Smooth and Efficient Real-time Video Transport in Presence of Wireless Errors

GUANG YANG
University of California, Los Angeles
LING-JYH CHEN
Academia Sinica
TONY SUN, MARIO GERLA and M. Y. SANADIDI
University of California, Los Angeles

In this paper we study a smooth and efficient transport protocol for real-time video over wireless networks. The proposed scheme, named the Video Transport Protocol (VTP), has a new and unique end-to-end rate control mechanism that aims to avoid drastic rate fluctuations while maintaining friendliness to legacy protocols. VTP is also equipped with an Achieved Rate estimation scheme and a Loss Discrimination Algorithm, both end-to-end, to cope with random errors in wireless networks efficiently. We show by analysis that VTP preserves most of the convergence properties of AIMD and converges to its fair share fast. VTP is compared to two recent TCP Friendly Rate Control (TFRC) extensions, namely TFRC Wireless and MULTFRC, in wired-cum-wireless scenarios in Ns-2. Results show that VTP excels in all tested scenarios in terms of smoothness, fairness and opportunistic friendliness. VTP is also implemented to work with a video camera and an H.263 video codec as part of our hybrid testbed, where its good performance as a transport layer protocol is confirmed by measurement results.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms: Video Transport Protocol, TCP Friendly Rate Control, Real-time Video, Wireless Networks

1. INTRODUCTION

Real-time video is becoming increasingly important on the Internet. Unlike conventional applications, it usually requires a minimum, continuous bandwidth guarantee, as well as stringent bounds on end-to-end delays and jitters. TCP [Allman

Authors' address: Guang Yang, Computer Science Department, UCLA, Los Angeles, CA 90095, email: yangg@cs.ucla.edu; Ling-Jyh Chen, Institute of Information Science, Academia Sinica, Taipei 11529, Taiwan, email: cclljj@iis.sinica.edu.tw; Tony Sun, Computer Science Department, UCLA, Los Angeles, CA 90095, email: tonysun@cs.ucla.edu; Mario Gerla, Computer Science Department, UCLA, Los Angeles, CA 90095, email: gerla@cs.ucla.edu; M. Y. Sanadidi, Computer Science Department, UCLA, Los Angeles, CA 90095, email: medy@cs.ucla.edu.

et al. 1999], the *de facto* transport protocol on the Internet, has been considered for video applications [Mehra and Zakhor 2003]. However, the instantaneous sending rate of TCP changes drastically such that large buffering is needed at the receiver to accommodate rate fluctuations [Rejaie et al. 1999a]. Buffering smoothes the video playback, but also brings up two concerns. First, it causes delays. For Video-on-Demand (VoD) applications, playback delays of seconds or slightly longer are tolerable, but for real-time, especially interactive applications, e.g. video conferencing and online gaming, playback delays must be tightly bounded [Sun and Reibman 2001]. The second concern is that more and more mobile/wireless devices are now connected to the Internet. These devices are often small and inexpensive with limited resources where large buffering is impractical. Therefore, real-time video needs a more appropriate *rate control* mechanism that requires little data buffering.

There have been two classes of solutions: cross-layer and end-to-end. On the Internet, deployment of cross-layer approaches [Kazantzidis 2002] typically requires modifications to the network infrastructure, thus end-to-end solutions are the preferred choice. End-to-end rate control can be furthered categorized based on how the rate is adjusted. For example, some schemes [Rejaie et al. 1999b][Bansal and Balakrishnan 2001] adjust the rate in a step-wise fashion, while others use equations derived from analytical models to compute the appropriate sending rate. TCP Friendly Rate Control (TFRC) [Floyd 2000], the most representative of the latter, has gained much popularity as a reference scheme for real-time video transport on the Internet.

TFRC attempts to match the long-term throughput of TCP[1] and is smooth, fair and TCP friendly in wired networks. However, with the increasing popularity of wireless and mobile devices, it is highly desirable to have video transport schemes also work properly across wireless networks. Unfortunately, wireless links are usually error-prone and lossy. Legacy TCP does not handle error-induced packet loss well; it tends to treat every loss as congestion-induced and over-cut its window, leading to severe performance degradation. Since TFRC attempts to faithfully match the throughput of TCP, it suffers the same efficiency problem in the presence of moderate to high random errors [Yang et al. 2001].

Recently, TFRC has been extended for better efficiency in wireless networks. Among the solutions are TFRC Wireless [Cen et al. 2003] and MULTFRC [Chen and Zakhor 2004]. In the meantime, step-wise rate control specifically designed for wireless video is also under wide investigation. In particular, we have proposed the Video Transport Protocol (VTP) in [Yang et al. 2004] and presented preliminary comparison between VTP and TFRC extensions in [Yang et al. 2005]. The newly designed rate control mechanism in VTP keeps monitoring the end-to-end Achieved Rate (AR) to achieve smoother rate adaptation while maintaining friendliness to legacy TCP. Moreover, a Loss Discrimination Algorithm (LDA) is used to distinguish congestion and wireless loss and minimize the impact of random errors. In this paper we perform a comprehensive study on VTP, with the following contributions:

1) Design and analysis of the unique rate control mechanism in VTP that provides smooth, efficient and friendly transport functionality for real-time video in wireless

[1]In this paper we use "TCP" and "TCP Reno" interchangeably unless stated otherwise.

networks. Following the methodology in [Loguinov and Radha 2003], we show that although VTP may temporarily violate the strict monotonous convergence property seen in AIMD, it does preserve other convergence properties of AIMD and converge to its fair share as fast as TCP.

2) Comprehensive evaluation of VTP and comparison to other similar protocols. We implement VTP as an Ns-2 [Ns2] module and assess it in various wired-cum-wireless scenarios. Ns-2 simulations show that VTP meets our design goals and performs well in all tested cases, with good smoothness, efficiency, fairness and friendliness properties.

3) Implementation and measurements of VTP in a networked testbed. We implement VTP in C++ on top of an RTP [Schulzrinne et al. 2003] module and an H.263 codec, and assess its performance in a hybrid network testbed. The measurements match the simulation results, further convincing us that the unique design of VTP is a desirable feature for real-time video in wireless networks.

The rest of the paper is organized as follows. Section 2 presents some background information and work related to VTP. Section 3 briefly recapitulates the design of VTP, especially its unique rate control mechanism. Convergence properties of VTP is studied in Section 4, followed by performance evaluation in simulation in Section 5 and testbed measurements in Section 6. Finally Section 7 concludes the paper.

2. BACKGROUND AND RELATED WORK

2.1 Target Scenario and Design Goals

Wireless LAN (WLAN) is an important target scenario of VTP. In this case, the shared wireless link is the last hop and also the bottleneck. It is much more error-prone than a wired link, where random packet loss may occur frequently.

We envision three design goals of VTP. First, it must provide *smooth rate adaptation*. VTP is proposed for real-time video applications. Tolerable response time for such applications is typically less than 150 msec [Kurose and Ross 2004], thus limited buffering is allowed on the client side. Compared to video-on-demand (VoD) applications, real-time video requires rate control to be smoother and more adaptive to the network dynamics.

Secondly, VTP must be *efficient and robust to errors*. Although error recovery techniques such as Automatic Repeat reQuest (ARQ) and Forward Error Correction (FEC) can hide many losses from the transport layer, they do not completely solve this problem. ARQ increases both end-to-end delay and its variance, and may incur head-of-the-line blocking in a single FIFO queue as performed in the majority of existing devices. FEC is most effective when errors are sporadic. In practice, random errors in wireless networks are bursty [Hsu et al. 1999]. After limited ARQ/FEC where appropriate, packet loss rates of a few percent are still expected to be seen in wireless networks [Tang et al. 2001].

