuzzy logic controlled video rate and the time-averaged rate of a TCP cross-connection (FTP). (The individual time plots are difficult to interpret graphically due to burstiness.) The fuzzy-controlled video rate has an increasing trend tracking the ideal rate presented by a perfect TCP-friendly source.

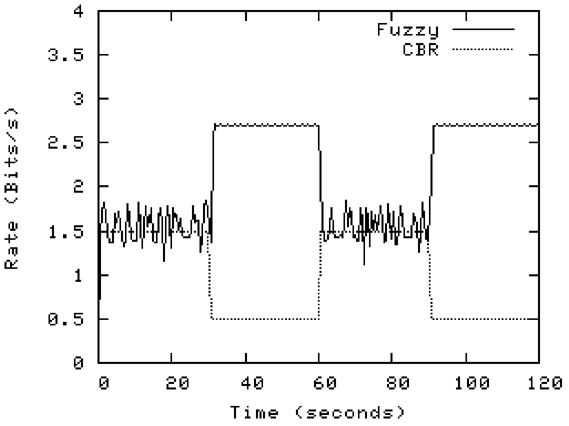

Fig. 2 Instantaneous rate for a 3.0 Mbit/s bottleneck with CBR traffic and CBR cross traffic.

4. Conclusion

This paper has presented a set of experiments that indicate that fuzzy control does not induce network instability, especially in conditions of high congestion across a tight link. In fact, fuzzy control achieves this in conditions of nonlinearity, whereas some control systems assume that feedback is independent of the controlled stream. Achieving TCP-friendliness implies backwards compatibility with TCP-dominated general networks and forwards compatibility with networks with a low or high bandwidth latency product [22].

Experiments are on-going investigating both the response to aggregated long-lived TCP flows (as in FTP and peer-to-peer) and short-lived TCP flows (typical of web traffic). Triangular membership functions can be changed to bell-shaped (at a cost in computation time) and the model optimized by neuro-fuzzy adaptation, further tuning the responsiveness to other traffic.

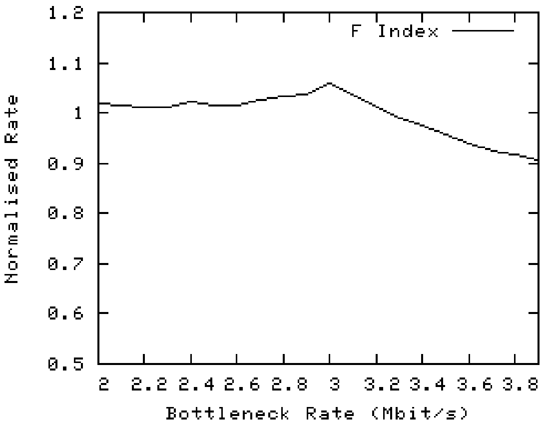

Fig. 3 Normalised average rate for varying bottleneck rates with CBR traffic and CBR cross traffic.

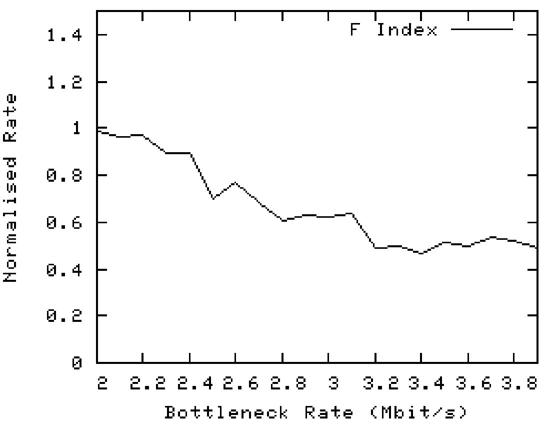

Fig. 4 Normalised average rate for varying bottleneck rates with CBR traffic and FTP cross traffic.

