ARTICLE IN PRESS

The Journal of Systems and Software xxx (2008) xxx–xxx

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

A receiver-centric rate control scheme for layered video streams in the Internet

Panagiotis Papadimitriou *, Vassilis Tsaoussidis, Lefteris Mamatas

Democritus University of Thrace, Electrical and Computer Engineering Department, 12 Vas. Sofias Street, Xanthi, Greece

ARTICLE INFO

Article history:
Received 20 February 2007
Received in revised form 29 February 2008
Accepted 29 February 2008
Available online xxxx

Keywords:
End-to-end protocols
Rate control
Quality of Service
Video streaming

ABSTRACT

We present a new end-to-end protocol, namely *Dynamic Video Rate Control* (DVRC), which operates on top of UDP and enables the adaptive delivery of layered video streams over the Internet. The protocol optimizes the performance on video delivery with concern to friendliness with interfering traffic. DVRC enables a closed-loop control between server and client, where the receiver detects the state of congestion, determines the proper transmission rate, and eventually opts for the optimal number of layers that should be delivered according to this rate. The protocol relies on a hybrid *Additive Increase Additive Decrease* (AIAD)/*Additive Increase Multiplicative Decrease* (AIMD) algorithm (namely AIAMD) that manages to differentiate congestive and non-congestive loss by utilizing history in its control rules. AIAMD combines the most desirable features of AIAD and AIMD, reacting gently to random loss and more aggressively to congestion and adapting effectively to the dynamics of the network. Therefore, DVRC enables the desired smoothness for video streaming applications and at the same time avoids significant damage during congestion. Exploring DVRC's potential through extensive simulations, we identify notable gains in terms of bandwidth utilization and smooth video delivery. Furthermore, our results indicate that the protocol allocates a well-balanced amount of network resources maintaining friendliness with corporate TCP connections.

1. Introduction

Multimedia applications gain popularity and streaming media data is expected to compose a considerable portion of the overall data traffic traversing the Internet. These applications generally prefer timeliness to reliability. Video streaming, in particular, calls for strict requirements on end-to-end latency and delay variation. Long end-to-end delays commonly affect the timely delivery of video-data causing data unavailability and unintelligible real-time interaction with frustrating consequences to the end-user. Furthermore, reliability parameters, such as packet loss and bit errors, usually compose an impairment factor, since they cause a perceptible degradation on video quality. Unlike bulk data transfers, video streaming seeks to achieve smooth playback quality rather than simply transmit at the highest attainable bandwidth.

Multimedia applications commonly *rely* on the unreliable transport services provided by *User Datagram Protocol* (UDP). UDP is a fast, lightweight protocol without any transmission or retransmission control. The protocol does not have functionality to override application characteristics, such as its transmission rate. It simply transmits at application rate and pattern. However, the lack of congestion control poses a threat to the network: had such applications dominated the Internet, it would have faced risk of congestion collapse. In this context, Internetworking functionality evolves towards punishing free-transmitting protocols.

On the other hand, *Transmission Control Protocol* (TCP), based on the principles of congestion management (Jacobson, 1988), *Slow-Start* (Stevens, 1997), and *Additive Increase Multiplicative Decrease* (AIMD) (Chiu and Jain, 1989), provides a reliable data delivery service to Internet applications and is in large part responsible for the remarkable stability of the Internet. However, the protocol introduces arbitrary delays, since it enforces reliability and in-order delivery. Furthermore, the process of probing for bandwidth and reacting to the observed congestion induces oscillations in the achievable transmission rate. The variations in sending rate can be theoretically smoothed out with application-level buffering, but this could result in huge client buffers and unacceptable end-to-end delays depending on the extent of fluctuations.

Several TCP protocol extensions (Bansal and Balakrishnan, 2001; Jin et al., 2004; Mascolo et al., 2001; Rhee et al., 2000; Tsaoussidis and Zhang, 2002) have emerged to overcome the standard TCP limitations providing more efficient bandwidth utilization and sophisticated mechanisms for congestion control, which preserve the fundamental *Quality of Service* (QoS) guarantees for multimedia traffic. TCP-friendly protocols, presented in Floyd et al. (2000), Yang et al. (2001), and Yang and Lam (2000), achieve smooth window adjustments while they manage to compete fairly with TCP flows. In order to achieve smoothness, they use gentle

* Corresponding author.
E-mail addresses: ppapadim@ee.duth.gr (P. Papadimitriou), vtsaousi@ee.duth.gr (V. Tsaoussidis), emamatas@ee.duth.gr (L. Mamatas).

0164-1212/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2008.02.067

backward adjustments upon congestion. However, this modification has a negative impact on protocol responsiveness. In Papadimitriou et al. (2005) and Tsaoussidis and Zhang (2005), we showed that TCP-friendly protocols are unable to effectively recover from excessive congestion incidents resulting in increased packet drops, which eventually degrade application performance. Equation-based rate control (Floyd et al., 2000), which can be construed as the opposite end of linear congestion control algorithms (e.g., AIMD), has been promoted as an attractive option for multimedia Internet transmission. It has the desirable feature of delivering maximally smooth, TCP-fair transmission rate. However, it is characterized by very slow responsiveness to network dynamics, as well.

Considering TCP's limitations and the impending threat of unresponsive UDP traffic, we need sophisticated congestion control that interacts efficiently with other flows on the Internet. An overview of Internet's current congestion control paradigm reveals that routers play a relatively passive role: they merely indicate congestion through packet drops or *Explicit Congestion Notification* (ECN). It is the end-systems that perform the crucial role of responding appropriately to these congestion signals. Furthermore, since video encoding involves inter-frame dependencies, the random dropping of packets by routers can seriously degrade video quality. In MPEG, for example, dropping packets from an independently encoded *I* (*intra*-picture) frame causes the following dependent *P* (*predictive*), and *B* (*bidirectional*) frames not to be fully decodable. In practice, inter-frame dependencies may render a 3% packet loss rate up to a 30% frame loss rate (Boyce and Gaglianello, 1998).

Without explicit feedback, end-to-end congestion measure can only be loss probability, as used in TCP Reno and *TCP-friendly Rate Control* (TFRC) (Floyd et al., 2000), or queuing delay, as used in TCP Vegas (Brakmo and Peterson, 1995) and FAST TCP (Jin et al., 2004). However, recent studies (Martin et al., 2003) uncovered that delay and packet loss can have a weak correlation, especially when packet losses occur due to other reasons than buffer overflow (i.e., wireless errors). As a result, using delay as a measure of congestion may cause undesirable effects in terms of bandwidth utilization, as well as network stability, especially if it is not augmented with loss information.

At the same time, numerous video-streaming applications have implemented their own congestion control mechanisms, usually on a case-by-case basis on top of UDP. Most applications exploit *Real-Time Control Protocol* (RTCP) (Schulzrinne et al., 2003) and *Real-Time Streaming Protocol* (RTSP) (Schulzrinne et al., 1998) to enable controlled delivery of video. RTCP, in particular, allows the video application to exploit feedback of reception statistics (i.e., received and lost packets, jitter, round-trip delay) and adjust the sending rate accordingly. In the presence of such an end-to-end protocol, a video streaming server can adapt the quality of its transmission. The manipulation of a compressed video stream can be attained by techniques, such as *temporal* and *quality* scalability (Liu and Zhang, 2003). According to temporal scaling, the streaming server selectively discards frames prior to transmission. *Simulcast* and *layered adaptation* are the most remarkable quality scalability techniques, which directly adjust the bitrate of the transmitted video stream. Despite the presence of such mechanisms, implementing application-level congestion control is still difficult and eventually not part of most applications needs. We believe that a new transport protocol is needed, which would combine unreliable datagram delivery with built-in congestion control. This protocol would act as an enabling technology: new and existing applications could use it to timely transmit data without destabilizing the Internet.

In this context, we have been working on a rate control scheme for adapting outgoing video streams to the characteristics of the end-to-end network path. Rate adaptive video streams offer the clients the benefit of being resilient to changing network conditions and allow a large number of streams to concurrently share network resources. Multimedia streams can be adaptive, since user-perceived QoS is often satisfactory over a range of stream compression levels. Although this adaptivity is limited (i.e., multimedia streams have minimum subscription levels, below which service quality is unacceptable), they have the capability of adjusting their subscription levels in response to congestion, such as elastic flows do. Along these lines, we designed a new end-to-end protocol, namely *Dynamic Video Rate Control* (DVRC), which operates on top of UDP and desirably provides out-of-order delivery. DVRC is intended to interact with a plethora of multi-rate streaming applications that rely on cumulative layered transmission (Liu and Zhang, 2003). This approach is based on information decomposition. The video stream is encoded at a *base layer* and one or more *enhancement layers*, which can be combined to render the stream high quality. Layered adaptation is performed by adding or dropping enhancement layers depending on the prevailing network conditions.

DVRC enables a closed-loop control between server and client, monitoring the transport's progress separately for transient and persistent rate changes, and in response actuating the video stream quality-rate trade-off. The protocol employs a receiver-centric congestion control mechanism and does not rely on QoS functionality in routers, such as *Random Early Drop* (RED), ECN or other *Active Queue Management* mechanisms. Inline with Hsieh et al. (2003), Rhee et al. (2000) and Tsaoussidis and Zhang (2002), certain protocol functionalities can be moved from the sender to the receiver. The sender merely acts based on the requests from the receiver. Generally, delegating the protocol's control functions to the receivers composes an elegant and functional approach in several occasions:

- Receiver-oriented error control incarnates the property of the receiver to determine with better accuracy the data delivery rate and the potential level of data loss. This abrogates the impact of false assessments at the sender due to lost or delayed acknowledgements.
- In wired/wireless environments, the receiver is adjacent to the wireless last-hop and has first knowledge about the characteristics of the wireless links.
- Receiver-centric transport protocols can significantly reduce the complexity of the server implementation, since they distribute the state management across a large number of clients.

DVRC relies on a hybrid *Additive Increase Additive Decrease* (AIAD)/AIMD scheme, namely AIAMD, in order to adapt its sending rate. In contrast to the *memory-less* AIAD and AIMD algorithms, AIAMD utilizes *history* information in its control rules. A very limited number of proposals relying on linear congestion control schemes exploit history information (e.g., Jin et al., 2003). In summary, AIAMD has the following salient attributes: (i) utilizes history of receiving rates in order to distinguish between congestion-induced and random loss; (ii) reacts gracefully to random loss in order to keep the sending rate variation to minimum, but reacts quickly to the onset of congestion; and (iii) is both efficient and fair to the network environment. As a result, DVRC enables the desired smoothness for video streaming applications and at the same time avoids significant damage during congestion. Our simulations demonstrate that DVRC provides low variation in the transmission rate in steady state, and at the same time is reactive and TCP-friendly. We also show that satisfactory performance can be achieved with a small number of layers (4–5 layers).

The remainder of this paper is organized as follows: Section 2 provides an overview of related work, while in Section 3 we discuss the design and implementation details of the proposed rate control

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

scheme. In Section 4, we present the parameters of our evaluation methodology, followed by Section 5, where we demonstrate conclusive performance studies based on extensive simulations. Finally, in Section 6 we highlight our conclusions.

2. Related work

The literature includes numerous studies and proposals towards efficient rate/congestion control for multimedia applications in the Internet. Generally, we can distinguish two approaches for rate control of multimedia traffic in the Internet:

- TCP-like protocols (e.g., Rejaie et al., 1999): Rate control is performed in TCP fashion, where packet loss infers congestion, and subsequently the transmission rate is reduced multiplicatively.
- Equation-based protocols (e.g., Floyd et al., 2000): The available bandwidth is estimated based on statistics of the *Round Trip Time* (RTT) and packet loss probability. In response to the bandwidth estimates obtained, the source adjusts the transmission rate accordingly.

Rate Adaptation Protocol (RAP) (Rejaie et al., 1999) is a rate-based protocol that employs an AIMD-oriented algorithm for the transmission of real-time streams. The sending rate is continuously adjusted by RAP in a TCP-friendly fashion using feedback from the receiver. However, since RAP employs TCP's congestion control parameters (i.e., 1, 0.5), it causes short-term rate oscillations, primarily due to the multiplicative decrease. Furthermore, RAP occasionally does not result in inter-protocol fairness. *TCP Emulation at the Receivers* (TEAR) (Rhee et al., 2000) enables the receiver to emulate the congestion window modifications of a TCP sender, based on the congestion signals observed at the receiving host. The receiver maintains an exponentially weighted moving average of the congestion window, and divides this amount by the estimated RTT to obtain the TCP-friendly sending rate.

TFRC (Floyd et al., 2000) is a representative equation-based protocol, which adjusts its transmission rate in response to the level of congestion, as estimated based on the calculated loss rate. Multiple packet drops in the same RTT are considered as a single loss event, allowing the protocol to follow a more gentle congestion control strategy. More precisely, the TFRC sender uses the following TCP response function:

$$T(p,\mathrm{RTT},\mathrm{RTO}) = \frac{1}{\mathrm{RTT}\sqrt{\frac{2p}{3}} + \mathrm{RTO}\left(3\sqrt{\frac{3p}{8}}\right)p(1+32p^2)} \quad (1)$$

where p is the steady-state loss event rate and *RTO* is the retransmission timeout value. Eq. (1) enforces an upper bound on the sending rate T. However, the throughput model is quite sensitive to parameters (e.g., p, RTT), which are often difficult to measure efficiently and to predict accurately. The long-term TCP throughput equation does not capture the transit and short-lived TCP behaviors, and it is less responsive to short-term network and session dynamics. According to Floyd et al. (2000), TFRC's increase rate never exceeds 0.14 packets per RTT (or 0.28 packets per RTT when history discounting has been invoked). In addition, the protocol requires five RTTs in order to halve its sending rate. Consequently, the instantaneous throughput of TFRC has a much lower variation over time. TFRC eventually achieves the smoothing of the transmission gaps and therefore, is suitable for applications requiring a smooth sending rate. However, this smoothness has a negative impact, as the protocol becomes less responsive to bandwidth availability (Tsaoussidis and Zhang, 2005).

