INTERVAL TYPE-2 FUZZY LOGIC CONGESTION CONTROL OF VIDEO STREAMING

E. Jammeh, M. Fleury, C. Wagner, H. Hagras, and M. Ghanbari

Computing and Electronics Department, University of Essex, United Kingdom

Keywords: Fuzzy logic, congestion control, video streaming

Abstract

Intelligent congestion control is vital in encoded video streaming of a clip or film, as network traffic volatility requires constant adjustment of the bit-rate. Equation-based solutions to congestion control are prone to fluctuations in the delivery rate and may respond only when packet loss has already occurred, while both fluctuations and packet loss affect the end-user's perception of the delivered video. A type-1 fuzzy logic controller can operate at video display rates and can cope with uncertainties in packet delay measurements but this paper proposes an interval type-2 fuzzy logic congestion controller, as this has the ability to anticipate un-modeled network states, besides potentially reducing training time prior to deployment. The paper demonstrates an order of magnitude improvement in delivered video quality using type-2 fuzzy logic in respect to type-1 logic, when the control inputs are subject to noise, and reduced packet loss compared to an equation-based controller.

1 Introduction

This paper proposes intelligent control of video transport by means of interval type-2 (IT2) fuzzy logic. IT2 fuzzy logic is more robust to network traffic volatility and uncertain traffic measurements than traditional (type-1) fuzzy logic, and also can outperform analytic congestion controllers in terms of delivered video quality, as this paper demonstrates in respect to the established TCP-friendly Rate Control (TFRC) [9]. Video transport across a network is achieved by determining the instantaneous available bandwidth along a network path and adapting the bit-rate of the encoded video stream. In an Internet Protocol (IP) network arriving traffic streams may contend for bandwidth if there is no admission control. If the IP network is an internet[1] then packet delivery is best-effort and packets are dropped at intermediate router buffers if congestion becomes too great resulting in loss of delivered video quality. For encoded video[2], the bit-rate is adapted either by changing the compression quantization parameter at a live-video encoder or an intermediate transcoder after partial decoding. In a congested network, a key problem for video is the fragile nature of the compressed stream, which means that the loss of particular packets and more generally particular pictures or frames has a knock-on effect at the decoder.[3] Video streaming is a delay-intolerant application implying that it is better to deliver a lower-quality video clip or film than to resend packets. Because of its real-time nature, measurement of available bandwidth is generally performed by observing in real-time the packet arrival statistics of the video stream packets. A network path's available bandwidth is volatile as differing data flows including video streams arrive and leave network links along the path. This uncertainty means that an equation-based congestion controller, especially those that rely on packet loss feedback, may be unsuitable.

However, TFRC has the advantage that, at least in principle, its basis appears as a mathematical formula, giving confidence in the ability to predict its response in different network conditions. On the other hand, a IT2 fuzzy-logic congestion controller (FLCC) can rely on packet delay feedback, allowing queue build-up at intermediate buffers to be anticipated, hopefully before packet loss occurs. It is also robust to uncertainty in input measurements and uncertainty in the range of its membership functions. This latter quality brings confidence in the FLCC's ability to cope with a range of possibly un-modeled network traffic conditions at training time.

A traditional, type-1 FLCC is not completely fuzzy, as the boundaries of its membership functions are fixed. This implies that there may be unforeseen traffic scenarios for which the existing membership functions do not suffice to model the uncertainties in the video stream congestion control task. An IT2 FLCC can address this problem by extending a Footprint-of-Uncertainty (FOU) on either side of an existing type-1 membership function. In IT2 fuzzy logic, the variation is assumed to be constant across the FOU, hence the designation 'interval'. Though the possibility of type-2 fuzzy systems has been known for some time [20], only recently [13] have algorithms to calculate an IT2 output control value at video rate become available. The first IT2 controllers [8] are now emerging, in which conversion or retyping from fuzzy IT2 to fuzzy type-1 takes place before output. Not only does such a controller bring confidence that re-tuning will not be needed for when arriving traffic displays unanticipated or un-modeled behavior but the off-line training period required to form the mem-

[1] Rather than a network that merely uses IP packet framing.

[2] Most networks will not support the transport of very high bit-rates of raw video.