The third goal of VTP is to be *friendly to legacy TCP*. [Floyd 2000] defined the term of *TCP compatibility*. This definition should be followed when legacy TCP is efficient. In wireless networks where TCP cuts its window in excess and loses the efficiency, however, we argue that this conventional definition should be revised, allowing a more efficient protocol to exploit the unused bandwidth. In this paper, we adopt the notion of *opportunistic friendliness* [TCPW]. A new protocol *NP*

is said to be opportunistically friendly to TCP if TCP flows coexisting with *NP* obtain no less throughput than what they would achieve if all flows were TCP.

2.2 Related Work

Rate control for Internet video transport has been well studied in the literature. The Streaming Control Protocol (SCP) [Cen et al. 1998] and Rate Adaptation Protocol (RAP) [Rejaie et al. 1999b] were among the early efforts to provide end-to-end TCP friendly rate control for media streaming applications. SCP invokes a mixed rate- and window-based control policy. It largely mimics the TCP behavior while trying to smooth out the instantaneous rate by maintaining an appropriate amount of buffering in the network. RAP is a rate-based mechanism and employs the Additive Increase Multiplicative Decrease (AIMD) algorithm to adjust its sending rate in a step-wise fashion. It is TCP friendly in a wide range of parameters. However, as later discussed in [Zhang and Loguinov 2004][Yang et al. 2004], the AIMD window adjustment in TCP does not directly translate into AIMD rate control. Therefore, rate-based AIMD schemes have to deal with extra issues such as dynamic RTTs, etc. in order to maintain TCP friendliness.

Binomial algorithms are a class of end-to-end, TCP friendly congestion control algorithms [Bansal and Balakrishnan 2001] proposed to address the rate fluctuation problem in TCP. Binomial algorithms generalize AIMD by changing the amount of window increases/decreases. Two particular algorithms, IIAD and SQRT, were shown to be well-suited to video applications. [Feamster et al. 2001] further studied the interaction between binomial algorithm congestion control and video qualities.

TCP Friendly Rate Control [Handley et al. 2003] is another end-to-end protocol trying to achieve the equivalent long-term throughput of TCP with less short-term fluctuations. Unlike the aforementioned schemes, TFRC largely relies on the equation below to compute the appropriate sending rate:

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \tag{1}$$

where T, s, R, p and t_{RTO} are the upper bound of the sending rate, packet size, RTT, loss rate and Retransmit Time Out value (RTO), respectively.

RAP, binomial algorithms and TFRC were proposed for the wired Internet. Like legacy TCP, they all assume that packet loss is caused by network congestion and reduce the sending rate unconditionally. Therefore, they share with TCP the same inefficiency problem in the presence of random errors and are not directly applicable to wireless networks.

Rate Control Scheme (RCS) was proposed in [Tang et al. 2001] to handle random packet loss over wireless links. RCS uses low-priority dummy packets to probe the available bandwidth on the path, then estimates the admissible rate. RCS requires intermediate gateways to implement a multiple-priority mechanism. This feature is costly and currently unavailable on the Internet.

[Yang et al. 2004] studied end-to-end TCP friendly streaming in 3G systems. They proposed deducing the cause of packet loss and the error rate based on information provided in the RLC and RRC layers. Pure transport-layer rate control for wireless networks have also been investigated. Cen et al. [Cen et al. 2003] studied

different Loss Discrimination Algorithms (LDAs), e.g. Biaz [Biaz and Vaidya 1999], Spike [Tobe et al. 2000], mBiaz, ZigZag, ZBS [Cen et al. 2003], etc., in various wireless topologies. They proposed adding an end-to-end LDA to TFRC to improve its robustness against errors. MULTFRC [Chen and Zakhor 2004] is another TFRC extension for wireless networks. It creates multiple simultaneous TFRC connections on the same path when a single connection cannot utilize the bandwidth efficiently. We will see how VTP compares to TFRC Wireless and MULTFRC later in this paper.

3. THE VIDEO TRANSPORT PROTOCOL (VTP)

The Video Transport Protocol (VTP) was proposed in [Yang et al. 2004] as a simple end-to-end scheme for smooth, efficient and friendly rate control in error-prone wireless networks. VTP consists of two key components: an Achieved Rate (AR) based rate control mechanism and a Loss Discrimination Algorithm (LDA). For the sake of completeness, we now briefly recapitulate the design of VTP in this section.

3.1 TCP Behavior in Terms of Instantaneous Rate

A number of rate-based protocols mimic TCP by adjusting their instantaneous rates in the AIMD fashion. It is often seen as obvious that AIMD *window* adjustment would directly translate into AIMD *rate* adjustment; many times the terms "window" and "rate" were used interchangeably.

Recently, researchers have pointed out [Zhang and Loguinov 2004][Yang et al. 2004] that due to buffering, TCP behavior in terms of rate can be very different from the well-known "sawtooth" shape. Considering TCP Reno operating in congestion avoidance. Due to the self-clocking mechanism, the sender normally does not send out a data packet until an ACK comes in. Before the bottleneck link on the path is fully utilized, the rate of ACKs keeps increasing along with the sending rate of data packets. However, after the bottleneck link is saturated, the rate of ACKs is limited by the bottleneck capacity and will no longer increase, which in turn limits the TCP sending rate. An exception is that TCP transmits one extra packet every RTT to probe the bandwidth. This raises the average sending rate to $C + 1/RTT$ where C is the bottleneck capacity. It has been shown that, under the assumption that the buffer size B is equal to the "pipe" size P, TCP congestion window and instantaneous sending rate can be illustrated as in Figure 1.

3.2 Achieved Rate Estimation

The Achieved Rate (AR) in VTP is defined as the rate that the sender has successfully pushed through the bottleneck on the path. This is the rate that the receiver measures, plus the fraction corresponding to packet loss at the exit of the bottleneck due to random errors. The receiver samples and filters the receiving rate, then reports periodically to the sender. The sender updates the smoothed AR value with an Exponentially Weighted Moving Average (EWMA). If portions of the packets were lost at the exit of the bottleneck due to errors, VTP is able to estimate the fraction of lost packets via the LDA (explained shortly) and prorate AR accordingly.

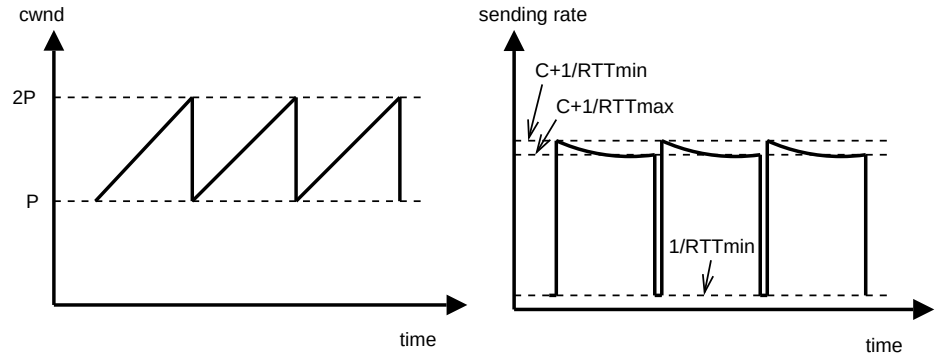


Fig. 1. Congestion window and instantaneous sending rate of TCP NewReno.

Rate estimation is useful to transport layer protocols, especially in error-prone wireless scenarios. In [Gerla et al. 2004] we have studied a few estimation techniques in TCP. Due to the characteristics of real-time video, such as more paced traffic compared to TCP, simple EWMA estimation works sufficiently well.

