Computer Communications 34 (2011) 1836–1847

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

TCP-Friendly Rate Control (TFRC) for bursty media flows

Arjuna Sathiaseelan *, Gorry Fairhurst

School of Engineering, University of Aberdeen, Aberdeen, AB24 3UE, UK

ARTICLE INFO

Article history:
Received 4 June 2010
Received in revised form 2 May 2011
Accepted 3 May 2011
Available online 11 May 2011

Keywords:
TFRC
Congestion control
Multimedia
Variable rate traffic

ABSTRACT

TCP-Friendly Rate Control (TFRC) was originally designed for multimedia streaming applications where continuous data was available at the sender. However, TFRC is not well-suited to the variable rate traffic presented by many modern adaptive media codecs. One way to counter this deficiency would be for the sender to continue to transmit at the media rate during periods of silence, known as padding. This use of padding can ensure acceptable application performance. However, it also degrades network performance, and decreases the usefulness of TFRC congestion control. Recent standardisation has resulted in a new revised TFRC specification. This paper describes candidate methods that were evaluated as a part of this revision and presents the first analysis of the new TFRC specification including a comparison this with the proposed Faster Restart method. It evaluates behaviour both in terms of the application performance benefit and the implications on other network traffic that share an Internet bottleneck and shows that the new methods improve the performance of bursty media. Although Faster Restart allowed TFRC to better support bursty applications, the additional gain was determined to be small when combined with the revised TFRC specification. Finally, revised TFRC is shown to remove the former incentive for padding, substantially improving the performance of other network traffic sharing a congested network.

1. Introduction

Internet multimedia has seen unprecedented growth in the past few years. In 2010 alone, Internet multimedia accounted for 30% of overall Internet traffic [1]. This growth is expected to accelerate in the coming years. Most interactive media applications currently use the User Datagram Protocol (UDP) [2] transport protocol. This offers a connectionless service with non-guaranteed datagram delivery. Many UDP-based multimedia applications can (and often do) transmit at a constant rate, irrespective of the available network capacity. Uncontrolled multimedia traffic is becoming increasingly problematic to network operators, who are driven to rely on deep packet inspection and other heuristics to classify and control Internet traffic. A growth in long-lived non-congestion-controlled traffic may be seen to pose a real threat to the overall health of the Internet [3], and, in particular, forms an obstacle to the deployment of triple-play services (voice, video and data) over bandwidth-constrained technologies such as mobile Internet, wide-area wireless, and broadband satellite systems.

Many current applications implement a form of congestion control, but typically do not use a standards-based method. For example, Adobe's Flash Player uses the Real Time Media Flow Protocol (RTMFP) with a proprietary form of congestion control [4]. One disadvantage of not using standards-based methods is that it is difficult to analyse whether new mechanisms effectively share the Internet with other congestion-responsive flows. This can make it hard to judge their expected impact on other traffic sharing a bottleneck along the path. It is therefore in the interest of the Internet community to update standards specifications so that they match the needs of prevailing applications and to facilitate research, evaluation and standardization of new methods for congestion control.

Congestion control for datagram transport was defined by the TCP-Friendly Rate Control (TFRC) algorithm first standardized by the Internet Engineering Task Force (IETF) in 2003, published as RFC 3448 [5]. TFRC defined a rate-paced protocol that requires both the sender and receiver to participate in determining an allowed sending rate. The receiver periodically sends a feedback report, indicating the recent loss event rate experienced by the session. Absence of a feedback report indicates an idle period where an application has stopped sending data, or data/feedback packets dropped in the network [6]. The sender uses the feedback reports to calculate the allowed sending rate for the next round trip time (RTT) period using an equation that models the equivalent throughput that would have been obtained by a bulk TCP flow. TFRC was designed as a congestion control method for multimedia applications such as streaming video and audio, since it allows a smooth variation in sending rate by gradually decreasing and increasing the sending rate. Various publications [7–10] have analysed the performance of classical streaming applications using RFC 3448, and concluded that TFRC is safe for use with these applications in the general Internet. TFRC can also be used for multicast traffic [11], although this is not the focus of this paper.

* Corresponding author. Tel.: +44 7971329880.
E-mail addresses: arjuna@erg.abdn.ac.uk (A. Sathiaseelan), gorry@erg.abdn.ac.uk (G. Fairhurst).

0140-3664/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2011.05.003

The TCP throughput equation [6] uses the packet size for calculating a sending rate measured in packets over a RTT. TFRC was intended for applications that send a fixed packet size, thus it was designed to be reasonably fair when competing for capacity with TCP connections that also used the same packet size. This has consequences: a low bandwidth TFRC flow using small-packets that shares a network bottleneck with a high-bandwidth TCP flow using large-packets may be forced to slow down, even though the nominal rate in bytes per second (Bps) of the TFRC flow is less than the rate achieved by the TCP flows.

To allow TFRC to also offer acceptable performance for small-packet flows, a Small-Packet (SP) variant of TFRC, TFRC-SP [12], has been defined. This was designed to acquire the same capacity in Bps as a TCP flow that uses packets of up to 1500 bytes. To control misuse at high rates, TFRC-SP also enforces a minimum interval of 10 ms between packets. The core congestion control algorithm is similar to TFRC. This prevents a single flow from sending small packets arbitrarily frequently. TFRC-SP was published as an experimental specification, and as such the IETF therefore invites experimentation and simulation to examine its performance across a wide range of Internet topologies and styles of application behavior. The paper is intended to contribute a response to this invitation.

The analysis is also appropriate to Datagram Congestion Control Protocol Congestion (DCCP) Control Identifiers, CCID-3 [13] and CCID-4 [14]. Implications for these DCCP CCIDs are reviewed in the discussion section of this paper.

The original specification of TFRC (in RFC 3448) [5] suited many traditional multimedia streaming applications where continuous data is available at the sender. It does not suit the traffic generated by many modern applications characterised by periods of higher (but limited) transmission rate, separated by periods in which much less (which we call 'data-limited'), or no data (which we call 'idle periods') is sent. Examples of such bursty applications include variable-rate video or modern silence-suppressed Voice over IP (VoIP) codecs [15,16], where the media sent by the application is not a continuous flow. Similarly, video encoders employing motion compensation may result in varying media rates. Applications that switch video or audio content from a number of input streams, can also exhibit significant burstiness. It is therefore important to understand the behaviour of TFRC when supporting bursty applications.

The paper seeks to capture the story of why RFC 3448 was not suited to bursty multimedia traffic. It evaluates an enhancement to improve RFC 3448, known as "Faster Restart" (FR) [17], a mechanism that accelerates slow start after idle and long data-limited periods. FR was intended as a contribution to the DCCP working group (WG) to drive work to evaluate new alternate methods, and as such contributed to the process that finally updated the algorithms in RFC 3448, leading to RFC 5348 [6]. This paper is also the first paper to analyse bursty applications and FR with TFRC, and the first to report on the performance of RFC 5348.

The paper is structured as follows: Section 2 discusses the behavior of RFC 3448 for carrying bursty media flows and the need for new methods. Section 3 describes the revised TFRC specification, RFC5348, and compares this with a new method, Faster Restart, FR, intended to address this performance shortfall. Section 4 analyses the benefit offered by these algorithms and also discusses whether FR and RFC 5348 are safe to be deployed in the general Internet. Section 5 provides a general discussion followed by the conclusions.

2. Challenges presented by bursty media over TFRC

This section explores the implications arising from the design of TFRC, as specified in RFC 3448. Fig. 1 shows the growth of the allowed sending rate [5] for a bursty application that uses TFRC. The figure assumes content is encoded at a specified media rate. In active periods, data is generated at the media rate of the encoder plus associated overhead, interspersed by short idle times when the encoder is inactive. Each encoded media packet is placed in a transmit buffer, with the rate at which packets are sent to the network controlled by TFRC. The sending rate to the network may therefore be characterised by a series of phases:

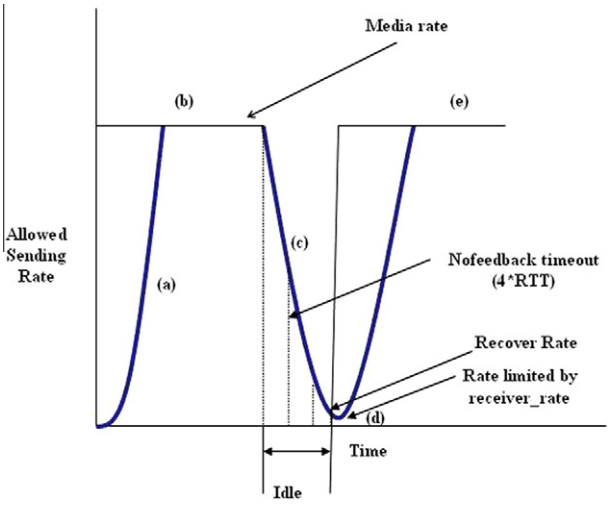

Fig. 1. Mismatch between the media rate and TFRC sending rate.

(a) Initial slow-start: TFRC starts with a maximum of 1 packet per RTT [5]. The initial rate (for the first few RTTs) will usually be less than the media rate of a multimedia application. For a small RTT, this start-up delay will be acceptable; however for high latency paths (e.g. greater than 200 ms) this can severely degrade the perceived performance [15].

(b) A period of transmission at the media rate: Once the TFRC allowed sending rate reaches the media rate and if outstanding packets were backlogged in the transmit buffer (for example following the initial slow-start phase), there may be a transient overshoot, where the sending rate rises above the media rate. If there are no outstanding packets in the transmit buffer, TFRC sends at a rate determined by the media rate.