[3] A group of pictures (GOP) commonly consists of 12 or 15 pictures with the first intra-coded picture forming a decoding anchor for the others in the GOP.

bership functions can be reduced. This paper extends an existing FLCC [10] for video streaming to an IT2 FLCC and compares the performances in the presence of measurement noise, which is artificially injected to test the relative robustness. Encouragingly, the delivered video quality[4] is equivalent to the successful type-1 FLC when the measurement noise is limited and under test results in a considerable improvement when the perturbations are large.

In our application, FLCC is a sender-based system for unicast video streams. The receiver returns a feedback message indicating changes to the delay experienced by video stream packets crossing the Internet. This allows the sender to compute the network congestion level and from that the FLCC estimates the response. An FLCC can be efficiently implemented by means of a look-up table of quantized model output values or directly in hardware, including for the IT2 FLCC of this paper [12]. The need for a hardware congestion controller implies that once developed the FLCC models should have wide applicability *without* retuning, whatever the traffic conditions at a bottleneck or tight link[5] The same controller should also be able to cope with a range of Internet path delays and with video streams with differing characteristics in terms of scene complexity, motion, and scene cuts.

The remainder of this paper is organized as follows. Section 2 reviews other applications of fuzzy logic to network traffic control and Section 3 demonstrates the type-2 extension to fuzzy logic. Section 4 describes the simulation methodology employed to demonstrate the basic tracking capability of an FLCC and the advantages of an IT2 FLCC, while Section 5 reports the results of the simulations in comparisons between an IT2 and type-1 FLCC and between our IT2 FLCC and TFRC. Finally, Section 6 draws some wider conclusions.

2 Related work

In a survey of congestion control through computational intelligence, the authors of [16] observe that not much work has been reported on deploying natural algorithms within the Internet. Asynchronous Transfer Mode (ATM) networks, which employ access control to virtual circuits, are one domain to which fuzzy logic has been more extensively applied [3][6]. Because of its resemblance to ATM admission control, fuzzy logic bitrate control has been applied to Bluetooth (IEEE 15.4.1) wireless links [17]. For the same reason, in a number of papers, the authors of [18] have explored fuzzy logic to improve the performance of the Random Early Discard (RED) router queue algorithm and in [19] DiffServ buffer occupancy for each class of layered video packets. Within video coding fuzzy logic has found an application [11][7] in maintaining a constant video rate by varying the encoder quantization parameter according to the output buffer state, which is a complex control problem without an analytical solution.

3 FLCC control

The FLCC determines incipient congestion from queuing delay. For each received packet indexed by i

$$OWD_i = T_r - T_s, \tag{1}$$

where T_r is the receive time of the current packet and T_s is the time the packet was sent. When it is appropriate, the computed OWD_i updates the minimum and maximum one-way delays (OWDs), OWD_{min} and OWD_{max}, on a packet-by-packet basis. Subsequently, the maximum queuing delay is found as $maxQD = OWD_{max} - OWD_{min}$. The queuing delay over the network path, QD_i is computed from the measured delay and the minimum delay:

$$QD_i = OWD_i - OWD_{min} \tag{2}$$

and an exponentially weighted average of the queuing delay for the i^{th} received packet is formed by,

$$avgQD_i = (1-\alpha) \times avgQD_{i-1} + \alpha \times QD_i \tag{3}$$

where $\alpha \leq 1$ is the forgetting constant. In simulations, α was set to 0.1.

The queuing delay is a measure of network congestion, and the ratio of the average queuing delay to the maximum queuing delay is a measure of bottleneck link buffer fullness. A delay factor (d_f) is computed from the average queuing delay and the maximum queuing delay,

$$d_f = \frac{avgQD_i}{maxQD} \tag{4}$$

where d_f ranges between [0,1] with 0 indicating no incipient congestion, 1 indicating full-blown congestion, with shades of incipient congestion between 0 and 1. d_f is an early notification of congestion and is the first input to the IT2 FLCC.

A trend analysis method is used to determine the general trend of the average delay. In each measurement epoch, a number k of queue delay samples are grouped into τ groups where $\tau = \sqrt{k}$. We use the pairwise comparison test (PCT) to determine the overall trend of the queueing delay as shown in (5).

$$T_{PCT} = \frac{\sum_{i=2}^{\tau} I(M^i > M^{i-1})}{\tau - 1} \tag{5}$$

where M^i is the median of group i and I(X) is 1 if X holds and 0 otherwise. The value of T_{PCT} is sent back to the sender where a fuzzifier determines the level was increasing or not according to a membership function.