3.3 VTP Rate Control

VTP must avoid drastic rate reduction while responding to congestion in a TCP friendly way. The tradeoff is between the amount of rate reduction and the length of time this reduced rate is maintained.

Simply speaking, VTP reduces the rate by less but stays at it for longer. Figure 2 illustrates the concept of VTP rate control and compares it to TCP. Upon congestion, TCP chooses the "deep but short" strategy where its rate is cut to near zero and then quickly restored to $C + 1/RTT$. In contrast, VTP reduces its rate by a smaller portion but stays at it for longer. The two shaded areas $A1$ and $A2$ represent the amount of extra data that TCP and VTP would transmit if the loss did not happen. Let $A1 = A2$, the interval over which the reduced rate should be maintained can be calculated. After that, VTP enters congestion avoidance.

During congestion avoidance, VTP tries to match the TCP behavior. Define an "equivalent window" *ewnd* in VTP as the number of packets transmitted by the sender during one RTT. VTP computes its sending rate as follows:

(1) At the beginning of congestion avoidance, *ewnd* is initially set equal to the window size that TCP would have in the same situation.

(2) The sender measures its current *ewnd* by counting the packets transmitted within the current RTT. Letting R be the current transmit rate, we have

$$ewnd = R \cdot RTT \tag{2}$$

(3) Following Additive Increase, the new window $ewnd'$ is given as

$$ewnd' = R \cdot RTT + 1 \tag{3}$$

(4) Converting from window to rate, we have the new rate R' as

$$R' = (R \cdot RTT + 1)/(RTT + \Delta RTT) \tag{4}$$

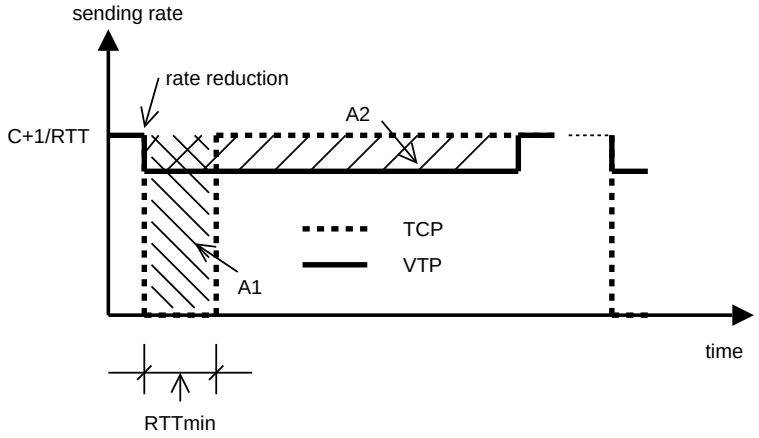


Fig. 2. Comparison of rate control between TCP and VTP.

where ΔRTT is the RTT increase after a round.

(5) Assuming RTT increases linearly in each round, then

$$R' = (R + 1/RTT)/(2 - RTT^{[-1]}/RTT) \tag{5}$$

where $RTT^{[-1]}$ is the round-trip time during the previous round.

3.4 Loss Discrimination

VTP is equipped with an end-to-end LDA to distinguish between congestion and error loss. [Cen et al. 2003] concluded that in the WLAN scenario, *Spike* is a simple and effective algorithm; its efficacy is very close to the much more complex hybrid algorithms. Therefore, VTP has chosen a variant of Spike as its LDA. Simply speaking, the LDA keeps track of the RTT. If the current RTT is close to the minimum RTT, the bottleneck is unlikely to be congested. On the contrary, if RTT is close to the maximum RTT, the path is heavily loaded and congestion is imminent.

We do not intend to further discuss LDAs in this paper. To summarize, the VTP rate control only applies to congestion-induced packet loss. Upon a loss, VTP calls the LDA to determine the cause of the loss; error-induced random loss will be ignored from the rate control action and have no direct impact on it.

4. CONVERGENCE PROPERTIES OF VTP

4.1 Diagrammatic Argument of Convergence

In this section we study the convergence properties of VTP. Following the widely used approach in [Chiu and Jain 1989][Bansal and Balakrishnan 2001][Loguinov and Radha 2003], we first show through the diagrammatic argument that VTP converges to its "fair share" almost identically as TCP, and the speed is as fast as TCP. For convenience, consider one VTP and one TCP connections with the same RTT in an error-free situation. Assume TCP always operates in congestion

avoidance, and for the sake of consistency we consider the equivalent window *ewnd* of VTP.

Two convergence diagrams are drawn in Figure 3. In the left diagram, starting at an arbitrary point s, both TCP and VTP probe the bandwidth linearly, reflected in the diagram as the 45^o line $\overline{sa}$. After the buffer overflow at point a, TCP cuts its window by half. Reflected on the diagram, the starting point of the new cycle is on the medial line of $\overline{ae}$. At the same time, VTP reduces its equivalent window to the value corresponding to no queuing. This is reflected on the diagram as drawing a horizontal line through the intersection of $\overline{oa}$ and the "zero backlog" line, namely g. Therefore, the new probing cycle will start from point b.

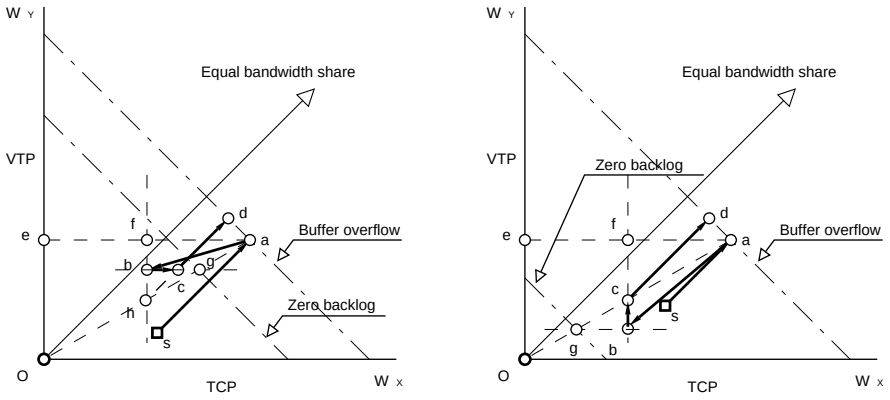


Fig. 3. Two cases of convergence of VTP equivalent window, where VTP is in temporary advantage (left) or disadvantage (right) after the window cut.

If the VTP connection was replaced by TCP, the new probing cycle would start from the midpoint of $\overline{oa}$, denoted as h. The length of $\overline{bh}$ is the gain that VTP obtains over TCP during the window cut. As per VTP rate control, it must refrain from window increase until the "virtual TCP" catches up. On the diagram, it is represented as the flat line $\overline{bc}$. Note that $\overline{bc} = \overline{bh}$. Finally after VTP has given up the right amount of time, both VTP and TCP increase their windows linearly, resulting in the 45^o line $\overline{cd}$. Overall, VTP and TCP converge to the equal bandwidth share as the polyline of $s \rightarrow a \rightarrow b \rightarrow c \rightarrow d$. From basic geometry, we know $\overline{gcd}$ is actually a 45^o straight line. Therefore, if we replaced VTP with TCP, the two TCP connections would converge to the equal bandwidth share as $s \rightarrow a \rightarrow h \rightarrow c \rightarrow d$. Comparing the two cases, it is clear that the temporary disadvantage of TCP (and thus the advantage of VTP) upon buffer overflow is quickly compensated. In other words, VTP converges to its fair share as friendly and fast as TCP.