Datagram Congestion Control Protocol (DCCP) (Kohler et al., 2006) is a new transport protocol that provides a congestion-controlled flow of unreliable datagrams. DCCP is intended for delay-sensitive applications which have relaxed packet loss requirements. The protocol aims to add to a UDP-like foundation the minimum mechanisms necessary to support congestion control. DCCP provides the application with a choice of congestion control mechanisms via *Congestion Control IDs* (CCIDs), which explicitly name standardized congestion control mechanisms. CCIDs are negotiated at connection startup. Currently, two CCIDs have been developed supporting TCP-like and TFRC congestion control.

Since TCP is rarely chosen to transport delay-sensitive traffic over the Internet, numerous TCP-friendly protocols (Yang et al., 2001; Yang and Lam, 2000) constitute an elegant framework for multimedia applications. We consider as TCP-friendly any protocol whose long-term arrival rate does not exceed the one of any conformant TCP in the same circumstances (Floyd and Fall, 1999). *GAIMD* (Yang and Lam, 2000) is a TCP-friendly protocol that generalizes AIMD congestion control by parameterizing the additive increase rate and multiplicative decrease ratio. For the family of AIMD protocols, Yang and Lam (2000) derive a simple relationship between α and β in order to be friendly to standard TCP:

$$\alpha = \frac{4(1-\beta^2)}{3}$$

Based on experiments, they propose an adjustment of $\beta = 0.875$ as an appropriate smooth decrease ratio, and a moderated increase value $\alpha = 0.31$ to achieve TCP friendliness.

TCP Westwood (Mascolo et al., 2001) is a TCP-friendly protocol that emerged as a sender-side-only modification of TCP Reno congestion control. TCP Westwood exploits end-to-end bandwidth estimation in order to adjust the values of slow-start threshold and congestion window after a congestion episode. The protocol incorporates a recovery mechanism which avoids the blind halving of the sending rate of TCP Reno after packet losses and enables Westwood to achieve high link utilization in the presence of wireless errors. The specific mechanism considers the sequence of bandwidth samples *sample_BWE*[n] obtained using the acknowledgments (*ACK*) arrivals and evaluates a smoothed value, *BWE*[n], by low-pass filtering the sequence of samples, as described by the following pseudocode:

Algorithm 1. TCP-Westwood
if *ACK* an has been received
  sample_BWE[n] = (*acked* * *pkt_size* * 8)/(*now-last_ACK_time*)
  BWE[n] = (1 − *beta*) * (*sample_BWE*[n] + *sample_BWE*[$n-1$])/2
  + *beta* * *BWE*[$n-1$]
end if

where *acked* is the number of segments acknowledged by the last *ACK*; *pkt_size* is the segment size in bytes; now is the current time; *last_ACK_time* is the time the previous *ACK* was received; β is the pole used for the filtering (a value of 19/21 is suggested). TCP Westwood+ is a recent extension of TCP Westwood that computes one sample of available bandwidth every RTT, using all data acknowledged in the specific RTT (Grieco and Mascolo, 2004).

TCP-Real (Tsaoussidis and Zhang, 2002) is a high-throughput transport protocol that incorporates a congestion avoidance mechanism in order to minimize transmission-rate gaps. The protocol approximates a receiver-oriented approach beyond the balancing trade of the parameters of additive increase and multiplicative decrease. TCP-Real introduces another parameter, namely γ, which determines the window adjustments during congestion avoidance. This parameter can be adaptive to the detected conditions. Generally, TCP-Real can be viewed as a TCP (α, β, γ) protocol, where γ captures the protocol's behavior prior to congestion when congestion boosts up.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

Bansal et al. (2001) provide a detailed study of the behavior of slow-responsive congestion control algorithms, including TCP-friendly protocols such as TFRC. A self-clocking mechanism is incorporated into TFRC in order to limit the amount of sending rate increase. In the presence of packet loss, the subsequent transmission rate cannot exceed the data receiving rate in the previous RTT. An upper bound to the sending rate is also applied in the absence of packet loss, preventing a drastic rate increase in the case of bandwidth availability. This modification renders TFRC's congestion control more efficient, especially in dynamically changing environments. Note that TCP's self-clocking has a different effect on protocol behavior compared to TFRC (Grieco and Mascolo, 2004). More precisely, TCP's self-clocking is more restrictive, since it always enforces packet conservation.

Numerous studies for adaptive video delivery appear in Feamster and Balakrishnan (2002), Feamster et al. (2001), Liu and Zhang (2003), Rejaie et al. (2000), Saparilla and Ross (2000). An overview of existing solutions for video adaptation is presented in Liu and Zhang (2003). Feamster et al. (2001) analyze the impact of selected congestion control algorithms on the performance of streaming video delivery. They concentrate on binomial congestion control (Bansal and Balakrishnan, 2001) and especially on SQRT, which responds to packet drops by reducing the congestion window size proportional to the square root of its value instead of halving it. However, binomial schemes are not able to achieve TCP-friendliness independent of link capacity (Cai et al., 2005). In Feamster and Balakrishnan (2002) a *Real-Time Control* (RTP) (Schulzrinne et al., 2003) compatible protocol (i.e., SR-RTP) is proposed, which adaptively delivers high quality video in the face of packet loss. SR-RTP effectively enables selective reliability, retransmitting only the important data. Finally, Rejaie et al. (2000) propose a layered mechanism to adapt the quality of congestion-controlled video. The mechanism is able to control the level of smoothing in order to improve the quality of the delivered video stream.

3. Design and implementation

The design principles of the proposed rate control scheme mainly rest on the assumption that user's perception is sensitive to smooth and timely playback of the received video. Despite the degradation in visual quality, smooth video of lower bitrate is considered more preferable than inconsistent and jerky video of higher rate. In this context, the primary goal of DVRC is to deliver the optimal number of layers that the client can manage according to the prevailing network conditions. Layered video adaptation is performed in terms of user-perceived quality, as well as from the perspective of inter-protocol friendliness.

3.1. Sender and receiver functionality

DVRC, in a complementary role, operates on top of UDP and supports rate control relying on sender and receiver interaction. DVRC acknowledges each datagram received by transmitting a control packet. The protocol does not integrate reliability to UDP datagrams, so control packets do not trigger retransmissions. Although DVRC uses the control packets to update the sending rate, data transmissions themselves are not directly triggered by control packets, but instead are sent out based on the determined rate at the receiving end.

We have encapsulated additional header information to UDP datagrams (Fig. 1), including *packet type*, *length* and *sequence number*, *frame type*, *timestamp* and *video layer*. The DVRC header includes all necessary fields to add RTP functionality, and to enable rate control. Therefore, we do not use RTP avoiding its overhead. *Packet type* field denotes whether a segment with video-data or a control packet is transmitted. *Video layer* field includes the cumulative number of layers that should be delivered based on current conditions. *Timestamp* field is used to handle RTT computation. More precisely, when the sender transmits a video-packet, it updates the specific field with current time T_{S_n}. As soon as the receiver acquires the packet, it generates a control packet attaching T_{S_n} to the *timestamp* field. Upon the receipt of the corresponding feedback, the sender subtracts the included *timestamp* from current time in order to estimate the RTT sample. If T'_{S_n} denotes the time the control packet has been received, the sender gets the observed value of each RTT, as follows:

$$\mathrm{SampleRTT} = T'_{S_n} - T_{S_n}$$

Therefore, DVRC obtains an accurate approximation of RTT, which does not require synchronization between sender's and receiver's clocks.

Since UDP is a protocol without reliability, some datagrams may be lost due to congestion or inability of the receiving host from reading the packets rapidly enough. The receiver uses packet drops or re-ordering as congestion indicator. Consequently, congestion control is triggered, when

- a packet is received carrying a sequence number greater than the expected sequence number;
- the receiver does not acquire any packets within a timeout interval.

Along these lines, the proper adjustment of the timeout interval is critical. A timeout interval that is set too short will claim false packet losses resulting in a wasteful reduction of the transmission rate. On the other hand, a long and consequently conservative timeout interval will inevitably impact the protocol responsiveness. In order to properly adjust the timeout, we exploit RTT measurements (*SampleRTT*) and based on this quantity we further compute the weighted average of RTT:

$$\mathrm{EstimatedRTT} = \gamma \times \mathrm{EstimatedRTT} + (1 - \gamma) \times \mathrm{SampleRTT} \tag{2}$$

setting the smoothing factor γ to 0.9. After RTT estimation, the timeout interval for DVRC (*DTO*) can be calculated by

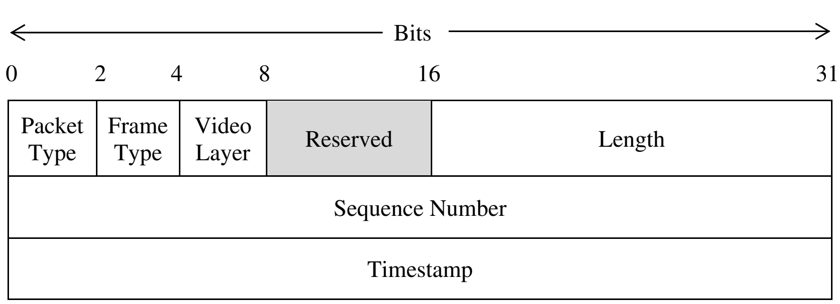

Fig. 1. DVRC header.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

$$\text{DTO} = \text{EstimatedRTT} + \delta \times \text{Deviation} \quad (3)$$

where δ is set to 4 and *Deviation* is the smoothed estimation of the variation of RTT. Deviation is expressed as

$$\text{Deviation}_n = \varepsilon \times \text{Deviation}_{n-1} + (1-\varepsilon) \times |\text{EstimatedRTT} - \text{EstimatedRTT}'| \quad (4)$$

where $Deviation_{n-1}$ and $EstimatedRTT'$ are the variation of RTT and the estimated RTT in the last round respectively, while ϵ is set to 0.25.

3.2. Rate adjustment

The rate adjustment of the video stream is performed on a receiver-oriented fashion. The sender merely acts based on the requests from the receiver. The selection of the rate control parameters is based on the following directions: (i) to enable the desired smoothness for video streaming applications, (ii) to avoid significant damage during congestion, and (iii) to maintain friendliness with coexisting TCP flows. Although some damage is inevitable at periods of congestion, a smooth backward adjustment has the potential to enhance application performance. On the contrary, multiplicative decrease causes transmission gaps that hurt the performance of multimedia applications, which subsequently experience jitter and degraded throughput.

According to the model in Chiu and Jain (1989), we consider n users sharing a single bottleneck link. In order to simplify our analysis and focus on the behavior of AIAMD control, our model inherits the assumptions of *Chiu-Jain* model: (i) all users have the same RTT and (ii) adjust their loads simultaneously. In Tsaoussidis and Zhang (2005) we extended the *Chiu-Jain* model by taking into consideration the role of the bottleneck queue. Furthermore, Gorinsky and Vin (2002) provide an extension of this model for flows with different RTTs. Incorporating such extensions into the AIAMD control is not in the scope of this paper and will compose future work.

Following the *Chiu-Jain* model, we consider a discrete timescale where every instant t corresponds to the moment when each user adjusts its load. If during time slot t, the ith user's load is $x_i(t)$, then the total load at the bottleneck resource would be

$$x(t) = \sum_{i=1}^{n} x_i(t) \quad (5)$$

The network provides the users with a binary feedback $y(t)$, which indicates whether the total load $x(t-1)$ after the previous adjustment exceeds an optimal value X_{goal}:

$$y(t) = \begin{cases} 1 & \text{if } x(t-1) > X_{\text{goal}} \\ 0 & \text{if } x(t-1) \leqslant X_{\text{goal}} \end{cases} \quad (6)$$

Linear congestion control algorithms are governed by the following update function:

$$x_i(t) = \begin{cases} \alpha_I + \beta_I x_i(t-1) & \text{if } y(t) = 0 \\ \alpha_D + \beta_D x_i(t-1) & \text{if } y(t) = 1 \end{cases} \quad (7)$$

In the case of AIAD ($\beta_I = \beta_D = 1$), user i responds to binary feedback $y(t)$, as follows:

$$x_i(t) = \begin{cases} \alpha_I + x_i(t-1) & \text{if } y(t) = 0 \\ \alpha_D + x_i(t-1) & \text{if } y(t) = 1 \end{cases} \quad (8)$$

where $\alpha_I > 0$ and $\alpha_D < 0$. Following AIMD ($\beta_I = 1$, $\alpha_D = 0$), the corresponding response for user is

$$x_i(t) = \begin{cases} \alpha_I + x_i(t-1) & \text{if } y(t) = 0 \\ \beta_D x_i(t-1) & \text{if } y(t) = 1 \end{cases} \quad (9)$$

where $\alpha_I > 0$ and $0 < \beta_D < 1$. We note that neither of these controls utilizes history information; the increase and decrease rules depend solely on current load and parameters α_I, α_D, β_I and β_D.