IT2 input membership functions for d_f and trend are constructed, Fig. 1, as an extension of the type-1 FLCC through an FOU at the boundaries of the formerly crisp (fixed)

[4]Calculated as the luminance Peak Signal-to-Noise Ratio (PSNR) measured in dB, i.e. $10\log_{10} x$, where x is a normalized, mean-square error summation of the pixel-wise difference between transmitted and received video frames.

[5]The tight link is the link with lowest available bandwidth at any one time (owing to arriving cross-traffic) and effectively determines buffer congestion. Such tight links commonly occur at the edges of networks in the transition from the core network to a campus or corporate network or in an access network to the home.

membership functions. Assuming the usual singleton input of d_f (or T_{PCT}), an interval set requires just an upper and lower value to be resolved to form the resulting FOU in the corresponding output set. For example, Fig. 2 shows two IT2 membership functions for input sets A and B, each with an FOU. Singleton input X is a member of each with different degrees of membership. Strictly, an infinite number of membership functions (not all necessarily triangular) can exist within the FOUs of sets A and B, but IT2 sets allow the upper and outer firing levels to be taken, as shown in Fig. 2. The minimum operator (min) acts as a t-norm[6] on the upper and lower firing levels to produce a firing interval. The firing interval serves to bound the FOU in the output triangular membership function shown to the right in Fig. 2. The lower trapezium outlines the FOU, which itself consists of an inner trapezoidal region that is fixed in extent. The minimum operator, also used by us as a t-norm, has the advantage that it requires less hardware circuitry than a product t-norm. Once the FOU firing interval is established, Center-of-Sets type reduction was applied by means of the Karnik-Mendel algorithm, which is summarized in [13]. Type reduction involves mapping the IT2 output set to a type-1 set. In practice, defuzzification of this type-1 output fuzzy set simply consists of averaging maximum and minimum values. The result of defuzzification is a crisp value that determines the change in the video rate.

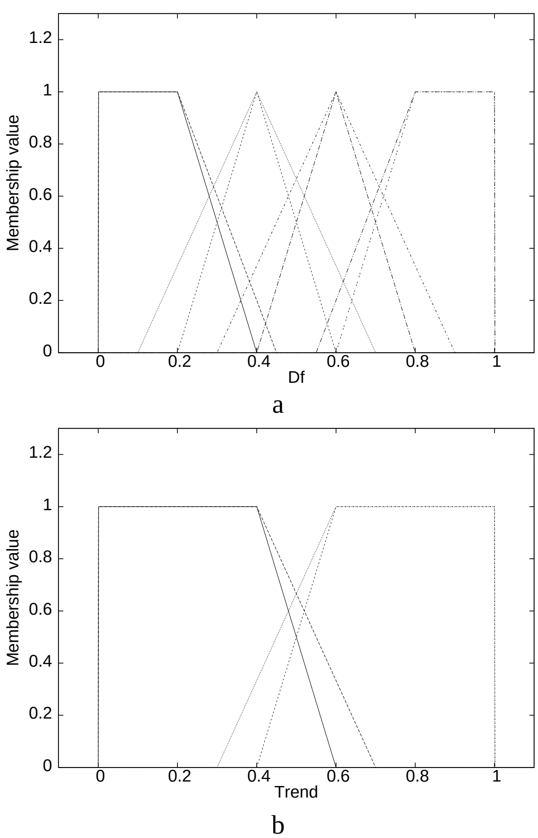


Figure 1: IT2 FLCC (a) Delay factor (Df) (b) Trend membership functions.

[6] A t-norm or triangular norm is a generalization of the intersection operation in classical logic.

