TRANSMISSION PROTOCOLS FOR STREAMING VIDEO OVER WIRELESS

Minghua Chen and Avideh Zakhor

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley, CA 94720
{minghua, avz}@eecs.berkeley.edu

ABSTRACT

Rate control is an important issue in video streaming applications for both wired and wireless networks. A widely accepted rate control method in wired networks is TCP friendly equation based rate control [1], i.e. TFRC. However, it assumes that packet loss in wired networks is primarily due to congestion, and as such is not applicable to wireless networks in which the bulk of packet loss is due to error at the physical layer. In this work, we show multiple TFRC connections is an efficient end-to-end rate control solution for wireless video streaming applications. The approach not only avoids modifications to the network infrastructure or network protocol, but also results in full utilization of the wireless channel. Video streaming related simulations are carried out to show the efficiency of our proposed approach.

1. INTRODUCTION

Rate control is an important issue in both wired and wireless streaming applications. A popular rate control scheme over wired networks is equation based rate control [1], in which the TCP Friendly rate is determined as a function of packet loss rate, round trip time and packet size. This approach is known as TCP Friendly Rate Control (TFRC). For streaming over wireless where packets can be corrupted by wireless channel errors at the physical layer, rate control is still an open issue. TFRC can not distinguish between packet loss due to buffer overflow and that due to physical layer errors. Specifically, it has been designed to deal with buffer overflow in wired networks and as such, treats any loss as a sign of congestion. Consequently, a number of techniques have been combined with TFRC to improve its performance over wireless [4, 5]. These methods either hide end-hosts from packet loss caused by wireless channel error, or provide end-hosts the ability to distinguish between packet loss caused by congestion and that caused by wireless channel error. For example Cen et. al. present an end-to-end based approach to facilitate streaming over wireless [5]. Their approach is based on two observations; first, relative one way delay increases monotonically if there is congestion; second, inter-arrival time is expected to increase if there is packet loss caused by wireless channel errors. Therefore, by examining these two quantities they differentiate between congestion and wireless channel errors. However, the high wireless error misclassification rate may result in under-utilizing the wireless bandwidth, as shown in [5]; in addition, their approach requires modifications to congestion control protocol.

Another way to achieve rate control for streaming over wireless is to insert a TFRC-aware Snoop-like module, similar to [4], into the network for local retransmissions when packets are corrupted by wireless channel errors, and to apply TFRC at end-hosts. This way, streaming rate is not affected by wireless channel errors. The advantages of this approach are its simplicity, and robustness to unpredictable wireless channel conditions. The main disadvantage is that it requires modifications to the network infrastructure.

In this paper, we explore the necessary and sufficient condition under which using one TFRC connection in wireless streaming applications results in under-utilization of the wireless bandwidth. We then propose the use of multiple simultaneous TFRC connections for a given wireless streaming application. The advantages of our approach are as follows: first, it is an end-to-end approach, and does not require any modifications to network infrastructure and protocols, except at the application layer. Second, it has the potential to fully utilize the wireless bandwidth provided the number of connections and packet size are selected appropriately. A more detailed exposition of our proposed approach can be found in [7].

The rest of the paper is structured as follows. In Section 2, we present the problem formulation together with an optimal strategy based on multiple TFRC connections. In Section 3, we propose an implementation of the optimal strategy called MULTFRC. Video related simulations are included in Section 4 to demonstrate the effectiveness of MULTFRC. Section 5 concludes the paper.

2. PROBLEM FORMULATION

2.1. Setup and Assumptions

The typical scenario for streaming over wireless is shown in Figure 1 where a video server s in the wired network is streaming video to a receiver r in the wireless network. The wireless link is assumed to have available bandwidth B_w, and packet loss rate p_w, caused by wireless channel error. There could also be packet loss caused by congestion at node 2, denoted by p_c. The end-to-end packet loss rate observed by receiver is denoted by p, and the streaming rate is denoted by T. We refer to the wireless channel as underutilized if the streaming throughput is less than the maximum possible throughput over the wireless link, i.e. $T(1-p) < B_w(1-p_w)$. Without loss of generality, we assume there are no cross traffics at either node 1 or node 2; for the case with cross traffics, see [7].

Given this scenario, we assume the following. First, in the long term, the wireless link is assumed to be the bottleneck. By this, we mean there is no congestion at node 1. Second, we assume there is no congestion and queuing delay at node 2 if and only if wireless bandwidth is underutilized, i.e. we achieve $p_c = 0$ and minimum round trip time, defined as RTT_{min}, if and only if $T \leq B_w$. When $T > B_w$, we have $p_c \geq 0$ and $rtt \geq RTT_{min}$. Third, B_w and p_w are assumed to be constant, and the packet loss caused

This work was supported by NSF grant ANI-9905799 and AFOSR contract F49620-00-1-0327.