Before analyzing the behavior of AIAMD, we investigate the efficiency of hybrid congestion controls algorithms in terms of fairness. Consider a system in a steady state with synchronous congestion signals and static bandwidth. Following a linear congestion control algorithm, each flow would experience a periodic cycle of increases and decreases in its rate R. We assume that such a sequence consists of S_I increases followed by S_D decreases. Based on the generalized model of linear congestion control, as expressed in (7), we formulate an expression for the rate variation in steady state. After a sequence of S_I increases, the rate evolves at $\alpha_I \sum_{h=1}^{S_I} \beta_I^h + \beta_I^{S_I} R$, while after S_D decreases, it becomes $\alpha_D \sum_{h=1}^{S_D} \beta_D^h + \beta_D^{S_D} R$. Since the transmission rates before and after the sequence of S_I increases and S_D decreases are equal, rate R in steady state can be expressed by the following equation:

$$\begin{aligned} R &= \alpha_D \sum_{h=1}^{S_D} \beta_D^h + \beta_D^{S_D} \left(\alpha_I \sum_{h=1}^{S_I} \beta_I^h + \beta_I^{S_I} R \right) \\ &= \alpha_D \sum_{h=1}^{S_D} \beta_D^h + \alpha_I \beta_D^{S_D} \sum_{h=1}^{S_I} \beta_I^h + \beta_D^{S_D} \beta_I^{S_I} R = A + BR \end{aligned} \quad (10)$$

where $A = \alpha_D \sum_{h=1}^{S_D} \beta_D^h + \alpha_I \beta_D^{S_D} \sum_{h=1}^{S_I} \beta_I^h$ and $B = \beta_D^{S_D} \beta_I^{S_I}$. We differentiate three cases:

(i) $A = 0$ which gives $\alpha_I = \alpha_D = 0$ (Multiplicative Increase/Multiplicative Decrease). Therefore Eq. (10) is written as $R = BR$. Since $R > 0$, $B = 1$ allowing any value of R.
(ii) $B = 1$, thus $\beta_1 = \beta_D = 1$ (AIAD) or a specific case of $\beta_I \beta_D = 1$. Eq. (10) now gives $R = A + R$ which enforces $A = 0$. Likewise, there is no restriction to the value of R.
(iii) $A \neq 0$ and $B \neq 1$. In this case, $R = A/(1-B)$, and subsequently there is a single value of R in steady state where all flows can converge.

Following these observations, hybrid (linear) controls achieve fairness and can be safely used without implications on network stability.

DVRC employs a hybrid AIAD/AIMD (AIAMD) scheme beyond the conventional approach of purely additive increase and multiplicative decrease. According to AIAD, in the absence of packet loss, the rate is gracefully increased in order to probe for additional bandwidth; otherwise, the transmission rate is gently decreased in order to alleviate congestion. The graceful rate adjustments of AIAD overcome a number of problems associated with AIMD rate control. More precisely, AIAD is less susceptible to random loss,[1] results in higher link utilization and does not induce significant fluctuations in the transmission rate. However, AIAD's responsiveness is poor upon sudden congestion, due to its additive decrease policy. More precisely, invoking an additive decrease in response to severe congestion, the sender will not throttle aggressively enough and the loss will persist. In such case, AIMD responds more aggressively in order to confine packet loss. The integrated rate control algorithm combines the strengths of AIAD and AIMD reacting gently to random loss and more aggressively to congestive loss, adapting effectively to the dynamics of the network.

AIAMD is able to differentiate congestive and non-congestive loss by maintaining a history of the receiving rates throughout the connection. Observations of the network dynamics and event losses are frequently assumed within a time period of an *epoch*. We consider an epoch as the time period between two observed loss events (i.e., during an epoch the transmission rate evolves uninterrupted). The receiving rates at the end of each epoch are useful, since they compose a good predictor for the congestion

[1] Random loss is not uncommon in wireless/mobile systems, due to temporary loss of lower-level connectivity (e.g., fading, handoffs).

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

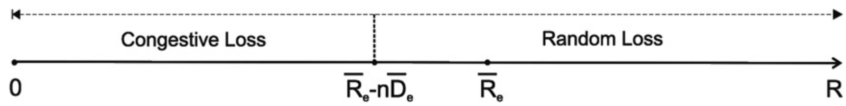

Fig. 2. AIAMD loss differentiation.

state for the following epochs. We define the state variable $\mathscr{R}$, which is the receiving rate updated at the end of each epoch. DVRC monitors the progress of the receiving rate, and subsequently keeps track of the profile of $\mathscr{R}$ by maintaining two variables: the moving average $\overline{\mathscr{R}}$ of the receiving rate and the average deviation $\overline{D}$ of the receiving rate. We also use a constant n which gives a pre-determined weight to $\overline{D}$. Based on the progress of $\overline{\mathscr{R}}$ and $\overline{D}, \overline{\mathscr{R}} - n\overline{D}$ can be used as a threshold between congestion and random loss. On the occurrence of packet loss, the instant value of $\overline{\mathscr{R}} - n\overline{D}$ is compared to the currently measured rate, determining the appropriate recovery strategy. A receiving rate within the $\overline{\mathscr{R}} - n\overline{D}$ bound indicates a small deviation from the average rate $\overline{\mathscr{R}}$, allowing the interpretation of a temporary loss. In this case, AIAMD invokes an additive decrease in the transmission rate. On the other hand, a measured rate below $\overline{\mathscr{R}} - n\overline{D}$ indicates a considerable rate decrease due to a congestion incident. Subsequently, the protocol infers congestion and triggers a multiplicative decrease. Fig. 2 illustrates the loss differentiation applied by AIAMD, where $\overline{\mathscr{R}}_e$ and $\overline{D}_e$ denote the average receiving rate and the average deviation of that rate at the end of eth epoch, respectively.

With respect to Eqs. (5)–(9), the transmission rate for DVRC is determined based on the following algorithm:

$$R(t) = \begin{cases} \alpha_I + R(t-1) & \text{if } y(t) = 0 \\ \alpha_D + R(t-1) & \text{if } y(t) = 1 \text{ and } R(t-1) \geqslant \overline{\mathscr{R}} - n\overline{D} \\ \beta_D R(t-1) & \text{if } y(t) = 1 \text{ and } R(t-1) < \overline{\mathscr{R}} - n\overline{D} \end{cases} \quad (11)$$

We adopt the conventional AIAD/AIMD parameters, $\alpha_D = 1$, $\alpha_D = -1$ and $\beta_D = 0.5$. We have also set n experimentally to 1.5; however, it can be adjusted differently in order to modify the transient behavior of the control. AIAMD tries to preserve AIAD's property of gentle variations in the transmission rate for random loss, enabling the desired smoothness for video streaming applications. At the same time, AIAMD reacts more aggressively in response to the reduction of network resources or the advent of new connections. Therefore, the protocol prevents multimedia applications from significant damage during congestion.

In order to provide more insight to the behavior of AIAMD control, Fig. 3 shows the variation in the receiving rate and in the $\overline{\mathscr{R}} - n\overline{D}$ bound. In this simulation, an AIAMD flow shares a 1 Mbps bottleneck link with one TCP flow. The AIAMD flow experiences infrequent congestion events where the measured rate falls below $\overline{\mathscr{R}} - n\overline{D}$. At the same time, simulated random loss with 3% packet error rate enforces several loss events which are accurately interpreted as error-induced, since the measured rate exceeds the current value of $\overline{\mathscr{R}} - n\overline{D}$.

3.3. Delivery of video layers

DVRC is optimized for layered video streaming. The server encodes raw video into k cumulative layers using a layered coder: layer 1 is the base layer and layer k is the least important enhancement layer. The layer rates are given by l_i, $i = 1,2,\ldots,k$. Let c_j denote the cumulative layer rate up to layer j, i.e., $c_j = \sum_{i=1}^{j} l_i,\ i = 1,2,\ldots,k$ while $r_k = (c_1,c_2,\ldots,c_k)$ denotes the rate vector of the cumulative layer rates. With the cumulative subscription policy, this discrete set offers all possible video rates that a client could receive, and the maximum number of layers that can be delivered to a receiver with an expected bandwidth B is $K = \max\{j : c_j \leqslant B, c_j \in r_k\}$. DVRC can

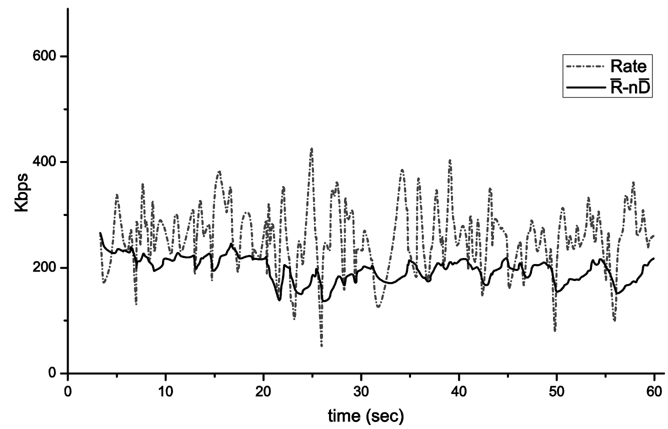

Fig. 3. Receiving rate and $\overline{\mathscr{R}} - n\overline{\mathscr{D}}$ variation.

effectively interact with static layer-rate allocation schemes, as well as with dynamic source allocation, i.e., dynamically allocating the layer rates.

DVRC enables a closed-loop control between server and client, where the receiver detects the state of congestion, determines the proper transmission rate and eventually opts for the optimal number of layers that should be delivered according to this rate. More precisely, the protocol performs the following control loop in order to actuate the video stream adaptation:

(1) The receiver acquires the rate vector r_k[2] and monitors the progress of reception statistics $\overline{\mathscr{R}}$ and $\overline{D}$.
(2) It determines the subsequent transmission rate R, based on Eq. (11).
(3) It calculates K using $K = \max\{j : c_j \leqslant R(t),\ c_j \in r_k\}$.
(4) It generates a control packet attaching K to the *video layer* field.
(5) Upon receiving the control packet, the sender joins or leaves layers until the subscription level is K.

In order to explore the available bandwidth, transmission initiates with the lowest rate that can accommodate the base layer (i.e., $R(t_0) = c_1$). Although DVRC does not specify any particular coding algorithm in the application layer, the protocol can interact efficiently with coders of wide dynamic range and fine granularity in terms of rate control, such as *Fine Granularity Scalability* (FGS) (Li, 2001).

In addition, exploiting the field *frame type* in the DVRC header, the protocol can optionally deliver layered video using a prioritized transmission. According to this scheme, the sender assigns different priorities to the layers according to their levels of importance, and during congestion the routers discard low priority packets first, securing the successful delivery of the most important video frames. In the simulation experiments of this paper, DVRC does not utilize any type of prioritization.

[2] It is not necessary for the receiver to periodically acquire the rate vector, when layer-rates have been allocated statically. In this case, the server is only required to advertise the rate vector to the client at the beginning of the connection.

4. Evaluation methodology

4.1. Experimental settings

The evaluation plan was implemented on the NS-2 Network Simulator. In order to assess the performance of DVRC, we implemented an experimental MPEG-4 video streaming server which is capable of distributing layered video. The server transmits video-data by dividing each frame into fixed size packets. The traffic generated closely matches the statistical characteristics of an original MPEG-4 video trace. MPEG-4 coding standard is based on *I*, *P* and *B* frames. The compression initiates by encoding a single I frame, followed by a group of *P* and *B* frames. *P* frames carry the signal difference between the previous frame and motion vectors, while *B* frames are interpolated; the encoding is based on the previous and the next frame. The model developed is based on *Transform Expand Sample* (TES). We used three separate *TES* models for *I*, *P*, and *B* frames, respectively. The resulting MPEG-4 stream is generated by interleaving data obtained by the three models. In the case of DVRC, the protocol header (Fig. 1) is encapsulated in each segment during the packetization process. Since the number of layers is quite limited in a layered coder in practice, we maintain a base plus four enhancement layers (i.e., $k = 5$) that further refine video quality. The experimental layered adaptation scheme is illustrated in Fig. 4.

We performed extensive evaluations of DVRC along with TFRC and the measurement-based TCP Westwood+(TCPW+) (Westwood NS-2 Implementation). TFRC, in particular, is designed for efficiency on media delivery over a wide range of network and session dynamics. We also included in our experiments the combined approach of RTCP over UDP. In this case, the receiver utilizes *RTCP Receiver Reports* in order to provide feedback to the sender at periodic intervals and whenever congestion is detected. When the MPEG-4 server transmits over TFRC or TCPW+, the video is streamed at the optimal quality (i.e., adaptation is disabled). As a result, we evaluate the transmission rate control performed by each protocol at the transport layer. Since DVRC enables rate control based on layer distribution, the streaming server performs layered adaptation based on DVRC's feedback loop. Likewise, RTCP/UDP is tested with the specific layered adaptation scheme.

The experiments were conducted based on realistic scenarios which address the heterogeneity of the Internet. Initially, we enabled simulations on a single-bottleneck *dumbbell* topology with a round-trip link delay of 70 ms in order to evaluate single DVRC flow performance, as well as inter-protocol friendliness. The bottleneck link is shared by competing MPEG and TCP connections. The link capacities are configured depending on the experiment performed. We also used a network topology (Fig. 5a) which includes cross traffic, wireless links and varying RTTs. The router *R*1 is the bottleneck for MPEG traffic, while the router *R*2 is another bottleneck for competing MPEG and cross traffic. Cross traffic includes a number of FTP connections over TCP Reno. The propagation delays of the access links from all the source nodes, as well as the links to the TCP sink nodes range from 5 ms to 15 ms, while the corresponding bandwidth capacities range from 2 Mbps to 10 Mbps. The capacity of all access links to the MPEG sink nodes is set to 2 Mbps. By randomizing RTTs, we avoid synchronization effects. In addition, we used the topology in Fig. 5b which employs the characteristics of the previous topology along with FTP/Reno flows both in the forward and backward direction. Random burst UDP traffic with the Pareto distribution has been also introduced to the forward direction. Finally, we enabled simulations on a multi-hop topology where MPEG flows traverse through *N* hops, as shown in Fig. 5c. Apart from MPEG traffic, each hop is used by two TCP connections, one in the forward and one in the backward direction of the main traffic. The capacity *bw* of the links between the gateways is configured depending on the experiment.