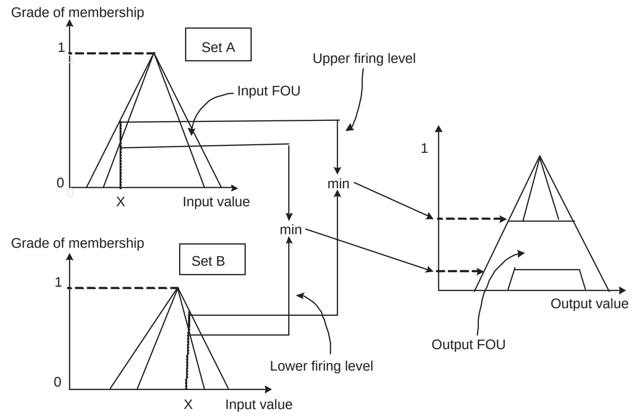


Figure 2: IT2 FL calculation of output FOU

4 Methodology

Fig. 3 shows the video streaming architecture modeled in our experiments. A video transcoder at the server is necessary for pre-encoded video-rate adaptation, while a video decoder at the client decodes the received video stream in real time, prior to display. The in-house frequency-domain video transcoder reported in [1] was employed in this paper's experiments and applied to MPEG-2 pre-encoded video streams. This transcoder obtains a new quantizer scale through a linear rate-quantization dependency. For on-line video clip streaming, it is the video encoder's rate that is adapted. Algorithms for rate adaptation in the spatial domain for the standard codecs are documented in [14] and in the standards documents themselves.

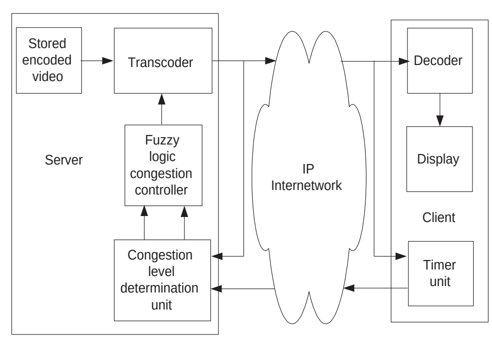


Figure 3: Video-streaming architecture

The client-side timer unit monitors the delay of incoming packets and relays this information to the congestion level determination (CLD) unit. The CLD module monitors the departure times of the outgoing packets. This unit also estimates the trend of packet delay based on feedback from the timer unit at the receiver. The timer unit monitors the arriving packet delays before finding a time-smoothed and normalized estimate of the packet delay. The FLCC takes inputs from the CLD unit and computes a sending rate that reflects the network's state. The appropriate change in the transcoder (or video encoder) quantization level is then calculated. Transported packets are received by the client, depacketized, decoded and displayed at video rate.

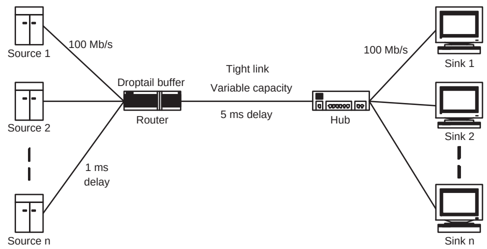


Figure 4: Network configuration simulated in the evaluation of the FLCCs.

The performance evaluation of the FLCC consisted of simulations using models within the well-known ns-2 network simulator (v. 2.31 used) [2], with FLCC implemented as a new protocol within ns-2. The simulated network, with a typical dumbbell topology, had a tight link between a router and a hub. In Fig. 4, one or more video sources stream across the tight link. Critical network behavior is focussed at the tight link, though the identity of that tight link will change over time. Therefore, in the simulations it is assumed that the tight link characteristics are static over time. All side link bandwidths were configured such that congestion will only occur at the tight link (by setting the sidelink capacities to 100 Mbit/s). The one way delay of the tight link was set to 5 mS (representing a total delay across a network path), and the side-link delays were set to 1 mS. The router heading the tight link was configured with a buffer with its maximum queue size set to twice the delay bandwidth product, as is normal in such experiments to avoid packet loss from simply setting too small a buffer size. The default First-In First-Out (FIFO) or droptail queueing policy was set at the buffer.

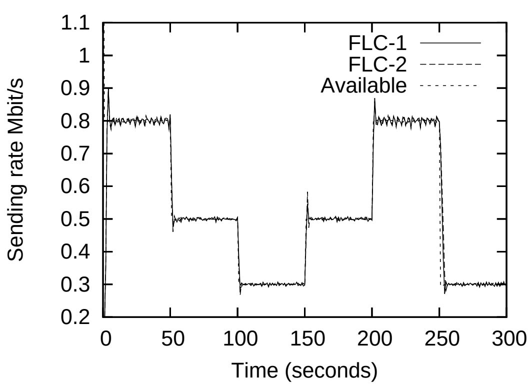


Figure 5: IT2 and type-1 FLC sending rates for stepped available bandwidths.