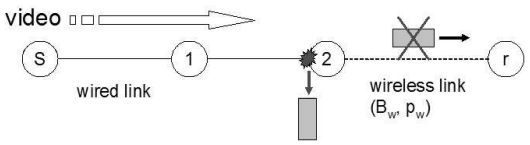

Fig. 1. Typical scenario for streaming over wireless.

by wireless channel error is assumed to be random and stationary. Fourth, for simplicity, the backward route is assumed to be error-free and congestion-free.

Based on this scenario, the two goals of our rate control can be stated as follows. First, the streaming rate should not cause any network instability, i.e. congestion collapse. Second, it should lead to the optimal performance, i.e. it should result in highest possible throughput and lowest possible packet loss rate.

TFRC can clearly meet the first goal, because it has been shown (a) to be TCP-friendly, and (b) not to cause network instability. In the remainder of this paper, we propose ways of achieving the second objective listed above, using a TFRC-based solution, without modifying the network infrastructure and protocols.

2.2. A Sufficient and Necessary Condition for Under-utilization

We use the following model for TFRC to analyze the problem [2]:

$$T = \frac{kS}{rtt\sqrt{p}}, \tag{1}$$

where T represents the sending rate, S is the packet size, rtt is the end-to-end round trip time, p is the end-to-end packet loss rate, and k is a constant factor. Although this model has been refined to improve accuracy [1, 3], it is simple, easy to analyze, and more importantly, it captures all the fundamental factors that affect the sending rate. Furthermore, the results we derive based on this simple model can be extended to other more sophisticated models, such as the one used in [1].

The overall packet loss rate is p, which is a combination of p_w and p_c, and can be written as: $p = p_w + (1 - p_w)p_c$. This shows that p_w is a lower bound for p, and that the bound is reached if and only if there is no congestion, i.e. $p_c = 0$. Combining this observation and (1), an upper bound, T_b, on the streaming rate of one TFRC connection can be derived as follows:

$$T \leq \frac{kS}{RTT_{min}\sqrt{p_w}} \equiv T_b \tag{2}$$

If there is no congestion, i.e. $p_c = 0$, and hence no queuing delay caused by congestion, we get $rtt = RTT_{min}$, $p = p_w$, and therefore $T = T_b$ in (2). In this case, the throughput is $T_b(1 - p_w)$, which is the upper bound of throughput given one TFRC connection for the scenario shown in Figure 1. Based on these, we can state the following:

Theorem 1 Given the assumptions in Section 2.1, sufficient and necessary condition for one TFRC connection to under-utilize wireless link is

$$T_b < B_w. \tag{3}$$

Proof: See the proof of Theorem 1 in [7].

If the condition in (3) is satisfied, then direct application of TFRC to wireless scenario results in under-utilization. In essence, the approaches taken in [4, 5] ensure the condition in (3) is not satisfied, through modifications to network infrastructure or protocols. For example in the TFRC-AWARE Snoop-like solution, p_w becomes effectively zero after local retransmissions, and thus (3) can never be satisfied. By effectively setting $p_w = 0$, Snoop-like module translates the new problem, i.e. rate control for streaming over wireless, into an old one, i.e. rate control for streaming over wired network, for which a known solution exists. Similar observations can be made for the end-to-end statistics based approaches such as [5].

2.3. A Strategy to Reach the Optimal Performance

It is not necessary to avoid the condition in (3) in order to achieve reasonable performance for one *application*. This is because it is conceivable to use multiple simultaneous connections for one application. The total throughput of the application is expected to increase with the number of connections until it reaches the hard limit of $B_w(1 - p_w)$.

In general, given B_w, p_w, and the packet size S for each connection, it can be shown that when full wireless channel utilization occurs, the optimal number of connections, n_{opt}, satisfies:

$$B_w(1-p_w) = \frac{n_{opt}\ kS(1-p_w)}{RTT_{min}\sqrt{p_w}} \Rightarrow n_{opt}S = B_w\frac{RTT_{min}\sqrt{p_w}}{k} \tag{4}$$

Thus what really matters is the product of n_{opt} and S, and as such, it is always possible to achieve full wireless channel utilization by choosing n_{opt} to be an integer, and selecting S accordingly[1]. It is also possible to analyze the case with different packet sizes for different connections, but this is harder to analyze, and is not fundamentally different from the case with the same packet size for all connections. For the rest of the paper, we assume the packet size S is fixed. Then, the optimal number of connections is given by

$$\left\lfloor B_w\ \frac{RTT_{min}\sqrt{p_w}}{kS} \right\rfloor \equiv \hat{n}_{opt} \tag{5}$$

resulting in throughput of $\hat{n}_{opt}\frac{kS}{RTT_{min}\sqrt{p_w}}(1 - p_w)$ and packet loss rate of p_w.