(c) During an idle period: When the application stops sending data for a period of 4 RTTs (nofeedback timeout) or more, TFRC reduces the allowed sending rate by one half every no-feedback timer expiry (4 RTTs) [5,6]. This rate can be reduced to a minimum of 2 packets per RTT. We term this as the recover_rate, as suggested in [6].

(d) Rate limited by receiver rate: When the application restarts, TFRC must slow-start back from the recover_rate to the media rate. The allowed sending rate can increase to at most twice the current receiver rate (rate received over the last RTT)) [6]. This will take a significant time when operating over a high latency path.

When an application becomes idle or does not send at the allowed rate (known as "data-limited" or "application-limited" [6,18]), a TFRC receiver calculates the receiver rate by including time periods where data was not sent. This results in a low receiver rate. For several RTTs the low receiver rate will limit the allowed sending rate to less than that required to maintain the media rate [15,16]. A TFRC receiver would also calculate a low receiver rate for its first feedback packet [6].

(e) Return to media rate: Once the allowed sending rate reaches the media rate, and the backlog in the transmit buffer is cleared, the sender rate is again controlled by the media rate.

A sender that transmits bursts of data followed by idle periods therefore oscillates between periods in which the allowed sending rate is sufficient to satisfy the media rate, and periods where that rate is insufficient. This oscillation is a result of TFRC's conservative congestion control and it is not triggered by an indication of observed loss/congestion (such as a loss report or loss of a feedback message). Such oscillations of sending rate would affect delay-sensitive interactive flows, such as VoIP. One common way to mitigate this poor performance is for a sender to continue to send at the media rate during periods of silence. This is known as "padding" [6].

Although padding a VoIP flow can improve the performance (as shown in Section 4), it is undesirable for the network, since this consumes more network capacity than necessary to support the application, and hence reduces the capacity available for other flows that share a common network bottleneck [19]. However, the poor application performance of the original TFRC specification has made this attractive for application designers [17]. Hence mechanisms were needed to mitigate the problems imposed by RFC 3448 and at the same time provide an incentive to the application designers to remove padding.

3. Techniques to improve TFRC

A TFRC flow that fairly achieves a sending rate (i.e. with no reported loss) has proved that the path between the hosts can at least support this rate [17]. This rate may be safely sustained while the sender continues to receive feedback, but may need to be reduced if path characteristics change, for example due to a change of route, mobility handover, or the level of traffic increases (e.g. additional flows start).

This section examines two methods that can allow a previously achieved sending rate to be as a basis for sending after an idle period. Although Balakrishnan et al. argued that network conditions are often relatively stable for periods of several minutes [20], this is not always the case, and we assert that it is important that any standardized method is robust to such changes. To ensure TFRC is safe for deployment in the Internet, requires examination of the impact of a sudden change in the path. Any path change experienced while a sender was idle could potentially result in a significant reduction in the sender's fair rate (i.e. following a reduction in path capacity while the sender was idle). It is therefore important to assess whether it is reasonable to allow an application to send faster after idle, which can improve application responsiveness, but can also contribute to transient congestion in times of change [17].

3.1. Faster Restart (FR)

This section first considers Faster Restart (FR), a change to TFRC that was proposed in [17]. FR accelerates slow start after idle and long data-limited periods. It does not modify the response at the beginning of a connection, since at this time the properties of the network path are usually unknown. Following an idle period or a long data-limited period, FR allows a flow to quadruple its sending rate in every congestion-free RTT, instead of doubling towards the previously achieved rate. It sets the recover_rate to 8 packets per RTT and does not reduce the allowed sending rate to less than the recover_rate in the absence of congestion (i.e. reported loss). When a FR sender detects congestion, the allowed sending rate is reduced appropriately to reflect the congestion.

3.2. Revised TFRC specification (RFC 5348)

The IETF recently revisited the TFRC protocol specification defined in RFC 3448 resulting in a new standard for TFRC (RFC 5348) [6], which therefore obsoleted RFC 3448 [5]. RFC 5348 is intended to provide better support to media flows with idle and data-limited periods. During the initial slow-start phase, the initial rate of the sender was increased to 4 packets per RTT (depending on the Maximum Segment Size, MSS, specified in [21]). The behaviour after an idle period was also updated, so that in the absence of loss, the sending rate is not reduced below 4 packets per RTT, equal to the initial rate. The key differences between RFC 4338 and RFC 5348 are highlighted in Section 9.1 of the revised specification. RFC 5348 also updates the TFRC-SP specification [12], DCCP CCID-3 [13] and CCID-4 [14].

RFC 5348 sought to address the performance issues identified in Section 2 by maintaining a set of recent receiver rate reports (over two RTTs) and using the maximum value from this set when calculating the allowed sending rate. This ensures that a low receiver rate is reported after an idle or data-limited period, the low receiver report is discounted when calculating the current sending rate, unless the feedback packet reported a new loss event or an increase in the loss event rate. In the latter case, the values in the receiver rate set are reduced by one half to reflect the loss, ensuring an appropriate response to congestion.

During an idle period (a period greater than or equal to the no-feedback timeout) an RFC 5348 sender still reduces the allowed sending rate by one half after each no-feedback timer interval (at least four RTTs). This sending rate is however, not reduced below the allowed initial sending rate during idle periods (the recover_rate [6]). When the sender emerges from the idle period, it effectively has to slow-start to the previous sending rate. The mismatch between the TFRC sending rate and the application media rate may still degrade performance of the application, if it had strict requirements for timeliness, for example interactive video and audio. This paper evaluates the performance of interactive video and audio when using RFC 5348 to determine whether the revised TFRC, RFC 5348, is able to mitigate the key problems posed by RFC 3448, or whether a new mechanism, such as FR, is better suited to improve the performance of this class of application.

3.3. Comparison of methods

To understand the dynamics of these methods, a 64 Kbps VoIP flow was simulated using the ns-2 [22] network simulator for a path with 250 ms one way delay. The application modeled a G.711 VoIP codec with 160 byte packets and the capacity was 2 Mbps, hence there was no congestion. The application became idle for a period starting at 10 s with a duration of 10 s. This duration was chosen to illustrate the effect of the sending rate reducing and restarting from the recover_rate. Fig. 2 presents a comparison of the allowed sending rates of RFC 3448, RFC 5348 and FR (since the flow was in slow-start during the entire period of the simulation, the allowed sending rate was always twice the actual sending rate). RFC 3448 (with and without FR) performed similarly during the initial slow-start phase. However, the sending rate grew more slowly compared to RFC 5348 (with and without FR), due to the low receiver rate problem identified in Section 2. After the idle period, the mechanisms started with a different recover_rate. It took approximately 4.5 s for RFC 3448 to reach the target media rate.

Although FR with RFC 3448 starts with a rate similar to FR with RFC 5348, the low receiver rate problem affected the growth of the sending rate, whereas FR with RFC 3448 took 3.5 s to reach the target media rate. RFC 5348 solved the low receiver rate problem and was able to start at a recover_rate of 4 packets/RTT, and hence was faster in achieving the target media rate, in 2.5 s. FR with RFC 5348

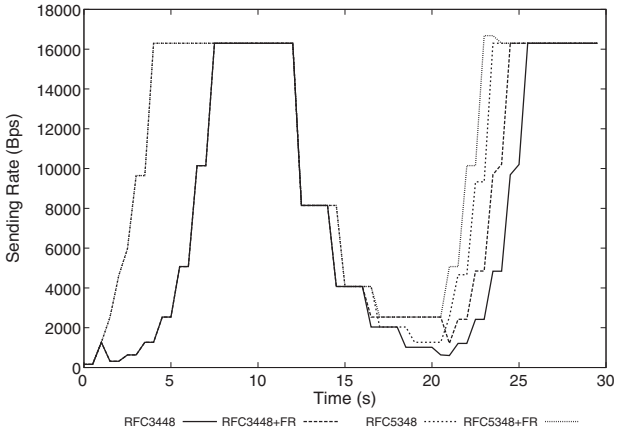

Fig. 2. Sending rate dynamics of RFC 3448, RFC 5348 and FR.

Table 1
Features of RFC 3448, RFC 5348 and faster restart.

Features	RFC 3448	RFC 5348	Faster restart
Initial Slow-start rate	1 packet/RTT	4 packets/RTT	4 packets/RTT
Idle period recover rate	2 packets/RTT	4 packets/RTT	8 packets/RTT (small packets) 4 packets/RTT (large packets)
After idle period behaviour	Double sending rate	Double sending rate	Quadruple sending rate
Data-limited period behaviour	Rate limited by receiver rate	Rate not limited by receiver rate	Rate limited by receiver rate (if used with RFC 3448) Rate not limited by receiver rate (if used with RFC 5348)

was able to start with a higher recover_rate of 8 packets/RTT and hence was able to grow its sending rate faster than RFC 5348 achieving the media rate in 2 s.

Table 1 summarises the important characteristics of the original TFRC specification (RFC 3448), the revised version (RFC 5348) and FR.

4. Evaluation of new methods

This section evaluates the benefits provided by FR and the new methods in RFC 5348. The analysis examines both the impact on the application and the performance of other flows sharing a common network bottleneck.

Performance was explored using a classic dumb-bell topology with two routers connected via a bottleneck link. Each router was connected to one or more access nodes. A range of bottleneck capacities and delays were considered with a droptail queue set to the bandwidth-delay product. The background traffic on the forward path comprised a number of web flows simulated by the ns-2 web traffic generator. 1 session comprised 30 web pages, each with 10 objects, using HTTP/1.0-like transactions with one web object per TCP connection. The web objects were distributed according to a Pareto II distribution with an average parameter of 400 packets and shape parameter of 1.002 [23]). The reverse path carried FTP traffic with a maximum window of 6 segments. The flow of small control packets on reverse-path served to break-up phase effects [23].