The right diagram in Figure 3 is different only in that VTP is in temporary *disadvantage* after the window cut. However, VTP is able to adjust its window at the beginning of congestion avoidance, based on the behavior of the "virtual TCP". Thus it is able to make up the deficit and still converge to its fair share quickly.

4.2 Convergence Properties

[Loguinov and Radha 2003] studied in detail the convergence properties of binomial algorithms including AIMD. One important conclusion was that AIMD is the only TCP-friendly binomial algorithm with monotonic convergence to fairness, a nice and desirable property. We now show that the rate control algorithm in VTP may temporarily violate this property, but will promptly be rectified and, in general, preserve most of the convergence properties of AIMD.

According to [Loguinov and Radha 2003], four conditions must hold in the decrease function, and three must hold in the increase function, to guarantee monotonous convergence. Specifically, the four conditions for the decrease function are[2]:

(1) strictly decreasing efficiency, i.e. $e_{i+1} < e_i$;
(2) non-decreasing fairness, i.e. $f_{i+1} \geq f_i$;
(3) no arbitrary crossing of the line of equal bandwidth share, i.e. $(y_i > x_i) \Rightarrow (y_{i+1} > x_{i+1})$ and vice versa;
(4) positive window sizes, i.e. $(x_i > 0 \wedge y_i > 0) \Rightarrow (x_{i+1} > 0 \wedge y_{i+1} > 0)$.

Similarly, the three conditions for the increase function are:

(1) strictly increasing efficiency, i.e. $e_{i+1} > e_i$;
(2) non-decreasing fairness, i.e. $f_{i+1} \geq f_i$;
(3) no arbitrary crossing of the line of equal bandwidth share, i.e. $(y_i > x_i) \Rightarrow (y_{i+1} > x_{i+1})$ and vice versa.

Assume the bottleneck buffer size and the "pipe" size are B and P, respectively. In terms of windows, TCP and VTP congestion/rate control algorithms differ only after congestion loss is detected. Immediately after the detection, we have,

$$x_{i+1} = \frac{1}{2} \cdot x_i, \; y_{i+1} = \frac{P}{B+P} \cdot y_i$$

As B and P can take arbitrary values, we can no longer guarantee that $f_{i+1} \geq f_i$ and/or $(y_i > x_i) \Rightarrow (y_{i+1} > x_{i+1})$ still hold at any moment. For example, if $1 < (x_i/y_i) < (2P/(B+P))$, we have $x_i > y_i$ but $x_{i+1} < y_{i+1}$. This means the fairness measure may temporarily decrease, and/or the line of equal bandwidth share may be crossed.

However, as we have seen in Figure 3, upon congestion loss VTP is able to "compare" its equivalent window to the window that a TCP connection would have in the same situation, and adjust its window accordingly after the rate reduction. More specifically, in the left diagram of Figure 3, windows of the "VTP-TCP" combination evolve along the polyline of $s \rightarrow a \rightarrow b \rightarrow c \rightarrow d$, while windows of the "TCP-TCP" combination would evolve along the polyline of $s \rightarrow a \rightarrow h \rightarrow c \rightarrow d$. In other words, the overall effect after this cycle remains the same if VTP is replaced by TCP or vice versa. Similar is in the right diagram of Figure 3 where both

[2]We define x and y as the window size of TCP and VTP, respectively, $e = x + y$ as the efficiency, $f = min\{x/y, y/x\}$ as the fairness. We use subscript i to indicate a value in the i^{th} interval.

$s \rightarrow a \rightarrow b \rightarrow c \rightarrow d$ ("VTP-TCP" combination) and $s \rightarrow a \rightarrow c \rightarrow d$ ("TCP-TCP") lead to the same point of d when the current cycle ends. Therefore, although VTP loses the strict monotonous convergence property of the AIMD algorithm, the discrepancy between VTP and TCP is quickly fixed, and the long-term effect on the window size remains the same. Thus VTP preserves most of the convergence properties of AIMD and converges to its fair share as fast as TCP.

5. PERFORMANCE EVALUATION

We now evaluate VTP and compare it to other rate control schemes in the WLAN scenario through Ns-2 [Ns2] simulations. We begin with a description of our simulation setup, followed by the evaluation based on various metrics. We will present our testbed measurements in the next section.

5.1 Simulation Setup

We use throughout this section the wired-cum-wireless topology in Figure 4 for simulations. For simplicity, the wired segment is abstracted as one-hop error-free links. We choose to assign each wired link a capacity of 100 Mbps and one-way delay of 20 msec. The wireless segment is set as an 802.11 WLAN running at the 11 Mbps data rate, where nodes are connected to the Internet via the base station. The wireless segment is error-prone; a two-state Markov Chain error model is applied to simulate various error patterns. Unless otherwise stated, in this paper the average time in the "good" state is set to 1 second, while the average time in the "bad" state is varied to achieve the target overall error rate. Error probabilities in these states are set to 0 and 1, respectively.

Previously in [Yang et al. 2004][Yang et al. 2005], we imposed an error model on a wired link to mimic the shared wireless medium. That old setup was not as sophisticated and did not take any PHY or MAC characteristics of an 802.11 WLAN into account. When the error rate is set to zero, the topology essentially becomes a normal wired network. [Yang et al. 2005] has shown that in such a wired network, VTP, TFRC and TFRC Wireless all perform better than MULTFRC, with VTP demonstrating the best smoothness in the rate.

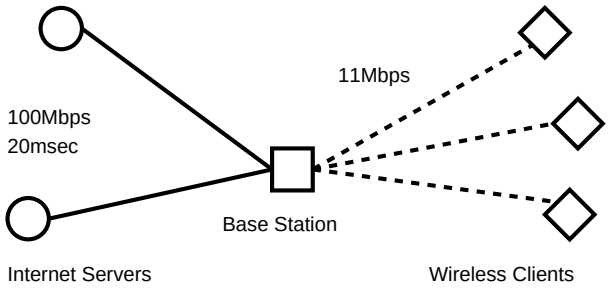


Fig. 4. Simulation setup.

In Figure 4, the foreground traffic is directed from the Internet servers (left) to the wireless clients (right). Since the wired links have relatively high capacities, the

WLAN becomes the bottleneck where packets can be buffered at the base station. We set the queue size to be equal to the pipe size, i.e. the round-trip bandwidth-delay product. Cross traffic is also introduced in some runs to test the adaptivity of the foreground traffic to available bandwidth changes.

5.2 Comparison to Internet Rate Control Schemes

We first compare VTP to four well-known rate control schemes on the Internet. Among them, TFRC is the most popular equation-based transport protocol, TCP represents the traditional AIMD algorithm, IIAD and SQRT are taken from the family of binomial algorithms.

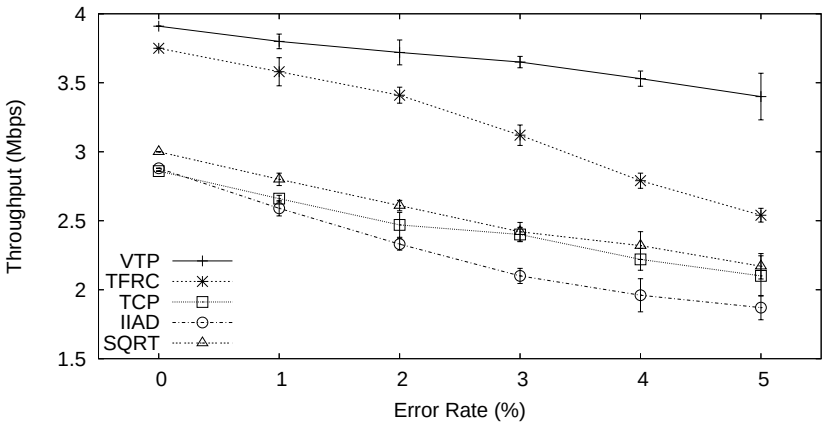


Fig. 5. Comparison of VTP throughput to TFRC, TCP (AIMD), IIAD and SQRT, with 99% confidence intervals.