Opening more than n_{opt} connections results in larger rtt, or possibly higher end-to-end packet loss rate. The intuition here is that as number of connections exceeds n_{opt}, the sending rate of each connection has to decrease. Thus by (1), the product $rtt\sqrt{p}$ has to increase, so either rtt increases or p increases, or they both increase [7].

To summarize, if the number of TFRC connections is too small so that the aggregate throughput is smaller than $B_w(1-p_w)$, wireless channel becomes under-utilized. If the number of connections is chosen optimally based on (4), then wireless channel becomes fully utilized, the total throughput becomes $B_w(1 - p_w)$, with $rtt = RTT_{min}$, and the overall packet loss rate achieves the lower bound p_w. However, if the number of connections exceeds n_{opt}, even though the wireless channel continues to be fully utilized at

[1] Of course p_w may also change when packet size changes, but for the sake of simplicity, we assume p_w is fixed as packet size changes. Analysis can be extended given a relation between p_w and S. The point here is to exploit packet size as a way to achieve finer granularity in rate increase/decrease.

$B_w(1-p_w)$, the rtt will increase beyond RTT_{min} and later on packet loss rate can exceed the lower bound p_w. For NS-2 simulations and actual experiments to validate this, see [7].

Thus a strategy leading to optimal performance can be described as follows: *Keep increasing the number of connections until an additional connection results in increase of end-to-end round trip time or packet loss rate.* In Section 3, we use this observation to develop a practical scheme called MULTFRC to determine the optimal number of connections.

3. MULTIPLE TFRC (MULTFRC)

The basic idea behind MULTFRC is to measure the round trip time, and adjust the number of connections accordingly so as to (a) utilize the wireless bandwidth efficiently, and (b) ensure fairness between applications. There are two components in the system: rtt measurement sub-system (RMS), and connections controller sub-system (CCS), both of them residing at the sender.

RMS measures average rtt over a window, denoted by ave_rtt, and reports it to the CCS. Specifically, RMS receives reports from receiver every round trip time, containing the average rtt_{sample} measured in the past round trip time window. RMS then further computes a smoothed version of these average rtt's every m reports, i.e. $ave\_rtt = \frac{1}{m}\sum_{i=1}^{m} rtt\_sample_i$. Setting m to large values can reduce the noise in ave_rtt, while setting it to small values makes the system more responsive to changes in round trip time.

CCS's basic functionality is to Inversely Increase and Additively Decrease (IIAD(α, β)) the number of connections n, based on the input from RMS with α and β being preset constant parameters. Specifically, it first sets the rtt_min as the minimum ave_rtt seen so far, and then adapts the number of connection n as follows:

$$n = \begin{cases} n-\beta, & \text{if } ave\_rtt - rtt\_min > \gamma\ rtt\_min; \\ n+\alpha/n, & \text{otherwise.} \end{cases} \quad (6)$$

where γ is a preset parameter. The reason for this is fair and efficient sharing among multiple MULTFRC applications, and between MULTFRC and TCP or TFRC connections.

For a given route, $ave\_rtt - rtt\_min$ corresponds to current queuing delay, and γrtt_min is a threshold on the queuing delay that MULTFRC can tolerate before it starts to decrease the number of connections. Ideally, ave_rtt becomes larger than rtt_min if and only if the link is fully utilized, and the queue on bottleneck link router is built up, introducing additional queuing delay. Thus by evaluating the relation between ave_rtt and rtt_min, MULTFRC detects full utilization the wireless link, and controls the number of connections accordingly.

When there is a route change either due to change in the wireless base station, or due to route change within the wired Internet, the value of rtt_min changes, affecting the performance of MULTFRC. Under these conditions, it is conceivable to use route change detection tools such as traceroute [6] to detect the route change, in order to reset rtt_min to a new value. Furthermore, it can be argued that the overall throughput of MULTFRC will not go to zero, resulting in starvation; this is because MULTFRC always keeps at least one connection open. We have selected the following parameters empirically: $\alpha = \beta = 1$, $\gamma = 0.2$ and $m = 50$.