5 Simulation Results

5.1 Bandwidth tracking

The objective of initial tests was to establish whether an IT2 FLCC should be preferred to a type-1 controller. The type-1 FLCC was the same one as reported in [10], with the extension to fuzzy trend models. The IT2 FLCC employed the *same* membership functions as in [10] but with type-2 FOUs, as reported in Section 3. Initial tests measured the ability of IT2 and type-1 FLCC to track a stepped available bandwidth across the tight link. In Fig. 5, in separate tests both controllers vary the rate of a Constant Bit Rate (CBR) source, with a rate before control of 1 Mbit/s, Both controllers achieve minimal oscillations in the sending rate, which would not be the case for a bandwidth probing congestion controller [10]. There are small over-shoot peaks at the available bandwidth transition points but such drastic changes in background traffic rate are unlikely to occur across a live link. The similarity in response is expected given that the difference between the two controllers should be proportional to the degree of uncertainty in detecting a network's congestion and its trend.

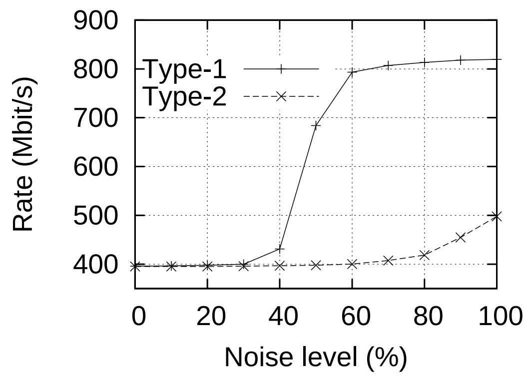


Figure 6: Average sending rate for an increasing noise level.

5.2 Response to measurement noise

In the following simulations, a 40 s video clip was input consisting of a newsreader and changing backdrop with moderate motion, which we designate 'News'. The clip was MPEG-2 encoded [5] at a Variable Bit Rate with a mean rate of 1 Mbit/s over 40 s. The rate was changed through a transcoder and PSNR was found by reconstructing with a reference MPEG-2 decoder. The display rate was 25 frame/s, resulting in 1000 frames (pictures) in each run. The source video was Common Intermediate Format (CIF)-sized (366×288 pixel resolution) with a GOP structure of N = 12, and M = 3. For error resilience purposes, there was on slice per packet, resulting in 18 packets per frame.

FLCC uses delay and its variation to gauge the state of the network. There is, however, inherent noise in the measurement of delay, including packet timestamps with limited resolution and unresolved clock drift between sender and receiver.[7] These uncertainties in the input to an FLCC will potentially impact its performance. A normal distribution generated a random noise value with zero mean and a specified standard deviation, determined by the level of noise required and dynamically adjusted relative to the measured (simulated) value. Thus, with the simulated value for a

[7]The Network Time Protocol is normally employed to synchronize clocks to a universal time but at periodic intervals of the order of milliseconds.

Noise Level %	Type-1	Type-2
0	77.527	76.722
10	78.192	76.607
20	78.986	77.098
30	80.281	77.677
40	102.927	77.747
50	193.612	78.244
60	227.173	80.238
70	230.016	84.294
80	230.651	93.822
90	230.924	113.355
100	231.082	124.652

Table 1: Standard deviation of Type-1 and Type-2 sending rates (kbit/s)

packet's delay, a random variate was added to that value so that in the long-term that additional value would form a given percentage of the delay value. For each simulation the level of additional noise was incrementally increased. At each incremental step, the performance of the two controllers was compared in terms of rate adaptation accuracy, standard deviation (s.d.) of the sending rate, packet loss rate, and delivered video quality (PSNR). The smoothness of the transmission rate (measured by a reduction in the s.d. of the delay on a per-packet basis) is important in video transport as a fluctuating compressed bit-rate implies a fluctuation video quality, which is more disconcerting to a viewer than a stream of consistent quality, even if that average quality was lower than that of a fluctuating stream.