Figure 5 compares the throughput to these schemes under different error rates. When no errors are present, all five protocols are able to utilize the bandwidth efficiently[3]. Note that TCP (AIMD), IIAD and SQRT are tested as reliable protocols, i.e., they enforce every byte to be received correctly at the receiver, therefore they achieve slightly less throughput due to retransmission overhead, compared to VTP and TFRC which are best-effort protocols.

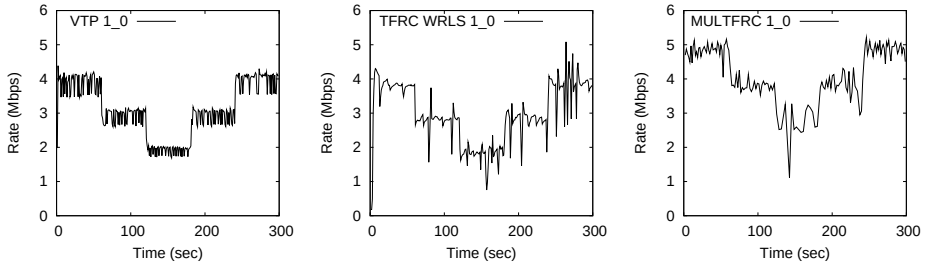
As the error rate grows, VTP is able to stay at high throughput, while others start to drop their efficiency. This is consistent with observations from other researchers. Schemes originally designed for the wired Internet, without wireless extensions, often suffer from efficiency problems and are not applicable in wireless scenarios.

5.3 Efficiency and Rate Adaptivity

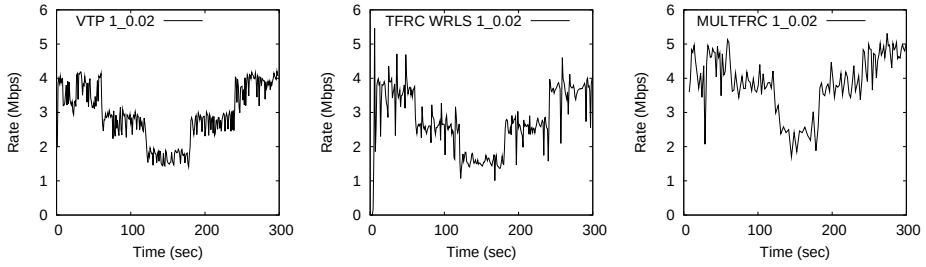
We then evaluate the efficiency and rate adaptivity properties of VTP and compare it to two TFRC extensions, TFRC Wireless and MULTFRC. Figure 6 shows the

[3]Although the nominal data rate in the configured 802.11 WLAN is 11 Mbps, the actual achievable throughput is typically less than half of that, due to PHY/MAC overhead, etc.

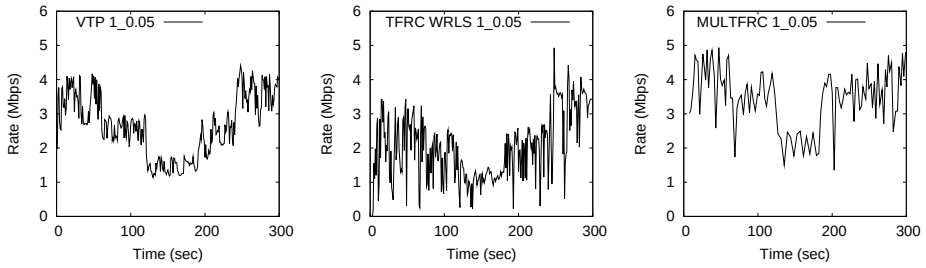
instantaneous sending rates of VTP, TFRC Wireless and MULTFRC during 300-second simulations. To better study the behavior of each protocol, only a single foreground flow is configured, sharing the bandwidth of the WLAN with constant bit-rate (CBR) cross traffic. The aggregate amount of cross traffic is 1 Mbps between the 60^{th} and 120^{th} second and between the 180^{th} and 240^{th} second, and 2 Mbps between the 120^{th} and 180^{th} second.



(a) no errors (avg. time in good state = 1 sec, avg. time in bad state = 0 sec).



(b) 2% errors (avg. time in good state = 1 sec, avg. time in bad state = 0.02 sec).



(c) 5% errors (avg. time in good state = 1 sec, avg. time in bad state = 0.05 sec).

Fig. 6. Instantaneous sending rates of VTP, TFRC Wireless and MULTFRC under different error rates.

In Figure 6, VTP consistently performs better than TFRC extensions in terms of efficiency, smoothness and adaptivity, under various error rates. Interestingly, in contrast to previous results [Yang et al. 2005], MULTFRC performs better than TFRC Wireless. This is mainly due to the following reasons:

1) The effective bandwidth of the WLAN is much lower than 11 Mbps as used in [Yang et al. 2005]. As we have previously pointed out, when the available bandwidth

is low, MULTFRC only needs to manipulate a small number of TFRC connections to reach an equilibrium state, leading to fast response and convergence speed.

2) LDAs based on end-to-end delay monitoring, as used in TFRC Wireless, are less effective in the 802.11 WLAN scenario. PHY and MAC characteristics such as interference, fading, backoff, RTS/CTS, etc. all introduce unpredictable and non-negligible delays not related to queuing. Therefore, long delays may be observed when no significant queuing (i.e. congestion) is actually happening. In this case, the LDA tends to misclassify error loss as congestion-induced and cut the rate unnecessarily, leading to degraded efficiency and more rate fluctuations.

It is worth mentioning that although VTP and TFRC Wireless both use a delay-based LDA, AR estimation helps VTP retain much better performance when the LDA goes wrong. When the LDA misclassifies an error loss as congestion-induced, VTP sending rate is cut to AR, which in most cases is a more accurate estimate of the "eligible" rate than simply cutting by half. Lacking this feature forces the flow to mistakenly overcut its rate as we have seen in TFRC Wireless.

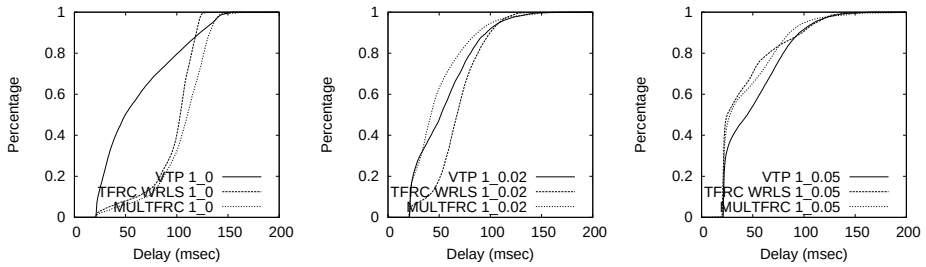


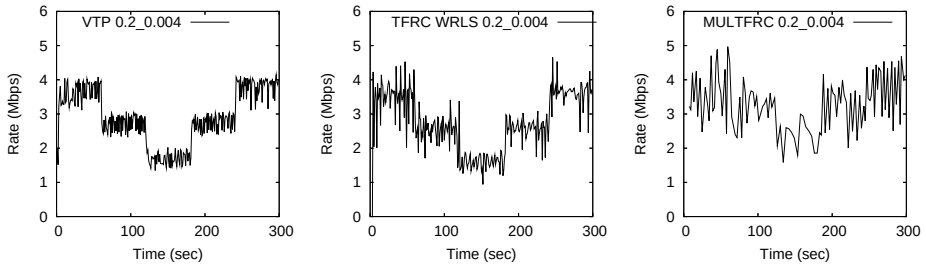
Fig. 7. End-to-end delays of VTP, TFRC Wireless and MULTFRC under different error rates (0, 2%, 5% from left to right, respectively).