In [7], we have evaluated the performance of MULTFRC system through NS-2 simulations and actual experiments over Verizon Wireless 1xRTT CDMA data network. We have shown via simulations that MULTFRC can achieve reasonable utilization of the wireless bandwidth, and does not starve applications that use one TCP connection.

For the actual experiments over 1xRTT, we stream from a desktop connected to Internet via 100 Mbps Ethernet in EECS domain at U.C. Berkeley, to a notebook connected to Internet via Verizon Wireless 1xRTT CDMA data network. In this case it is quite likely that the 1xRTT CDMA link is the bottleneck for the streaming connection. The 1xRTT CDMA data network is advertised to operate at data speeds of up to 144 kbps for one user. As we explore the available bandwidth for one user using UDP flooding, we find the average available bandwidth averaged over eight 30 minutes-long streaming sessions to be between 80 kbps to 97 kbps. The packet size S is 1460 bytes. As we cannot control p_w in actual experiments, we measure the average throughput, average number of connections, and packet loss rate. We compare the performance of MULTFRC system and one TFRC connection in Table 1. As seen, MULTFRC on average opens 1.8 connections, and results in 60% higher throughput at the expense of a larger round trip time, and higher packet loss rate.

Table 1. Actual experimental results over 1xRTT CDMA.

scheme	**throughput (kbps)**	**rtt (ms)**	**packet loss rate**	**ave. # of conn.**
one TFRC	54	1624	0.031	N/A
MULTFRC	86	2512	0.045	1.8

Table 2 shows packet loss details of MULTFRC for a 30 minutes long experiment with packet size of 760 bytes. As expected, both the packet loss rate and burstness of the loss increase as the number of connections increases.

Table 2. Packet loss details of MULTFRC

# of conn. opened	% of time	pkt loss rate	avg. burst error length	snd. dev.	max. burst length
one	24.6	0.015	2.86	3.43	7
two	60.1	0.047	2.41	3.63	10
three	15.4	0.083	3.25	9.93	11

4. VIDEO STREAMING SIMULATIONS

To evaluate the performance of MULTFRC in video streaming applications, we simulate streaming of a 60 second long video clip through a channel, with throughput trace corresponding to one of the the traces obtained from actual experiments over 1xRTT CDMA as described in Section 3. Our goal is to compare the quality of video streaming achievable using one TFRC connection with that of MULTFRC.

We encode 300 frames of *news.cif* sequence using MPEG-4 at bit rates varying from 50kps to 100 kbps as controlled by TMN-5 [8]. The frame rate is 10 frame per second; the I-frame refresh rate is once every fifteen frames. The coded video bit stream is packetized with fixed packet size of 760 bytes. The packets are then protected using Reed-Solomon (RS) codes with different protection levels for one TFRC and MULTFRC. This is because packet loss statistics are different in the two cases. Specifically, the statistics of 30 minutes long trace indicates the longest burst loss to be 6 packets long for one TFRC and 11 packets long for MULTFRC. Thus, we apply RS(56,50) to one TFRC case, and RS(61,50) to MULTFRC case in order to sufficiently protect packets in both cases.

The RS-coded packets are then passed through channels simulated using one TFRC, and MULTFRC packet level traces each lasting 70 seconds, selected from the 30 minutes long actual experiments described in Section 3. The throughput and packet loss details for a 70 second long segment of one TFRC and MULTFRC connections are shown in Fig. 2. As Seen, both the throughput and the packet loss rate are higher for MULTFRC than for one TFRC case.

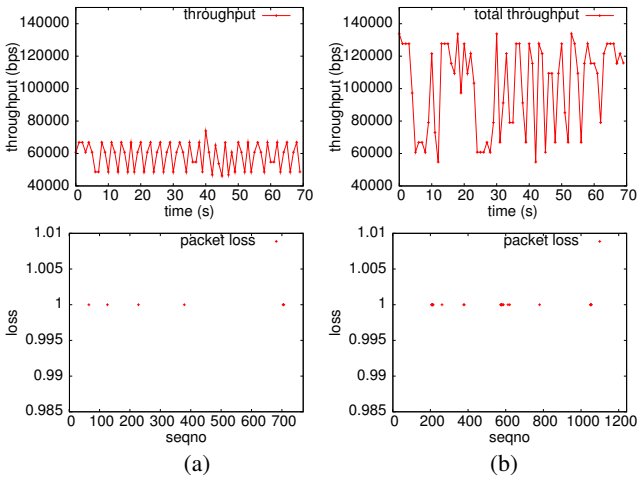

Fig. 2. Throughput and packet loss details for (a) one TFRC; (b) MULTFRC.