These simulations considered two types of media:

- Low rate media flows: These flows represented VoIP traffic with an encoding rate of 64 Kbps (packet size, 160 bytes). TFRC-SP was used for transport congestion control. Although the TFRC-SP standard is based on the RFC 5348, for completeness TFRC-SP was also simulated based on RFC 3448. FR was evaluated in both cases.
- High rate media flows: These flows used a rate appropriate to medium quality video over IP with an encoding rate of 512 Kbps (packet size of 1000 bytes). Applications were modelled using RFC 3448 and RFC 5348. FR was evaluated in both cases.

The performance of the media flows was also compared with an Additive Increase Multiplicative Decrease (AIMD) protocol such as TCP. Since interactive media such as VoIP flows do not use TCP [3], due to TCP's reliability constraints, these simulations used DCCP CCID-2 [24]. The congestion control mechanism of DCCP CCID-2 is similar to that of TCP SACK, but DCCP does not retransmit lost packets [24] and hence is better suited to multimedia. Tests were also repeated with a non-congestion-responsive UDP transport.

4.1. Performance analysis

4.1.1. VoIP Performance

This section analyzes the performance of low rate media flows using a simulated bottleneck capacity of 6 Mbps and a range of path delays (symmetric delay was applied to the feedback path) and packet drop rates. The packet drop rates were random, based on a uniform distribution.

A G.711 VoIP codec (64 Kbps media rate) was modelled with silence suppression and an ON/OFF model for conversational data, with periods of voice activity of variable length separated by periods of silence with an exponentially distributed mean burst parameter of 1 s and silence parameter of 1.5 s [25]. The media packet size was 160 bytes.

A TFRC-SP sender started sending packets, following setup of each call. However, initially the sender refrained from sending packets with encoded media data, since the allowed sending rate was less than the required media rate. Media packets were sent once the allowed rate reached the media rate, in this case 50 packets per second (pps). A transmit buffer of 5 packets was used (corresponding to 100 ms).

The E-Model defined in [26,27] was used to determine a Quality Factor "RScore" for each voice call. For VoIP traffic, the RScore was calculated based on sub-factors, that consider the impact of loss (I_e) and impact of delay (I_d) using the formula

$$R = 94.2 - I_e - I_d \tag{1}$$

where

$$I_e = \lambda_1 + \lambda_2 ln(1 + \lambda_3 e) \tag{2}$$

and

$$I_d = 0.024d + 0.11(d - 177.3)I(d - 177.3) \tag{3}$$

I (x) is a unity function. The simulation reported the mean RScore with a confidence interval of 2% over several runs. The RScore may be translated to determine an equivalent Mean Opinion Score (MOS), which was used to determine the quality of the VoIP flow (e.g. an RScore less than 50 is equivalent to a "poor" MOS score, 70–80 to "medium" quality, and an RScore greater than 90 is required for "best" quality).

Fig. 3 shows the performance of the small packet variant of TFRC (TFRC-SP) compared to TFRC for a path delay of 50 ms. In the presence of packet loss, TFRC-SP was designed to acquire the same capacity as a TCP flow using packets of up to 1500 bytes. Hence, TFRC-SP has a more aggressive response function than TFRC, leading to better application performance for VoIP flows in

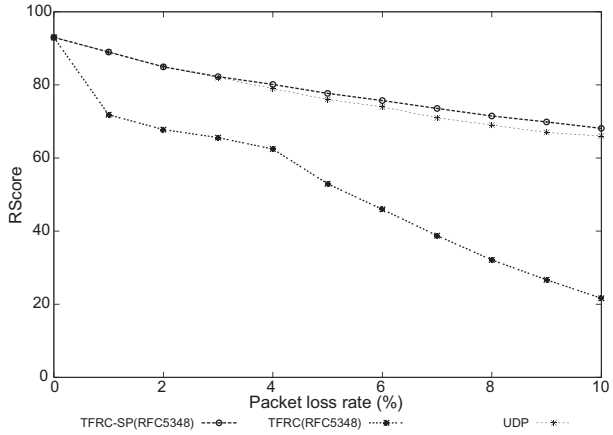

Fig. 3. VoIP quality: benefit of using TFRC-SP.

the presence of packet loss. The remainder of this paper therefore considers TFRC-SP as the congestion control method for VoIP flows.

Fig. 4 presents the quality of the VoIP flow (with a mean burst parameter of 1 s and a mean idle parameter of 1.5 s) with no congestion packet loss (Fig. 4a) and with 1% congestion packet loss (Fig. 4b), for various one-way delays.

The quality of the VoIP flow using FR with RFC 5348 was better than using only RFC 5348. When the VoIP traffic was characterised by long idle periods (greater than 4 RTTs), FR offered performance benefits (Fig. 4a and b). The quality of the VoIP flow using RFC 3448 was poor for delays greater than 50 ms. FR with RFC 3448 improved the quality of a VoIP flow compared to RFC 3448. However the low receiver rate report problem affected FR with RFC 3448. Although the quality was higher, performance remained poor for delays larger than 50 ms. VoIP flows using RFC 5348 and FR with RFC 5348 were able to achieve excellent to medium quality for respective delays of 10 ms and 250 ms, but for a delay of 300 ms, the quality was low. CCID-2 was able to achieve a good quality for delays up to 200 ms, but this quality reduced when the delay was more than 200 ms.

The R-Score model suggests that a 1% packet loss rate does not significantly impair the performance of the VoIP codec, as evidenced by the consistent performance of the UDP VoIP flow across the range of delays. With 1% packet loss, the quality of the VoIP flow using CCID-2 was poor. This is because the AIMD behaviour of CCID-2 is not suited to interactive flows, such as VoIP. Upon packet loss, CCID-2 would reduce its congestion window by half and then increase linearly. A mismatch between the media rate of the application and the congestion window of the CCID-2 sender would cause packets to be dropped at the transmit buffer. In all cases, RFC 3448 offered poor quality for a delay over 50 ms. FR with RFC 3448 improved performance, but the quality also remained poor for delays greater than 50 ms. A VoIP flow using RFC 5348 and FR with RFC 5348 achieved better quality. FR offered benefit to applications such as VoIP that currently use RFC 3448, although it was not observed to offer significant additional benefit when combined with RFC 5348.

4.1.2. Self benefit: incentive to remove padding

This section describes a set of simulations to analyse whether FR and/or RFC 5348 was able to offer acceptable performance when the application did not pad data during idle or data-limited periods. Multiple VoIP flows were simulated for a range of path delay using two scenarios:

- With padding: The VoIP flow was padded during idle or data-limited periods to produce a near constant media rate.
- Without padding: The VoIP flow was not padded, presenting bursty traffic.

Fig. 5 presents the average RScore achieved by varying the number of VoIP flows over a 2 Mbps, 150 ms bottleneck link, with a bottleneck queue size of 100 packets. The burst and idle parameters were respectively set to a mean of 1 s and 1.5 s. A path carrying many VoIP flows using padding resulted in a low average RScore, due to padding contributing to congestion and the resultant packet loss. When RFC 3448 was used to carry a bursty VoIP flow, the mismatch of the sending rate and the media rate, coupled with the impact of path delay resulted in packet drops at the transmit buffer, which degraded performance. There was no congestion at the bottleneck link, since RFC 3448 was unable to increase to a rate that would overwhelm the bottleneck router. Performance using RFC 3448 improved when padding was introduced, at least for non-congested paths.

FR with RFC 3448 improved performance, but this was still poor, since the allowed sending rate was constrained by the low receiver rate. RFC 5348 addressed the idle period and low receiver rate problem. FR with RFC 5348 achieved an excellent quality, but the gain was not appreciable compared with the benefits already offered by RFC 5348. Although CCID-2 provided good quality in the absence of congestion, when the number of VoIP flows increased, causing congestion, the performance of the VoIP flows degraded.

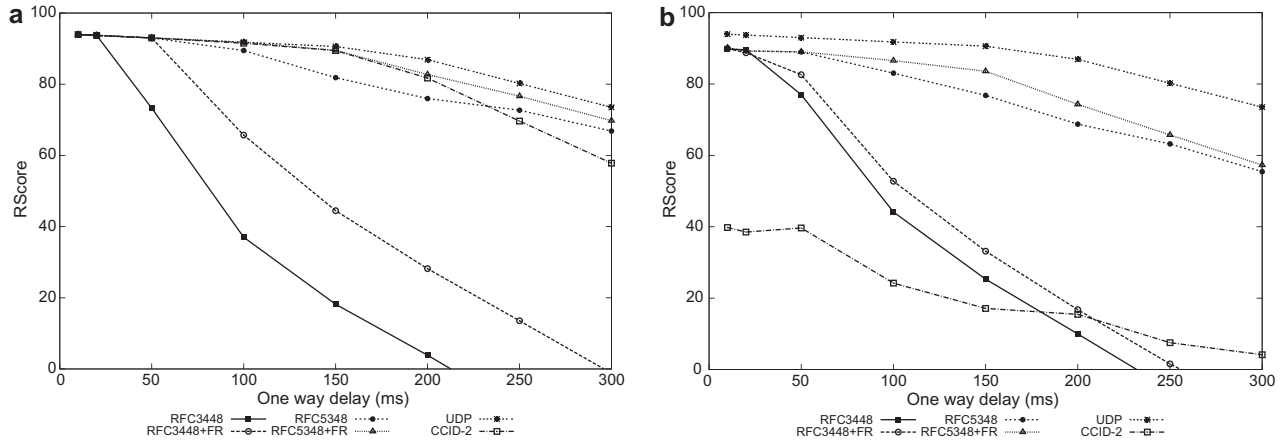

Fig. 4. (a) VoIP quality with no congestion loss. (b) VoIP quality with 1% congestion loss.