In additional to the sending rate, we also evaluate other QoS metrics. Figure 7 presents the cumulative distribution function (CDF) of the end-to-end delay in VTP, TFRC Wirelss and MULTFRC, respectively. In the zero error case, VTP incurs the lowest delay. This difference becomes less obvious in the 2% and 5% error cases, where in general TFRC Wireless and MULTFRC actually have shorter delays. This is largely due to the fact that their efficiency drops in these cases, leading to less MAC contention in the WLAN. All three protocols have very small end-to-end jitters, mostly within 5 msec between consecutive packets in the tested cases.

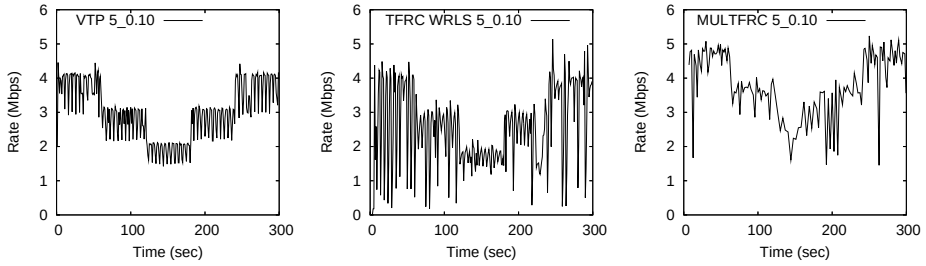
5.4 Burst Errors

To evaluate the impact of different "error burstiness" on VTP, we fix the overall error rate to 2% and change the average time in the "bad" state to 4 and 100 msec, respectively[4]. We plot the instantaneous sending rate in Figure 8 with TFRC Wireless and MULTFRC as comparison.

[4]Previously we had it as 20 msec for the 2% error case.



(a) shorter bursts (avg. time in good state = 0.2 sec, avg. time in bad state = 0.004 sec).



(b) longer bursts (avg. time in good state = 5 sec, avg. time in bad state = 0.1 sec).

Fig. 8. Instantaneous sending rates of VTP, TFRC Wireless and MULTFRC under different error burstiness.

Error burstiness seems to have a negative impact on VTP and TFRC Wireless but be neutral or even positive on MULTFRC. When errors are more bursty, it is more difficult for VTP and TFRC Wireless to maintain a smooth and efficient rate during the prolonged "bad" state. On the contrary, prolonged "bad" and "good" states are beneficial to MULTFRC since the number of transitions between states is reduced, meaning there are fewer connection creations/deletions. Moreover, smoothness and efficiency are less an issue in the "bad" state if multiple connections are used. This indicates that MULTFRC could be most useful in highly bursty error-prone scenarios.

5.5 TCP Friendliness

Fairness and friendliness of VTP as well as TFRC Wireless and MULTFRC have been extensively discussed in [Yang et al. 2005]. In this paper, we look at the property of TCP friendliness of these protocols in the WLAN configuration. In each run, one VTP, TFRC Wireless or MULTFRC connection coexists with one TCP NewReno connection, sharing the WLAN, and their respective throughput is recorded and plotted in Figure 9. MULTFRC achieves a noticeably higher throughput than VTP and TFRC Wireless in all cases. This is because the number of parallel TFRC connections in MULTFRC keeps fluctuating and frequently exceeds one; this grants MULTFRC a significant advantage in bandwidth utilization[5]. However, none of the streaming protocols impair the performance of the coexisting TCP

[5]Similar phenomena are also observed in Figure 6 where MULTFRC usually has a slightly higher instantaneous sending rate than VTP and TFRC Wireless.

connection, therefore all protocols demonstrate good TCP friendliness in our tests.

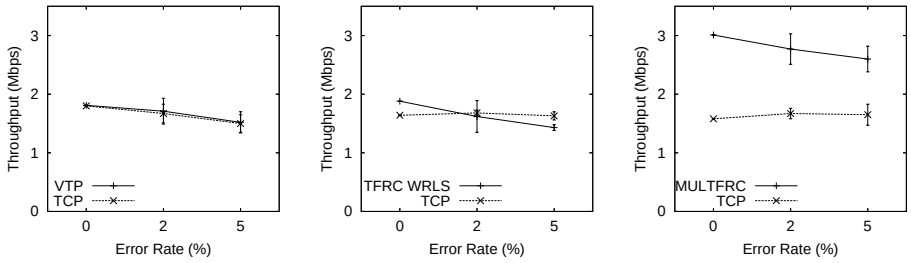


Fig. 9. TCP friendliness of VTP, TFRC Wireless and MULTFRC under different error rates, with 99% confidence intervals.

6. TESTBED IMPLEMENTATION AND MEASUREMENTS

We have implemented VTP in C++ as part of a hybrid testbed. In this section, we first describe our implementation and the concept of hybrid testbeds, then present measurement results from hybrid testbed experiments.

6.1 VTP Implementation and Hybrid Testbed

Our VTP implementation is written in Microsoft Visual C++ 6.0 and runs on Microsoft Windows. This implementation is built upon two previously available modules developed by UCLA researchers: a C++ implementation of the RTP protocol suite [Schulzrinne et al. 2003], and an H.263 video codec. Both modules conform to their respective standards.

The implementation consists of two pieces of software: a video server and a video client, running on separate computers. The server is connected to a camera that captures real-time video, in our case a Logitech QuickCam Zoom working at the 320×240 resolution. Captured video is then encoded by the H.263 codec into frames, with designated encoding parameters, and sent to the video client via RTP. When video frames arrive at the client, they are decoded and played back on the screen. Statistics such as the achieved rate, loss rate, etc. are calculated by the client and reported periodically to the server via RTCP. The server then adjusts its encoding and sending rates according to the VTP algorithms, forming a closed feedback loop.

To avoid the complex, costly, time-consuming and hard-to-repeat field tests, we have been evaluating wireless protocols with an alternative hybrid technique. Simply speaking, a hybrid testbed consists of both simulated models and real implementations. Shown in Figure 10, our VTP hybrid testbed is made up of three laptops: the video server, the video client, and an Ns-2 simulation workstation; they are interconnected through Ethernet. The video server and client are the C++ implementation of VTP as described above, while the rest of the network is simulated on the Ns-2 workstation. Packets are not directly transmitted between the server and the client, but sent to the Ns-2 workstation, where a desired network scenario is simulated in real time with one node corresponding to the physical video server