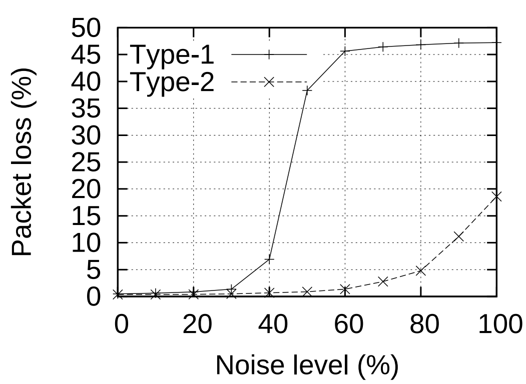


Figure 7: Packet loss rate for an increasing noise level.

The results are gathered in Fig. 6, Table 1, and Figs. 7–8. Below 30% additional noise, the two controllers do not significantly deviate. However, beyond 30% of additional noise, the IT2 FLCC showed significant improvement over the type-1 FLCC in terms of reduced fluctuation (s.d.) in the sending rate and a reduced packet loss rate, both of which will reflect themselves in better average video quality. In fact, Fig. 8 confirms that delivered average video quality is improved, though, for very high levels of measurement noise, the encoded video stream is so corrupt it matters little which FLCC is in control, the quality is very poor.

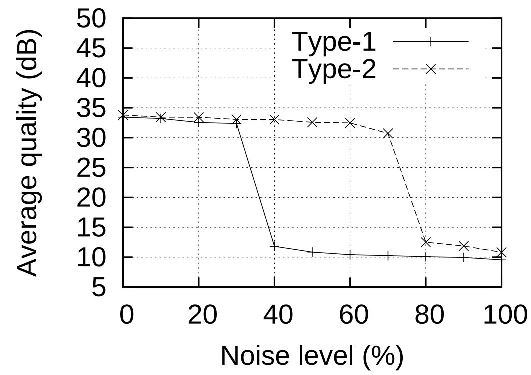
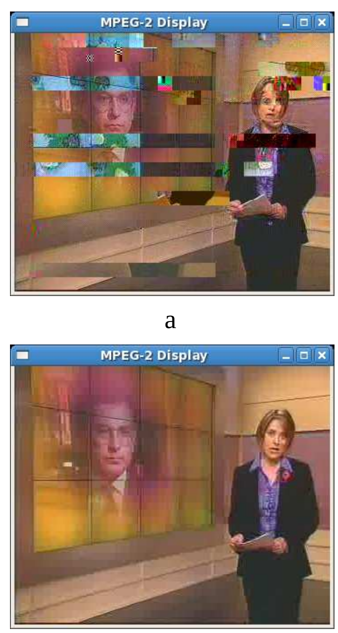


Figure 8: Average received video quality for an increasing noise level.

As a visual comparison, the same video frame is taken from the delivered video stream after decoding and shown in Fig. 9(a) & (b), when the video stream was under the control of the type-1 FLCC and the IT2 FLCC respectively. The improvement from employing the IT2 FLCC is self-evident. The blocky artifacts displayed are typically the result of macroblock errors. Macroblocks are the units of motion estimation to remove temporal redundancy in compression.



a

b

Figure 9: Received video frame after a 40% noise level addition to delay measurements with a) a type-1 FLCC and b) an IT2 FLCC.

5.3 Comparison with TFRC

Comparison was made with the TCP-friendly Rate Control (TFRC) protocol, the subject of an RFC [9] and a prominent method of congestion control from the originators of the 'TCP-friendly' concept [4]. To ensure fairness the publicly available TFRC ns-2 simulator model (in the form of `object tcl` scripts to drive the simulator) was availed of from `http://www.icir.org/tfrc/`. In TFRC, the sending rate is made a function of the measured packet loss rate during a single RTT duration measured at the receiver. The sender then calculates the sending rate according to the TCP throughput equation given in [15].