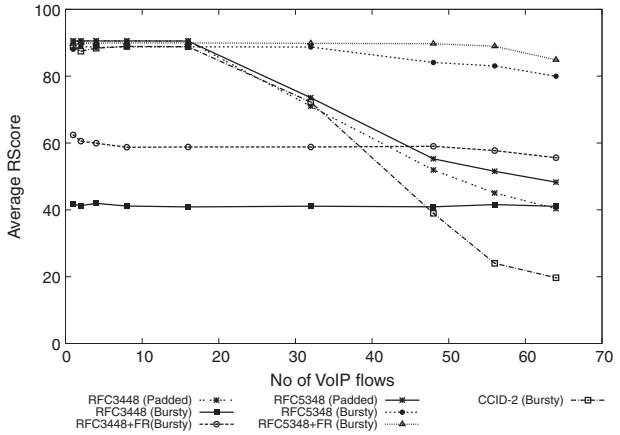

Fig. 5. Quality of multiple VoIP flows.

In summary, these results suggest the new methods provide acceptable performance when congestion control is used and therefore could remove the previous incentive for application designers to pad bursty flows during an idle or data-limited period. The next section analyses the impact of these new mechanisms on example Internet scenarios.

4.1.3. Video performance

FR offers benefit to applications such as VoIP that currently use RFC 3448, although it was not obsereved to give significant benefit when combined with RFC 5348. It is useful therefore to further evaluate how much benefit FR provided to the application. This analyzed the benefit offered by RFC 5348 and FR for video over IP using the first 30 s of a variable bit rate video trace from [28] recorded at 25 fps. This trace had a minimum rate of 8 Kbps, an average of 456 Kbps and a peak rate of 1.5 Mbps. For the first 15 s the media rate was less than the average media rate, and then increased to a peak around 20 s. A packet size of 1000 bytes and transmit buffer of 180 were used, corresponding to 400 ms buffering – a value suited to videoconferencing [29]. The bottleneck link was set to 100 Mbps. The packet discard rate at the transmit buffer was used as the metric to measure the performance of the video over IP flow with a range of one way delay (Fig. 6).

The video flows using RFC 3448 (with and without FR) were most impacted. When the application reduced the rate, RFC 3448 reduced the sending rate to reflect the current rate, since the recent reported receiver rate was used to calculate the allowed sending rate. Hence the sender was not able to rapidly grow the sending rate to match the higher media rate. This mismatch caused packets to be dropped in the transmit buffer, shown by the higher packet discard rate in Fig. 6. Although FR sought to increase the rate, the low reported receive rate using RFC 3448 resulted in minimal overall improvement.

The video flows using RFC 5348 were less impacted, since when the media rate reduced, a RFC 5348 sender considered this a data-limited period and hence discounted the last reported receiver rate by taking the maximum value from the receiver set [6]. This allowed the sender to maintain a higher allowed sending rate (in the absence of any detected congestion), ensuring a lower discard rate (Fig. 6). FR with RFC 5348 could not significantly improve performance over that of RFC 5348.

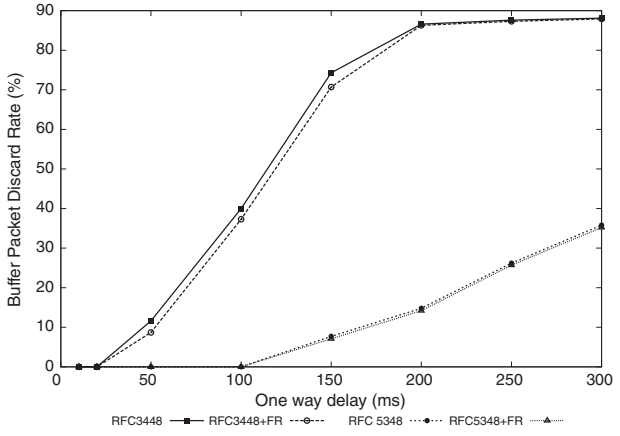

Fig. 6. Transmit buffer packet discard rate.

4.2. Impact on sharing with Internet traffic

There is often a design tradeoff between application performance *versus* network performance. When a method gives better performance for applications, it can be also be aggressive to the network. This section first evaluates whether the revised TFRC specification is TCP-friendly. It then evaluates cases where the revised TFRC and FR mechanisms are expected to be more aggressive compared to RFC 3448 and TCP. In particular, we identify the following questions to consider the safety of the new method in RFC 5348:

1. What is the effect of restarting with a large recover rate after a long idle period during transient conditions, where the network characteristics could have changed during the flow was idle?
2. What is the effect of restarting with a large sending rate after a long data-limited period during transient conditions?

4.2.1. Fairness of TFRC with TCP

This section provides an analysis considering whether TFRC is fair to a TCP flow that traverses multiple bottleneck routers. Fig. 7 presents the "parking-lot" topology used for the simulations. This comprises two bottleneck links (R1-R2) and (R2-R3), two horizontal access links (A-R1) and (B-R3) and three (vertical) access links. The bottleneck links have a rate of 2 Mbps, and the access links have a rate of 100 Mbps, with a queue size of 100 packets. All flows had a RTT of approximately 200 ms, to enable the effect of traversing multiple bottlenecks to be distinguished from that of different RTTs [30]. Source A introduced a bulk FTP flow using TCP SACK to destination B. Two equal sets of high rate media flows, each 512 Kbps using TFRC, traversed between (C-D) and (D-E). The first set of TFRC flows started with a uniformly-distributed random start time over the first 1 s, then at 5 s a TCP flow started and after 10 s a second set of TFRC flows started, also uniformly-distributed over 1 s. The TFRC flows were first simulated using RFC 3448 and

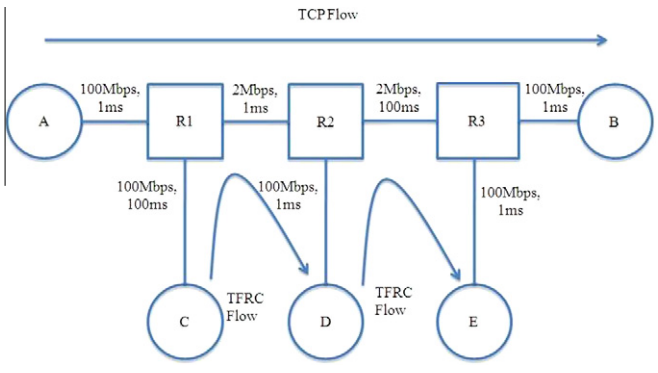

Fig. 7. Simulation toplogy.

then the tests were repeated using RFC 5348. FR was not used because idle and data-limited behaviour was not simulated.

Fig. 8 presents the expected fair share and the average throughput achieved by the TCP flow when it competed with multiple TFRC flows. The x-axis represents the total number of TFRC flows, with links (C-D) and (D-E) having equal number of TFRC flows i.e. if the total number of TFRC flows is 64, link (C-D) and (D-E) each having 32 TFRC flows. When TCP competed with TFRC (RFC 3448 or RFC 5348), the TCP flow was able to acquire more than its fair share rate (during mild congestion) and was able to get its fair share (during heavy congestion). Under heavy congestion, TCP was able to obtain more than its fair share competing with RFC 3448. This was due to the inability of RFC 3448 to grow its sending rate due to a lower initial window and low receiver rate after the first feedback packet.

TCP was able to achieve its fair share when the TFRC flows carried constant rate media flows of 512 Kbps. When there were less than 4 flows, the media rate of 512 Kbps prevented flows acquiring a fair share rate, and hence ensured they did not adversely compete with TCP. An increase in the number of flows reduced the available fair share of the bottleneck capacity, resulting in a TFRC fair share less than the 512 Kbps media rate (i.e. if a TFRC flow were to be greedy, it could send at higher than its fair share, starving any competing TCP flow). The observed results showed the TFRC flows did not acquire more than their fair share rate, allowing TCP to acquire a fair share of the bottleneck capacity. If the media flows had became idle or data-limited and restarted back to the same media rate, the impact on the TCP flow would not be more than this, since the volume of data sent by the TFRC flows during restart would be limited by the throughput equation.

The next section simulates a set of pathological conditions and examines the short-term impact on a bottleneck router when media flows restart at different rates.

4.2.2. Response to transient change of path

This section analyzes RFC 5348 and FR in a shared environment, to discover whether the new methods may be aggressive if used during transient conditions. Since throughput is not a reasonable measure for bursty multimedia, aggressiveness is analyzed in terms of packet arrival rates at the bottleneck router.

When TCP is used with a bursty application, the behaviour specified in RFC 2581 requires TCP to reduce its congestion window (cwnd) to one segment after a long idle period. This conservative behaviour is known to be unattractive to bursty applications. An experimental method was specified in RFC 2861, known as TCP Congestion Window Validation (TCP CWV). The TCP CWV proposal would allow TCP to respond differently when used to carry bursty applications [18] and reduces the cwnd by one half for every TCP Retransmission Time out (RTO) when the sender is idle. After a long idle period, this reduces cwnd to a minimum of 1 segment, (i.e. the recover window). TFRC has a similar goal and behaviour to that of TCP CWV. It reduces the sending rate during every nofeedback timeout [6], but uses a higher recover_rate than would be allowed with TCP CWV's recover window. A flow using RFC 3448 reduces the sending rate to 2 packets per RTT and slow-starts to the media rate. An RFC 5348 flow reduces the sending rate to 4 packets per RTT and then slow-starts to the media rate, whereas one using FR reduces the sending rate to 8 packets per RTT and quadruples to the media rate. The UDP flow always restarts at the full rate (determined by the media rate).