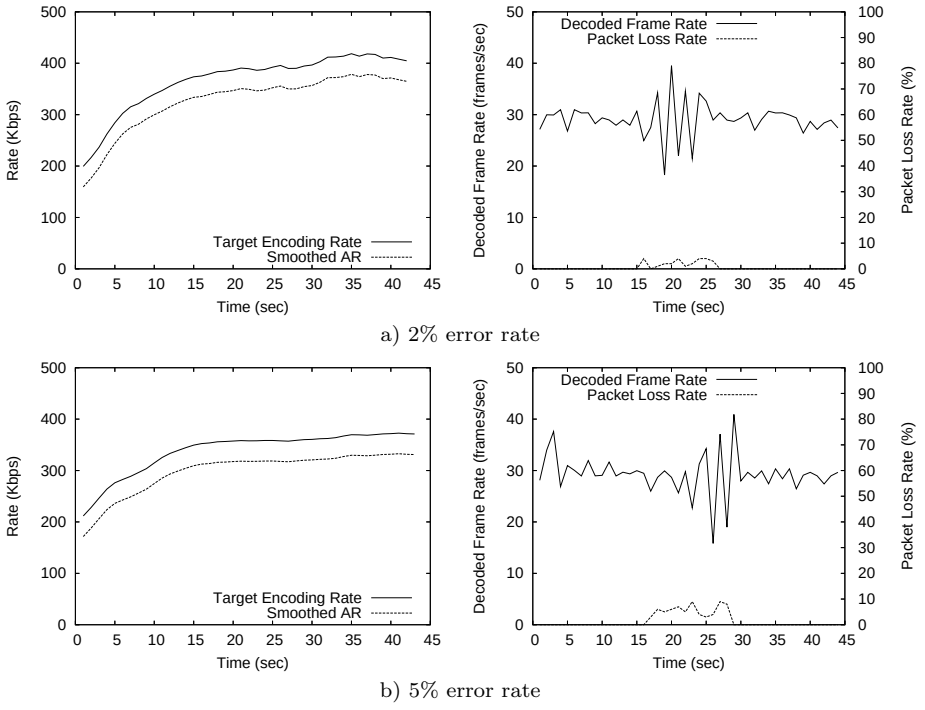
Fig. 10. Hybrid testbed consisting of three laptops: video server, Ns-2 simulation workstation, and video client (from left to right).

and another corresponding to the physical video client. In other words, these two simulated nodes are on behalf of the physical laptops in the simulation. Packets sent to the Ns-2 workstation are delayed or dropped based on simulation results, then forwarded to the intended physical destination if needed. Hybrid experiments are a quick and effective way to evaluate new protocols. Next we will show how VTP performs in our testbed.

6.2 Measurements

We reuse the WLAN topology in testbed measurements. One change is made though, to set the data rate of 802.11 to 2 Mbps instead of 11 Mbps. The main reason is that the camera and video codec currently used in the testbed jointly produce a video stream at an average bit rate between 200 and 500 Kbps, nicely fitted into the 2 Mbps nominal data rate for convenient performance assessment. Similar to the previous section, the video is streamed from a wired server, through the base station, to a wireless client. We study two cases where either random wireless loss is present or cross traffic is contending for the bandwidth.

In the first case, we impose a 2% or 5% error rate on the wireless channel roughly between the 15^{th} and 28^{th} second of an experiment. The results are shown in Figure 11. Although packet loss causes the decoded frame rate at the video client to fluctuate (the target frame rate is 30 fps), the target encoding rate and smoothed achieved rate at the video server remain unaffected. We have also observed, not shown here, that the LDA largely works correctly in this situation, showing the connection is in the error mode. Subjectively, we have seen rather smooth video on the client throughout the entire session of an experiment.



a) 2% error rate

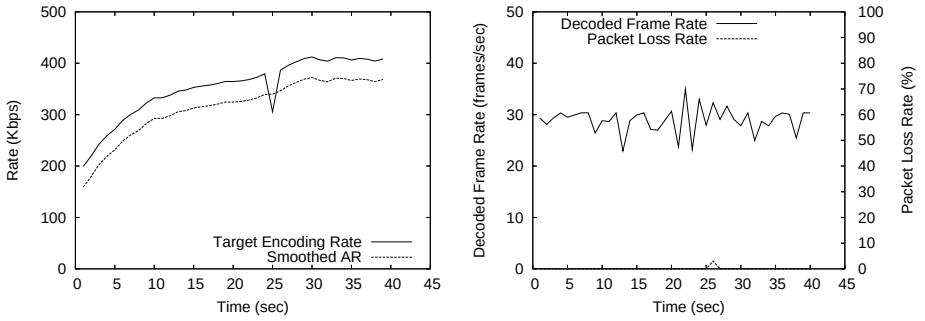
b) 5% error rate

Fig. 11. The target encoding (bit) rate and smoothed achieved rate at the video server (left), and the decoded frame rate and packet loss rate at the video client (right), under different error rates.

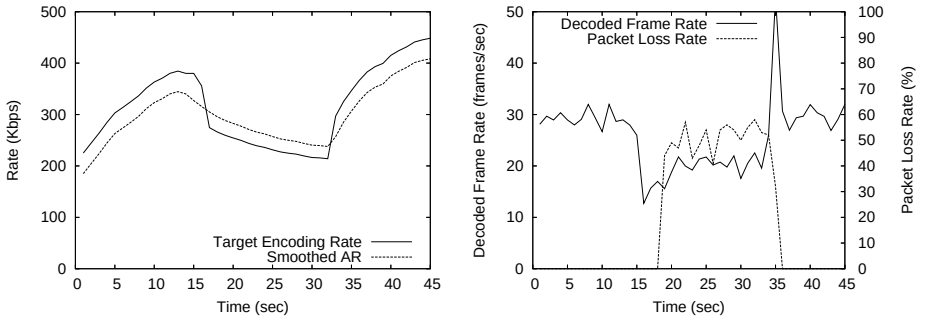
In the second case, a CBR flow at the rate of 30 or 80 Kbps is activated roughly between the 15^{th} and 32^{nd} second and goes through the shared WLAN; see Figure 12 for results. VTP is barely affected when the cross traffic is light (30 Kbps). The target encoding rate briefly dips but quickly bounces back. In the heavy cross traffic situation (80 Kbps), however, VTP consistently experiences congestion and gradually decreases its encoding rate as expected. Still, around half of the video packets are lost or late due to congestion. VTP per se is able to manage a frame rate of 15 - 20 fps; we have seen degraded playback quality with glitches on the client. We deem it as what an end-to-end rate control mechanism can do in this tough situation. Error protection techniques can be applied in parallel with VTP to improve the video quality; study of such interaction is beyond the scope of this paper and left for future investigation.

7. CONCLUSION

In this paper we have studied the Video Transport Protocol (VTP) for real-time video over the wired-cum-wireless network scenario. VTP has a unique rate control mechanism that measures the end-to-end Achieved Rate (AR), and adjusts its sending rate based on congestion conditions detected by a Loss Discrimination Algorithm (LDA). Rate decreases and increases are carefully designed so as to mimic the AIMD algorithm in TCP but with smoother rate adaptation. VTP has



a) 30 Kbps cross traffic



b) 80 Kbps cross traffic

Fig. 12. The target encoding (bit) rate and smoothed achieved rate at the video server (left), and the decoded frame rate and packet loss rate at the video client (right), with different cross traffic. Note that the spike in the decoded frame rate in the lower right graph is due to the relatively coarse measurement resolution.

demonstrated good intra-protocol fairness and opportunistic friendliness to legacy protocols. It can be either implemented as a stand-alone transport protocol, or integrated into existing protocols, such as the Real-time Transport Protocol (RTP), Datagram Congestion Control Protocol (DCCP) [Kohler et al. 2004] and Stream Control Transmission Protocol (SCTP) [Stewart et al. 2000], as a congestion/rate control option.