In cross- and reverse-traffic topologies, we used a link error model in the access links to the MPEG sink nodes. Error models were configured on both directions of the link traffic. Paxson (1999) provides evidence for correlated packet loss. In order to model temporally correlated loss observed in a fading wireless channel, we used the correlated *Bernoulli* model which characterizes the loss pattern as a Bernoulli distribution of loss rounds. Each round consists of a group of consecutive packets, the length of which is approximated by a geometric distribution. The first packet in the round is lost with probability *p*. Every other packet is lost with probability, if the previous packet has not been lost; otherwise, the loss probability is *q*. In our simulations we adjusted $p = 0.01$ and $q = 0.15$.

In all topologies, the routers are drop-tail with buffer size adjusted in accordance with the *bandwidth-delay* product. We set the packet size to 1000 bytes for all system flows (with the exception of UDP flows in the reverse-traffic topology which have 250 bytes packet size) and the maximum congestion window to 64 kB for all TCP connections. Diverse randomization seeds were used in order to reduce simulation dynamics. All the results are collected after 2 s in order to avoid the skew introduced by the startup effect.

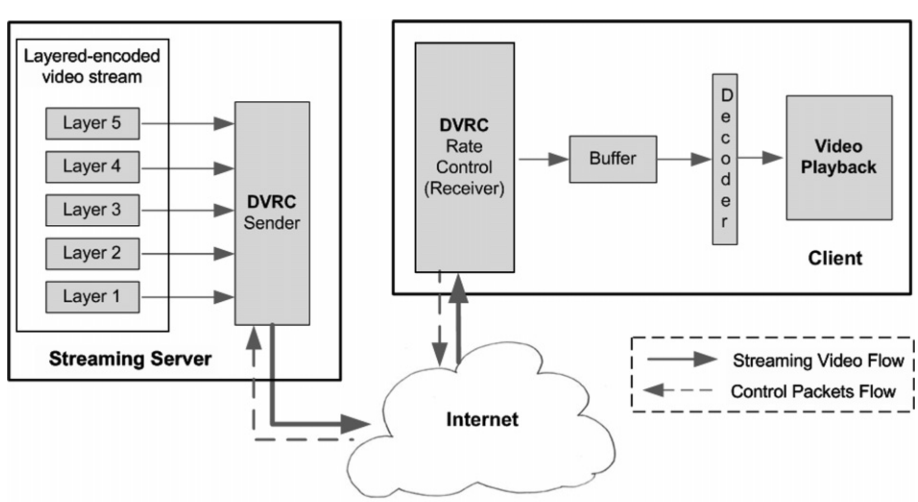

Fig. 4. Experimental layered adaptation scheme under DVRC.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

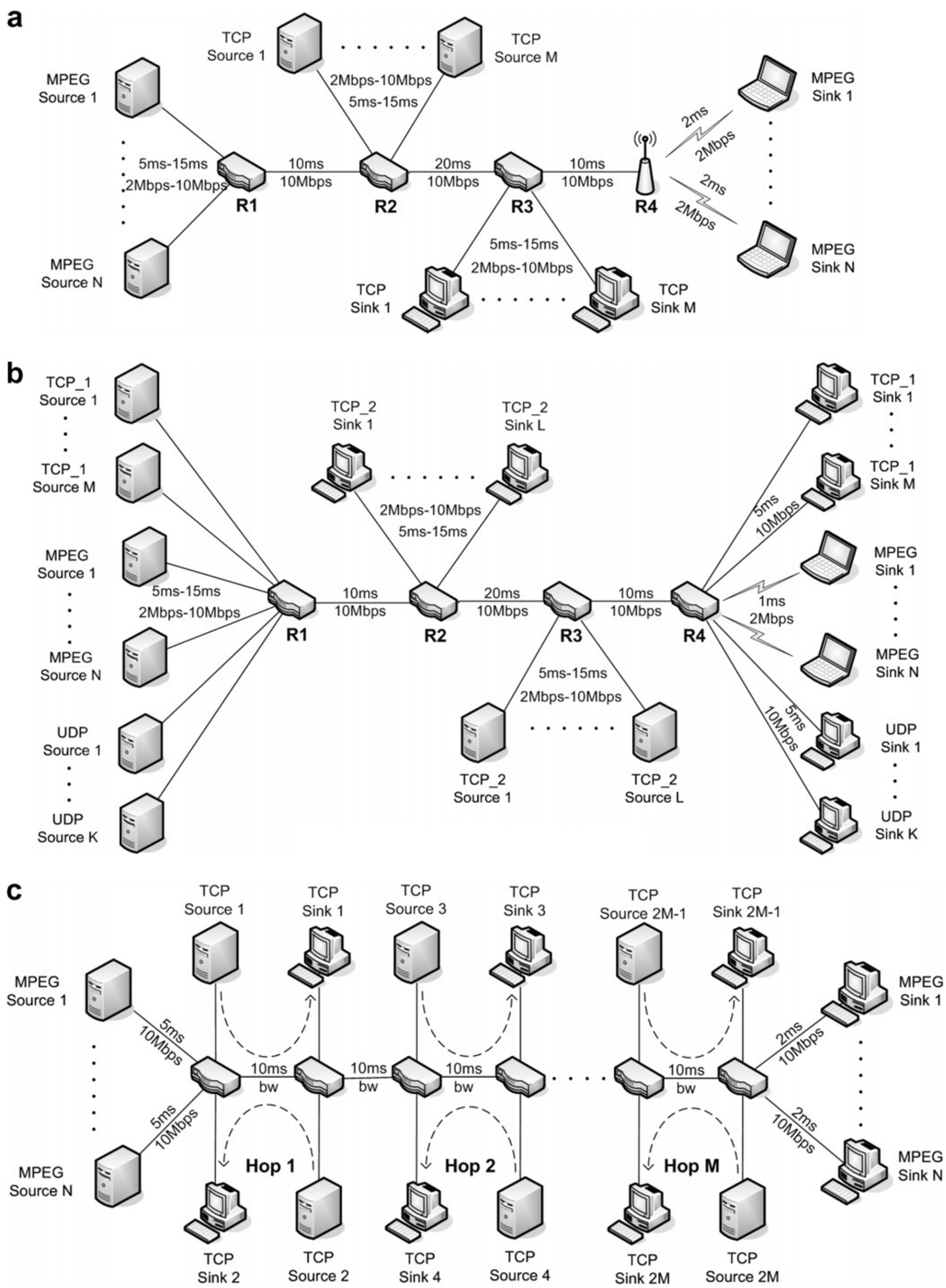

Fig. 5. Simulation topologies: (a) cross-traffic topology, (b) reverse-traffic topology and (c) multi-hop topology.

4.2. Measuring performance

We hereby refer to the performance metrics supported by our simulation model. Since both topologies include competing MPEG and FTP connections, our performance metrics are applied separately to the MPEG and FTP traffic. *Goodput* is used to measure the overall system efficiency in bandwidth utilization and is defined as

$$\text{Goodput} = \frac{\text{Original data}}{\text{Connection time}}$$

where *Original data* is the number of bytes delivered to the high-level protocol at the receiver (i.e., excluding retransmitted packets and overhead) and *Connection time* is the amount of time required for data delivery. Following the metric in Yang et al. (2001), we use *Coefficient of Variation* (*CoV*) in order to gauge the throughput smoothness experienced by flow *i*:

$$\text{CoV}_i = \frac{\sqrt{E_t\{\text{throughput}_i^2(t)\} - E_t\{\text{throughput}(t)\}^2}}{E_t\{\text{throughput}_i(t)\}}$$

where $E_t\{\cdot\}$ denotes the computation of the mean along time. For the *i*th flow, $throughput_i(t)$ is sampled at a time scale of a few RTTs throughout the entire connection. In our simulations, the sampling period is set to 150 ms. The frequency of throughput samples is dependent on RTT, as well as the type of network traffic. Generally, a time period of 150 ms composes a plausible candidate for a minimum interval over which throughput variations would begin to be

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

noticeable to multimedia users. Lower sampling periods can be considered for network paths with very low RTT, while higher intervals may conceal an amount of throughput variation. For a system with multiple flows, we present the average CoV of all flows.

Long-term fairness is measured by the *Fairness Index*, derived from the formula given in Chiu and Jain (1989), and defined as

$$\text{Fairness index} = \left(\sum_{i=1}^{n} \text{throughput}_i\right)^2 \Big/ \left(n \sum_{i=1}^{n} \text{throughput}_i^2\right)$$

where $Throughput_i$ is the throughput of the ith flow and n is the total number of flows. As a supplementary fairness metric, we use *Worst-Case Fairness*:

$$\text{Worst-case fairness} = \frac{\min_{1 \leqslant i \leqslant n}(\text{throughput}_i)}{\max_{1 \leqslant i \leqslant n}(\text{throughput}_i)}$$

in order to conduct a *worst-case* analysis and provide a tight bound on fairness. Similarly to *Fairness Index*, the range of *Worst-Case Fairness* is in $[0,1]$ with 1 representing the absolute fairness. Inter-protocol friendliness measurements were conducted based on *Normalized Throughput* which is the ratio of the average throughput received by each flow over the bandwidth fair-share on each case. *Normalized Throughput* also ranges in $[0,1]$ and is defined as

$$\text{Normalized throughput} = \frac{\text{Throughput}}{\text{Bandwidth Fair\_Share}} = \frac{\text{Throughput}}{\frac{B}{n}} = \frac{n \cdot \text{Throughput}}{B}$$

where B is the bottleneck capacity.

The task of specifying the effects of network QoS parameters on video quality is challenging. Transmission rate fluctuations, increased delays, delay jitter and packet loss commonly deteriorate the perceived quality or fidelity of the received video content. We note that these network QoS parameters do not affect quality in an independent manner; they rather act in combination or cumulatively, and ultimately, only this joint effect is detected by the end-user. Delay jitter composes a critical factor in the performance of video delivery. Let $D(i,j)$ denote the value of packet spacing at the receiver compared with packet spacing at the sender for a pair of packets i and j. $D(i,j)$ is represented as

$$D(i,j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i) \tag{12}$$

where S_i, S_j, R_i and R_j denote the sending and receiving times for packets i and j, respectively. In the absence of jitter, the spacings will be the same and $D(i,j)$ will be zero. Delay jitter is calculated continuously as a weighted average of the observed values of $D(i,j)$:

$$J(i,j) = \frac{15}{16} J(i,j) + \frac{1}{16} |D(i,j)| \tag{13}$$

Based on Papadimitriou et al. (2005), we use a metric for the performance evaluation on video delivery, called *Video Delivery Index*, which captures the joint effect of jitter and packet loss on perceptual quality. The metric monitors packet inter-arrival times and distinguishes the packets that can be effectively used by the client application (i.e., without causing interruptions) from delayed packets according to a configurable packet inter-arrival threshold. The proportion of delayed packets is denoted as *Delayed Packets Rate*. *Video Delivery Index* is defined as the ratio of the number of *jitter_free* packets over the total number of packets sent by the application:

$$\text{Video delivery index} = \frac{\#\ \text{jitter\_free packets}}{\#\text{sent packets}} \leqslant 1$$

According to the streaming video guidelines, playback quality is notably degraded when delay variation exceeds 75 ms. Buffering can eliminate the effects of delay variation by smoothing out jitter; however, additional delays are incurred to the video playback. Furthermore, buffering exhibits certain limitations, such as application delay tolerance and buffer memory constraints. Along these lines, we adjusted the packet inter-arrival threshold at 75 ms, which specifies the point where delay variation becomes perceptible and possibly disturbing. Since MPEG traffic is sensitive to packet drops, we additionally define *Packet Drop Rate*, as the ratio of the number of lost packets over the number of packets sent by the application. For a system with multiple flows, we present the average *Video Performance Index* of all MPEG flows.

5. Performance evaluation

In the sequel, we demonstrate conclusive performance studies based on extensive simulation results. More precisely, we evaluate the efficiency of DVRC in terms of video delivery and fairness, and we further examine DVRC's friendliness with coexisting TCP traffic.

5.1. Single DVRC flow

Initially, we evaluate DVRC's efficiency in terms of smooth video delivery. We simulated a single MPEG flow over DVRC competing with two FTP flows of TCP Reno. We enabled the simulations on the dumbbell topology, where all link capacities are set to 1 Mbps. Fig. 6 illustrates the variation in the sending rates of the three flows throughout the entire experiment. The transmission rate is averaged over 150 ms intervals, each one including approximately two RTTs for the specific topology. In the case of DVRC, the fluctuations in the sending rate are of low magnitude in comparison with the highly variable rates of TCP congestion control. Apart from the oscillations, Fig. 6 depicts occasional abrupt rate reductions induced by the TCP sources in response to coarse timeouts, which inevitably cause interruptions on data delivery. Generally, TCP's strategy of rapid backward and graduated upward adjustments hurts the performance of TCP-based applications, and especially damages delay-sensitive traffic. On the contrary, DVRC's gentle rate adjustments result in a smoothed flow that optimizes video delivery and playback on the receiver. Only in a few situations, such as the occasional bursts of a corporate TCP flow, AIAMD invokes a multiplicative decrease in order to confine packet loss for the MPEG flow. Generally, DVRC remains relatively immunized from the disturbances caused by the interfering TCP connections.

In order to quantify the smoothness observed by the end-user, we measured CoV for the single DVRC connection, which is $CoV_{DVRC} = 0.1854$. In addition, we carried out the same experiment simulating the MPEG transfer with TFRC. The corresponding CoV measurement is $CoV_{TFRC} = 0.2229$. A lower CoV expresses a lower variation in throughput rates, and consequently higher smoothness; hence, DVRC appears to be smoother than TFRC. In the remainder of Section 5, we provide more extensive performance studies, where we compare DVRC with TRFC, TCP Westwood+, and RTCP/UDP in dynamic networks with high link-multiplexing.