The analysis considers the effects of a transient change where the current path has much less available capacity, e.g. as a result of a route change to a more constrained path, a mobility handover to lower capacity, or a sudden increase of cross traffic.

The first scenario considered a set of flows in slow-start, prior to a large idle period, where no flows had recently experienced packet loss. During the idle period, the path capacity was reduced. Fig. 9 presents the simulation topology. The bottleneck link (R-C) was 100 Mbps, with 100 ms delay, to ensure that no packets were lost prior to becoming idle. The simulations were performed with equal numbers of media flows using the TFRC methods. TCP CWV was used as a benchmark for comparing the congestion response. Source A carried TFRC flows and Source B carried an equal number of TCP CWV flows. Each high rate media flow started at random time distributed over 1 s. After 30 s the flows stopped for 5 s and the bottleneck link (R-C) capacity reduced to 2 Mbps. After 5 s, the flows re-started at a random time distributed over 1 s.

The average arrival rate at the bottleneck was measured over 5 s. If no packets were lost prior to idle, the arrival rate of a flow would not be less than the recover_rate. The aggressiveness of a flow was assessed by considering the flow arrival rate during periods of heavy or persistent congestion, when packets were lost. The maximum sending rate for a TCP connection is T Bps [31], given by:

$$T \leqslant \frac{1.5 * \sqrt{2/3} * B}{R * \sqrt{p}} \tag{4}$$

where B is the size of the packet, R is the minimum RTT, and p is the average packet loss rate for the measured time interval.

If the arrival rate of a competing flow did not exceed T (given by Eq. (4)), then the flow was considered to be TCP-friendly [31]. The average arrival rate and the maximum allowed sending rate T (plotted as vertical bars) were compared (Fig. 10). The expected congestion behaviour of a flow may vary depending on other network traffic, path dynamics and the application behaviour. A flow

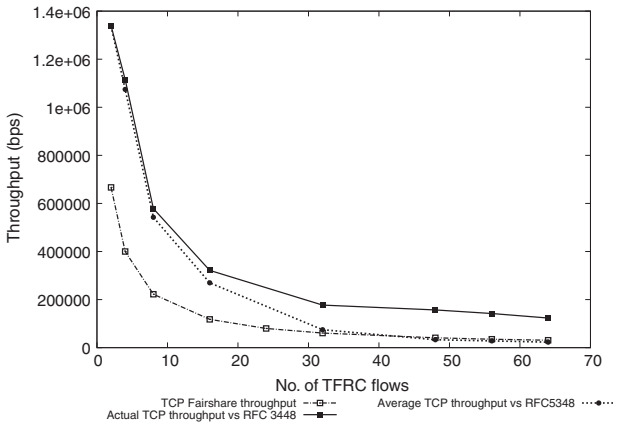

Fig. 8. Throughput of single TCP flow.

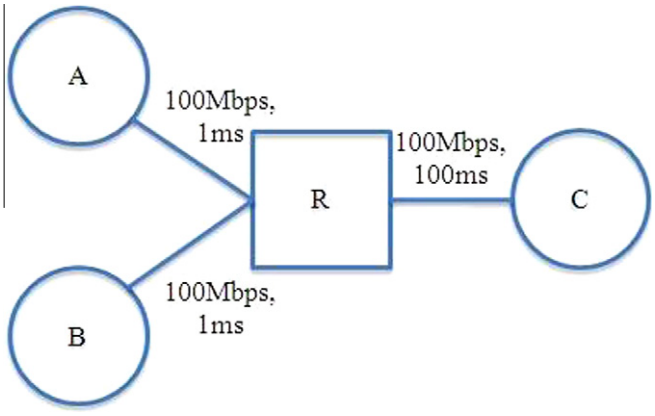

Fig. 9. Simulation toplogy.

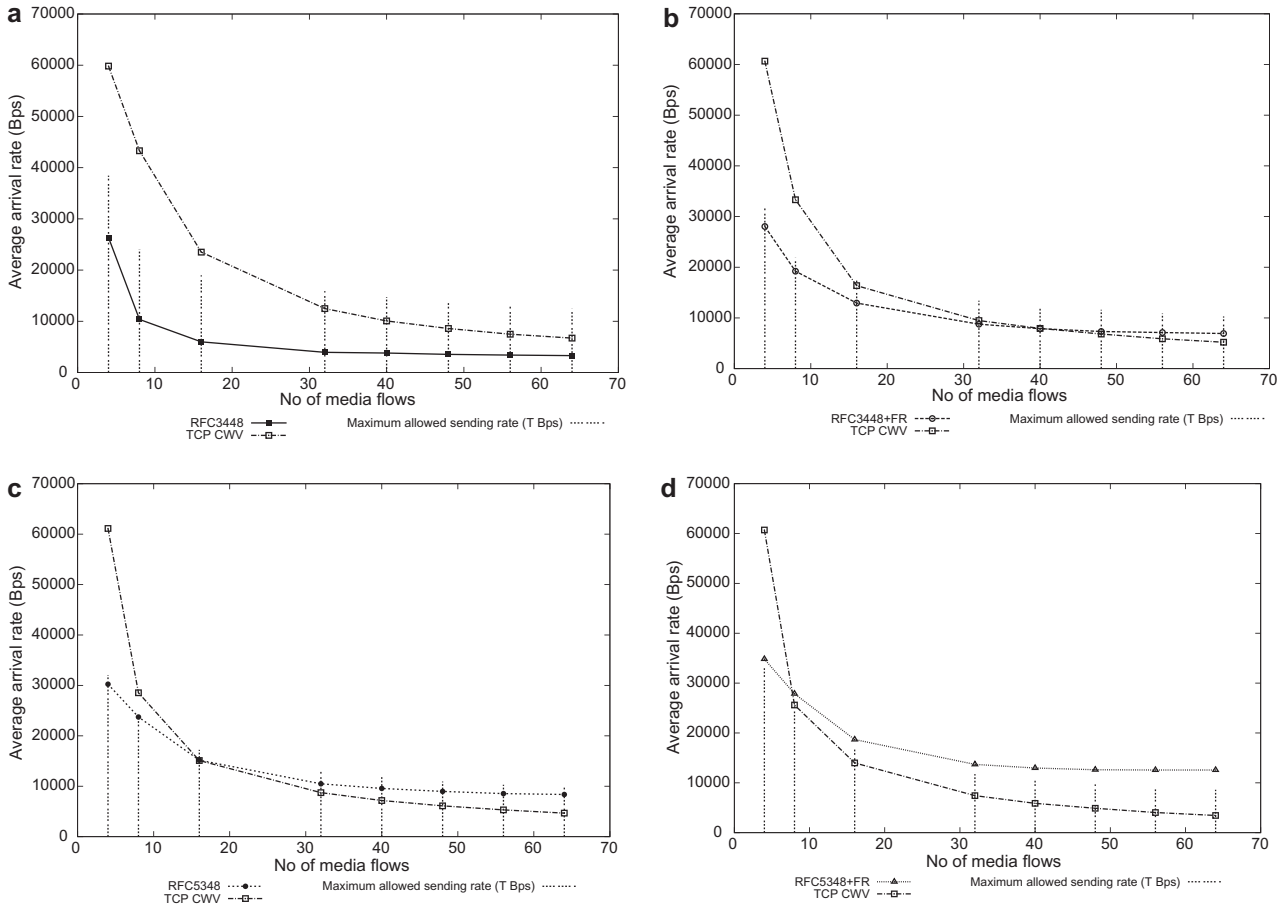

Fig. 10. (a) Average arrival rates for RFC 3448. (b) Average arrival rates for RFC 3448 + FR. (c) Average arrival rates for RFC 5348. (d) Average arrival rates for RFC 5348 + FR.

that consumes more than twice T over a period of several seconds was considered to be significantly more aggressive than TCP [32]. That is, although a flow with a rate between T and 2T achieved more than its fair share, the flow would not be judged to be significantly aggressive. Fig. 11a presents the aggregate packet loss rates at the bottleneck router over the 5 s period and Fig. 11b presents an example of the packet loss rate evolution over time for a single simulation with 64 media flows. The packet loss rate graph shows that in the first few seconds when the flows restarted with a larger recover rate, RFC 5348 and FR could be more aggressive than RFC 3448. This is the drawback of flows using a larger recover rate during transient conditions. However, as the flows experience congestion, the loss decreases and stabilises over time, similar to RFC 3448, and hence we consider this response is acceptable for use across the Internet.