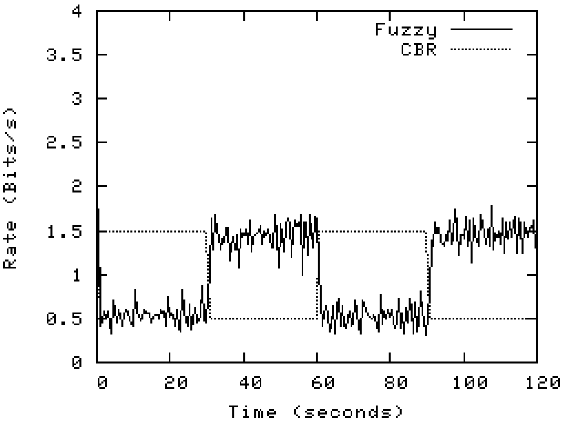

Fig. 5 Instantaneous rate for a 2.0 Mbit/s bottleneck with MPEG2 video traffic and CBR cross traffic.

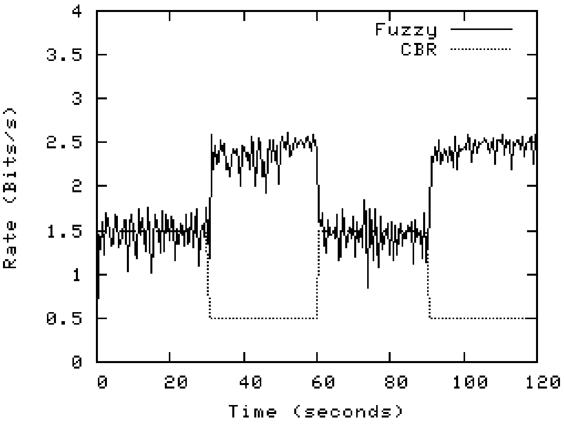

Fig. 6 Instantaneous rate for a 3.0 Mbit/s bottleneck with MPEG2 video traffic and CBR cross traffic.

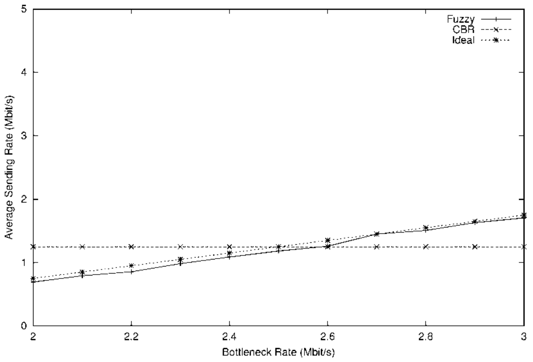

Fig. 7 Time-averaged (over 50 s) sending rates for varying bottleneck rates with video traffic and CBR cross traffic.

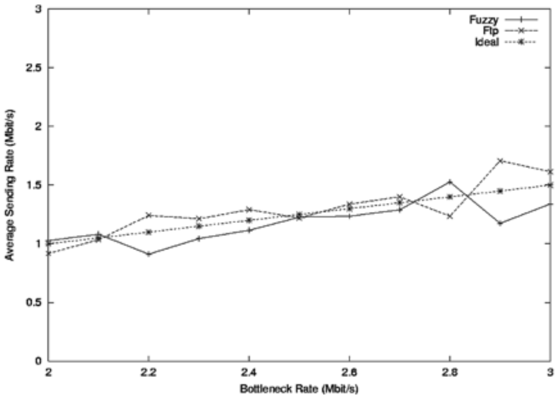

Fig. 8 Average sending rates for varying bottlenecks with video traffic and FTP cross traffic.

Acknowledgement

This work is being carried out with assistance from an EPSRC Grant, Reference No. EP/C538692/1.

References

[1] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seeley and C. Diot, “Packet-level traffic measurements from the Sprint IP backbone”, *IEEE Network,* 17(6):6-17, 2003.

[2] P. Gevros, J. Crowcroft, and S. Bhatti. “Congestion control mechanisms and the best effort service model”, *IEEE Network*, 15(3):16-26, 2001.