The performance on video delivery for the DVRC connection is depicted in Fig. 7. The DVRC flow shares the bottleneck link with two FTP connections, which are expected to cause notable disturbances on perceptual video quality. Nevertheless, delay jitter does not exceed the frustrating limit of 75 ms, while a high proportion of inter-arrival times are monitored with values below 50 ms, where jitter effects are not noticeable by the majority of multimedia users. Such a performance does not necessitate the use of deep playback buffers in order to ameliorate the effect of jitter. Only in conditions of scarce bandwidth, a playback buffer may notably improve the delivered video quality and prevent potential interruptions on video playback.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

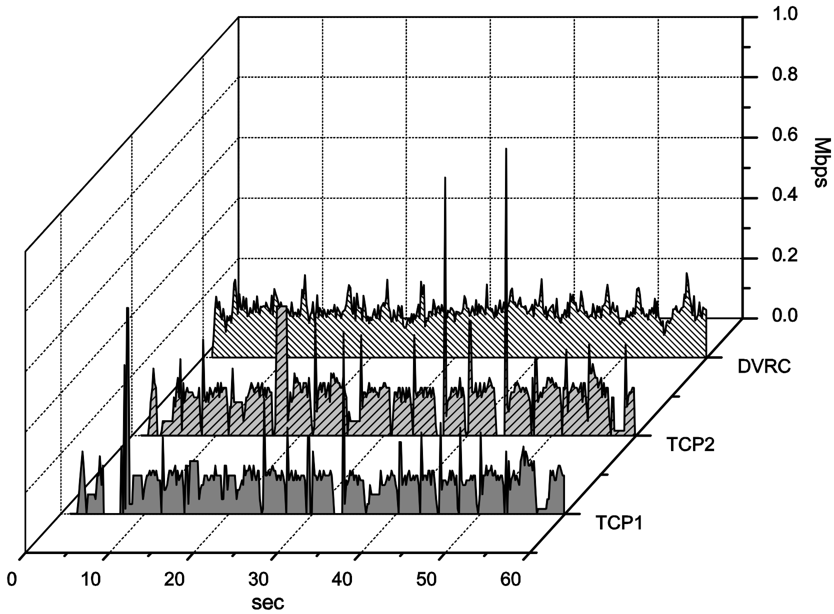

Fig. 6. Sending rates of competing DVRC and TCP flows.

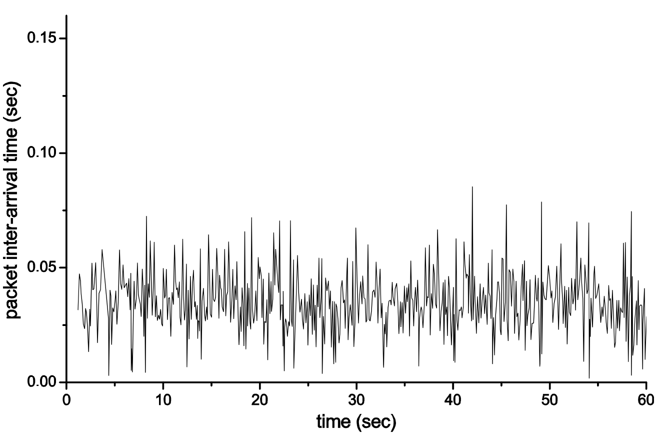

Fig. 7. DVRC delay jitter.

5.2. DVRC performance with multiple flows

Departing from the performance study of a single DVRC flow, we carried out a series of experiments in order to assess the performance of our approach versus TCP-friendly traffic. We simulated a wide range of MPEG flows (1–50) of (i) DVRC, (ii) TFRC, and (iii) TCP Westwood+ competing with 10 FTP connections of TCP Reno, successively. The corresponding experiments were conducted on the cross-traffic topology. We measured *Goodput*, *Video Delivery Index*, *Fairness Index* and *Worst-Case Fairness*, and we additionally demonstrate statistics from delayed and lost packets, since both compose influencing factors for perceived video quality (Figs. 8–10). Furthermore, we present traces of the queue-length of router *R*2 (Fig. 11) in the presence of 40 MPEG flows.

According to Fig. 8a, DVRC yields effective bandwidth utilization, especially for high link-multiplexing. The protocol exploits its rate control algorithm reacting gently to the link errors across the last-hop wireless channel. Only congestion-induced loss enforces DVRC to significantly reduce its transmission rate in order to adapt to current network dynamics. Inline with DVRC, TFRC manages to utilize a high fraction of the available bandwidth. TCPW+ achieves remarkable goodput performance for relatively low link-multiplexing. However, at high contention (40–50 flows) the protocol does not achieve full utilization of the available bandwidth. Despite the improvements over the initial version of Westwood, TCP Westwood+'s algorithm occasionally underestimates the available resources, since the estimation filter is slow, needing time to converge to the available bandwidth. Apparently, TCPW+ would perform more efficiently in the presence of higher bandwidth, unfolding its potential.

Fig. 8b reveals that DVRC's transmission rates exhibit small fluctuations in average, achieving the desired smoothness required by most video streaming applications. The protocol adjusts the sending rate in response to the prevailing conditions, as well as congestion history, trying to preserve AIAD's property of graceful variation in the sending rate.

From the perspective of video delivery, DVRC exhibits remarkable efficiency, delivering smooth video which is slightly affected by contention (Fig. 9a). The protocol achieves the timely delivery of most packets, enabling stable playback quality without interruptions (Fig. 9c). In addition, the low packet drop rate (Fig. 9b) in conjunction with the relaxed packet loss requirements of streaming video justifies our choice not to integrate reliability to DVRC. AIAMD is therefore less susceptible to random packet loss and loss synchronization. Our simulations (including a more extensive set of experiments with a diverse number of layers, not presented here) demonstrate that DVRC can achieve satisfactory performance with a small number of layers (4–5 layers).

According to Fig. 9c, TFRC delivers a smoothed flow and eventually confines the short-term oscillations in the sending rate, inline with DVRC. However, Fig. 9b illustrates relatively increased packet losses for TFRC, which deteriorate the perceptual video quality (Fig. 9a). In dynamic environments with wireless errors, TFRC occasionally fails to obtain accurate estimates of the loss event rate, invoking an inappropriate equation-based recovery, since TFRC's throughput model, as expressed in Eq. (1), is sensitive to the packet loss rate. TCPW+ exhibits increased packet loss (Fig. 9b), while a considerable proportion of the packets that are not dropped, reach the recipient later than required (Fig. 9c). The combined effect degrades the playback quality of the video stream (Fig. 9a). However, we note that TCPW+, as a modification of TCP Reno, is not optimized for multimedia applications, justifying its inferior performance on media delivery.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

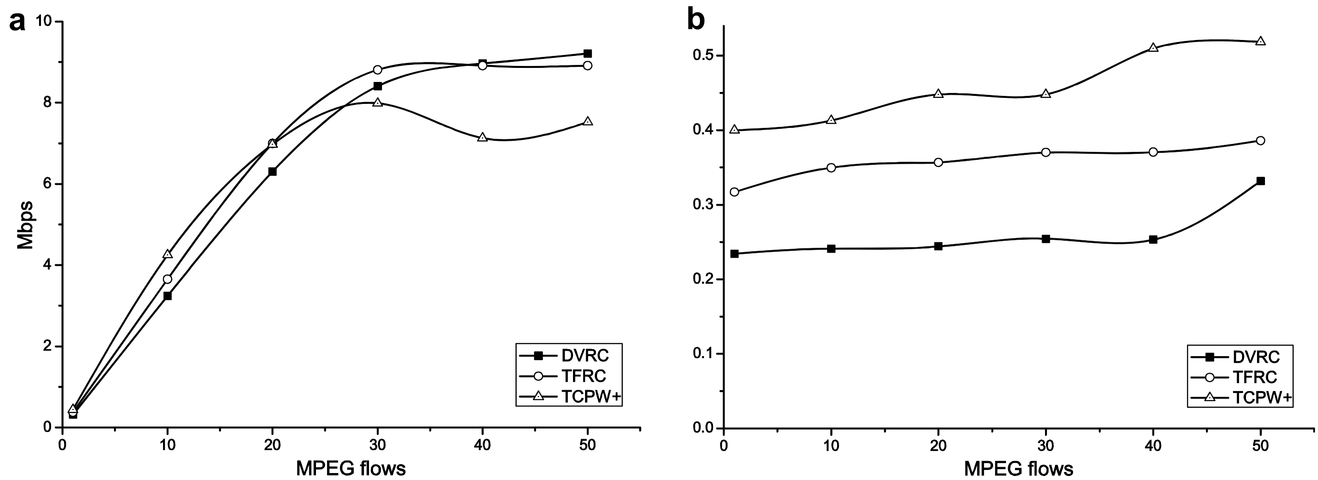

Fig. 8. Protocol performance: (a) goodput of MPEG flows and (b) CoV of MPEG flows.

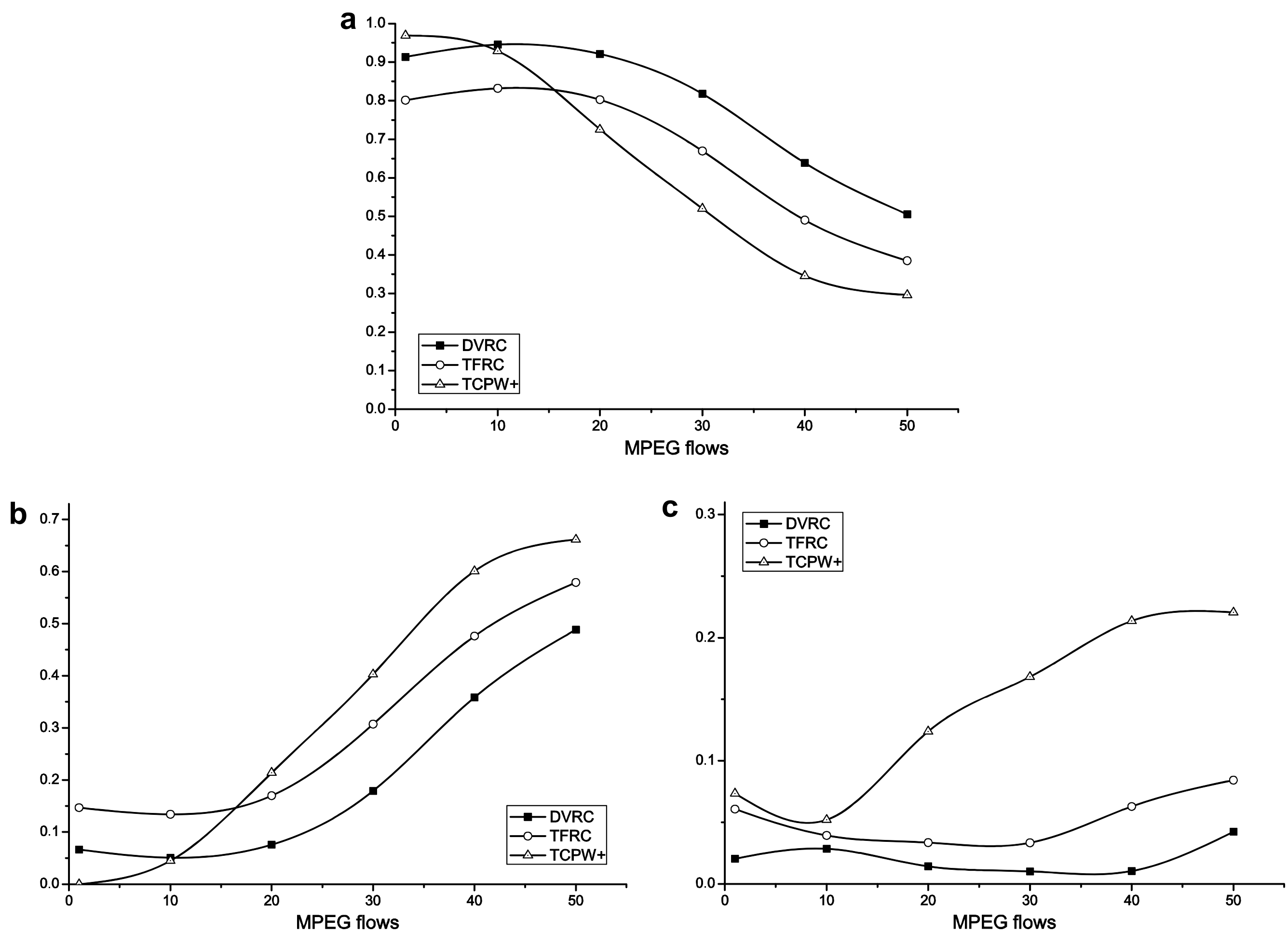

Fig. 9. Performance on video delivery: (a) average video delivery index, (b) packet drop rate and (c) delayed packets rate.