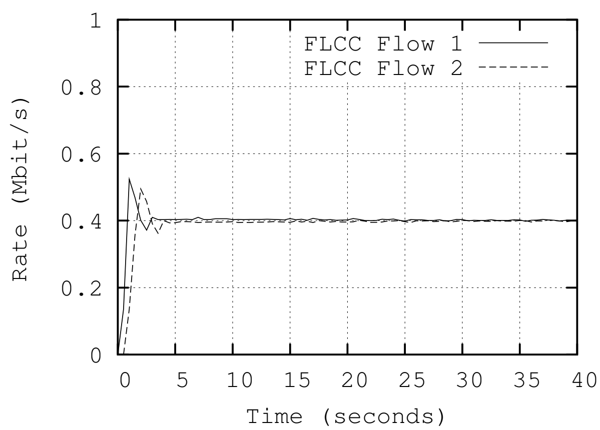
In Table 2, the number of controlled video sources (replicating the source described in Section 5.2) was incrementally increased. Owing to the nature of the simulator, the flows are started one after the other and the non-stationary nature over time of a compressed video's bit-rate results in differing and difficult to control flows. Each video stream is allocated 400 kbit/s bandwidth capacity across the tight link, so that if there are n video streams sharing the link then the bandwidth capacity is $b \times 10$ kbit/s. The bandwidth capacity utilization was calculated by averaging over the mean bit-rates of each stream and then equating that average to the capacity. Thus, a utilization of one indicates that on average the combined sending rates did not exceed the capacity of the link. If the combined sending rates instantaneously exceed the capacity for long enough then packet loss will occur if the router buffer overflows. Consequently, the percentage packet loss rate across all the sources for the duration of the streaming session was calculated in each experiment. Utilization of the IT2 FLCC and TFRC is approximately equivalent. However, packet loss is much reduced when the FLCC was deployed, even though take-up of the bandwidth capacity across the link was almost complete under either controller. The packet loss reduction arises from a reduction in sending rate fluctuations in the case of the FLCC. This is illustrated in plots of instantaneous sent bit-rate over the duration of the clip for the case of $n = 2$ in Fig. 10. Just two streams are plotted, as otherwise the behavior of individual steams is difficult to discern. Apart from initial settling down to steady state behavior, one of the TFRC plots varies around that of the other, whereas this is not the case for the IT2 FLCC streams.

Number of Flows	FLCC		TFRC	
	Utilization	Loss (%)	Utilization	Loss (%)
2	1.001	0.06	1.004	0.46
4	0.999	0.11	1.005	0.48
6	1.000	0.30	1.008	0.65
8	1.009	0.42	1.012	1.01
10	1.001	0.67	1.032	1.26

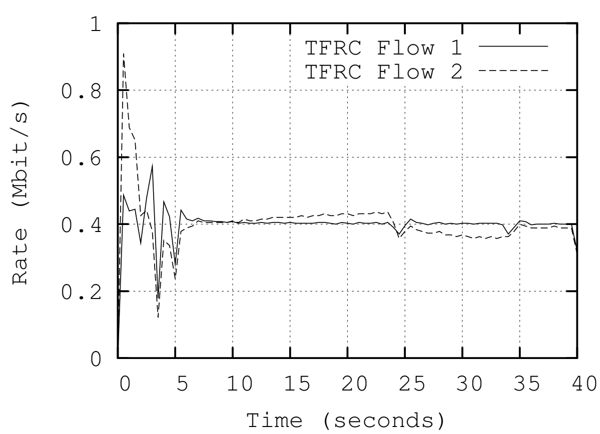
Table 2: Comparison of capacity utilization and packet loss between an IT2 FLCC and TFRC

6 Conclusion

Intelligent control of network traffic flows has been little explored, though policing of networks that have an access control mechanism has received some attention. However, the streaming of encoded video clips is taking an increasing share of bandwidth on the Internet. Video streams are brittle flows in the sense that they are sensitive to packet loss and packet queueing delay. TCP transport is unsuitable as a means of controlling these flows because its very reliability results in delay variation unless large buffers are deployed at the receiver. Unfortunately, such buffers are unsuitable for mobile devices because of the energy drain, even if the click-and-stream culture would permit the start-up delay. Therefore, UDP transport with an application layer congestion controller is the normal solution. Though mathematical modeling of TCP at the application layer as a way of preserving its average behavior has gained ground, this still results in fluctuations in the sending rate and larger packet losses than necessary. Fuzzy logic has been applied to congestion control with satisfactory results. However, the lack of a convenient way to demonstrate the robustness of this solution is an impediment to wider adoption. In this paper we have shown that an interval type-2 fuzzy logic controller preserves all the qualities of a a traditional fuzzy logic controller but is also able to respond to uncertainty in the packet delay measurements that form the principle feedback to the controller. In fact, the ability to cope with considerable corruption of the input was quite dramatic in our results. It was also found that the interval type-2 fuzzy logic congestion controller was able to achieve minimal packet loss (which is highly desirable for delivered video quality) in comparison to the established TFRC controller.



a



b

Figure 10: Illustration of the behavior of individual stream sending rates for (a) IT2 FLCC (b) TFRC

Acknowledgements

This work was supported by the EPSRC, UK under grant no. EP/C538692/1.