[3] T. J. Shanableh and M. Ghanbari, “Hybrid DCT/pixel domain architecture for heterogeneous video transcoding”, *Signal Processing: Image Communication, Special Issue on Multimedia Adaptation*, 18(8):601-620, 2003.

[4] S. Floyd and K. Fall, “Promoting the use of end-to-end congestion control in the Internet”, *IEEE/ACM Transactions on Networking*, 7(4):458-472, 1999.

[5] K. Claffy, G. Miller, and K. Thompson,“The nature of the beast: Recent traffic measurements from an internet backbone”, In *Internet Society Conf., INET'98*, 1998.

[6] Y. Richard and S. L. Lam. “General AIMD congestion control”, In *8*th *International Conference on Network Protocols, (ICNP 2000)*, pages 187-198, 2000.

[7] R. Rejaie, M. Handley, and D. Estrin, “RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet”, In *IEEE INFOCOM'99*, vol. 3, pp. 1337-1345, 1999.

[8] J. Widmer, R. Denda, and M. Mauve, “A survey on TCP-friendly congestion control”, *IEEE Network*, 15(3):28-37, 2001.

[9] E. Jammeh, M. Fleury, and M. Ghanbari. “Smoothing transcoded MPEG-1 video streams for Internet transmission”, *IEE Proceedings: Vision, Image and Signal Processing*, 151(4):298-305, 2004.

[10] S. McCanne, V. Jacobson, and M. Vetterli. “Receiver-driven layered multicast” In *ACM SIGCOMM '96*, pp. 117-130, 1996.

[11] T. Shanbleh and M. Ghanbari, “Multilayer transcoding with format portability for multicasting of single-layered video”, *IEEE Transactions on Multimedia*, 7(1):1-15, 2005.

[12] M. Handley, S. Floyd, J. Padyhe, and J. Widmer. “TCP friendly rate control (TFRC): Protocol specification”, 2003. RFC 3448.

[13] J. Padhye, V. Firoiu, D. Towsley, and J. Kursoe. “Modeling TCP throughput: A simple model and its empirical validation”, In *ACM SIGCOMM '98*, pp. 303-314, 1998.

[14] J. Widmer, C. Boutremans, and J.-Y. Le Boudec, “End-to-end congestion control for TCP-friendly flows with variable packet size”, *Computer Communication Review*, 34(2):137-151, 2004.

[15] M. Jain and C. Dovrolis, “Pathload: A measurement tool for end-to-end available bandwidth”, In *Passive and Active Measurements (PAM) Workshop*, 2002.

[16] B. Melander, M. Björkman, and P. Gunningberg, “A new end-to-end probing and analysis method for estimating bandwidth bottlenecks”, In *IEEE Globecom Internet Symposium*, pp.415-420, 2000.

[17] M. Jain and C. Dovrolis, “End-to-end available bandwidth: measurement methodology, dynamics, and relation with

TCP throughput", *IEEE/ACM Transactions on Networking*, 11(4):537.549, 2003.

[18] V. Ribeiro, R. Riedi, R. Barabanuik, J. Navratil, and L. Cottrell, "pathchirp: Efficient available bandwidth estimation for network paths", In *Passive and Active Measurement Workshop (PAM)*, 2003.

[19] E. H. Mamdani and H. Assilian, "An experiment in linguistic synthesis with a fuzzy logic control", *International Journal of Man-Machine Studies*, 7(1):1-13, 1975.

[20] N. R. Sastry and S. S. Lam, "CYRF: A theory of window-based unicast congestion control", IEEE/ACM Transactions on Networking, 13(2):330-342, 2005.

[21] D. Sisalem and A. Wolisz, "LDA+ TCP-friendly adaptation: A measurement and comparison study", In *10*th *International Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2000.

[22] D. Katabi, M. Handley, and C. Rohrs, "Internet congestion control for high bandwidth-delay product networks", In *ACM SIGCOMM*, pp. 89-102, 2002.