As shown in Fig. 10a, DVRC excels in bandwidths sharing, regardless of link-multiplexing. The AIMD-based responses during congestion enforce competing DVRC flows to converge to the fairness point. Similarly, TFRC and TCPW+ achieve relatively high levels of fairness, if assessed by the traditional *Fairness Index*. However, according to Fig. 10b the worst-case fairness for TFRC and TCPW+ fluctuates and gradually degrades, representing a notable difference between the minimum and maximum throughput rates achieved by both protocols. Therefore, in the case of TFRC and TCPW+, the system can be significantly unfair to a small fraction of flows. Generally, competing TFRC flows may have different throughput rates, since their observed loss event rates can differ. This observation is profound in the case of increased contention (Fig. 10b). Note that the oscillations in the curves of Fig. 10 are

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

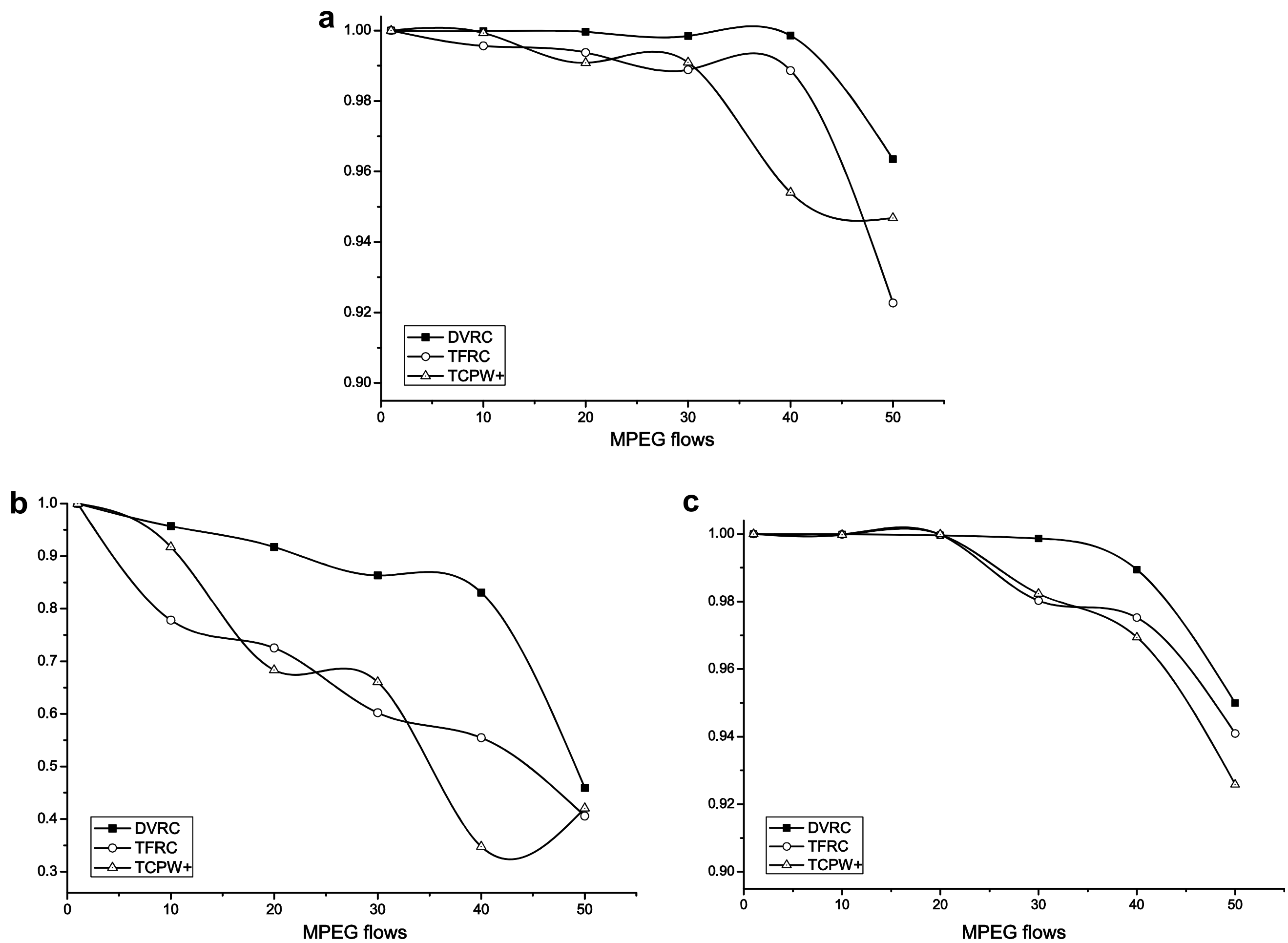

Fig. 10. Fairness: (a) fairness index, (b) worst-case fairness and (c) fairness index (no cross traffic and highly varying RTTs).

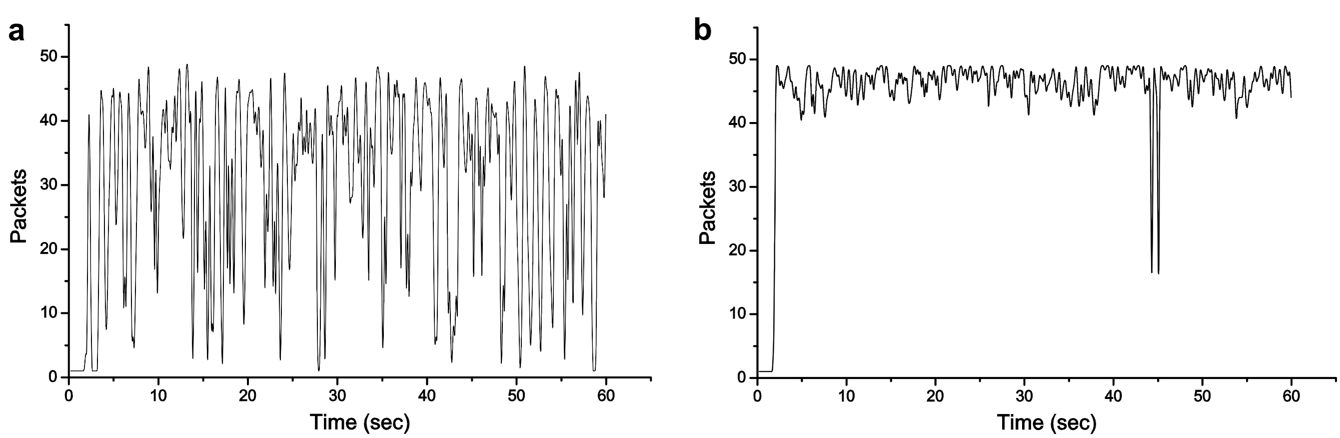

Fig. 11. Queue length of router $R2$ (40 flows): (a) DVRC and (b) UDP.

caused by the presence of random loss, which affects protocol behavior especially if the underlying congestion control is not able to detect the nature of loss (e.g., TFRC). Due to random errors, competing MPEG flows may experience different packet loss rates, affecting flow throughput and subsequently intra-protocol fairness.

Fig. 10c illustrates intra-protocol fairness results from a modified scenario, where the simulated topology includes only varying MPEG flows without cross TCP traffic and link errors. Furthermore, flow RTTs vary significantly, allowing the study of protocol behavior in such conditions. DVRC maintains adequate levels of intra-protocol fairness with a perceptible degradation when MPEG flows increase beyond 40. TFRC exhibits the most notable reduction in its *Fairness Index* compared to Fig. 10a, since TFRC flow throughputs are inversely proportional to RTT. Due to the RTT variation, individual throughput rates differ degrading fairness.

According to Fig. 11a, DVRC results in variable buffer conditions (due to the increased contention which inevitably leads to congestion), while UDP alone (i.e., without any supporting end-to-end protocol such as DVRC or RTCP) results in rapidly growing queues and eventually in buffer overflows (Fig. 11b). In the case of DVRC, the buffer is utilized efficiently and the oscillations are apparently of smaller magnitude than in conventional AIMD protocols. Furthermore, Fig. 11a reflects the overall behavior of DVRC: there are gentle reductions in the queue length (additive decrease), while DVRC occasionally enforces a persistent buffer draining phase by promptly responding to congestion (multiplicative decrease).

5.3. DVRC performance with heterogeneous networks and reverse traffic

We enabled additional simulations on the reverse-traffic topology, which allowed us to draw further conclusions on DVRC efficiency at conditions of increased contention with various types of traffic, including TCP reverse flows and UDP burst traffic. We specifically simulated 1–50 MPEG flows competing with 15 TCP (5 in the forward and 10 in the reverse direction) and 10 UDP flows. The peak sending rates for UDP flows range from 64 to 256 kbps, each one occupying up to 2.5% of the bottleneck bandwidth. The MPEG flows were simulated with DVRC, TFRC, as well as RTCP on top of UDP. Recall that DVRC and RTCP/UDP utilize layered adaptation, while TFRC transfers MPEG streams at optimal quality.

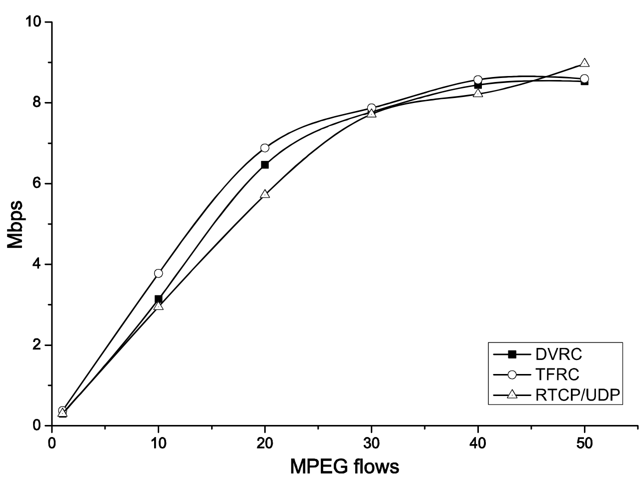

Fig. 12. Goodput of MPEG flows.

Fig. 12 shows that DVRC and TFRC achieve satisfactory bandwidth utilization, although the available resources are relatively limited due to the presence of interfering TCP and UDP flows. Furthermore, DVRC is not affected by reverse traffic, achieving high goodput rates even for high link-multiplexing. RTCP/UDP yields inferior performance in terms of goodput compared with DVRC and TFRC. Even if RTCP could compare more favorably with these protocols, its role is still limited: it simply provides information to the application allowing the source to reduce its transmission rate on the occurrence of congestion. It does not implement congestion control on top of UDP, as DVRC does. Hence, the task of rate adaptation is delegated to the application itself. The efficiency of RTCP/UDP is therefore highly dependent on the application-level rate control. Allowing the application to directly adapt the outgoing video flow may cause implications on corporate TCP flows, since TCP friendliness may not be attained. The inherent difficulty and risks of application-level congestion control make it less attractive, even for multimedia applications. Recent studies discourage such solutions and propose modifications or new mechanisms at the transport layer (e.g., Kohler et al., 2006). On the contrary, DVRC provides the exact level of stream adaptation based on the subsequent transmission rate, as determined by the AIAMD algorithm.

Fig. 13a illustrates the remarkable efficiency of DVRC from the perspective of video delivery. Inline with the corresponding results of previous section (Fig. 9), the protocol delivers video of acceptable quality regardless of link-multiplexing. DVRC's interaction with the layered scheme sustains relatively low packet loss, delivering a large amount of the video-data sent by the application without any retransmission effort (Fig. 13b). Apart from packet loss statistics, *Video Delivery Index*, as depicted in Fig. 13, reflects DVRC's smooth video delivery in a wide range of network dynamics. A combined overview of Figs. 12 and 13 reveals that competing flows in the forward and backward direction do not cause notable implications on DVRC efficiency and perceptual video quality. The rest of the protocols exhibit a noticeable degree of performance degradation (Fig. 13), which becomes more evident as contention increases. Apparently, packet loss composes a limiting factor for TFRC. The protocol yields inferior performance, since it is not able to respond rapidly to the onset of congestion, decreasing its sending rate gently and allowing a considerable amount of packet loss.

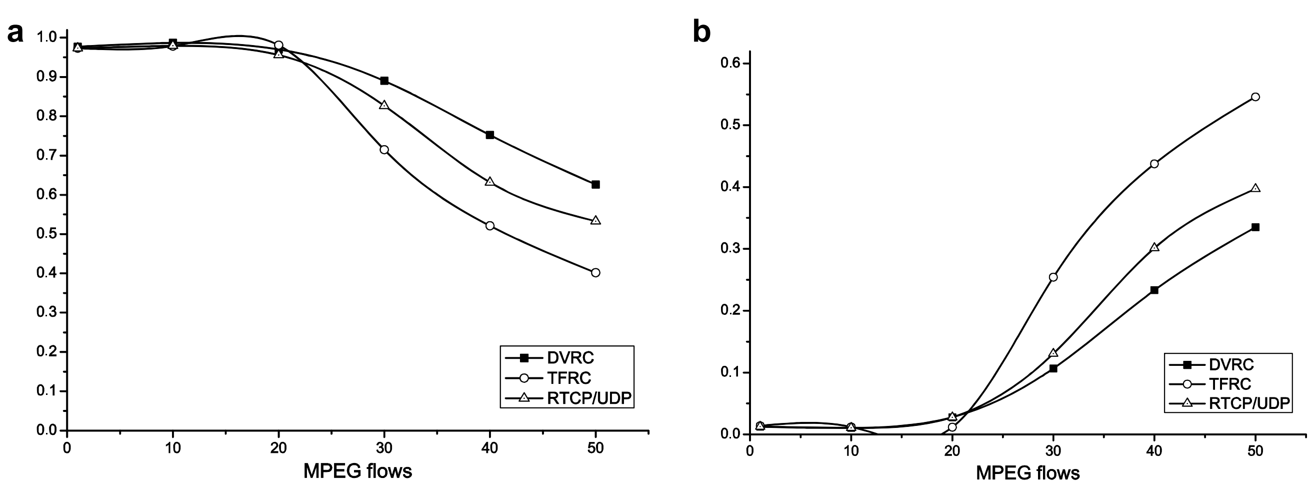

Fig. 13. Performance on video delivery: (a) average video delivery index and (b) packet drop rate.

In terms of intra-protocol fairness, DVRC and TFRC achieve a fair behavior among their flows (Fig. 14). Fig. 14b shows relatively small deviations between the minimum and maximum throughput rate of the DVRC flows. On the contrary, RTCP does not enable UDP to overcome its well-known fairness issues, as the unresponsive protocol does not employ any mechanism to converge to fairness. Furthermore, the presence of application-level rate control in collaboration with RTCP can hardly provide guarantees for fairness. As already mentioned, the observed oscillations in the curves of Fig. 14 are the effects of random loss, which causes a perceptible throughput imbalance among the competing MPEG flows.

5.4. DVRC friendliness

We provide further performance studies investigating the interactions of a diverse range of MPEG flows (1–50) with interfering TCP traffic. We demonstrate results from DVRC and TFRC competing with TCP Reno. We performed the simulations on the dumbbell topology configuring all link capacities at 10 Mbps. We demonstrate *Normalized Throughput* results from experiments with 20 and 40 TCP connections (Figs. 15 and 16).