When a TCP CWV flow experienced packet loss after a restart from a large idle period with a recover window of 1 segment, the ssthresh reduced to 2 segments. Hence when TCP CWV experienced persistent congestion, it was unable to increase the cwnd. RFC 3448 exhibited a lower average arrival rate when it competed

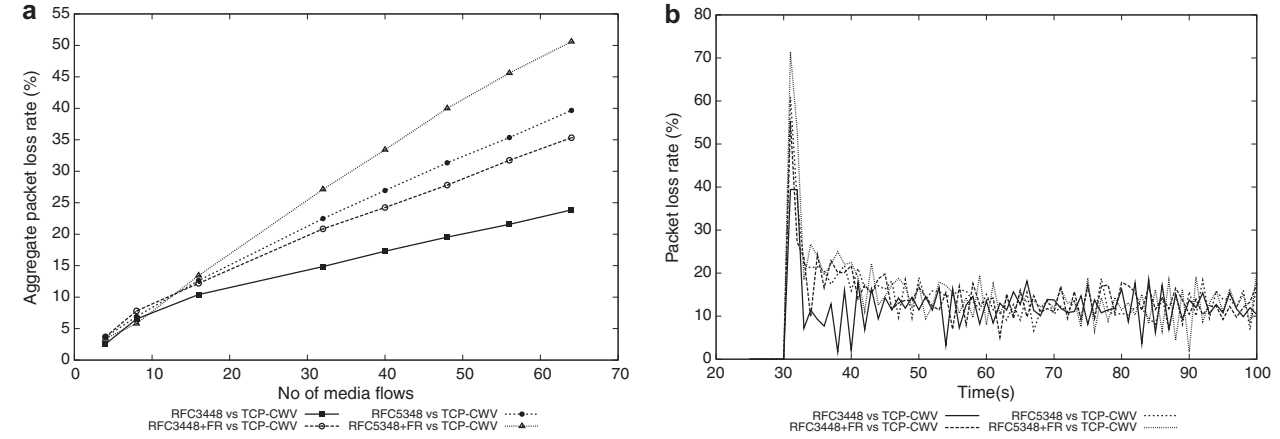

Fig. 11. (a) Aggregate packet loss rates over a 5 s period. (b) Packet loss rate evolution over time.

with TCP CWV flows and the arrival rates was less than T, for all levels of congestion. Although RFC 3448 with FR had a higher recover_rate of 8 packets per RTT, this did not exceed T Bps over all congestion levels. For all congestion levels, the average arrival rate of RFC 5348 was less than T.

FR with RFC 5348 did exceed T, achieving a higher rate because of the higher recover_rate of 8 packets/RTT and the accelerated behaviour of FR, allowing the sender to quadruple to the rate after an idle period. This behaviour caused a serious congestion episode reflected in the higher packet loss rates experienced at the bottleneck router in Fig. 11a and b. However, in periods of congestion, the sending rate of FR was limited by the throughput equation. In this scenario, FR was not more aggressive than 1.5∗T, and hence is not considered significantly more aggressive than TCP [32]. In this example scenario, no flows experienced loss and flows remained in slow-start prior to becoming idle and hence had larger recover_rates. If flows had experienced packet loss prior to going idle, then the effect would be less severe, as demonstrated in the next scenario.

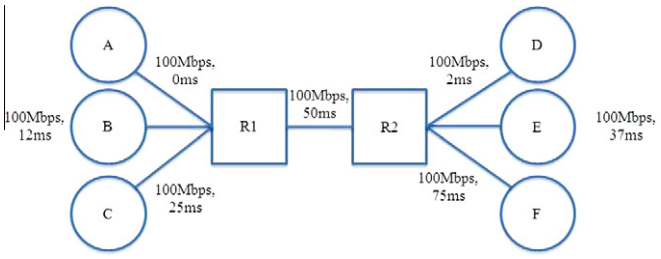

Fig. 12. Simulation toplogy.

The behaviour of TFRC is dependent on the ratio of the idle period to the path RTT. Fig. 12 shows a more complex network topology and considers a range of RTTs. The set of link delays were the same as those in [30]. The values were used to analyse the impact of restarting with different sending rates. Perfomance was examined by considering an application sending 512 Kbps, with one packet approximately every 15 ms. The behavior of a TFRC sender considers the application to be idle only for an RTT much greater than 15 ms. Smaller RTTs were therefore not considered.

Simulations were performed with equal numbers of media flows carried by both TCP with CWV and TFRC. Each high rate media flow started at a uniformly-distributed random time over 1 s. At 25 s, the link (R1-R2) capacity reduced to 2 Mbps forming a bottleneck. This ensured that most flows experienced loss prior to going idle. After 25 s the media flows were stopped and restarted at random times over 3 s, producing a wider range of idle periods. This ensured that the TFRC flows reduced their sending rates to different rates based on their idle period. Fig. 13 shows the arrival rate for the TCP flows and TFRC flows at the bottleneck, measured over a period of 5 s and the average arrival rate plotted with the maximum sending rate T. The maximum RTT, R in equation (4), was 300 ms. Fig. 14 presents the aggregate packet loss rates experienced at the bottleneck router over the 5 s period.

When the new TFRC methods experienced loss prior to becoming idle, the sending rate was lower than the recover_rate, and hence their average arrival rates were comparable to those for RFC 3448. Fig. 14 reports packet loss rates at the bottleneck router. For all methods, the average arrival rate was less than T, showing

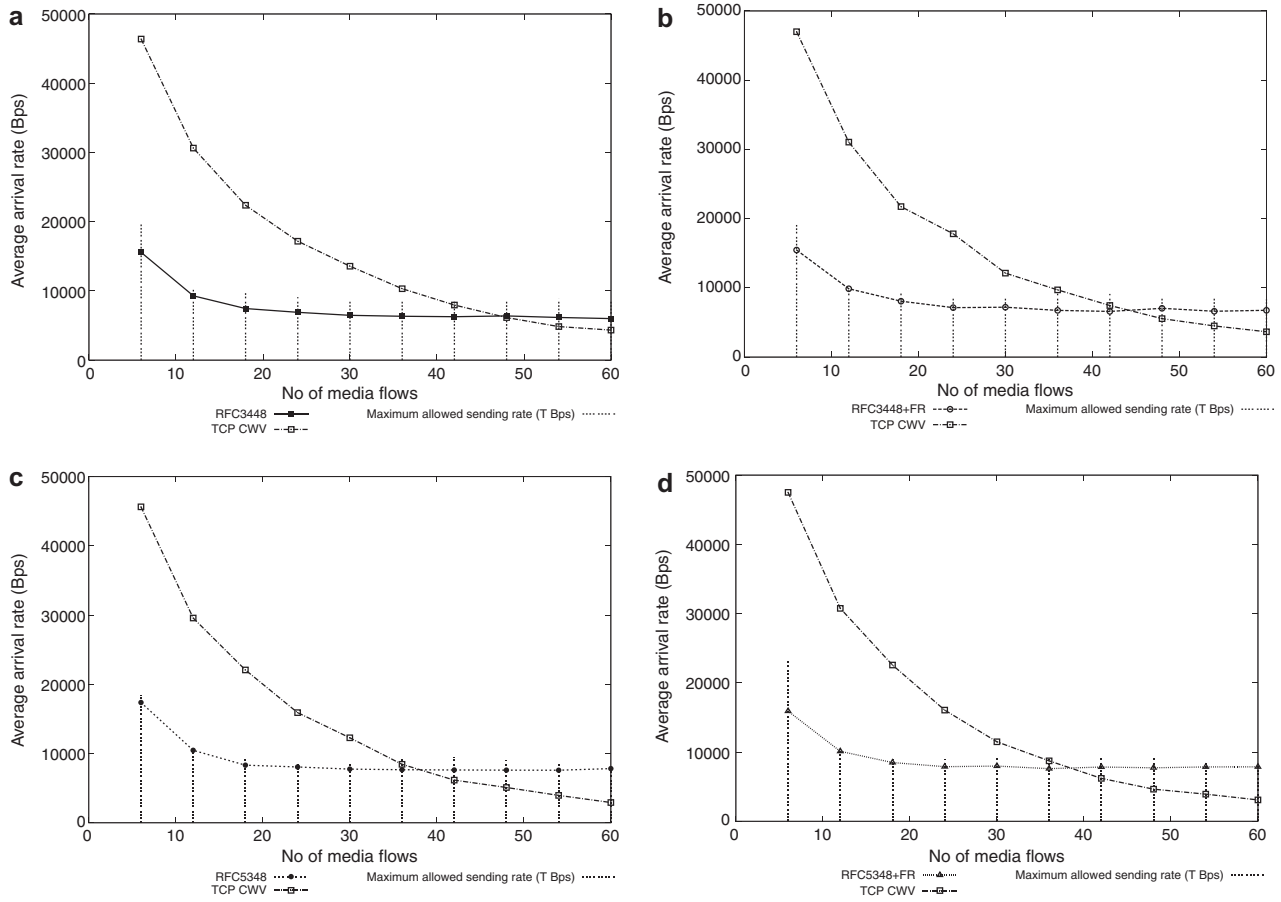

Fig. 13. (a) Average arrival rates for RFC 3448. (b) Average arrival rates for RFC 3448 + FR. (c) Average arrival rates for RFC 5348. (d) Average arrival rates for RFC 5348 + FR.

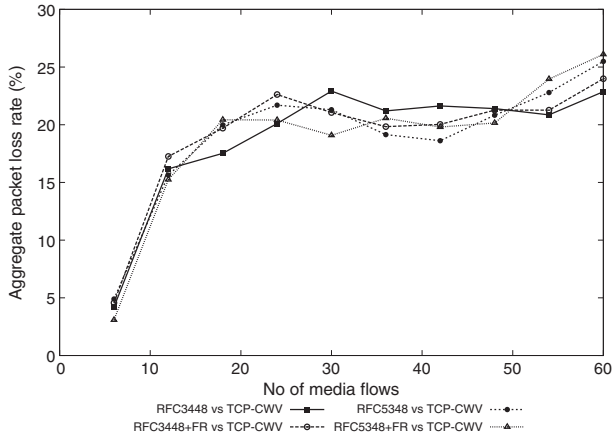

Fig. 14. Comparison of packet loss rates.

that in the presence of heavy congestion, where flows experience loss prior to restart. The impact of the methods on the bottleneck router was comparable for all considered TFRC methods.

These results are also applicable when a data-limited application using the new TFRC methods experiences a sudden reduction in path capacity. This would lead to packet loss during the data-limited period, but afterwards the new methods resume sending at a rate limited by the throughput equation, and hence would not be aggressive.