References

[1] P. A. A. Assunção and M. Ghanbari. A frequency domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(8):953–967, December 1998.

[2] L. Breaslu, K. Estrin, D. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, X. Ya, and H. Yu. Advances in network simulation. *IEEE Computer*, 33(5):59–67, May 2000.

[3] A. Şekercioglu, A. Pitsillides, and A. Vasilakos. Computional intelligence in management of ATM networks: A survey of current state of research. *Soft Computing Journal*, 5(4):257–263, August 2001.

[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, August 1999.

[5] M. Ghanbari. *Standard Codecs: Image Compression to Advanced Video Coding*. IEE, London, UK, 2003.

[6] S. Ghosh, Q. Razouki, H. J. Schumacher, and A. Celmins. A survey of recent advances in fuzzy logic in telecommunications networks and new challenges. *IEEE Transactions on Fuzzy Systems*, 6(3):443–447, 1998.

[7] P. M. Grant, Y.-S. Saw, and J. M. Hannah. Fuzzy rule based MPEG video rate prediction and control. In *Eurasip ECASP Conference*, pages 211–214, June 1997.

[8] H. Hagras. Type-2 FLCs: A new generation of fuzzy controllers. *IEEE Computational Intelligence*, pages 30–43, February 2007.

[9] M. Handley, S. Floyd, J. Padyhe, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol specification, 2003. IETF RFC 3448.

[10] E. A. Jammeh, M. Fleury, and M. Ghanbari. Delay-based congestion avoidance for video communication with fuzzy logic control. In *17th Int. Workshop on Packet Video*, November 2007.

[11] A. Leone, A. Bellini, and R. Guerrieri. An H.261 fuzzy-controlled coder for videophone sequences. In *IEEE World Conf. on Computational Intelligence*, pages 244–248, June 1994.

[12] M. Melgarejo, A. Garcia, and C. Pena-Reyes. Protwo: A hardware-based platform for real-time type-2 fuzzy inference. In *IEEE Int. Conf. on Fuzzy Systems*, pages 977–982, 2004.

[13] J. M. Mendel. Type-2 fuzzy sets and systems: An overview. *IEEE Computational Intelligence*, pages 20–29, February 2007.

[14] H. Suna nd X. Chen and T. Chiang. *Digital Video Transcoding for Transmission and Storage*. CRC Press, Boca Raton, FL, 2005.

[15] J. Padyhe, V. Firoiu, D. Towsley, and J. Krusoe. Modeling TCP throughput: A simple model and its empirical validation. In *ACM SIGCOMM '98*, pages 303–314, September 1998.

[16] A. Pitsillides and A. Sekercioglu. Congestion control. In W. Pedrycz and A. Vasiliakos, editors, *Computational Intelligence in Telecommunications Networks*, pages 109–158. CRC Press, Boca Raton, FL, September 2000.

[17] R. Razavi, M. Fleury, and M. Ghanbari. Fuzzy logic control of adaptive ARQ for video distribution over a Bluetooth wireless link. *Advances in Multimedia*, 2007. 13 pages, online volume.

[18] L. Rossides, C. Chrysostemou, A. Pitsillides, and A. Sekercioglu. Overview of Fuzzy-RED in Diff-Serv networks. In D. W. Bustard, W. Liu, and R. Sterritt, editors, *Soft-Ware 2002*, pages 2–14, April 2002. LNCS # 2311.

[19] X. Wang, D. Ye, and Q. Wu. Using fuzzy logic controller to implement scalable quality adaptation for stored video in DiffServ networks. In *12$^{\underline{th}}$ Int. Workshop on Packet Video*, April 2002.

[20] L. A. Zaddeh. The concept of linguistic variable and its application to approximate reasoning. *Information Sciences*, 8:199–249, 1975.