To our knowledge, VTP is one of the few truly end-to-end schemes that perform well in the wireless environment without requiring the support from lower layer feedback and AQM mechanisms, among which also exist TFRC Wireless and MULTFRC, two recent extensions to the well known TFRC protocol. We have compared the performance of all three protocols via Ns-2 simulations. MULTFRC creates multiple simultaneous TFRC connections to better utilize the bandwidth when a single connection is inefficient. However, frequent increases/decreases in the number of connections trigger a fluctuating behavior, and its convergence speed to the "fair share" is slow. We believe that MULTFRC is more appropriate for networks with low-bandwidth wireless links such as 1xRTT, and/or with very bursty errors. TFRC Wireless and VTP are similar in that they are both equipped with a delay-based LDA and target the TCP throughput in zero error situations. Unfortunately, the LDA may misclassify random loss as congestion-induced due to

wireless PHY/MAC characteristics. In this case, the advantage of AR estimation helps VTP achieve a better performance than TFRC Wireless.

VTP has been designed to provide smooth, efficient and friendly video transport in presence of wireless errors. So far we have been focusing on WLAN scenarios, where wireless devices directly connect to a base station. In other scenarios, devices may connect to a base station via multiple hops of wireless links, e.g. in an mobile ad hoc or mesh network. Due to multi-hopping, node mobility, issues of hidden terminals, etc., these scenarios are more challenging that those that VTP is currently targeting. We have not discussed this in the paper, but are very interested to see in our future work how end-to-end rate adaptation and loss discrimination can be tweaked to work in such multi-hop wireless scenarios.

A popular method for video transmission is to allow multicasting from one server to multiple clients, and/or to enable one client to receive streams from multiple servers. These one-to-many and many-to-one setups can save/balance bandwidth consumption and/or improve user-perceived video quality. As VTP is an end-to-end unicast protocol, its direct applicability in such scenarios is limited. However, the concepts of rate estimation and loss discrimination can be generalized to accommodate one-to-many and many-to-one cases, like TFRC has been extended to the TCP-Friendly Multicast Congestion Control (TFMCC) [Widmer and Handley 2003]. We plan to investigate this in our future work.

REFERENCES

Allman, M., Paxson, V., and Stevens, W. 1999. Tcp congestion control. *RFC 2581*.

Bansal, D. and Balakrishnan, H. 2001. Binomial congestion control algorithms. In *Proceedings of the IEEE Infocom*. IEEE, Anchorage, AK.

Biaz, S. and Vaidya, N. 1999. Discriminating congestion losses from wireless losses using inter-arrival times at the receiver. In *Proceedings of the IEEE Symp. Application-Specific Systems and Software Engineering and Technology*. IEEE, Richardson, TX.

Cen, S., Cosman, P., and Voelker, G. 2003. End-to-end differentiation of congestion and wireless losses. *IEEE/ACM Transactions on Networking 11,* 5 (Oct.), 703–717.

Cen, S., Pu, C., and Walpole, J. 1998. Flow and congestion control for internet media streaming applications. In *Proceedings of the SPIE MMCN*. SPIE, San Jose, CA.

Chen, M. and Zakhor, A. 2004. Rate control for streaming video over wireless. In *Proceedings of the IEEE Infocom*. IEEE, Hong Kong, China.

Chiu, D.-M. and Jain, R. 1989. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems 17,* 1 (June), 1–14.

Feamster, N., Bansal, D., and Balakrishnan, H. 2001. On the interactions between layered quality adaptation and congestion control for streaming video. In *Proceedings of the International Packet Video Workshop*. Kyongju, Korea.

Floyd, S. 2000. Congestion control principles. *RFC 2914*.

Gerla, M., Ng, B. K. F., Sanadidi, M. Y., Valla, M., and Wang, R. 2004. Tcp westwood with adaptive bandwidth estimation to improve efficiency/friendliness tradeoffs. *Computer Communications 27,* 1 (Jan.), 41–58.

Handley, M., Floyd, S., Padhye, J., and Widmer, J. 2003. Tcp friendly rate control (tfrc): Protocol specification. *RFC 3448*.

Hsu, C. Y., Ortega, A., and Khansari, M. 1999. Rate control for robust video transmission over burst-error wireless channels. *IEEE Journal on Selected Areas in Communications 17,* 5 (May), 756–773.

Kazantzidis, M. 2002. Adaptive wireless multimedia. Ph.D. thesis, University of California, Los Angeles, Los Angeles, CA.

Kohler, E., Handley, M., and Floyd, S. 2004. Datagram congestion control protocol (dccp). *Internet Draft*.

Kurose, J. F. and Ross, K. W. 2004. *Computer Networking: A Top-Down Approach Featuring the Internet (3rd ed)*. Addison Wesley.

Loguinov, D. and Radha, H. 2003. End-to-end rate-based congestion control: Convergence properties and scalability analysis. *IEEE/ACM Transactions on Networking 11,* 4 (Aug.), 564–577.

Mehra, P. and Zakhor, A. 2003. Tcp-based video streaming using receiver-driven bandwidth sharing. In *Proceedings of the International Packet Video Workshop*. Nantes, France.

Ns2. The network simulator. *http://www.isi.edu/nsnam/ns/*.

Rejaie, R., Handley, M., and Estrin, D. 1999a. Quality adaptation for congestion controlled video playback over the internet. In *Proceedings of the ACM SIGCOMM*. ACM, Cambridge, MA.

Rejaie, R., Handley, M., and Estrin, D. 1999b. Rap: An end-to-end rate-based congestion control mechanism for realtime streams in the internet. In *Proceedings of the IEEE Infocom*. IEEE, New York, NY.

Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V. 2003. Rtp: A transport protocol for real-time applications. *RFC 3550*.

Stewart, R., Xie, Q., Morneault, K., Sharp, C., Schwarzbauer, H., Taylor, T., Rytina, I., Kalla, M., Zhang, L., and Paxson, V. 2000. Stream control transmission protocol. *RFC 2960*.

Sun, M. and Reibman, A. 2001. *Compressed Video over Networks*. Marcel Dekker, Inc., New York, NY.

Tang, J., Morabito, G., Akyildiz, I., and Johnson, M. 2001. Rcs: A rate control scheme for real-time traffic in networks with high bandwidth-delay products and high bit error rates. In *Proceedings of the IEEE Infocom*. IEEE, Anchorage, AK.

TCPW. Tcp westwood home page. *http://www.cs.ucla.edu/NRL/hpi/tcpw/*.

Tobe, Y., Tamura, Y., Molano, A., Ghosh, S., and Tokuda, H. 2000. Achieving moderate fairness for udp flows by path-status classification. In *Proceedings of the IEEE LCN*. IEEE, Tampa, FL.

Widmer, J. and Handley, M. 2003. Tcp-friendly multicast congestion control (tfmcc). *Internet-Draft: draft-ietf-rmt-bb-tfmcc-02*.

Yang, F., Zhang, Q., Zhu, W., and Zhang, Y. 2004. End-to-end tcp-friendly streaming protocol and bit allocation for scalable video over wireless internet. *IEEE Journal on Selected Areas in Communications 22,* 4 (May), 777–790.

Yang, G., Chen, L., Sun, T., Gerla, M., and Sanadidi, M. Y. 2005. Real-time streaming over wireless links: A comparative study. In *Proceedings of the IEEE ISCC*. IEEE, Cartagena, Spain.

Yang, G., Gerla, M., and Sanadidi, M. Y. 2004. Adaptive video streaming in presence of wireless errors. In *Proceedings of the IPIF/IEEE MMNS*. Springer-Verlag, San Diego, CA.

Yang, Y., Kim, M., and Lam, S. 2001. Transient behaviors of tcp-friendly congestion control protocols. In *Proceedings of the IEEE Infocom*. IEEE, Anchorage, AK.

Zhang, Y. and Loguinov, D. 2004. Oscillations and buffer overflows in video streaming under non-negligible queueing delay. In *Proceedings of the ACM NOSSDAV*. ACM, Cork, Ireland.