The experiments in this section show the level of congestion is a function of the bottleneck link capacity. In these simulations a sudden reduction in link capacity was used to explore whether it is reasonable to allow an application to send faster after idle, and we found that the contribution to transient congestion in times of change, was an acceptable tradeoff in return for improving application responsiveness [17]. Therefore, it was safe to use RFC 5348 and FR in all the scenarios we considered.

5. Discussion

Congestion control is essential when bandwidth-limited paths need to be shared with other traffic or where path characteristics can change (e.g. through mobility, routing changes, changes at the link-layer, wireless propagation changes). Although the TFRC algorithm defined in RFC 3448 has been deployed, this algorithm was based on a model for TCP that was designed for continuous flows. Previous research identified that RFC 3448 was not well-suited to bursty applications such as VoIP [15,16]. Hence, it was important for TFRC to be revised. In response to this need, an experimental extension to improve recovery after a long idle or data-limited period, FR [17], was introduced to the Datagram Congestion Control (DCCP) working group who maintain TFRC. This inspired the working group to redesign the TFRC protocol to better suit bursty applications, in an update finally published as RFC 5348. This paper presents the first detailed analysis of the performance examining the two new TFRC methods and assessing whether they would be suitable for use in the Internet.

The impact of a RFC 3448 flow carrying bulk traffic sharing the Internet has already been extensively evaluated (e.g. [8]). Since RFC 5348 did not update this bulk behaviour, we expect the impact of RFC 5348 flows sharing the Internet to not be worse than that using RFC 3448. For bursty flows, the results in Section 4.1 show that RFC 5348, was able to perform better than RFC 3448, even in the presence of small transmit buffers and largely varying media rates, and we suggest this indicates suitability for bursty interactive applications, such as VoIP and video.

FR was effective in improving the performance of restart during idle periods for VoIP traffic carried by RFC 3448. However, it did not provide significant performance benefit for a data-limited application, such as bursty video. This justifies why it was appropriate for RFC 5348 to update its restart behaviour. Furthermore, since FR did not offer additional performance when used in combination with RFC 5348, we conclude that RFC 5348 alone is sufficient.

One impact of RFC 5348 is that the sender retains the previous sending rate when entering a data-limited period. Even though the receive rate is low during this period, it may resume at the previous rate at some arbitrary later time [14]. This approach follows the method standardised for TCP. If a long-lived TFRC flow suffered a path change while data limited for a period of minutes or hours, it could therefore restart using a high sending rate that may be fair for the current path. To counter this, it would seem desirable to decay the value of the previous sending rate as this becomes out-of-date, similar to the experimental CWV method proposed for TCP [18]. A method to achieve this decay was suggested in Appendix C of RFC 5348 standard [6]. In this proposal, the sending rate is reduced by a half for every 4 RTT of data-limited period, ensuring a sender that remains data-limited for a significant interval will eventually reduce the sending rate to the media rate. Fig. 15 presents the sending rate for the variable rate video flow in Section 4.1c using RFC 5348 updated with this CWV-style behaviour. This shows that the CWV-style RFC 5348 responds to the data-limited period and reduces the sending rate to reflect the current media rate of an application.

To verify whether such a behavior would be beneficial to the network, a data-limited scenario was simulated using the topology in Fig. 9. A set of 512 Kbps media flows became data-limited for a period, during which the bottleneck capacity was reduced from 100 Mbps to 2 Mbps. The flows then restarted at their initial sending rate. Fig. 16 shows the aggregate packet loss rate measured over a 5 s period. When RFC 5348 was modified to use the TCP-CWV style behavior i.e., the sending rate reduced by a half for every 4 RTTs of data-limited period, the aggregate packet loss rate reduced, improving network performance.

The simulations in Sections 4.1a and c were repeated to verify whether such a conservative approach lead to poor performance for an application. The results are not presented in this paper, since they did not show any notable difference (less than 0.2%) in performance for either VoIP or video. There were periods when the video flow was data-limited (e.g. during the period 20 s–25 s in Fig. 15) then increased to a higher media rate, but the change was minimal (less than twice TFRC's current sending rate). Hence this did not

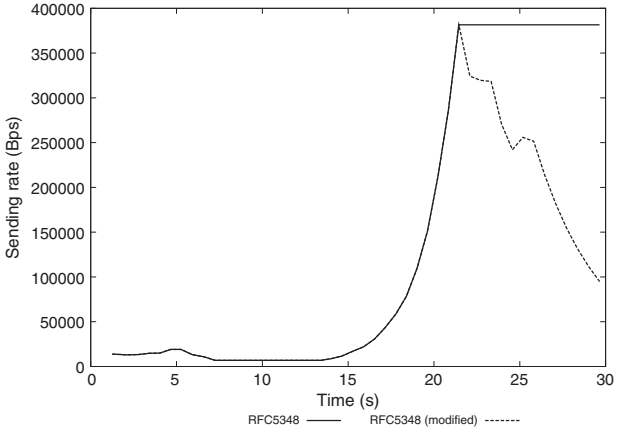

Fig. 15. Sending rate comparison.

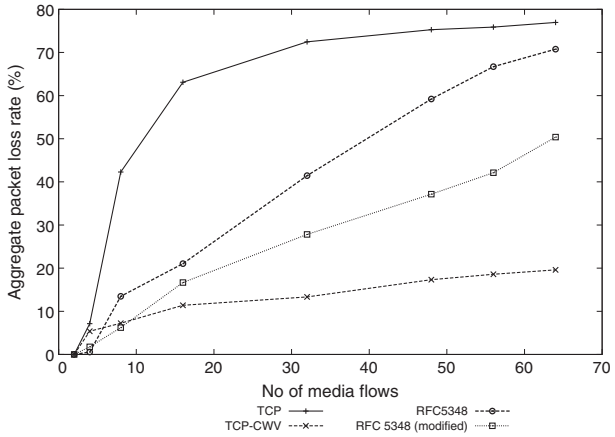

Fig. 16. Aggregate packet loss rate.

impair the video performance, since the sending rate was sufficient to accommodate the change.

These results show that it is desirable to decay the previous sending rate for a data limited period, and that this need not impair the application performance. However, we were not able to suggest the most appropriate function to reduce the sending rate during a data-limited period, or to explore the impacts of different choices of network topology or application design. This remains future work.

To improve the performance of interactive flows, a TFRC (RFC 5348) sender does not reduce the sending rate during a data-limited period, but TFRC does reduce the sending rate to one half every 4 RTTs during an idle period. There have been recent requests to update TCP congestion control during initial slow-start [35] and during idle and data-limited periods [36]. If TFRC were also to use a larger initial sending rate, as in [35], this could improve performance of both streaming and interactive flows. To offer acceptable performance in a shared Internet, these methods may need to rapidly reduce the sending rate in the presence of negative feedback (e.g. following the advice offered for use of TFRC with Quick-start [37]). However, the congestion response will depend on the accuracy of the RTT measurement and we suggest further analysis is needed to ensure a congestion response comparable to [36] in scenarios where feedback may be lost. Furthermore, if the method in [36] is standardized, then there could be a case for TFRC to also preserve the sending rate during (short) idle period.

There are cases where loss on a path is not a direct function of congestion. Complexities can be introduced by queuing strategies to support differentiated QoS, by load-balancing, and by lower-layer mechanisms, such as mobility hand-overs, link-layer retransmission, wireless propagation effects, etc. These effects can change the loss rate and round trip time, and may not necessarily be the same on the forward and return paths. For example packet loss in wireless networks may be induced by link errors. Misinterpretation of loss/RTT variation could unnecessarily reduce the TFRC sending rate and may not be sufficient to accommodate the media rate of an application. RFC 5348 does not offer any mechanisms to mitigating this. Future work could explore incorporating methods such as TFRC-Veno [33], which propose a new TFRC equation based on TCP Veno [34]. Such a mechanism may enable RFC 5348 to improve performance in lossy wireless networks. However, it is important to note that this is not just an issue for TFRC: Non-congestion losses also impact performance of a flow using TCP [34] or DCCP.

While TFRC is an algorithm that may be used by any UDP application, the IETF also wanted to encourage a more standard approach to multimedia transport, which culminated in the DCCP transport protocol (see Section 1) [3]. DCCP provides a standard way to implement congestion control and parameter negotiation. Applications that may in future be expected to use DCCP include those that prefer timeliness of data to reliability (example interactive video and Voice over IP). These applications benefit from the flow-based semantics of TCP, but do not require TCP's in-order delivery and reliability semantics.

DCCP offers a choice of congestion control algorithms (identified by congestion control ID, CCID). CCID-2 is based on the TCP SACK-like congestion control. This is appropriate for DCCP flows that would benefit from receiving as much capacity as possible over the long term, consistent with end-to-end congestion control, and that can tolerate the large sending rate variations characteristic of AIMD congestion control, including halving the congestion window in response to a congestion event. CCID-3 is an instantiation of TFRC, as defined in RFC 5348. CCID 4 is based on TFRC-SP, updated by the features introduced in RFC 5348. Simulations in this paper have confirmed that DCCP CCID-4 is more suited than CCID-3 for VoIP applications in the presence of loss (Section 4). DCCP is intended to be applicable to all types of multimedia traffic and therefore we believe the outcomes of this paper would be beneficial to the DCCP community.

Further work is needed to examine TFRC behaviour over links with varying propagation delay, especially in the presence of loss. TFRC's congestion response relies on accurate estimation of the RTT to determine the receive rate, loss event rate and to determine idle periods. Overestimation of the receiver rate and underestimation of the loss interval or RTT would inflate the sending rate, which could lead to aggressive behavior. There has been a related recently reported problem for the DCCP specifications for CCID-3/CCID-4, where the use of a window counter to provide the receiver-side RTT estimation was shown to result in inaccurate RTT estimation [38]. This issue was resolved by introducing a timestamp option to improve performance over variable-RTT paths. RTT estimation is also important for setting an appropriate nofeedback timer. Too low a value may falsely trigger the nofeedback timer, unnecessarily reducing the TFRC sending rate, and reducing the performance of interactive applications. Such issues are particularly important for paths that include wireless links.