Overcoming the *greedy* nature of UDP, DVRC monitors the prevailing network conditions and adjusts the video transmission rate maintaining friendliness with corporate flows. At periods of congestion, the integrated rate control algorithm behaves as AIMD (which is admittedly TCP-friendly), enforcing DVRC to relinquish a respectable amount of its resources. As depicted in Fig. 15, DVRC allows interfering TCP flows to obtain network resources close to the fair share of the link (i.e., *Normalized Throughput* of 1), which is critical in heterogeneous systems with multiple flows and different protocols. This observation is more profound in the situation of high link-multiplexing, where DVRC coexists fairly with TCP. In lightly loaded environments, the protocol allocates only the resources required to transmit all the video layers that can be accommodated to the receiver's bandwidth. In congested networks, DVRC simply relies on delivering the base layer, which corresponds to the minimum subscription level that service quality is acceptable. DVRC performs its task without implications on background traffic, preventing the potential of a congestive collapse.

On the other hand, Fig. 16 reveals a significant discrepancy between the throughput rates achieved by competing MPEG and TCP connections. Fig. 16 illustrates that TFRC flows behave more aggressively. TFRC sources achieve higher throughput rates than the link fair share with a consequent starvation of the TCP Reno sources. Competing TFRC and TCP flows may observe loss event rates that can be significantly different, especially if they have different sending rates. Rhee and Xu (2005) study analytically the long-term throughput imbalance between competing TFRC and TCP connections, reporting that the throughput difference can be further amplified, as long as coexisting TCP and TFRC flows experi-

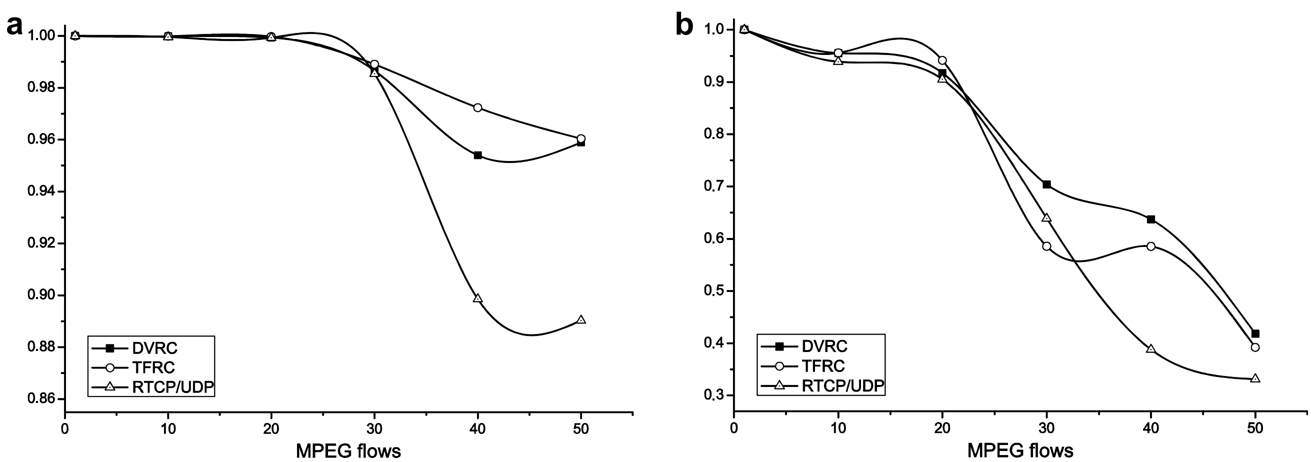

Fig. 14. Fairness: (a) fairness index and (b) worst-case fairness.

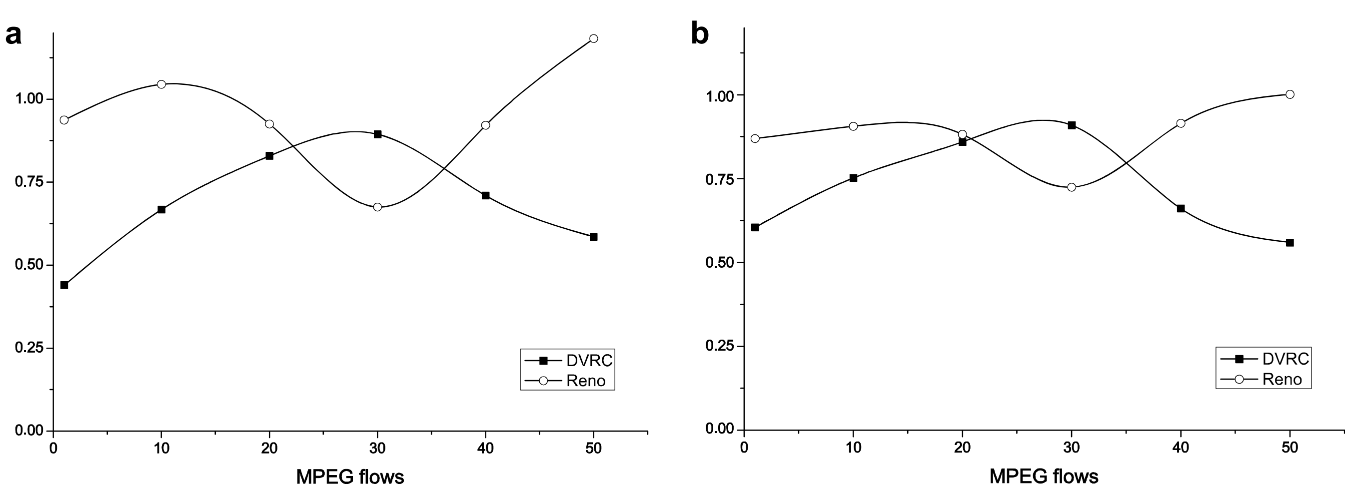

Fig. 15. Normalized throughput (DVRC): (a) 1–50 DVRC flows vs. 20 Reno flows and (b) 1–50 DVRC flows vs. 40 Reno flows.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

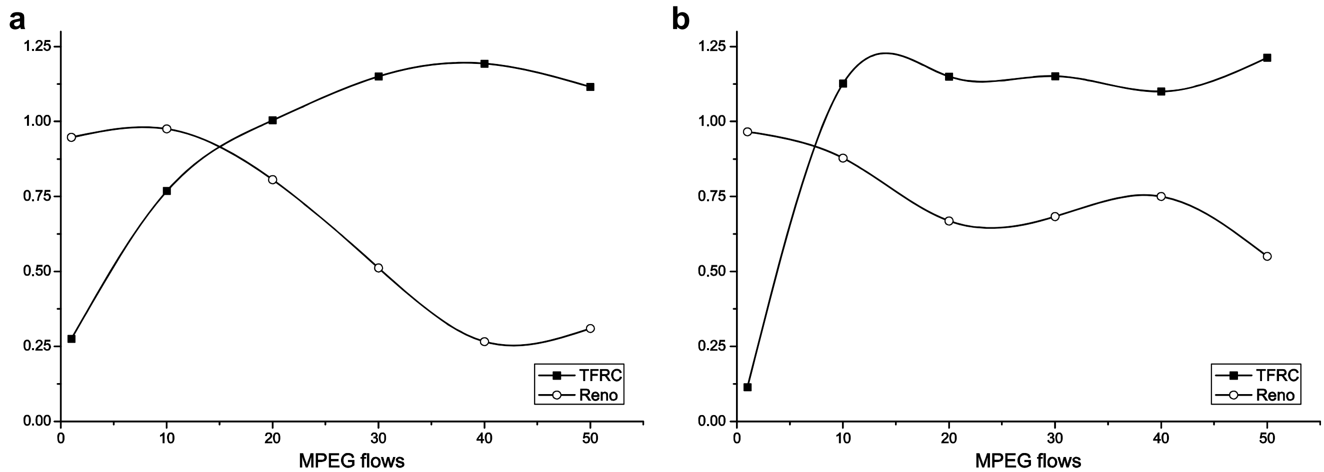

Fig. 16. Normalized throughput (TFRC): (a) 1–50 TFRC flows vs. 20 Reno flows and (b) 1–50 TFRC flows vs. 40 Reno flows.

ence different loss event rates. One fundamental factor for this throughput imbalance is the different RTO estimation schemes employed by TCP and TFRC. According to Rhee and Xu (2005), competing TFRC and TCP connections may end up having different RTO values depending on network delays.

5.5. DVRC performance and friendliness with multi-hop networks

We conclude our performance studies investigating DVRC's performance and friendliness in networks with multiple hops and cross TCP traffic. The simulations were conducted on the topology of Fig. 5c, where one or multiple MPEG flows from the main sources compete with MPEG cross traffic over TCP Reno along the network path. Initially, we simulated one MPEG flow of DVRC traversing through all hops ranging from 1 to 10. The capacity *bw* of the links between the routers was set to 1 Mbps. Fig. 17 illustrates the throughput achieved by the DVRC flow, as well as the average throughput of all forward TCP flows (ranging from 1 to 10 depending on the number of hops). The DVRC flow joins the network after 10 s and competes with TCP traffic for the remaining 190 s. However, the DVRC flow is still able to obtain its fair share, despite the presence of the TCP flows which have already allocated a considerable portion of the available bandwidth in the forward path during the first 10 s. Furthermore, according to Fig. 17 the average throughput of the TCP flows is close to the link fair share. Hence, DVRC does not cause any implications to the interfering flows, allowing them to allocate most of the remaining resources.

On the onset of congestion, both TCP and DVRC reduce their sending rate multiplicatively based on the principles of AIMD algorithm. Furthermore, DVRC estimates the timeout value similarly to TCP preventing potential misbehaviors when DVRC coexists with TCP flows. Rhee and Xu (2005) uncover that a main reason for the long-term throughput imbalance between competing TCP and TFRC flows is the different timeout estimation schemes between the two protocols.

We enabled additional simulations with diverse main MPEG connections (1–50) running over DVRC, TFRC and RTCP on top of UDP. DVRC and RTCP/UDP utilize layered adaptation, while TFRC transfers MPEG streams at optimal quality. The capacity of the links between the routers was set to 10 Mbps and the number of hops was fixed at 5. Thus, the main MPEG connections compete with five TCP flows, while there are also five TCP flows in the backward direction of the link traffic. Figs. 18–20 depict the corresponding goodput, video performance and intra-protocol fairness results. All these simulation results validate the efficiency and the remarkable performance of DVRC, inline with our previous performance studies.

Fig. 18 illustrates DVRC's efficiency in bandwidth utilization, as the DVRC flows traverse through multiple congested gateways. Even during high link-multiplexing, the MPEG flows allocate an adequate amount of the available resources facilitating the transportation of the streams. Besides the contention in the forward direction, DVRC manages to alleviate all the implications due to

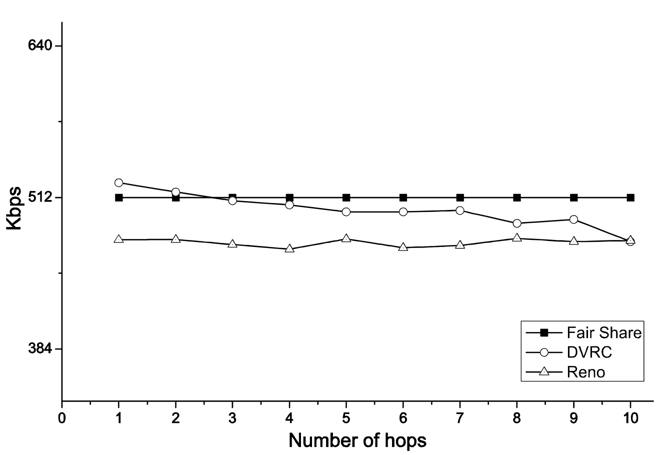

Fig. 17. DVRC throughput and average throughput of forward TCP flows.

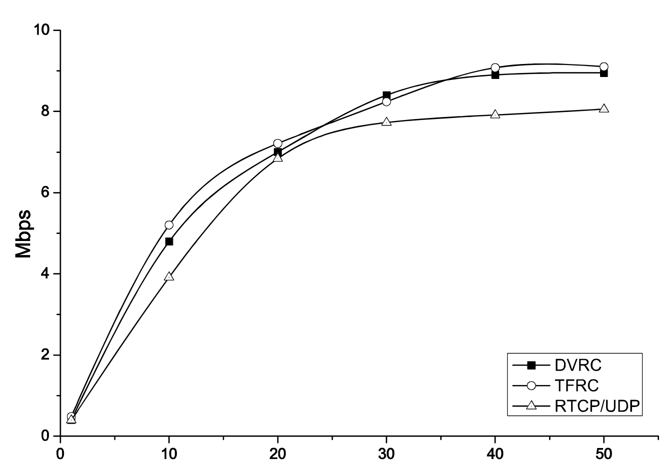

Fig. 18. Goodput of MPEG flows.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

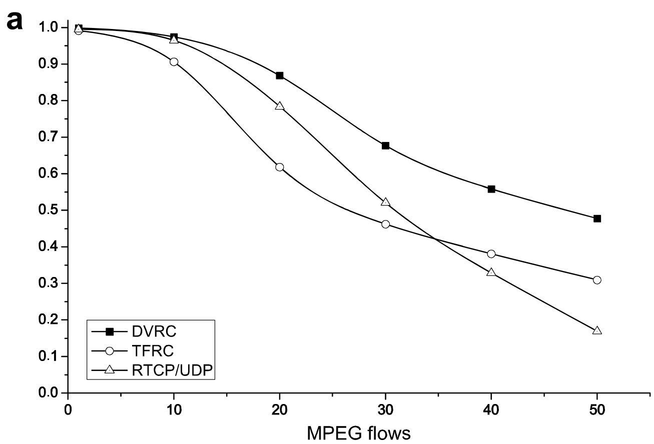

Fig. 19. Performance on video delivery: (a) average video delivery index and (b) packet drop rate.