The receiver decodes the received RS-coded packets and stores the MPEG-4 bit streams into a playback buffer. In this simulation, we fill the buffer with 10 seconds worth of data before starting the MPEG-4 decode and display process. The playback rate is fixed at 10 frames per second, and hence decoding process is stopped and the display is frozen whenever the playback buffer is empty.

To show the efficiency of MULTFRC, we compare the playback buffer occupancies of MULTFRC and one TFRC for several bit rates in Fig. 3. As seen, compared to one TFRC case, MULTFRC can sustain video streaming at higher bit rates and hence higher visual quality, despite the fact that it needs stronger FEC to combat the higher packet loss rate.

5. CONCLUSION

Rate control is an important issue in video streaming applications for wireless networks, where the channel error based packet loss degrades the performance of traditional rate control schemes, e.g. TFRC. We began this paper by reviewing new results on rate control over wireless. Specifically, we focused our attention on our recently proposed rate control scheme MULTFRC. MULTFRC opens appropriate number of TFRC connections to achieve highest possible wireless bandwidth, minimizing packet loss rate, at the same time avoiding to starve TCP based applications. We then carried out video simulations to demonstrate that MULTFRC can sustain video streaming at higher bit rates, despite the fact that it needs stronger FEC to combat the higher packet loss rate.

Future work includes the analysis on whether the changing the number of connections will introduce instability into the whole network, and investigating the possibility of applying the idea to improve the performance of TCP over wireless.

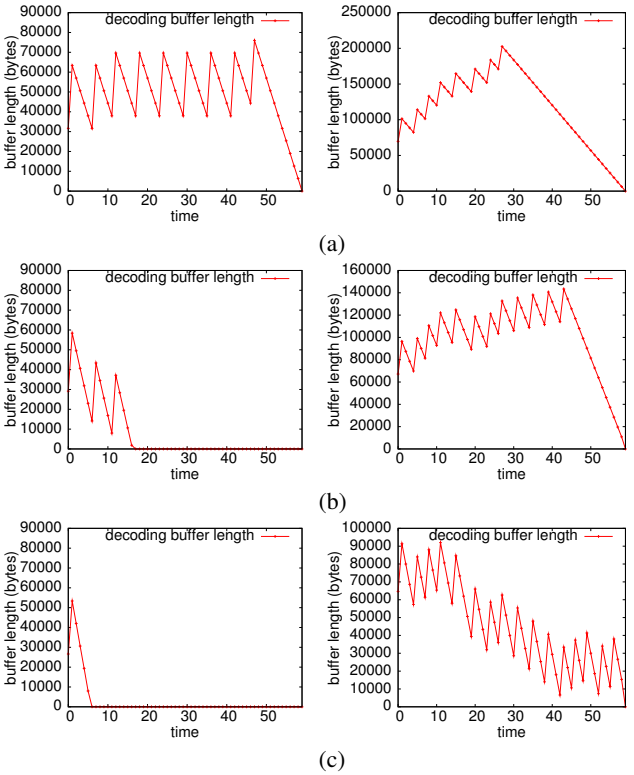

Fig. 3. Throughput and packet loss details for one TFRC (left) and MULTFRC (right): the streaming bit rate is at (a) 50kbps; (b) 70kbps; (c) 90kbps.

6. REFERENCES

[1] S. Floyd, M. Handley, J. Padhye and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", *Proc. ACM SIGCOMM 2000*, Aug. 2000, pp. 43 - 56

[2] S. Floyd and K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet", *IEEE/ACM Transactions on Networking*, Aug. 1999

[3] J.Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation", *ACM SIGCOMM 98*, pp. 303 - 314

[4] H. Balakrishnan, V. Padmanabhan, S. Seshan, R. Katz, "A comparison of mechanisms for improving TCP performance over wireless links", *ACM SIGCOMM'96*, pp.256 - 269

[5] S. Cen, P.C. Cosman, and G.M. Voelker, " End-to-end differentiation of congestion and wireless losses", *Proc. Multimedia Computing and Networking (MMCN) conf. 2002*, pp. 1-15, San Jose, CA, Jan 23-25, 2002

[6] tranceroute, *http://www.traceroute.org/*

[7] M. Chen and A. Zakhor, "Rate Control for Streaming Video over Wireless", Proceeding of Infocom 2004, Hongkong, China, March, 2004

[8] T. Research, "TMN (H.263) encoder/decoder, version 2.0, tmn (h.263) codec", June 1996