Sections 4.1 and 4.2 have examined the detailed performance across a range of scenarios. Although the results present a clear benefit to both the application and network, the authors encourage further research to explore the suitability of these methods across a wider range of network scenarios (including the use of wireless links). Application-level design choices can also impact the overall performance, e.g. choice of buffer size, encoding and decoding techniques. In particular, this paper made assumptions on the way that the media codec interfaced to the transport layer. The use of a small transmit buffer at this interface is attractive for interactive traffic such as a videoconferencing, and where the rate mismatch between the congestion-controlled rate and the media rate could result in transmit buffer overflow, with loss of media packets. Sophisticated multimedia applications may use a range of method to mitigate this effect, e.g.:

1. Re-buffering data at the receiver, to momentarily pause output until sufficient media has been accumulated to allow replay;
2. Advanced coding/concealment techniques at the receiver that hide the effects of loss;
3. Stream-thinning, where the sender selectively removes chosen media frames when the media rate exceeds the rate allowed by congestion control;
4. Adaptive video codecs that vary the media rate based on feedback from congestion control.

We suggest that future work should explore these methods, on their own, or in combination to improve the quality of multimedia during periods of network congestion.

6. Conclusion

The introduction of congestion control for multimedia traffic is important to ensure the stability of the next generation Internet. The TCP-Friendly Rate Control (TFRC) algorithm was first specified in RFC 3448. The specification in RFC 3448 poorly supported interactive multimedia applications, leading to common use of non-standard congestion control methods and an incentive to use padding to guarantee the required media rate for bursty applications. From a network perspective, padding consumes unnecessary capacity and is therefore undesirable for other flows that share an Internet bottleneck. It was thus important to revisit and revise the TFRC mechanism to support bursty media flows and make TFRC more suited to a wider range of multimedia flows.

This paper presents the first detailed analysis considering the performance of two new TFRC improvements designed to better support flows carrying bursty applications. Faster Restart allows more rapid increase of the sending rate over paths where a higher rate was previously used. Revised TFRC, specified in RFC 5348, also increases the sending rate compared to RFC 3448, but uses a different metric for calculating the allowed sending rate when an application has used less than the recent allowed rate. Simulation results demonstrate that both new methods allow TFRC to offer acceptable performance with bursty media over shared Internet paths. Although Faster Restart could benefit RFC 3448, the method does not offer significant additional benefit when used in combination with Revised TFRC. Our analysis also evaluates the performance of Revised TFRC and demonstrates that this can substantially improve the performance of other network traffic sharing a congested network. These results demonstrate that Revised TFRC has addressed the principle performance shortfalls of TFRC for bursty applications and has removed the previous incentive for applications to use padding. We therefore expect this new standard to further encourage the use of a standards-based congestion control for multimedia.

Acknowledgements

We thank Sally Floyd for her extensive feedback on this paper and members of the IETF DCCP working group for their contributions to RFC 5348 and FR. We would also like to thank Raffaello Secchi for providing valuable feedback during the earlier versions of this paper. This work was supported by the EU Information Society Technologies SATSIX project, IST-2–26950 and the Satellite Internet for Rural Access (SIRA) project at the Rural Digital Economy Hub, funded by the RCUK Digital Economy Programme.

References

[1] Cisco Visual Networking Index: Usage Study, White paper.
[2] J. Postel, User Datagram Protocol, IETF RFC 768, August 1980.
[3] S. Floyd, M. Handley, E. Kohler, Problem Statement for the Datagram Congestion Control Protocol (DCCP), IETF RFC 4336, March 2006.
[4] M. Kaufman, Real Time Media Flow Protocol Overview, <http://www.ietf.org/proceedings/10mar/slides/tsvarea-1.pdf>, IETF TSV Area, IETF 77, Anaheim, CA, USA, March, 2010.
[5] M. Handley, S. Floyd, J. Padhye, J. Widmer, TCP Friendly Rate Control (TFRC): Protocol Specification, IETF RFC 3448, January 2003.
[6] M. Handley, S. Floyd, J. Padhye, J. Widmer, TCP Friendly Rate Control (TFRC): Protocol Specification, IETF RFC 5348, April 2008.
[7] J. Padhye, Towards a Comprehensive Congestion Control Framework for Continuous Media Flows in Best Effort Networks, Ph.D. Thesis, University of Massachusetts Amherst, 2000.
[8] S. Floyd, M. Handley, J. Padhye, J. Widmer, Equation-Based Congestion Control for Unicast Applications, ACM SIGCOMM, Sweden, August 2000.
[9] L. Xu, J. Helzer, Media streaming via TFRC: An analytical study of the impact of TFRC on user-perceived media quality, Computer Networks 51 (17) (2007).
[10] M. Chen, A. Zakhor, Rate control for streaming video over wireless, IEEE INFOCOM, Hong Kong, March 2004.
[11] J. Widmer, M. Handley, Extending Equation-based Congestion Control to Multicast Applications, ACM SIGCOMM, San Diego, August 2001.
[12] S. Floyd, E. Kohler, TCP Friendly Rate Control (TFRC): the Small-Packet (SP) Variant, IETF RFC 4828, April 2007.
[13] S. Floyd, E. Kohler, J. Padhye, Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC), IETF RFC 4342, March 2006.
[14] S. Floyd, E. Kohler, Profile for DCCP Congestion Control ID 4: the Small-Packet Variant of TFRC Congestion Control, IETF RFC 5622, August 2009.
[15] A. Sathiaseelan, G. Fairhurst, Performance of VoIP using DCCP over DVB-RCS Satellite Network, IEEE ICC, Glasgow, June 2007.
[16] V. Balan, L. Eggert, S. Niccolini, M. Brunner, An Experimental Evaluation of Voice Quality Over the Datagram Congestion Control Protocol, IEEE INFOCOM, Alaska, May 2007.
[17] E. Kohler, S. Floyd, A. Sathiaseelan, Faster Restart for TCP Friendly Rate Control (TFRC), IETF Work in progress, Internet draft draft-ietf-dccp-tfrc-faster-restart-06.txt, July 2008.
[18] M. Handley, J., Padhye, S. Floyd, TCP Congestion Window Validation, IETF RFC 2861, Experimental, June 2000.
[19] S. Floyd, J. Kempf, IAB Concerns Regarding Congestion Control for Voice Traffic in the Internet, IETF RFC 3714, March 2004.
[20] H. Balakrishnan, S. Seshan, M. Stemm, R. Katz Analysing stability in wide-are network performance, ACM Sigmetrics, Seattle, USA, June 1997.
[21] M. Allman, S. Floyd, C. Partridge, Increasing TCP's Initial Window, IETF RFC 3390, October 2002.
[22] S. McCanne, S. Floyd, Network Simulator. <http://www.isi.edu/nsnam/ns/>.
[23] P. Sarolahati, M. Allman, S. Floyd, Determining an appropriate sending rate over an underutilized network path, Computer Networks 51 (7) (2007).
[24] S. Floyd, E. Kohler, Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control, IETF RFC 4341, March 2006.
[25] ITU-T Recommendation P.59: Artificial Conversational Speech, March 1993.
[26] Assessing VoIP Call Quality Using the E-model, white paper, available online at: <http://www.ixiacom.com/pdfs/library/white_papers/voip_quality.pdf>.
[27] G.107, The E-model, A Computational Model for Use in Transmission Planning, ITU-T, 2005.
[28] H.263 Video traces, Telecommunication Networks Group, <http://www.tkn.tu-berlin.de/research/trace/ltvt.html>.
[29] S.H. Kang, A. Zakhor, Packet scheduling algorithm for wireless video streaming, in: 12th International Packet Video Workshop, Pittsburgh, USA, April 2002.
[30] L. Andrew, C. Marcondes, S. Floyd, L. Dunn, R. Guillier, W. Gang, L. Eggert, S. Ha, I. Rhee, Towards a Common TCP Evaluation Suite, PFLDnet, Manchester, March 2008.
[31] S. Floyd, K. Fall, Promoting the use of end-to-end congestion control in the internet, IEEE/ACM Transactions on Networking 7 (4) (1999).
[32] S.Floyd, HighSpeed TCP for Large Congestion Windows, IETF RFC 3649, December 2003.
[33] B. Zhou, C. Fu, V. Li, TFRC Veno: an Enhancement of TCP Friendly Rate Control over Wired/Wireless Networks, ICNP, October 2007, pp. 216–225.
[34] C.P. Fu, S.C. Liew, TCP Veno: TCP enhancement for transmission over wireless access networks, IEEE Journal of Selected Areas in Communications 21 (2) (2003).
[35] J. Chu, N. Dukkipati, Y. Cheng, M. Mathis, Increasing TCP's Initial Window, IETF Work in progress, Internet draft-ietf-tcpm-initcwnd-00.txt, October 2010.
[36] G. Fairhurst, I. Biswas, Updating TCP to support Variable-Rate Traffic, IETF Work in progress, draft-fairhurst-tcpm-newcwv-01.txt, March 2011.
[37] Fairhurst, G. and A. Sathiaseelan, "Quick-Start for the Datagram Congestion Control Protocol (DCCP), RFC 5634, August 2009.
[38] G. Renker, G. Fairhurst, Sender RTT Estimate Option for DCCP, IETF Work in progress, Internet draft-ietf-dccp-tfrc-rtt-option-06.txt, April 2011.