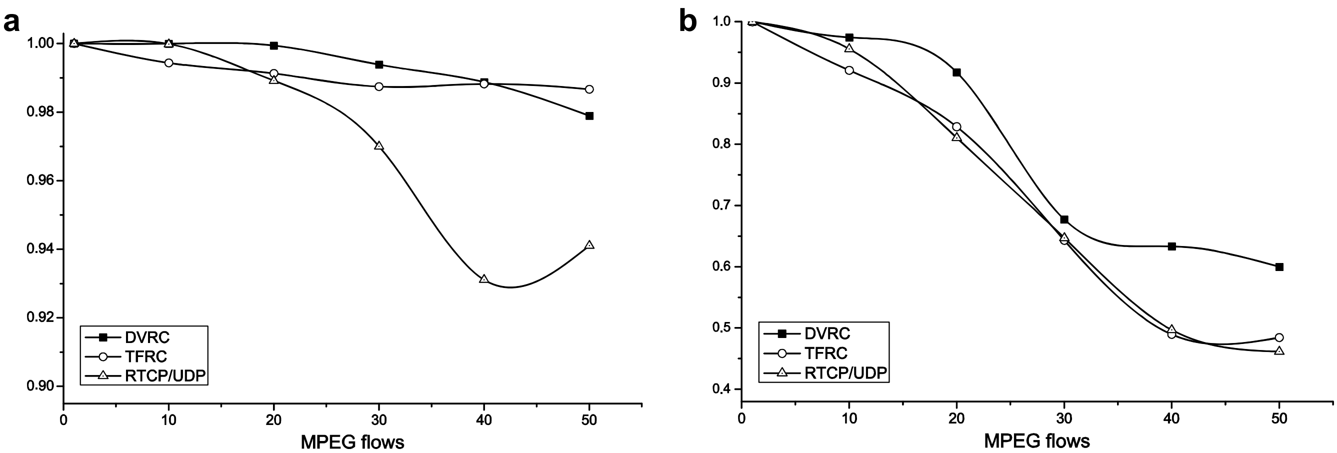

Fig. 20. Fairness: (a) fairness index and (b) worst-case fairness.

the considerable reverse TCP traffic that tends to interfere with the DVRC control packets flow. In the previous section as well as in Fig. 17, we showed that DVRC's remarkable goodput performance does not come at any cost: the corporate TCP flows always manage to obtain the fair share of the link. Furthermore, intra-protocol fairness is attained, as we show in the sequel. TFRC also achieves satisfactory bandwidth utilization (Fig. 18). However, the protocol's performance comes at the expense of inter-protocol friendliness, as we reported earlier in Fig. 16 and in accordance with the findings of Rhee and Xu (2005). The combined RCTP/UDP approach cannot effectively adapt the sending rate to the characteristics of the multi-hop network path and subsequently yields limited performance, as the level of contention increases.

According to Fig. 19a, DVRC exhibits superior performance on video delivery outperforming the rest of the protocols. DVRC interacts efficiently with the layered scheme adapting the video rate to the condition of the network path. The amount of packet loss is reduced (Fig. 19b), despite the implications caused by the multiple congested gateways. However, even in the presence of a scalable video scheme, the adaptivity is limited, since a portion of the video stream (i.e., base layer in the case of layered encoding) should be always delivered, so that video can be decoded the receiver. This limitation inevitably enforces a certain amount of packet loss in the presence of scarce bandwidth. The high *Video Delivery Index* also reflects DVRC's smooth delivery which enables a regular video flow without unnecessary delays. The *Video Delivery Index* for DVRC is slightly decreased compared to Fig. 13a, since video quality is perceptibly impaired due to the queuing delays experienced in the buffers of the congested routers. Both TFRC and RTCP/UDP suffer from increased packet loss which inevitably deteriorates the perceptual video quality (Fig. 19). TFRC, in particular, has a tendency to buffer overflow which is effectively captured in Fig. 19b, as its flows traverse through the gateways across the network path.

Fig. 20 illustrates DVRC's efficiency in terms of intra-protocol fairness. All competing DVRC flows achieve similar throughput rates, due to the presence of AIAMD control. Note that in this scenario the experienced losses are typically congestion-oriented and the employed AIAMD mechanism responds with multiplicative backward adjustments at congestion incidents. Admittedly (e.g., Chiu and Jain, 1989; Gorinsky and Vin, 2002), multiplicative decrease achieves fairness, and DVRC's fair behavior validates the accuracy of the applied loss differentiation. If the AIAMD algorithm had falsely invoked additive decrease during congestion, fairness would have been compromised, since additive decrease does not enforce fast convergence to the fairness point.

6. Conclusions

We have presented a rate control scheme for the adaptive delivery of layered video streams over the Internet. DVRC enables a more sophisticated loss-recovery strategy adjusted dynamically to the particular characteristics of the underlying network. The integrated rate control algorithm (AIAMD) combines the most desirable features of AIAD and AIMD, a graceful variation in the

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067

transmission rate and sensitivity to the onset of sudden congestion. Through extensive simulations, we identified significant gains for DVRC in highly-multiplexed dynamic networks. The protocol is less susceptible to random packet loss, effectively adapts to the vagaries of the network and eventually delivers smooth video. The corresponding performance studies reveal that the proposed rate control scheme compares very favorably with congestion control mechanisms that explicitly address delay-sensitive traffic, such as TFRC. DVRC has also demonstrated remarkable intra-protocol fairness, as well as friendliness with corporate TCP flows. DVRC is therefore a viable alternative to existing rate/congestion control schemes. In addition, DVRC can be integrated into existing protocols (e.g., DCCP), as a rate control option.

References

Bansal, D., Balakrishnan, H., April 2001. Binomial congestion control algorithms. In: Proceedings of the IEEE INFOCOM 2001, Anchorage, Alaska, USA.

Bansal, D., Balakrishnan, H., Floyd, S., Shenker, S., August 2001. Dynamic behavior of slowly-responsive congestion control algorithms. In: Proceedings of the ACM SIGCOMM 2001, San Diego, USA.

Boyce, J., Gaglianello, R., September 1998. Packet loss effects on MPEG video sent over the public Internet. In: Proceedings of the ACM Multimedia, Bristol, UK.

Brakmo, L., Peterson, L., 1995. TCP Vegas: end to end congestion avoidance on a global Internet. IEEE Journal on Selected Areas of Communications 13 (8), 1465–1480.

Cai, L., Shen, X., Pan, J., Mark, J., 2005. Performance analysis of TCP-friendly AIMD algorithms for multimedia applications. IEEE Transactions on Multimedia 7 (2), 339–355.

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

Feamster, N., Balakrishnan, H., April 2002. Packet loss recovery for streaming video. In: Proceedings of the 12th IEEE International Packet Video Workshop, Pittsburgh, USA.

Feamster, N., Bansal, D., Balakrishnan, H., April–May 2001. On the interactions between layered quality adaptation and congestion control for streaming video. In: Proceedings of the 11th IEEE International Packet Video Workshop, Korea.

Floyd, S., Fall, K., 1999. Promoting the use of end-to-end congestion control in the Internet. IEEE/ACM Transactions on Networking 7 (4), 458–472.

Floyd, S., Handley, M., Padhye, J., Widmer, J., August 2000. Equation-based congestion control for unicast applications. In: Proceedings of the ACM SIGCOMM 2000, Stockholm, Sweden.

Grieco, L., Mascolo, S., 2004. Adaptive rate control for streaming flows over the Internet. ACM Multimedia Systems 9 (6), 517–532.

Grieco, L., Mascolo, S., 2004. Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control. ACM Computer Communication Review 34 (2), 25–38.

Gorinsky, S., Vin, H., August 2002. Extended analysis of binary adjustment algorithms. Technical Report TR2002-39. Department of Computer Sciences, The University of Texas at Austin.

Hsieh, H., Kim, K., Zhu, Y., Sivakumar, R., September 2003. A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces. In: Proceedings of the ACM MOBICOM'03, San Diego, USA.

Jacobson, V., August 1988. Congestion avoidance and control. In: Proceedings of the ACM SIGCOMM'88, Stanford, USA.

Jin, C., Wei, D.X., Low, S.H., March 2004. TCP FAST: motivation, architecture, algorithms, performance. In: Proceedings of the IEEE INFOCOM 2004, Hong Kong, China.

Jin, S., Guo, L., Matta, I., Bestavros, A., 2003. A spectrum of TCP-friendly Window-based congestion control algorithms. IEEE/ACM Transactions on Networking 11 (3), 341–355.

Kohler, E., Handley, M., Floyd, S., September 2006. Designing DCCP: congestion control without reliability. In: Proceedings of the ACM SIGCOMM 2006, Piza, Italy.

Li, W., 2001. Overview of the fine granularity scalability in MPEG-4 video standard. IEEE Transactions on Circuits and Systems for Video Technology 11 (3), 301–317.

Liu, J., Zhang, Y., 2003. Adaptive video multicast over the Internet. IEEE Multimedia 10 (1), 22–33.

Martin, J., Nilsson, A., Rhee, I., 2003. Delay-based congestion avoidance for TCP. IEEE/ACM Transactions on Networking 11 (3), 356–369.

Mascolo, S., Casetti, C., Gerla, M., Sanadidi, M., Wang, R., July 2001. TCP Westwood: Bandwidth estimation for enhanced transport over wireless links. In: Proceedings of the ACM MOBICOM'01, Rome, Italy.

NS-2 Network Simulator. <http://www.isi.edu/nsnam/ns/>.

Papadimitriou, P., Tsaoussidis, V., Tsekeridou, S., June 2005. The impact of network and protocol heterogeneity on application QoS. In: Proceedings of the 10th IEEE International Symposium on Computers and Communications (ISCC), Cartagena, Spain.

Paxson, V., 1999. End-to-end packet dynamics. IEEE/ACM Transactions on Networking 7 (3), 277–292.

Rejaie, R., Handley, M., Estrin, D., 2000. Layered quality adaptation for internet video streaming. IEEE Journal on Selected Areas in Communications (JSAC) 18 (12), 2530–2544.

Rejaie, R., Handley, M., Estrin, D., March 1999. RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet. In: Proceedings of the IEEE INFOCOM 1999, New York, USA.

Rhee, I., Ozdemir, V., Yi, Y., April 2000. TEAR: TCP emulation at receivers – Flow control for multimedia streaming. NCSU Technical Report.

Rhee, I., Xu, L., August 2005. Limitations of equation-based congestion control. In: Proceedings of the ACM SIGCOMM 2005, Philadelphia, USA.

Saparilla, D., Ross, K., March 2000. Optimal streaming of layered video. In: Proceedings of the IEEE INFOCOM 2000, Tel-Aviv, Israel.

Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V., April 1998. Real time streaming protocol (RTSP). RFC 2326, IETF.

Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V., July 2003. RTP: A transport protocol for real-time applications. RFC 3550, IETF.

Stevens, W., January 1997. TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. RFC 2001.

TCP Westwood NS-2 Implementation. <http://c3lab.poliba.it/index.php/Westwood:NS2>.

Tsaoussidis, V., Zhang, C., 2002. TCP Real: receiver-oriented congestion control. Computer Networks, Elsevier 40 (4), 477–497.

Tsaoussidis, V., Zhang, C., 2005. The dynamics of responsiveness and smoothness in heterogeneous networks. IEEE Journal on Selected Areas in Communications (JSAC) 23 (6), 1178–1189.

Yang, Y.R., Kim, M.S., Lam, S.S., April 2001. Transient behaviors of TCP-friendly congestion control protocols. In: Proceedings of the IEEE INFOCOM 2001, Anchorage, Alaska, USA.

Yang, Y.R., Lam, S.S., November 2000. General AIMD congestion control. In: Proceedings of the 8th IEEE International Conference on Network Protocols (ICNP), Osaka, Japan.

Panagiotis Papadimitriou obtained a B.Sc. in Computer Science from University of Crete, Greece; and an M.Sc. in Information Technology from University of Nottingham. He is currently a Ph.D. Candidate in Electrical and Computer Engineering in Democritus University of Thrace, Greece. He is a Visiting Lecturer in the Information Management Department of Technological Educational Institute (TEI) of Kavala, Greece. His research interests include transport protocols for multimedia applications, video and voice over IP, congestion control, and space communications.

Vassilis Tsaoussidis received a B.Sc. in Applied Mathematics from Aristotle University, Greece; a Diploma in Statistics and Computer Science from the Hellenic Institute of Statistics; and a Ph.D. in Computer Networks from Humboldt University, Berlin, Germany (1995). He held faculty positions in Rutgers University, New Brunswick, SUNY Stony Brook and Northeastern University, Boston. In May 2003, he joined the Department of Electrical and Computer Engineering of Democritus University of Thrace, Greece. His research interests lie in the area of transport/network protocols, i.e., their design aspects and performance evaluation. He is editor-in-chief in the Journal of Internet Engineering and editor for IEEE Transactions in Mobile Computing, the Journal of Computer Networks (Elsevier), the Journal of Wireless Communications and Mobile Computing and the Journal of Mobile Multimedia. He participates in several Technical Program Committees in his area of expertise, such as INFOCOM, GLOBECOM, ICCCN, ISCC, EWCN, WLN, and several others.

Lefteris Mamatas received a diploma in Electrical and Computer Engineering from Democritus University of Thrace, Greece in 2003, where he is currently a Ph.D. candidate. He worked twice as a research intern at DoCoMo Eurolabs. His research interests lie in the area of Network Protocols in Heterogeneous Environments, QoS/Resource Allocation and Energy Efficiency of Communication Mechanisms. He published 12 conference and 5 journal papers.

Please cite this article in press as: Papadimitriou, P. et al., A receiver-centric rate control scheme for layered video streams in the Internet, J. Syst. Software (2008), doi:10.1016/j.jss.2008.02.067